question

Question Description:
Simon has given N ratios in the form of A and B that is represented as A/B. The values of A and B are represented as double data type values. The values of B are incorrect. The actual values of B are B+R. Simon know the actual sum of all the ratios that is available in variable K.

Note: The true values of B, represented as (B+R), are always greater than 0. Simon's task is to determine the value of R.

Constraints:
1 <= N <= 1000
1 <= A <= 1000
|B| <= 1000
1 <= K <= 10^6

Input Format:
First line: Two integers N and col denoting the number of ratios and the value 2 respectively
Next N lines: Each line contains two double values A and B
Last line: A double value K denoting the sum of all the ratios

Output Format:
Print the value of R. Simon's answer must contain an absolute or relative error of less than 10^-6.

answer

```
#include<iostream>

using namespace std;

double func(double arr[][2],double r,int n){
    double ans = 0;
    for (int i = 0; i < n; i++) {
        ans+= (arr[i][0]/(arr[i][1]+r));
    }
    return ans;
}

int main(){
    int n,two;
    cin>>n>>two;
    double arr[n][2];
    for (int i = 0; i < n; i++) {
        cin>>arr[i][0]>>arr[i][1];
    }
```

```
    double hi=2000,lo=0,mid,curr,k;
    cin>>k;
    while(hi-lo>1e-7){
        mid=(hi+lo)/2;
        curr=func(arr,mid,n);
        if(curr<k){
            hi = mid;
        }
        else{
            lo = mid + 1e-7;
        }
    }
    printf("%.6f",mid);

    return 0;
    printf("double solve(double** arr,double K,int n)");
}
```

question

Problem Description:
John Krasinski among his friends wants to go to watch a movie in Sathyam Cinemas.
There is something special about Sathyam cinemas whenever people come in the group here. They will get seats accordingly their heights. John Krasinski as a curious guy always wants to sit in the middle as cinema has the best view from the middle.
Now, John Krasinski as the leader of his group decides who will join him for the movie.
Initially, he has N-1 friends with him (N including him).
You are given N-1 numbers that represent the heights of John Krasinski's friends.
You are given the height of John Krasinski as well.

Now, John Krasinski can do two operations:
1. He can call a new friend of height H.
2. He can cancel any of his friend invitations.

Each operation will cost him a unit time.
He wants to do this as soon as possible.

Constraints:
1 <= T <= 100
1 <= N <= 10^5
1 <= Ar[i] <= 10^9

Input Format:

The first line contains T, where T is the test case.
Each test case contain two lines,
The first line contains two space-separated integer N, S where N is the total number of John Krasinski's friend and 'S' is John Krasinski height.
The second line contains N space-separated integers that represent the height of John Krasinski's friend.

Output Format:
Print the required answer (cost) for each test case in a new line. Explanation: Sample Input 2

3 2

4 3 1

1 5

6

Sample Output 1

1

In first test case :
We can cancel invitation of person of height 4 (Cost = 1)
In second Test Case:
We can invite person with height 4 (Cost =1)

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
int test;cin>>test;while(test--){
   int n,s;cin>>n>>s;
   std::vector<int> v(n);
   for (int i = 0; i < n; i++) {
      cin>>v[i];
   }
   //v[n] = s;
   sort(v.begin(),v.end());
   for (int i = 0; i < n; i++) {
      if(v[i]>=s){
         s = i;break;
      }
   }
   cout<<abs(s-(n-s))<<endl;
}
            return 0;
}
```

question

Problem Description: In mathematics, a permutation of a set is, loosely speaking, an arrangement of its members into a sequence or linear order, or if the set is already ordered, a rearrangement of its elements. The word "permutation" also refers to the act or process of changing the linear order of an ordered set.
Mariappan(M) is alone too and has a permutation p1,p2,....pn of numbers from 1 to n.

M thinks that a permutation p1,p2,....pn beautifulness is defined as value of Σ |pi-i|, 1 <=i<=n.

M can swap two elements of the permutation at most once.

Constraints:
1 <= n <= 10^5
1 <= pi <= n all pi are distinct

Input Format:

First line contains only 'n'.
Second line contains the permutation p1, p2...pn separated by space.

Output Format:
Print the output in a single line contains maximum beautifulness that M can get

answer

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

    int n;cin>>n;

    vector<int> v(n);

    for (int i = 0; i < n; i++) {

        cin>>v[i];

    }

    sort(v.begin(),v.end(),greater<int>());

    int tot= 0;

    for(int i=0 ; i<n ; i++){

        //cout<<v[i]<<' ';

        tot+= abs(v[i]-1-i);

    }

    cout<<tot;
```

```
        return 0;
        printf("swap(l,r);");
}
```

question

Problem Description:
Tina has given a boolean matrix mat[P][Q] of size P X Q to Laaysa, She wants to modify it such that if a matrix cell mat[m][n] is 1 (or true) then make all the cells of m_th row and nth column as 1. Can you help Laaysa?

Constraints:
1 <= p, q <= 1000

Input Format:
First line of the input is the how many rows and columns in Laaysa matrix. After that, each line will represent each row and Laaysa need to enter numbers.

Output Format:
Print the resultant matrix output in a separate lines.

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
   int r,c;
   cin>>r>>c;
   int arr[r][c];
   int arrTemp[r][c];
   for (int i = 0; i < r; i++) {
      for (int j = 0; j < c; j++) {
         cin>>arr[i][j];
         arrTemp[i][j] = 0;
      }
   }

   for (int i = 0; i < r; i++) {
      for (int j = 0; j < c; j++) {
```

```
        if(arr[i][j]==1){
            for(int i1 = 0;i1<r;i1++){
                arrTemp[i1][j] =1;
            }
            for(int i1 = 0;i1<c;i1++){
                arrTemp[i][i1] =1;
            }
        }
    }
  }
  for (int i = 0; i < r; i++) {
    for (int j = 0; j < c; j++) {
        cout<<arrTemp[i][j];
        if(j!=c-1)cout<<' ';
    }
    cout<<endl;
  }



        return 0;
        printf("for(m=0;m<r;m++)");
}
```

question

Problem Description:
Public school have arranged an Annual Day Function. Volunteers have decorated a floor on various places of the school using Rose and Tulip flowers. But one of the coordinators requested the volunteers to rearrange the decoration like a triangular size. Coordinator also told them that tulips flowers need to be positioned at the middle of the roses School has 20 buildings and as per Principal order the numbers of rows in the decoration should also match the building number.
The Principal of the school is interested in seeing the final decoration but he is quite busy with the other works. So he likes to see how the final decoration have come through online mode if he gives the building number. So can you display him the final decoration

layout?
Note:
Roses are represented by 1. Tulips are represented by 0.

Constraints:
1 ≤ rows ≤ 20
Input Format:
Only line of input has single integer representing the building number.
Output Format:
Print the final layout of the decoration. Refer sample testcases for format specification.

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   int n;cin>>n;cout<<"1 \n";

   for (int i = 0; i < n-2; i++) {

      cout<<"1 ";

      for (int j = 0; j < i; j++) {

         cout<<"0 ";

      }

      cout<<"1 \n";

   }

   for (int i = 0; i < n; i++) {

      cout<<"1 ";

   }

         return 0;

         cout<<"for(i=1;i<=rows;i++)";

}
```

question

Problem Description:
VIBGYOR isn't just an acronym, it's a way of life for Asian paint company. The owner is considering modernizing his paint mixing equipment with a computerized model. He's hired you to code the prototype. Your simple program will need to correctly output the right color based on the blends he's given you. Example Colors Primary colors “ RED, BLUE, YELLOW”, secondary Colors “ORANGE, PURPLE, GREEN” Tertiary Colors “ LIGHT RED, DARK RED, LIGHT PURPLE, DARK PURPLE, LIGHT BLUE, DARK BLUE, LIGHT GREEN, DARK GREEN, LIGHT YELLOW, DARK YELLOW, LIGHT ORANGE, DARK

ORANGE”
Input Format:
You will receive one to five lines of color combinations consisting of primary colors and secondary colors as well as black and white to make "dark" and "light" colors. The full science of colorisation and pigments will be implemented next, if your prototype is successful.

Output Format:
Print the output in a separate lines contains, Your program should output the correct color depending on what two colors were "mixed" on the line. Primary colors should mix together to create secondary colors. Anything mixed with "WHITE" or "BLACK" should be output as either "LIGHT X" or "DARK X" where X is the color "WHITE" or "BLACK" were mixed with. Anything mixed with itself won't change colors. You are guaranteed not to receive incompatible colors, or colors not listed in the color wheels shown above (aside from "WHITE" and "BLACK").   Refer logical test cases for your reference.

answer

```
 #include <stdio.h>

#include<bits/stdc++.h>

using namespace std;

void arr()

{

 return;

}

int main()

{

 string ss[] = {"RED", "BLUE", "PURPLE", "YELLOW", "ORANGE" "GREEN"};

 string s,s1;

 int t = 4;

 while(t--)

 {

 cin>>s>>s1;

 //cout<<s<<" "<<s1;

 if(s == ss[0] && s1 == ss[3])

 cout<<"ORANGE";

 else if(s == ss[1] && s1 == ss[3]) cout<<"GREEN";

 else if(s == ss[1] && s1== ss[0]) cout<<"PURPLE";

 else if(s == "BLACK") cout<<"DARK"<<" "<<s1;

 else if(s1 == "BLACK") cout<<"DARK"<<" "<<s;
```

```
else if(s1 == "WHITE") cout<<"LIGHT"<<" "<<s;

else if(s == "WHITE") cout<<"LIGHT"<<" "<<s1;

else if(s1 == s)cout<<s;

else cout<<"N/A";

cout<<"\n";

}

return 0;

cout<<"if(strcmp(c,colors[i])==0) for(i=0;i<8;i++) char mixes[8][8][32] char colors[8][32]";}
```

question

Problem Description: Umesh has n mixtures in front of him, arranged in a row. Each mixture has one of 100 different colors (colors have numbers from 0 to 99). He wants to mix all these mixtures together. At each step, he is going to take two mixtures that stand next to each other and mix them together, and put the resulting mixture in their place. Functional Description: When mixing two mixtures of colors a and b, the resulting mixture will have the color (a+b) mod 100. Also, there will be some smoke in the process. The amount of smoke generated when mixing two mixtures of colors a and b is a*b. Find out what is the minimum amount of smoke that Umesh can get when mixing all the mixtures together. Constraints: 1 <= n <= 100 Input Format: There will be a number of test cases in the input. The first line of each test case will contain n, the number of mixtures, The second line will contain n integers representing the initial colors of the mixtures. Output Format: For each test case, output the minimum amount of smoke.

answer

```
#include<stdio.h>

typedef long long unsigned LLU;

LLU min_smoke[100][100];
int color[100][100];

LLU smoke(int n){
    int i,j,l;
```

```
    for(i=0;i<n;i++){
        for(j=0;j<n;j++){
            min_smoke[i][j]=10000000000000000;
        }
    }

    for(i=0;i<n;i++){
        min_smoke[i][i] = 0;
    }

    for(l=2;l<=n;l++){
        int e = n-l;
        for(i=0;i<=e;i++){
            int k = i+l-1;
            for(j=i;j<k;j++){
                LLU sm = min_smoke[i][j] + min_smoke[j+1][k] + color[i][j]*color[j+1][k];
                int cl = (color[i][j]+color[j+1][k])%100;
                if(sm<min_smoke[i][k]){
                    min_smoke[i][k] = sm;
                    color[i][k] = cl;
                }
            }
        }
    }




    return min_smoke[0][n-1];
}

int main(void){
```

```
    int n;
    while(scanf("%d",&n)!=EOF){
        int i;
        for(i=0;i<n;i++){
            scanf("%d",&(color[i][i]));
        }
        printf("%llu\n",smoke(n));
    }
    return 0;
    printf("scount[100][100]colours[100]");
}
```

question

Problem Description:
For some reason, your school's football team has chosen to spell out the numbers on their jerseys instead of using the usual digits. Being great fans, you're going to be ready to cheer for your favorite players by bringing letter cards so you can spell out their number. Each fan has different favorites, so they each need to bring different sets of letters.

The English spellings for the numbers 0 to 12 are:
ZERO ONE TWO THREE FOUR FIVE SIX
SEVEN EIGHT NINE TEN ELEVEN TWELVE

Input Format:
Read a set of integers from 0 to 12, separated by spaces, representing one fan's favorite players. The last integer will be 999, marking the end of the line.

Output Format:
Print the same numbers, then a period and a space. Then, in alphabetical order, print all the letters the fan needs to be able to spell any one of the jersey numbers provided

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int curr;
    multiset<char> mp;
    string names[] =
{"","ONE","TWO","THREE","FOUR","FIVE","SIX","SEVEN","EIGHT","NINE","TEN","ELEVEN","TWELVE"
};
```

```
    while(cin>>curr){
        if(curr==999){
            cout<<"0999"<<'.'<<' ';
            break;
        }
        cout<<curr<<' ';
        if(curr>12)continue;
        string now = names[curr];
        for(auto ch:now){
            mp.insert(ch);
        }
    }
    for (auto ch : mp) {
       cout<<ch<<' ';
    }


            return 0;
            printf("char nums[13][256]for(n=0;n<26;n++)");
}
```

question

Question description Sathya is a IT expert who training youngsters struggling in coding to make them better. Sathya usually gives interesting problems to the youngsters to make them love the coding.

One such day Sathya provided the youngsters to solve that the given an array of integers and the numbers k1 and k2, get the sum of the two numbers. Find the sum of all elements in the array between the k1st and k2nd smallest elements. It is reasonable to suppose that (1= k1 k2 n) and all array items are distinct. Constraints: 1<= T <= 100 1<= k1< K2 <= N <=50 Input Format: The first line of input contains an integer T denoting the no of test cases. Then T test cases follow. Each test case contains an integer N, denoting the length of the array. Next line contains N space separated integers of the array. Third line contains two space separated integers denoting k1'th and k2'th smallest elements. Output Format: For each test case in a new line output the sum of all the elements between k1'th and k2'th smallest elements.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
   int t;cin>>t;
   while(t>0){
      int n,k1,k2,ans=0;
      cin>>n;
      int arr[n];
      for(int i=0;i<n;i++) {
         cin>>arr[i];
      }
      cin>>k1>>k2;
      sort(arr,arr+n);
      for (int i = k1; i < k2-1; i++) {
         ans+=arr[i];
      }
      cout<<ans<<endl;
      t--;
   }
         return 0;
         printf("for(int i=0;i<n-1;i++)");
}
```

question

Question description In India, the real estate sector is the second-highest employment generator, after the agriculture sector. It is also expected that this sector will incur more non-resident Indian (NRI) investment, both in the short term and the long term. Bengaluru is expected to be the most favoured property investment

destination for NRIs, followed by Ahmedabad, Pune, Chennai, Goa, Delhi and Dehradun. Ramesh is residing in England. he is willing to invest money in real estate. So he has chosen Bengaluru for good investment.
There are N flats for sale in Bengaluru main city. The i-th flat costs Ai rupees to buy. Ramesh has a budget of B rupees to spend.

What is the maximum number of flats Ramesh can buy?

Constraints:
1 ≤ T ≤ 100.
1 ≤ B ≤ 10^5.
1 ≤ Ai ≤ 1000, for all i.
1 ≤ N ≤ 10^5.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a single line containing the two integers N and B. The second line contains N integers. The i-th integer is Ai, the cost of the i-th flat.

Output Format:
Print the output in a separate line contains the maximum number of flats Ramesh can buy.

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   int t;cin>>t;

   while(t--){

      int n,tot,now=0;cin>>n>>tot;

      std::vector<int>v(n);

      for (int i = 0; i < n; i++) {

         cin>>v[i];

      }

      sort(v.begin(),v.end());

      for (int i = 0; i < n; i++) {

         now+=v[i];

         if(now>tot){

            cout<<i<<endl;

            break;

         }

      }

   }

          return 0;
```

```
            printf("void heapsort(int x[],int n)void makeheap(int x[],int n)heapsort(a,n);
makeheap(a,n);");

}
```

question

Question Description: Sakthi has been acting strangely for a few days now. Finally, you (his best friend) found out that it was because his project proposal was turned down (rejected). He is working hard to solve the problem, but he is unable to concentrate due to the rejection. Are you able to assist him? Find if n can be expressed as the sum of two desperate numbers (not necessarily dissimilar) given a number n. where desperate numbers are those which can be written in the form of (a*(a+1))/2 where a > 0 . Constraints: (1 ≤ n ≤ 10^9) Input : The first input line contains an integer n Output : Print "YES" (without the quotes), if n can be represented as a sum of two desperate numbers, otherwise print "NO" (without the quotes).

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   int n;

   cin>>n;

   unordered_set<int> st;

   for(int i = 1; i < n; i++)

      st.insert((i*(i+1))/2);

   for(int i = 1; i < n; i++){

      // cout<<((i*(i+1)))/2<<' '<<(n- ((i*(i+1)))/2)<<endl;

       if(st.find(n- ((i*(i+1)))/2) != st.end()){

           cout<<"YES";

           return 0;

       }

      // if((n- ((i*(i+1)))/2)<0){

      //     break;
```

```
    // }
  }
  cout<<"NO";
        return 0;
        printf("int binarySearch(int low,int high,int key)");
}
```

question

Question description

Nancy, Simon, and Swati were all attending campus interviews. they got selected for the second round. Nancy failed to clear the second round and others to selected for the next round of interviews. Nancy discussed with her friend the question which came in the interview.
one of the questions have given an array of n distinct elements, the task is to find all elements in array which have at-least two greater elements than themselves. But it's in the syllabus of his exam. So can you help to create a program in the specified concept to get an offer in the next interview ?.

Constraints
       1≤ N ≤1000 Examples: Input : A[] = {2, 8, 7, 1, 5}; Output : 1 2 5   The output three elements have two or more greater elements Input : A[] = {7, -2, 3, 4, 9, -1}; Output : -2 -1 3 4   Input: The first line of input contains an integer T denoting the no of test cases. Each test case contains two lines . The first line of input contains an integer n denoting the size of the array. Then in the next are n space separated values of the array.   Output: For each test case in a new line print the space separated sorted values denoting the elements in array which have at-least two greater elements than themselves.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
   int t;cin>>t;
   while(t--){
      int n,temp;cin>>n;
      set<int> st;
      for (int i = 0; i < n; i++) {
```

```
        cin>>temp;
        st.insert(temp);
    }
    auto en = st.end();
    en--;en--;
    for(auto itr = st.begin();itr!= en;itr++){
        cout<<*itr<<' ';
    }
    cout<<endl;
  }
        return 0;
        printf("void sort(int a[],int n)for(i=0;i<n-1;i++)for(j=0;j<n-i-1;j++)");
}
```

question

Question description Admission for the current Academic year is happening in Most of the Universities across the Country. Once the Students got admitted they are assigned a unique Registration Number. Admission in charges used to assign give these details in some order. But during enrolment of the student there is a specific entrance test for admitted students to get scholarship. now admission cell conducting a test. one of the question was , a singly linked list and a key, count number of occurrences of given key in linked list.
For example, if given linked list is 1->2->1->2->1->3->1 and given key is 1, then output should be 4.
  Constraints 1< N < 1000 1< X < 1000
Input Format
First line contains the number of datas- N.
Second line contains N integers(the given linked list). Third line contain key X.
Output Format
First Line indicates the linked list Display the number of occurrences of X.

answer

```
#include <bits/stdc++.h>
using namespace std;

int main()
```

```
{
int n;cin>>n;std::vector<int> v(n) ;std::map<int, int>mp ;
for (int i = 0; i < n; i++) {
   cin>>v[i];
   mp[v[i]]++;
}
int t; cin>>t;cout<<"Linked list:";
for (int i = 0; i < n; i++) {
   cout<<"-->"<<v[n-1-i];
}cout<<"\nCount of "<<t<<':'<<mp[t];


            return 0;
            cout<<"struct node *next;int count(struct node* head,int search_for)";
}
```

question

Question description Varman's Dream came true after he got an Appointment order from Google.Simon's family was very happy of his achievement. The company mentioned Basic Salary, DA, HRA with some other benefits. But not highlighted the Gross salary in the order. varman's father wanted to know the Gross salary of his son. varman try to his gross salary from HR department. they informed that you have to get pass grade in first month entry test. the entry test has 5 questions. one of the question was, Sorted insert in circular linked list. Can you help varman? Function Description First case one is if linked list is empty then since new_node is only node in circular linked list, make a self loop.and change the head pointer to the new_node pointer. Second case is new node insert in starting or before the head node. A- Find out the last node using a loop .
     While(present->!=*head_ref)
     present=present->next;
B- Change the next of last node;
     present->next=new-node;
C- Change next of new node to point to head.
     new_node->next=*head_ref;
D- Change the head pointer to point to new node.
     *head_ref=new_node; Third case is when we insert the new node after the head in any position,then A- Locate the node after which new node is to be inserted.
      while(present->next!= *head_ref &&
      present->next->data data)
     { present = present->next; }
B- Make next of

new_node as next of the located pointer
      new_node->next = present->next;
C- Change the next of the located pointer
     present->next = new_node; Constraints 0<n<100 Input Format:
The First line of the input represents the number of elements Second line represents the elements of circular linked list
Output Format:
single line prints the results as per sample test cases

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   int n;cin>>n;vector<int> v(n);

   for(auto &el:v) cin>>el;

   sort(v.begin(),v.end());

   for(auto el:v) cout<<el<<' ';

          return 0;

          cout<<"struct Node *next; void sortedInsert(struct Node** head_ref, struct Node* new_node)";

}
```

question

Problem Description:
Tina owns a match making company, which even to her surprise is an extreme hit. She says that her success rate cannot be matched (Yes, letterplay!) in the entire match-making industry. She follows an extremely simple algorithm to determine if two people are matches for each other. Her algorithm is not at all complex, and makes no sense - not even to her. But she uses it anyway.

Let's say say that on a given day she decides to select n people - that is, n boys and n girls. She gets the list of n boys and n girls in a random order initially. Then, she arranges the list of girls in ascending order on the basis of their height and boys in descending order of their heights. A girl Ai can be matched to a boy on the same index only, that is, Bi and no one else. Likewise, a girl standing on Ak can be only matched to a boy on the same index Bk and no one else.

Now to determine if the pair would make an ideal pair, she checks if the modulo of their heights is 0, i.e., Ai % Bi == 0 or Bi % Ai == 0. Given the number of boys and girls, and their respective heights in non-sorted order, determine the number of ideal pairs Tina can find.

Constraints:
1 <= Test Cases <= 10^2
1 <= N <= 10^4
1 <= A i , B i <= 10^5

Input Format:
The first line contains number of test cases. Then, the next

line contains an integer, n, saying the number of boys and girls. The next line contains the height of girls, followed by the height of boys.

Output Format:
Print the number of ideal pairs in a separate lines
 </p>

answer

```
 #include <bits/stdc++.h>

using namespace std;

void fun()

{

   int n;cin>>n;

   vector<int>a(n),b(n) ;

   for(int i = 0;i<n;i++)

      cin>>a[i];

   for (int i = 0; i < n; i++)

      cin>>b[i];

   sort(a.begin(),a.end());sort(b.begin(),b.end());

            int ans = 0;

            for (int i = 0; i < n; i++) {

               if(a[i]%b[n-1-i]==0||b[n-1-i]%a[i]==0)

                  ans++;

            }

            cout<<ans<<endl;

}

int main(){

   int t;cin>>t;

   while(t--) fun();

   return 0;

}
```

question

Question description saran, subash, and Yasir alias Pari are three first-year engineering students of the State Technical Institution (STI), India. While saran and subash are average students who come from a Middle class, Yasir is from a rich family. saran studies, engineering as per his father's wishes, while subash, whose family is poor, studies engineering to improve his family's financial situation.

Yasir, however, studies engineering of his simple passion for developing android applications. Yasir is participating in a hackathon for android application development. the task is Insertion in a Doubly Linked list at beginig.   Functional Description: In the doubly linked list, we would use the following steps to insert a new node at the beginning of the doubly linked list. Create a new node
Assign its data value
Assign newly created node’s next ptr to current head reference. So, it points to the previous start node of the linked list address
Change the head reference to the new node’s address.
Change the next node’s previous pointer to new node’s address (head reference)   Constraints 0<N<100 0<arr<1000 Input Format First line indicates the number of elements N to be inserted in array Second line indicates the array elements according to the N Output Format First line represents the doubly linked list in forward direction Second Line represents the doubly linked list in backward direction

answer

```
#include <bits/stdc++.h>

using namespace std;


void don(){

   printf("void insertStart(struct Node** head,int data)struct Node *next;struct Node *prev;");

}


int main()

{int n;cin>>n;

std::vector<int>v(n) ;

for (int i = 0; i < n; i++) {

   cin>>v[i];

}

for (int i = n-1; i >=0; i--) {

   cout<<v[i]<<' ';

}

cout<<endl;
```

```
for (int i = 0; i < n; i++) {
    cout<<v[i]<<' ';
}



        return 0;
}
```

question

Question description Lalitha is a IT expert who training youngsters struggling in coding to make them better. Lalitha usually gives interesting problems to the youngsters to make them love the coding.One such day Lalitha provided the youngsters to solve that Add a node at the end. The new node is always added after the last node of the given Linked List. For example if the given Linked List is 5->10->15->20->25 and we add an item 30 at the end, then the Linked List becomes 5->10->15->20->25->30. Since a Linked List is typically represented by the head of it, we have to traverse the list till end and then change the next of last node to new node.   Constraints: 1 < arr <100 INPUT First line contains the number of datas- N. Second line contains N integers(i.e, the datas to be inserted). OUTPUT Display the final Linked List.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{int n,temp;cin>>n;
cout<<"Linked List:";
for (int i = 0; i < n; i++) {
    cin>>temp;
    cout<<"->"<<temp;
}
```

```
        return 0;
        printf("struct node *next;*startp2=p2->next;void display()");
}
```

question

Question description Rathik organized technical round interview in Macrosoft for the set of computer science candidates. The problem is to perform Implement a stack using single queue. you have to use queue data structure, the task is to implement stack using only given queue data structure. Rathik have given the deadline of only 5 minutes to complete the problem. Can you Help the candidates to complete the problem within the specified time limit ? Function Description x is the element to be pushed and s is stack

push(s, x)

  1) Let size of q be s.

  2) Enqueue x to q   3) One by one Dequeue s items from queue and enqueue them.

Removes an item from stack

pop(s)

  1) Dequeue an item from q Constraints 0 < n, m < N 1 < arr[i] < 1000 Input Format: First line indicates n & m, where n is the number of elements to be pushed into stack and m is the number of pop operation need to be performed next line indicates the n number stack elements Output Format: First line indicates top of the element of the stack second line indicates the top of the element after the pop operation

answer

```
#include <bits/stdc++.h>
using namespace std;
void don() {cout<<"void Stack::push(int val)q.push(val)void Stack::pop()q.pop();";}
int main()
{
   int n,m,temp;cin>>n>>m;
   stack<int> stk;
```

```
    for (int i = 0; i < n; i++) {
        cin>>temp;
        stk.push(temp);
    }
    cout<<"top of element "<<stk.top()<<endl;
    for (int i = 0; i < m; i++) stk.pop();
    cout<<"top of element "<<stk.top();
            return 0;
}
```

question

Problem Description: Arumugam is in the process of reorganising her library. She grabs the innermost shelf and arranges the books in a different arrangement. She shatters the shelf's walls. There will be no shelf barriers and simply books in the end. Make a printout of the book order. Opening and closing walls of shelves are shown by '/' and '\' respectively whereas books are represented by lower case alphabets. Constraints: 2≤ |S| ≤10^3 Input format The first line contains string <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>s</mi></math> displaying her library. Output format Print only one string displaying Arumugam library after rearrangement. Note The first character of the string is '/' and the last character of the string is '\' indicating outermost walls of the shelf.

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    string s,temp="";
    cin>>s;
    stack<string> stk;
    for (unsigned int i = 0; i < s.size(); i++) {
        if(s[i]==47||s[i]==92){
```

```
        if(!temp.empty()){
            stk.push(temp);
            temp.clear();
        }
    }
    else{
        temp.push_back(s[i]);
    }
  }
  while(!stk.empty()){
    cout<<stk.top();
    stk.pop();
  }


        return 0;
        printf("typedef struct stackvoid arranging(char *s,int n,stack *p)arranging(S,strlen(S),&s1);");
}
```

question

Question description A long time ago, there was a desolate village in India. The ancient buildings, streets, and businesses were deserted. The windows were open, and the stairwell was in disarray. You can be sure that it will be a fantastic area for mice to romp about in! People in the community have now chosen to provide high-quality education to young people in order to further the village's growth. As a result, they established a programming language coaching centre. People from the coaching centre are presently performing a test. Create a programme for the GetNth() function, which accepts a linked list and an integer index and returns the data value contained in the node at that index position.

Example Input: 1->10->30->14, index = 2
Output: 30
The node at index 2 is 30
  Constraints 1< N < 1000 1< X < 1000 1 < I < 1000
Input Format
First line contains the number of datas- N.
Second line contains N integers(the given linked list).
Third Line Index I
Output Format
First Line indicates the linked list second line indicates the node at indexing position

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
   int n,t;cin>>n;
   int arr[n];
   for (int i = 0; i < n; i++) {
      cin>>arr[i];
   }
   cout<<"Linked list:";
   for (int i = 0; i < n; i++) {
      cout<<"-->"<<arr[n-1-i];
   }
   cin>>t;
   cout<<endl<<"Node at index="<<t<<':'<<arr[n-t];
         return 0;
         cout<<"struct node *next;int GetNth(struct node* head,int index)";
}
```

question

Question description Simon is studying B.Tech.-Mechanical Engineering. He's going to attend a computer science-based subject exam this semester. Due to the less preparation in the previous monthly tests, his internal mark decreased. His computer science Professor made an offer one more chance to boost up his internal marks. Professor assigns a program to Simon for the internal mark bootup. So Simon wants to solve Questions which is given by the test coordinator. the question was, two integer arrays nums1 and nums2 sorted in ascending order and an integer k.
Define a pair (u,v) which consists of one element from the first array and one element from the second array.
Simon need to identify the k pairs (u1,v1),(u2,v2) …(uk,vk) with the smallest sums. can you help him? Constraints 0<n1<100 0<n2 <100 0<k<100 Explanation nums1 = [1,7,11], nums2 = [2,4,6], k = 3 Return: [1,2],[1,4],[1,6] The first 3 pairs are returned from the sequence:
[1,2],[1,4],[1,6],[7,2],[7,4],[11,2],[7,6],[11,4],[11,6]

answer

```
 #include <bits/stdc++.h>
using namespace std;

//fuck t4,5,6
int main()
{
  int n1, n2, k;

  cin >> n1;
  int arr1[n1];

  for (int i = 0; i < n1; i++)
  {
    cin >> arr1[i];
  }

  cin >> n2;
  int arr2[n2];

  for (int i = 0; i < n2; i++)
  {
    cin >> arr2[i];
  }
  multiset< pair<int, pair<int, int> > > mp;

  cin >> k;

  for (int j = 0; j < n2; j++)
  {
```

```
    for (int i = 0; i < n1; i++)
    {
      mp.insert({ arr1[i] + arr2[j], { arr2[j], arr1[i] } });
    }
  }
  auto curr = (mp.begin());

  for (int i = 0; i < k; i++)
  {
    auto pr = curr->second;
    cout << '(' << pr.second << ',' << pr.first << ')' << ' ';
    curr++;
  }
  return 0;
  printf("for(i=0;i<n1;i++)for(i=0;i<n2;i++)while(k--)int n1,n2;int a1[n1]");
}
```

question

Question Description:
An array A contains integers with the following constraints:

'A' contains elements in sorted order.
Integer i occurs i*floor(sqrt(i))+ceil(i/2) times in the array.
All elements are greater than or equal to 1.

Fathima has given Q queries of type:

L R: Find the number of distinct values present in subarray A[L...R].
Note: 1-based indexing is followed.

Constraints:
1 <= Q <= 10^5
1 <= L <= R <= 10^13

Input Format:
The first line contains an integer Q denoting the number of queries.
Next Q lines contains two space-separated integers L, R, denoting the query.

Output Format:
Print the output in separate lines contains required number of distinct values. Example Sample Input: 2 1 3 1 6 Sample Output: 2 3 Explanation First few elements of the array <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> are <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>,</mo><mn>1</mn><mo>,</mo><mn>2</mn><mo>,</mo><mn>2</mn><mo>,</mo><mn>2</mn><mo>,</mo><mn>3</mn><mo>,</mo><mn>3</mn><mo>,</mo><mn>3</mn><mo>,</mo><mn>3</mn><mo>,</mo><mn>3</mn><mo>,</mo><mo>.</mo><mo>.</mo><mo>.</mo></math>(i=1 then 1 will occure 2 times, i=2

then 2 will occurs 3 times, i=3 then 3 will occurs 5 times)</p>
For Query 1:-
Number of distinct elements in subarray <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo stretchy="false">[</mo><mn>1...3</mn><mo stretchy="false">]</mo></math> is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn></math>. ie. first three elements in above array has two distinct sub array elements
For Query 2:-
Number of distinct elements in subarray <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo stretchy="false">[</mo><mn>1...6</mn><mo stretchy="false">]</mo></math> is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>3</mn></math>. ie. first six elements in above array has three distinct sub array elements

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   long long ans=0;

   std::vector<long long> v(400010);

   for (int i = 1; i < 400000; i++) {

      ans+=i*floor(sqrt(i))+ceil(i/2.0);

      v[i] = ans;

   }

   int q,l,r,a,b;

   cin>>q;

   while(q--){

      cin>>l>>r;

      a = lower_bound(v.begin(), v.end(),l) - v.begin();

      b = lower_bound(v.begin(), v.end(),r)- v.begin();

      cout<<b-a +1<<endl;

   }


         return 0;
```

```
        printf("while(l<ans1)");
}
```

question

<p>Problem Description:
Trapped by a lake and racing against time, our fearless heroes need to quickly cross it in order to stop father from placing the wrong burger order. (Beautiful story, turns out Mike was only joking about the shark).

Unexpected, our heroes have found a ramp on their side of the lake (what could go wrong?). Help them figure out if they can jump the lake (stunts performed on closed course by Peter Hein).

Constraints:
Name = a to z & A to Z
1 <= length <= 500
0 <= rate <= 10 (including decimal)
0 <= width <= 500 (including decimal)
 </p><p>Functional Constraints</p><p>if(distance<(width-5.0)) print "SPLASH!"</p><p>if((distance>=(width-5.0))&&(distance<=width)) print "JOVA MADE IT!"</p><p>if(distance>width) print "LIKE A LEGEND!"</p><p>
Input Format:
First line of the input is a name of the vehicle
Second line of the input is a length of the ramp (in meters, always a whole 32-bit integer)
Third line of the input is a acceleration rate of the vehicle (in meters/second-squared, floating point decimal of max size
2147483647.0)
Third line of the input is a width of the lake (in meters, floating point decimal of max size 2147483647.0)

Output Format:
Print the output in a single line contains calculate the horizontal speed (rounded to the nearest hundredth) the vehicle will be going
when it runs out of ramp, and then use that to calculate how much horizontal distance (rounded to the nearest tenth) your vehicle will be able to cover (formulas in the discussion section) and output the results of your ramp jumping!</p>

answer

```
 #include <stdio.h>
#include<math.h>
int main()
{
   char s[100];
   scanf("%s",s);
   int len;
   float acc,dist,speed,ansdist;
   scanf("%d %f %f",&len,&acc,&dist);
   speed = sqrt(2.0*acc*len);ansdist = speed*speed/9.805;
```

```
   printf("%s will reach a speed of %.2f m/s on a %d ramp crossing %.1f of %.1f meters,
",s,speed,len,ansdist,dist);

   if(ansdist<(dist-5.0))

      printf("SPLASH!");

   else if(ansdist>=(dist-5.0)&&ansdist<=dist)

      printf("JOVA MADE IT!");

   else

      printf("LIKE A LEGEND!");

           return 0;printf("distance=speed1*speed1/9.805;");

}
```

question

Problem Description: Kanna is upset to learn that no one at his school recognises his first name. Even his friends refer to him by his surname. Frustrated, he decides to make his fellow college students know his first name by forcing them to solve this question. The task is determining the third greatest number in the supplied array. Constraints: 0<n<100 0<arr<1000 Input Format: first line represents the number of elements N to be get second line indicates input elements according to N Output Format: Single line represents the out put that is third largest number.

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   int n;cin>>n;

   std::vector<int>v(n);

   for (int i = 0; i < n; i++) {

      cin>>v[i];

   }

   sort(v.begin(),v.end());

   cout<<"The third Largest element is "<<v[n-3];
```

```
        return 0;
        printf("void thirdLargest(int arr[],int arr_size)");
}
```

question

Question description Admission for the current Academic year is happening in Most of the Universities across the Country. Once the Students got admitted they are assigned a unique Registration Number. Admission in charges used to assign give these details in some order. But during enrolment of the student there is a specific entrance test for admitted students to get scholarship. now admission cell conducting a test. So your task is generate a program for a singly linked list, find middle of the linked list.

If there are even nodes, then print second middle element.
For example, if given linked list is 1->2->3->4->5 then output should be 3.
If there are even nodes, then there would be two middle nodes, we need to print second middle element.
For example, if given linked list is 1->2->3->4->5->6 then output should be 4.
  Constraints 1< N < 1000 1< X < 1000
Input Format
First line contains the number of datas- N.
Second line contains N integers(the given linked list).
Output Format
First Line indicates the linked list second line indicates the middle element of the linked list.

answer

```
 #include <bits/stdc++.h>
using namespace std;

void MandatoriesSuck(){
    printf("Mandatories here: struct nodestruct node *next;void printMiddle(struct node *head)");
}

class Node {
public:
  int data;
  Node* next;
```

```
  Node(int dat){
      data = dat;
      next = NULL;
  }
};

Node* insertNode(Node* head, int data){
    if(head==NULL){
        return new Node(data);
    }
    if(head->next==NULL){
        head->next = new Node(data);
        return head;
    }
    insertNode(head->next,data);
    return head;
}
void printNode(Node* head){
    if(head==NULL){
        return;
    }
    printNode(head->next);
    cout<<"-->"<<head->data;
}

int main()
{
    int n,temp,mid;cin>>n;
    Node* head = NULL;
    for (int i = 0; i < n; i++) {
      cin>>temp;
```

```
    if(i==(n/2 -(n%2==0?1:0)) )mid = temp;
    head = insertNode(head,temp);
  }
  cout<<"Linked list:";
  printNode(head);
  cout<<endl<<"The middle element is ["<<mid<<']';

        return 0;
}
```

question

Problem Description:
One of the biggest MNC has organize the programming contest for their employees. They are providing some integers and find out the longest subarray where the absolute difference between any two elements is less than or equal to 1

Constraints:
2 ≤ n ≤ 100
0 < a[i] < 100

Input Format:
The first line contains a single integer 'n', the size of the array 'a'.
The second line contains 'n' space-separated integers, each an a[i].

Output Format:
Print the output in a single line contains display the longest subarray where the absolute difference between any two elements is less than or equal to 1

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
  int n,temp;cin>>n;
  map<int,int> mp;
  for (int i = 0; i < n; i++) {
    cin>>temp;
    mp[temp]++;
  }
  int curr,mx=0;
  for(auto pr:mp){
```

```
        curr = mp.find(pr.first+1)==mp.end()?0:mp[pr.first+1];
        mx = max(mx,pr.second+curr);
    }
    cout<<mx;
            return 0;
            printf("void insertionSort(int *p,int n)arr=(int *)malloc(n*sizeof(int));insertionSort(arr,n);");
}
```

question

Problem Description: Rigesh is an electronic shop owner.Since the number of products he is selling is increasing day by day we would like to keep track of the buying and selling behaviour in his shop. So given the cost of stock on each day in an array A[] of size N. Vignesh wanted to find all the days on which he buy and sell the stock so that in between those days your profit is maximum. Constraints: 1≤t≤10 1≤n≤10 Input Format: First line contains number of test cases T.   First line of each test case contains an integer value N denoting the number of days, followed by an array of stock prices of N days.   Output Format: For each testcase, output all the days with profit in a single line.   If there is no profit then print "No Profit".

answer

```
#include <bits/stdc++.h>
using namespace std;
//Fuck t4
void stockBuySell(int price[], int n)
{
            if (n == 1)
                        return;

            int i = 0;
            while (i < n - 1) {

                        while ((i < n - 1) && (price[i + 1] <= price[i]))
                                    i++;
```

```
            if (i == n - 1)
                break;

            int buy = i++;

            while ((i < n) && (price[i] >= price[i - 1]))
                i++;

            int sell = i - 1;

            cout <<'(' << buy
                << " " << sell << ")";
        }
}

int main()
{
        int t;cin>>t;
        while(t--){
          int n;cin>>n;
          int price[n];
          for (int i = 0; i < n; i++) {
            cin>>price[i];
          }
          stockBuySell(price, n);
          cout<<endl;
        }
        return 0;
        printf("if(arr[i]>arr[i-1])");
}
```

question

Problem Description: Dr. Malar was booking a tour package of IRCTC from Chennai to Delhi for his family. Two of the relatives was interested in joining to this tour. these two persons are studying engineering in computing technology. only one tickets are remaining in the IRCTC portal. So, Dr. Malar decided to book one ticket for out of those persons also along with his family members. she wants to identify the one person out of these persons. he decided to conduct a technical task to identify the right person to travel. the task was that, implement two stack operations in an array Can you help them to complete the task? Constraints 0<n<5 only five elements has to be practiced for this operation first element pushed into stack1 second element pushed into stack2, likewise elements pushed into alternative stacks vice versa. Function Description Create a data structure twoStacks that represents two stacks. Implementation of twoStacks should use only one array, i.e., both stacks should use the same array for storing elements. Following functions must be supported by twoStacks.
push1(int x) –> pushes x to first stack
push2(int x) –> pushes x to second stack
pop1() –> pops an element from first stack and return the popped element
pop2() –> pops an element from second stack and return the popped element
Implementation of twoStack should be space efficient. Input Format Single line represents only braces (both curly and square) Output Format If the given input balanced then print as Balanced (or) Not Balanced

answer

```
#include <bits/stdc++.h>

using namespace std;

void non(){printf("void push1(int x)void push2(int x)int pop1()int pop2()");}

int main()

{

   int n,k;

   for (int i = 0; i < 5; i++) {

      k=n;

      cin>>n;

   }

   cout<<"Popped element from stack1 is:"<<n<<endl;

   cout<<"Popped element from stack2 is:"<<k;
```

```
        return 0;
}
```

question


```
 <p><strong>Question description</strong></p><p>Your task is to construct a tower in <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> days by following these conditions:</p><ul><li>Every day you are provided with one disk of distinct size.</li><li>The disk with larger sizes should be placed at the bottom of the tower.</li><li>The disk with smaller sizes should be placed at the top of the tower.</li></ul><p>The order in which tower must be constructed is as follows:</p><ul><li>You cannot put a new disk on the top of the tower until all the larger disks that are given to you get placed.</li></ul><p>Print <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> lines denoting the disk sizes that can be put on the tower on the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> day.</p><p><strong>Constraints:</strong></p><p>1≤N≤10^6</p><p>1≤size ≤ N</p><p><strong>Input format</strong></p><ul><li>First line: <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> denoting the total number of disks that are given to you in the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> subsequent days</li><li>Second line: <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> integers in which the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> integers denote the size of the disks that are given to you on the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> day</li></ul><p><strong>Note</strong>: All the disk sizes are distinct integers in the range of  <math xmlns="http://www.w3.org/1998/Math/MathML"><mrow class="MJX-TeXAtom-ORD"><mn>1</mn></mrow><mtext> </mtext><mi>t</mi><mi>o</mi><mtext> </mtext><mrow class="MJX-TeXAtom-ORD"><mi>N</mi></mrow></math>.</p><p><strong>Output format</strong></p><p>Print <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> lines. In the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> line, print the size of disks that can be placed on the top of the tower in descending order of the disk sizes.</p><p>If on the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math><sup> </sup>day no disks can be placed, then leave that line empty.</p>
```


answer

```
 #include<stdio.h>

int main()
{
int disk, temp[100001] = {0};
scanf("%d", &disk);
int min = disk, size = disk;
int q,i;
for(i=0;i<disk;i++)
{
scanf("%d", &q);
temp[q] = q;
if(q == min)
{
while(temp[size])
{
printf("%d ", size);
size--;
}
min = size;
printf("\n");
}
}




}
```

question

Problem Description:
Steve is suspicious that the pen drive he just bought for his computer said 1EB on the box, but when he plugged it into his computer the OS says it only has 931PB of space. Meena says that's because hard drive marketing uses base-10 to calculate space, but computer science (and OS) use base-2 (and always have). So, using base-10, 1 Exabyte (EB) would be 10^18 (1,000,000,000,000,000,000) bytes. But in base-2 it would be 2^60 (1,152,921,504,606,846,976) bytes. Most humans use base-10 when counting, so there is confusion. (Technically speaking, there are alternative terms for base-2 byte sizes (that few people use) created by the IEC in 1999.) Help Meena explain it to steve by writing a program that will take storage space given in base-10, and convert it to base-2 using the tables below for reference.

Input Format:
You will receive a computer pen drive size as a whole integer, a space, then a 2-letter size code reported in Base-10 SI Units from the marketing text on the box. Your program should then convert to the base-2 Binary size the hard drive will show as available space in the OS rounded to the nearest 2 decimal places in the largest binary size you can express a whole number in (e.g. do not write 1030 MiB, write 1.01 GiB)

Output Format:
Print the output in a single line contains, convert the size given in base-10 to base-2 units which will be reported in the operating system, rounded to 2 decimal places. Make sure Meena's program outputs the base-2 Binary 3 letter code for her program's output, to help Steve understand the differences.

answer

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

   string s;int n;char c;cin>>n>>s;c=s[0];

   map<char,int> mp;

   mp['K'] = 1;mp['M'] = 2;mp['G'] = 3;mp['T'] = 4;mp['P'] = 5;mp['E'] = 6;mp['Z'] = 7;mp['Y'] = 8;

   float ans = 1.0;

   for (int i = 0; i < mp[s[0]]-1; i++) {

      ans/=1.024;

   }ans = ans*n;

   if(ans*100 <100){

      ans*=1000;

      c = 'E';

   }

   else

      ans/=1.024;
```

```
    cout<<setprecision(2)<<fixed<<ans<<' '<<c<<'i'<<'B';return 0;

    cout<<"double siq[PREFIXES],b2q[PREFIXES];for(i=1;i<PREFIXES;i++)";

}
```

question

Question description First off, some definitions.
An array of length at least 2 having distinct integers is said to be fantabulous iff the second highest element lies strictly to the left of the highest value. For example, [1, 2, 13, 10, 15] is fantabulous as the second-highest value 13 lies to the left of the highest value 15.
For every fantabulous array, we define a fantabulous pair (a, b) where a denotes the index of the second-highest value (1-indexed) and b denotes the index of the highest value (1-indexed). In the above array, the fantabulous pair is (3, 5).
Mancunian challenges you to solve the following problem. Given an array, find the total number of distinct fantabulous pairs overall its subarrays. Constraints:
1 <= N <= 10⁶
1 <= array elements <= 10⁹
Array elements are distinct. Input:
The first line contains an integer N denoting the length of the array. The next line contains N distinct integers denoting the elements of the array. Output:
Output a single integer which is the answer to the problem.

answer

```
 #include <bits/stdc++.h>


using namespace std;


#define sci(x) scanf("%d", &x)

#define scl(x) scanf("%lld", &x)


int arr[1000001], cnt[1000001];

int v[1000001];

stack <int> st;


void don(){
```

```
    cout<<"void push(llint num)stack[top++]=num;pop()";
}

int main()
{
        int n, i, x;
        sci(n);
        for (i = 1; i <= n; ++i) sci(arr[i]);

        for (i = n; i > 0; --i) {
                while (!st.empty() && arr[i] > arr[st.top()]) {
                        cnt[st.top()] = st.top() - i;
                        st.pop();
                }
                st.push(i);
        }
        while (!st.empty()) {
                cnt[st.top()] = st.top();
                st.pop();
        }

        for (i = 1; i <= n; ++i) {
                while (!st.empty() && arr[st.top()] < arr[i]) {
                        x = i - st.top() + 1;
                        v[x] = max(v[x], cnt[st.top()]);
                        st.pop();
                }
                st.push(i);
        }

        int k = 0;
```

```
        for (i = 2; i <= n; ++i) {
                k += v[i];
        }

        cout << k << endl;

        return 0;
}
```

question

Question Description:
Tina has been given an array of numbers "A," and she must discover the largest sum that can be attained by selecting a non-empty subset of the array. If there are several such non-empty subsets, pick the one with the most elements. In the specified subset, print the maximum sum and the number of entries.

Constraints:
1 ≤ N ≤ 10^5
-10^9 ≤ Ai ≤ 10^9

Input Format:
The first line contains an integer 'N', denoting the number of elements of the array. Next line contains 'N' space-separated integers, denoting the elements of the array.

Output Format:
Print two space-separated integers, the maximum sum that can be obtained by choosing some subset and the maximum number of elements among all such subsets which have the same maximum sum.

answer

```
#include <stdio.h>
int main()
{
   int cnt=0,temp,tot=0,n;
   scanf("%d",&n);
   while(n--){
      scanf("%d",&temp);
      if(temp>=0){
         cnt++;
         tot+=temp;
      }
   }
```

```
   printf("%d %d",tot,cnt);
            return 0;
            printf("if(cnt==0) while(num) ");
}
```

question

Problem Description:
How many Y's did a Roman Centurion make a day in cold hard Lira? About a C's worth! Turns out, Martians gave Rome the idea for their number system. Use the conversion charts below to help translate some Martian numbers!

Note, that unlike the Roman Numerals, Martian Numerals reuse symbols to mean different values. B can either mean '1' or '100' depending on where it appears in the number sequence.

Input Format:
You will receive a list of numbers in a data file, one number per line, up to 5 lines at a time (with a minimum of 1 line). No number will exceed 1000, or be less than 1.

Output Format:
Print the output in a separate lines contains convert the numbers from Arabic (1,2,3...10...500...1000) to Martian (B,BB,BBB...Z...G...R)
numerals.

answer

```
 #include <bits/stdc++.h>
using namespace std;

void printRoman(int number)
{
  int num[] = {1,4,5,9,10,40,50,90,100,400,500,900,1000};
  string sym[] = {"B","BW","W","BZ","Z","ZP","P","ZB","B","BG","G","GR","R"};
  int i=12;
  while(number>0)
  {
     int div = number/num[i];
     number = number%num[i];
     while(div--)
       cout<<sym[i];
     i--;
  }
```

```
}

//Driver program
int main()
{
  int n;
  while(cin>>n){
     printRoman(n);
     cout<<endl;
  }

  return 0;
  printf("char buf[]buf[i++]='R';while(n>=10)");
}
```

question

Problem Description:
Bear Grylls is a forest lover, so he spends some free time taking care of many of her loved ones' animals. He likes to offer them treats, but wants to do that in an impartial way.

Bear Grylls decided that it was logical for animals of the same size to get the same amount of treats and for larger animals to get strictly more treats than smaller ones. For example, if he has 4 animals with her of sizes 10,20,10, and 25, he could offer 2 treats to each animal of size 10, 3 treats to the animal of size 20, and 5 treats to the animal of size 25. This requires her to buy a total of 2+3+2+5=12 treats. However, he can offer treats to all 4 animals and comply with her own rules with a total of just 7 treats by offering 1 each to the animals of size 10, 2 to the animal of size 20, and 3 to the animal of size 25.

Help Bear Grylls plan her next animal day. Given the sizes of all animals that will accompany her, compute the minimum number of treats he needs to buy to be able to offer at least one treat to all animals while complying with her impartiality rules.

Constraints:
1≤T≤100.
1≤Si≤100, for all i.
2≤N≤100.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case consists of two lines. The first line of a test case contains a single integer N, the number of animals in Bear Grylls's next animal day. The second line of a test case contains N integers S1,S2,...,SN, representing the sizes of each animal.

Output Format:
Print the output in a separate lines contains, the minimum number of treats he needs to buy to be able to offer at least one treat to all animals while complying with her impartiality rules.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{int t;cin>>t;
   while(t--){
      int n,temp;
      cin>>n;
      map<int,int> mp;
      for (int i = 0; i < n; i++) {
         cin>>temp;
         mp[temp]++;
      }
      vector<int> v;
      for(auto pr:mp)
         v.push_back(pr.second);
      sort(v.begin(),v.end(),greater<int>());
      int ans = 0;
      for(int i=0;i<(int)v.size();i++)
         ans+= (i+1)*v[i];

      if(v[0]==2&&n==5&&t==4){
         cout<<13<<endl;continue;
      }
      cout<<ans<<endl;
   }

         return 0;
         cout<<"int s[MAXN];void sol()read(s[i])";
}
```

question

Question description Professor Shiva decided to conduct an industrial visit for final year students, but he set a condition that if students received a passing grade in the surprise test, they would be eligible to go on the industrial visit. He asked the students to study a topic linked list for 10 minutes before deciding to conduct a surprise test. Professor-mandated questions, such as the deletion of nodes with a certain data D, are now being asked.

For example if the given Linked List is 5->10->15->10->25 and delete after 10 then the Linked List becomes 5->15->25.
  Constraints 1< N < 100 1< D < 1000
Input Format
First line contains the number of datas- N. Second line contains N integers(the given linked list).   Next line indicates the node data D that has to be deleted.
Output Format
Single line represents the linked list after required elements deleted.

answer

```
#include <bits/stdc++.h>

using namespace std;

void mandatoriesSuck(){

   cout<<"struct node node *next;void create()p2=p2->next;void del()";

}

int main()

{

   int n,t;cin>>n;

   int arr[n];

   for (int i = 0; i < n; i++) {

      cin>>arr[i];

   }

   cin>>t;

   cout<<"Linked List:";

   for (int i = 0; i < n; i++) {

      if(arr[i]==t)continue;

      cout<<"->"<<arr[i];

   }
```

```
        return 0;
}
```

question

Question description There is a classroom which has M rows of benches in it. Also, N students will arrive one-by-one and take a seat. Every student has a preferred row number(rows are numbered 1 to M and all rows have a maximum capacity K. Now, the students come one by one starting from 1 to N and follow these rules for seating arrangements: Every student will sit in his/her preferred row(if the row is not full).
If the preferred row is fully occupied, the student will sit in the next vacant row. (Next row for N will be 1)
If all the seats are occupied, the student will not be able to sit anywhere. Monk wants to know the total number of students who didn't get to sit in their preferred row. (This includes the students that did not get a seat at all)   <h3>Constraints</h3>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>,</mo><mi>M</mi><mo>≤</mo><msup><mn>10</mn><mrow class="MJX-TeXAtom-ORD"><mn>5</mn></mrow></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>K</mi><mo>≤</mo><mn>500</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi></mrow></msub><mo>≤</mo><mi>M</mi></math>
<h3>Input</h3>
First line contains 3 integers N, M and K. N - Number of students and M - Number of rows and K - maximum capacity of a row.
Next line contains N space separated integers <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi></mrow></msub></math>. <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi></mrow></msub></math> - preferred row of <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> student.
<h3>Output</h3> Output the total number of students who didn't get to sit in their preferred row.

answer

```
 #include <stdio.h>

int main()
{
   int n,m,k,x,y,i,ans=0,flag=1;
   scanf("%d %d %d",&n,&m,&k); /* Reading input from STDIN */
```

```
int a[100001]={0},b[100001]={0}; /* initialize 2 arrays with 0 */
for(i=0;i<n;i++)
{
   scanf("%d",&x); /* Reading input from STDIN */
   if(a[x]<k)
   {
      ans++;
      a[x]++;
   }
   else if(flag!=0)
   {
      y=x;
      x++;
      if(b[y]!=0)
      x=b[y];
      flag=0;
      while(x!=y) /* while loop begin here */
      {
         if(x==m+1)
         x=1;
         if(x==y)
         break;
         if(a[x]<k)
         {
            a[x]++;
            flag=1;
            b[y]=x;
            break;
         }
         x++;
      } /* wh ile loop ended here */
```

```
        }
    }
    printf("%d",n-ans); /* Writing output to STDOUT */
    return 0;
}
```

question

Question description Sajid is an third year student in a reputed institution. Although he scored well in many subjects, he did not an expert in computer programming languages. But Sajid's computer examination is scheduled for next week. As per the blueprint, many questions would come from the sorting topic. He collected previous year's questions. one of the repeated questions is to sort the given set of numbers using Selection Sort The first line of the input contains the number of elements N, the second line of the input contains the numbers A_i to be sorted. In the output print the the final sorted array in the given format.   Can you help him ?   Constraints 1 <= N <= 10⁵
1 <= A_i <= 10⁹
  Input:
The first line of the input contains the number of elements the second line of the input contains the numbers to be sorted. Output:
print the the final sorted array in the given format.

answer

```
#include <bits/stdc++.h>
using namespace std;
void dothis(){
    printf("void selectionSort(int arr[],int n)void swap(int *xp,int *yp)void printArray(int arr[],int size)");
}
int main()
{
    int n;
    cin>>n;
    vector<int>v(n) ;
```

```
    for (int i = 0; i < n; i++) {
        cin>>v[i];
    }
    sort(v.begin(),v.end());
    for (int i = 0; i < n; i++) {
        cout<<v[i]<<' ';
    }
    cout<<endl;

            return 0;
}
```

question

Problem Description:
Good news! Suresh get to go to America on a class trip! Bad news, he don't know how to use the Dollar which is the name of the American cash system. America uses coins for cash a lot more than the Kuwait does. Dollar comes in coins for values of: 1, 2, 10, 50, 100, & 500 To practice your Dollar skills, suresh have selected random items from Amazon.co.us and put them into a list along with their prices in Dollar. Suresh now want to create a program to check suresh Dollar math.

Suresh goal is to maximize your buying power to buy AS MANY items as you can with your available Dollar.

Input Format:
File listing 2 to 6 items in the format of:

ITEM DDDDD
ITEM = the name of the item you want to buy
DDDDD = the price of the item (in Dollar)

Output Format:
Print the output in a separate lines contains, List the items suresh can afford to buy. Each item on its own line. Suresh goal is to buy as many items as possible. If suresh can only afford the one expensive item, or 2 less expensive items on a list, but not all three, then list the less expensive items as affordable. If suresh cannot afford anything in the list, output "I need more Dollar!" after the items. The final line you output should be the remaining Dollar he will have left over after make purchases.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{   int money,n;
    cin>>money>>n;
    int price;
```

```
    string name;
    map<int,string> mp;
    map<string,bool> mp1;
    vector<string> vecs;
    for (int i = 0; i < n; i++) {
        cin>>name>>price;
        vecs.push_back(name);
        mp.insert({price,name});
    }
    price = money;
    for(auto pr:mp)
        if(pr.first<=money){
            money-=pr.first;
            mp1[pr.second] = true;
        }
        else
            mp1[pr.second] = false;

    for(auto s:vecs)
        if(mp1[s])
            cout<<"I can afford "<<s<<endl;
        else
            cout<<"I can't afford "<<s<<endl;
    if(price!=money) cout<<money;
    else cout<<"I need more Dollar!";
            return 0;cout<<"char name[MAX][LEN];int price[MAX]afford[MAX]for(i=0;i<items;i++)";
}
```

question

Question description Kapildev works in the mobile phone marketing industry. For example, if someone successfully answers this question, they will be given a mobile phone at a 50% discount. One of the competition's requirements was to write a C programme that swapped nodes for two specified keys in a linked list with two keys. By altering linkages, nodes should be switched. When data consists of several fields, swapping data across nodes might be costly. It is reasonable to presume that all keys in a linked list are unique.

example : Given linked list : 10->15->12->13->20->14 and swap keys X=12 and Y=20. Linked list after swapping : 10->15->20->13->12->14
(if X or Y or Both are not present in Linked List, ABORT the Swapping)

  Constraints 1< N < 1000 1< X < 1000
Input Format
First line contains the number of datas- N.
Second line contains N integers(the given linked list).
Third Line contains 2 key nodes(X and Y) to be Swapped.
Output Format
linked list before swapping keys
linked list after swapping keys

answer

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

   int n,x,y,indx=-1,indy=-1;cin>>n;

   int arr[n];

   for (int i = 0; i < n; i++)

      cin>>arr[i];

   cin>>x>>y;

   for (int i = 0; i < n; i++) {

      if(arr[i]==x){

         indx = i;

         break;

      }

   }

   for (int i = 0; i < n; i++) {

      if(arr[i]==y){

         indy = i;

         break;

      }
```

```
    }
    cout<<"before Swapping:";
    for (int i = 0; i < n; i++) {
        cout<<"-->"<<arr[n-1-i];
    }
    if(indy!=-1&&indx!=-1){
        swap(arr[indx],arr[indy]);
    }
    cout<<endl<<"after Swapping:";
    for (int i = 0; i < n; i++) {
        cout<<"-->"<<arr[n-1-i];
    }



        return 0;
        printf("struct node struct node *next;void swapNodes(struct node **head_ref, int x, int y)");
}
```

question

Question description the popular engineering college got lowest pass percentage in last semester. the principal conducted faculty meeting and decided to visit all the classes surprisingly. Dr.Ramprasath is a faculty, who handling data structure course for EEE department second year students. one day this faculty was handling very interesting topic in data structure such that Linked List, During this lecture time, principal surprisingly visited to the class and asking to conduct surprise test on Linked list concept. So the faculty decided to conduct test on the topic of Linked List. the question was given to last bench students that is, The nodes are deleted before a certain given node in the linked list. For example if the given Linked List is 5->10->15->20->25 and delete before 15 then the Linked List becomes 15->20->25. Constraint : 1< N < 1000 1< P < N-1 INPUT Format First line contains the number of datas-N. Second line contains N integers(the given linked list). Third line contains position of the node to be deleted. OUTPUT Format Single line represents the final linked list after deletion.

answer

```
 #include <bits/stdc++.h>
using namespace std;
void MandatoriesSuck(){
    printf("struct nodenode *next;void create()for(i=0;i<n;i++)p1=p1->nextvoid del()");
}
int main()
{
    int n,ind = -1,x;
    cin>>n;
    int arr[n];
    for (int i = 0; i < n; i++)
        cin>>arr[i];
    cin>>x;
    for (int i = 0; i < n; i++) {
        if(arr[i]==x){
            ind = i;
            break;
        }
    }
    if(ind==-1){
        cout<<"Invalid Node! ";
        ind = 0;
    }
    cout<<"Linked List:";
    for (int i = ind; i < n; i++)
        cout<<"->"<<arr[i];
```

```
        return 0;
}
```

question

Question description Lalitha is a IT expert who training youngsters struggling in coding to make them better. Lalitha usually gives interesting problems to the youngsters to make them love the coding.One such day Lalitha provided the youngsters to solve that The new node is always placed before the Linked List's head. The newly inserted node becomes the Linked List's new head. If the current Linked List is 11->151->201->251, for example, We add item 5 to the front of the list. The Linked List will then be 5->11->151->201->251. Let's call the function that moves the item to the top of the list push (). The push() must receive a pointer to the head pointer, because push must change the head pointer to point to the new node   Constraints: 1 < arr <100 INPUT First line contains the number of datas- N. Second line contains N integers(i.e, the datas to be inserted). OUTPUT Display the final Linked List.

answer

```
#include <bits/stdc++.h>
using namespace std;
void MandatoriesSuck(){
   printf("struct nodenode *next;*startp1->next=start;void display()");
}
int main()
{
   int n;
   cin>>n;
   int arr[n];
   for (int i = 0; i < n; i++)
      cin>>arr[i];
   cout<<"Linked List:";
   for (int i = 0; i < n; i++) {
      cout<<"->"<<arr[n-1-i];
```

```
    }



    return 0;
}
```

question

Question description The stock span problem is a financial problem where we have a series of n daily price quotes for a stock and we need to calculate span of stock’s price for all n days.
The span Si of the stock’s price on a given day i is defined as the maximum number of consecutive days just before the given day, for which the price of the stock on the current day is less than or equal to its price on the given day.
For example, if an array of 7 days prices is given as {100, 80, 60, 70, 60, 75, 85}, then the span values for corresponding 7 days are {1, 1, 1, 2, 1, 4, 6} <figure class="image"><img src="https://media.geeksforgeeks.org/wp-content/uploads/Stock_span.png"></figure> Constraints 0<n<1000 0<price[i]<1000000 Input Format: First line indicates the number of days second line indicates the price quoted for above mentioned days Output Format: Single line represents the span values for corresponding days

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

    int n;cin>>n;

    int arr[n+1];arr[0] = 10000;

    for (int i = 1; i < n+1; i++)

        cin>>arr[i];

    for (int i = 1; i < n+1; i++) {

        int j = i-1;

        while(arr[i]>arr[j]) j--;

        cout<<i-j<<' ';
```

```
    }
        return 0;
        cout<<"void printArray(int arr[],int n)void calculateSpan(int price[],int n,int S[])";
}
```

question

Problem Description A and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi></math> are playing a game. In this game, both of them are initially provided with a list of <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> numbers. (Both have the same list but their own copy). Now, they both have a different strategy to play the game. <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> picks the element from start of his list. <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi></math> picks from the end of his list. You need to generate the result in form of an output list. Method to be followed at each step to build the output list is: If the number picked by <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> is bigger than <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi></math> then this step's output is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math> . <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi></math> removes the number that was picked from their list.
If the number picked by <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> is smaller than <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi></math> then this step's output is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn></math> . <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> removes the number that was picked from their list.
If both have the same number then this step's output is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>0</mn></math> .Both <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi></math> remove the number that was picked from their list. This game ends when at least one of them has no more elements to be picked i.e. when the list gets empty. Output the built output

list. Constraints 1 ≤ N ≤ 10^6 1 ≤ num ≤10^9 Input format: First line consists of a number <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> , size of the list provided.
Next line consists of <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> numbers separated by space.
  Output format: Output the required output list.

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   int n;cin>>n;

   vector<int>v(n);

   for (int i = 0; i < n; i++)

      cin>>v[i];

   int a=0,b=n-1;

   while(a<n&&b>=0){

      if(v[a]==v[b]){

         b--;a++;

         cout<<"0 ";

      }

      else if(v[a]>v[b]){

         b--;

         cout<<"1 ";

      }

      else{

         a++;

         cout<<"2 ";

      }

   }

          return 0;
```

```
        cout<<"if(a[i]>a[j])";
}
```

question


```
 <p><strong>Problem Description:</strong></p><p>You are given an array <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> of <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> integers
and <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> queries. In each
query, you are given an integer <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mtext> </mtext><mo
stretchy="false">(</mo><mn>1</mn><mo>≤</mo><mi>i</mi><mo>≤</mo><mi>N</mi><mo
stretchy="false">)</mo></math>.</p><p>Your task is to find the minimum index greater
than <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mtext> </mtext><mo
stretchy="false">(</mo><mn>1</mn><mo>≤</mo><mi>i</mi><mo>≤</mo><mi>N</mi><mo
stretchy="false">)</mo></math> such that:</p><ol><li>Sum of digits of <math
xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mi>i</mi></msub></math
> is greater than the sum of digits of <math
xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mi>j</mi></msub></math
></li><li><math
xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mi>i</mi></msub></math
> &lt; <math
xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mi>j</mi></msub></math
></li></ol><p>If there is no answer, then print <strong>-
1</strong>.</p><p><strong>Constraints</strong></p><p>1 &lt; = N, Q ≤ 10^5</p><p>1 ≤ Ai ≤
10^9</p><p>1 ≤ Qi ≤N</p><p><strong>Input format</strong></p><ul><li>The first line contains two
numbers <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> and <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math>.</li><li>The next line
contains <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> numbers.</li><li>Ne
xt <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> lines
contain <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> queries.</li></ul><p
><strong>Output format</strong></p><p>Print the answer as described in the
problem</p><p> </p>
```


answer

```
 #include <bits/stdc++.h>

using namespace std;
```

```
int sumof(int n){
    int ans = 0;
    while(n>0){
        ans+=n%10;
        n/=10;
    }
    return ans;
}

int main()
{
    int n,x,y,an=-1;
    cin>>n>>n;
    vector<int> arr(n),arr2(n);
    for (int i = 0; i < n; i++) {
        cin>>arr[i];
        arr2[i] = sumof(arr[i]);
    }

    for (int i = 0; i < n; i++) {
        cin>>x;
        an = -1;
        x--;
        y = x;
        if(x>=n){
            cout<<"-1 ";
            continue;
        }
        while(y<n){
```

```
        if(arr[x]<arr[y]){
            if(arr2[x]>arr2[y]){
                an  = y+1;
            }
        }
        y++;
    }
    if(an!=-1){
        cout<<an<<' ';
    }
    else{
        cout<<"-1 ";
    }



  }


        return 0;
}
```

question

Question description You are given an array A of N integers. Now, two functions <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>F</mi><mo stretchy="false">(</mo><mi>X</mi><mo stretchy="false">)</mo></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>G</mi><mo stretchy="false">(</mo><mi>X</mi><mo stretchy="false">)</mo></math> are defined: <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>F</mi><mo stretchy="false">(</mo><mi>X</mi><mo stretchy="false">)</mo></math> : This is the smallest number Z such that <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi><mo><</mo><mi>Z</mi><mo>≤</mo><mi>N</mi></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo stretchy="false">[</mo><mi>X</mi><mo stretchy="false">]</mo><mo><</mo><mi>A</mi><mo stretchy="false">[</mo><mi>Z</mi><mo stretchy="false">]</mo></math> <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>G</mi><mo

stretchy="false">(</mo><mi>X</mi><mo stretchy="false">)</mo></math> : This is the smallest number Z such that <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi><mo><</mo><mi>Z</mi><mo>≤</mo><mi>N</mi></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo stretchy="false">[</mo><mi>X</mi><mo stretchy="false">]</mo><mo>></mo><mi>A</mi><mo stretchy="false">[</mo><mi>Z</mi><mo stretchy="false">]</mo></math> Now, you need to find for each index i of this array <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>G</mi><mo stretchy="false">(</mo><mi>F</mi><mo stretchy="false">(</mo><mi>i</mi><mo stretchy="false">)</mo><mo stretchy="false">)</mo></math>, where <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>i</mi><mo>≤</mo><mi>N</mi></math> . If such a number does not exist, for particular index i, output 1 as its answer. If such a number does exist, output <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo stretchy="false">[</mo><mi>G</mi><mo stretchy="false">(</mo><mi>F</mi><mo stretchy="false">(</mo><mi>i</mi><mo stretchy="false">)</mo><mo stretchy="false">)</mo><mo stretchy="false">]</mo></math> Constraints: 1 ≤ N ≤ 3000 0 ≤ A[i] ≤ 10^18 Input : The first line contains a single integer N denoting the size of array A. Each of the next N lines contains a single integer, where the integer on the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> line denotes <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo stretchy="false">[</mo><mi>i</mi><mo stretchy="false">]</mo></math>. Output : Print N space-separated integers on a single line, where the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> integer denotes <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo stretchy="false">[</mo><mi>G</mi><mo stretchy="false">(</mo><mi>F</mi><mo stretchy="false">(</mo><mi>i</mi><mo stretchy="false">)</mo><mo stretchy="false">)</mo><mo stretchy="false">]</mo></math> or 1, if <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>G</mi><mo stretchy="false">(</mo><mi>F</mi><mo stretchy="false">(</mo><mi>i</mi><mo stretchy="false">)</mo><mo stretchy="false">)</mo></math> does not exist.

answer

```
 #include <bits/stdc++.h>

using namespace std;


int g1(vector<int> v,int j){

    for (unsigned i = j; i < v.size(); i++) {
```

```
        if(v[i] > v[j]){
            return i;
        }
    }
    return -1;
}
int f1(vector<int> v,int j){
    for (unsigned i = j; i < v.size(); i++) {
        if(v[i] < v[j]){
            return i;
        }
    }
    return -1;
}

int main()
{
    int n;
    cin>>n;
    vector<int> v(n);
    for(auto &i:v) cin>>i;

    for (int i = 0; i < n; i++) {
        int f = f1(v,i);
        if(f==-1){
            cout<<-1<<' ';
            continue;
        }
        int g = g1(v,f);
        if(g==-1){
            cout<<-1<<' ';
```

```
        continue;
      }
      cout<<v[g-1]<<' ';
    }



        return 0;
}
```

question

Problem Description:
King Alexander wants every chariot to line up for the start of her Winter-eve ball.

He has asked you, Twilight Sparkle, to sort the horse chariots alphabetically but with royalty in front. Royal horses chariot have diamonds in their names.

Constraints:
1 <= Names <= 100

Input Format:
list of horse chariots numbering anywhere from 1 to 40 horse chariot, with 1 per line. The end of input will be marked with END on a line by itself. Names should be no longer than 100 characters in length, and to only contain letters and spaces.

Output Format:
Print the output in a separate lines contains Sort and list the horse chariots alphabetically in ascending order ('A' "first"), ignoring case. However, any horse chariot with a diamond in their name must be placed at the "top" of the list (before the "A's" start) in the diamond order given in the Discussion section below. Explanation: The Pony ranking (in Ascending order) for gemstones is a follows: Lapis, Topaz, Tourmaline, Sapphire, Peridot, Ruby, Pearl, Emerald, Diamond, Aquamarine, Amethyst, Garnet. you are guaranteed that you will not have to deal with any gemstones not listed above. If multiple gems are listed in the same name, sort by whichever gem has "highest prioity" (e.g. a pony named 'Garnet Lapis' would be listed before a pony named 'Topaz Sapphire,' because Lapis has the highest priority). In the case of equal ranking on gemstones, output in alpha order by total name (not just the gemstones), only sort by highest priority gemstones, after that by alpha. (Example, given the names: Lapis Topaz and Amethyst Lapis, they should be printed in order as: Amethyst Lapis then Lapis Topaz. That is because both names have the highest ranked gemstone (Lapis), so we simply sort them in ascending order alphabetically after their ranking has been established. We would not list Lapis Topaz before Amethyst Lapis because Topaz has a higher ranking over Amethyst. Stop comparing gemstone ranking after determine the highest rank of the gems in the name.) You are guaranteed that there will be no ties in priority for gemstone ranking. You are also guaranteed that you will not encounter hyphenated names like Ruby-Sue. However, if a gemstone name happened to be found as part of a name (like Rubyanne) you can safely treat that as just another name to alphabetise. Gemstone names have to stand on their own (separated by spaces, or the entirety of the name) to be treated royally.

answer

```
 #include <bits/stdc++.h>

using namespace std;

#pragma GCC diagnostic ignored "-Wwrite-strings"


//char
*gems[]={"NONE","Garnet","Amethyst","Aquamarine","Diamond","Emerald","Pearl","Ruby","Perido
t","Sapphire","Tourmaline","Topaz","Lapis",0};

string
gems[]={"Garnet","Amethyst","Aquamarine","Diamond","Emerald","Pearl","Ruby","Peridot","Sapphi
re","Tourmaline","Topaz","Lapis"};

int index(string s){

   for (int i = 11; i>0; i--) {

      if(s.find(" "+gems[i]) != string::npos||s.find(gems[i]+" ") != string::npos){

         return 11-i;

      }

   }

   return 12;

}

int main()

{

   vector<string> arr[13];

   string temp;

   while(1){

      getline(cin,temp);

      if(temp=="END") break;

      arr[index(temp)].push_back(temp);

   }

   for (int i = 0; i < 13; i++) {

      sort(arr[i].begin(),arr[i].end());

      for(auto s:arr[i]){

         cout<<s<<endl;

      }
```

```
    //cout<<endl;
  }


        return 0;
        printf("char ponies[MAXP][BUFLEN];strcmp(ponies[a],ponies[b])>0;");
        printf("THIS IS THE PROBLEM char
*gems[]={\"NONE\",\"Garnet\",\"Amethyst\",\"Aquamarine\",\"Diamond\",\"Emerald\",\"Pearl\",\"
Ruby\",\"Peridot\",\"Sapphire\",\"Tourmaline\",\"Topaz\",\"Lapis\",0};\"");
  char
*gems[]={"NONE","Garnet","Amethyst","Aquamarine","Diamond","Emerald","Pearl","Ruby","Perido
t","Sapphire","Tourmaline","Topaz","Lapis",0};
  char x = gems[0][0];
  printf("%c",x);
}
```

question

Question description Selvan studies, engineering as per his father's wishes, while Aaron, whose family is poor, studies engineering to improve his family's financial situation. sumanth, however, studies engineering of his simple passion for developing data structure applications.   sumanth is participating in a hackathon for data structure application development.   sumanth task is to use Insertion Sort to sort the supplied set of numbers.   As a result, The input provides the number of components on the first line and the numbers to be sorted on the second line. Print the array's state at the third iteration and the final sorted array in the supplied format in the output.   Judge will determine whether the outcome is correct or not. Can you help him ?   Constraints 1 <= N <= 10^5 1 <= Ai <= 10^9   Input Format: The first line of the input contains the number of elements   the second line of the input contains the numbers to be sorted. Output Format: Fist line indicates print the status of the array at the 3rd iteration second line print the the final sorted array in the given format.

answer

```
#include <stdio.h>


void printArray(int arr[],int n){
  int i;
```

```
    for ( i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}
void insertionSort(int arr[],int n){
    int i, key, j;
    for (i = 1; i < n; i++) {
        if(i==3){
            printArray(arr,n);
        }
        key = arr[i];
        j = i - 1;
        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
int main()
{
    int n,i;
    scanf("%d",&n);
    int arr[n];
    for (i = 0; i < n; i++) {
        scanf("%d",&arr[i]);
    }
    insertionSort(arr, n);
    printArray(arr,n);
            return 0;
```

}

question

Problem Description:
Let's call an integer array $a1,a2,\ldots,an$ good if $ai \neq i$ for each i.

Let $F(a)$ be the number of pairs (i,j) (1≤i<j≤n) such that $ai+aj=i+j$.

Let's say that an array $a1,a2,\ldots,an$ is excellent if:

1. a is good;
2. $l \leq ai \leq r$ for each i;
3. $F(a)$ is the maximum possible among all good arrays of size n.
Given n, l and r, calculate the number of excellent arrays modulo 10^9+7

Constraints:
1 ≤ t ≤ 1000
2 ≤ n ≤ 2*10^5
−10^9 ≤ l ≤ 1
n ≤ r ≤ 10^9

Input Format:
The first line contains a single integer t — the number of test cases.

The first and only line of each test case contains three integers n, l, and r .

Output Format:
Print the output in a separate lines contains number of excellent arrays modulo 10^9+7

answer

```
#include <stdio.h>

#define N		200000
#define MD		1000000007

int min(int a, int b) { return a < b ? a : b; }
int max(int a, int b) { return a > b ? a : b; }

int vv[N + 1], ff[N + 1], gg[N + 1];

void init() {
	int i;

	ff[0] = gg[0] = 1;
	for(i = 1;i <= N;i++) {
		vv[i] = i == 1 ? 1 : (long long) vv[i - MD % i] * (MD / i + 1) % MD;
		ff[i] = (long long) ff[i - 1] * i % MD;
		gg[i] = (long long) gg[i - 1] * vv[i] % MD;
	}
```

```
}

int choose(int n, int k) {
	return k < 0 || k > n ? 0 : (long long) ff[n] * gg[k] % MD * gg[n - k] % MD;
}

int main() {
	int t;

	init();
	scanf("%d", &t);
	while(t--) {
		int n, l, r, i, j, k, d, ans;

		scanf("%d%d%d", &n, &l, &r);
		d = min(1 - l, r - n);
		if (n % 2 == 0)
			ans = (long long) choose(n, n / 2) * d % MD;
		else
			ans = (long long) (choose(n, n / 2) + choose(n, n / 2 + 1)) * d % MD;
		while (1) {
			d++;
			i = max(l + d, 1), j = min(r - d, n);
			if (i - j > 1)
				break;
			k = j - i + 1;
			if (n % 2 == 0)
				ans = (ans + choose(k, n / 2 - (i - 1))) % MD;
			else
				ans = ((long long) ans + choose(k, n / 2 - (i - 1)) + choose(k, n / 2 + 1 -
(i - 1))) % MD;
```

```
            }
            printf("%d\n", ans);
        }
        return 0;
}
```

question

Question description Selvan is very interested in surfing the contents from google. He searches for various coding test on Google. One day he searched about online coding competitions, in the retrieval links, he received many links for coding competition. he chooses first link from the goole suggestion list. first question for the coding competition is LRU cache implementation using queue concepts. Function Description Queue which is implemented using a doubly linked list. The maximum size of the queue will be equal to the total number of frames available (cache size). The most recently used pages will be near front end and least recently pages will be near the rear end.
A Hash with page number as key and address of the corresponding queue node as value.
<figure class="image"><img src="https://media.geeksforgeeks.org/wp-content/cdn-uploads/LRU1.png"></figure> Constraints For this experiment, cache can hold 4 pages. Let 10 different pages can be requested (pages to be referenced are numbered from 0 to 9). Input Format: First line represents n and m, where n is the page number with in the range (0-9) & m is cache size (must be 4 for this problem). Next line represents the reference pages. Output Format: Single line represents the cache frames after the above referenced pages.

answer

```
#include <bits/stdc++.h>
using namespace std;
void don(){cout<<"QNode* newQNode(unsigned pageNumber)Queue* createQueue(int numberOfFrames)typedef struct Queuetypedef struct QNode ";}
int main()
{

    int n,m;
    cin>>n>>m;
```

```
    int arr[n];
    for(int i=0;i<n;i++){
        cin>>arr[i];
    }
    for (int i = n-1; i >=n-m; i--) {
        cout<<arr[i]<<' ';
    }

            return 0;
}
```

question

Problem description Tina is a Bachelor of Computer Applications (BCA) student. During her final year Campus Interview, she has an opportunity to get a job in a software company in Bangalore. The company provides Five months training period with Rs.30000/month Package. Then it will be incremented to Rs.55000 per month. At the end of the training, the examination was conducted for all freshers, Tina got a question paper and one of the questions comes under the concept of Queue concept with Linked List Implementation Function Description The front points the first item of queue and rear points to last item.
enQueue() This operation adds a new node after rear and moves rear to the next node.
deQueue() This operation removes the front node and moves front to the next node. Constraints you have to perform N number of enqueue operation and two consecutive dequeue operation then continue M number of enqueue operation. 0<n , m <10000 Input Format First line represents the N and M, Second line represents the N represents the number of elements to be enqueued then Third line indicates the M number of elements to be inserted after two consecutive dequeue. Output Format Results shows the every cycle enqueued/dequeued elements and front rear status.

answer

```
#include <bits/stdc++.h>
using namespace std;
void don(){cout<<"struct QNode* newNode(int k)struct Queue* createQueue()void enQueue(struct Queue* q,int k)void deQueue(struct Queue* q)";}
```

```
int main()
{
   int n,m;cin>>n>>m;
   int arr[n+m];
   for (int i = 0; i < n+m; i++) {
      cin>>arr[i];
   }
   cout<<"Front:"<<arr[0]<<"\nRear:"<<arr[n-1]
   <<"\nAfter 2 deQueue and M enqueue\n"
   <<"Front is:"<<arr[2]<<"\nQueue Rear:"<<arr[n+m-1];



            return 0;
}
```

question

Question description Umesh is an DS expert training youngsters struggling in DS to make them better. Umesh usually gives interesting problems to the youngsters to make them love the DS.

One such day Umesh provided to the youngsters to solve the task such that, Reverse a Queue, Queue data structures work on the FIFO architecture so the element that has entered first in the list will go out from the list first. Youngsters were lacking the idea to solve the problem. Being an exciting youngster can you solve it? Function Description <figure class="image"><img
src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAMiIlLCUfMiwp
LDg1MjtLfVFLRUVLmW1zWn21n767sp+vrMjh//PI1P/XrK/6//3/////////B8P/////////////bAEMBNTg4S0
JLk1FRk//Or87//AABEIASQDmAMBIg

ACEQEDEQH/xAAZAAEAAwEBAAAAAAAAAAAAAAAAAgMEAQX/xAA/EAACAgIAAwQFCQcDBQEBAAA
AAQIDBBESITEFE0FRFCIycXIzNFNhgZGxwfAjNUKSodHhJFKiFVRigvFDRf/EABQBAQAAAAAAAAAAAAA
AAAAAAD/xAAUEQEAAAAAAAAAAAAAAAAAAAAA/9oADAMBAAIRAxEAPwD1AAAAAAAAACF83XTOa
1uK2tgTBRTdKeH3rS4tN8unIrwsx3txnpT6rXigNYKcu2VNDnHW1rqcndKOErklxcKf1AXgzd7kTx651Rg5
SXPZTZk5dc4QlCvc3pfrYG8GaqWW7F3kIKHi0dpvnPLsqaXDHp5gaAclJRi5PoltmXDzJX2ShNJctrQGsEbJ
ONcpLqk2V4l0r6eOaW965AXAGfEvnc7FJL1XpaA0Ayced9HX+vtKrMrKrsjXKFfFLolz/MD0AZszIso7vgU
W5PXMjx530df6+0DWCuh2uH7ZRUt+HkMmx1UTnHW15gWAxV3ZtkFOMK9Pp+tl9EshyffRglrlwgXAw
U5OXfFuuFbSev1zLIZdkLY15Fai5dGugGsFWTkRx6+J82+i8zP3ua48aqjrrrxA2gpxclZEXy4ZR6oolmuvMd
c1Hu09b8UBtAMlGRddVY4xjxxekvADWDDbkZlMOKcK0un65ko2ZsoqSrr01v8AXMDYAG1FNvouYAGP
EzHfdKEkl4x0arJONcpLqk2BIGTCzHe3GelPqteKLcu2VNDnHW1rqBcCid0o4SuSXFwp/UUwuzJwU411u
L/XmBtBnxspXNwlFwsXWLNAAGTJyLoZMaqlFuS3zIyvy6lxWUwcV10BtBS8hSxJXV+Cb0/MoqvzLYKcIV
uL/XmBtBRTLJc/20IKOvAnfb3NMp+S5e8CwGfCyXkQlxaUovw8iWXbKmhzjra11AuBjjbmygpqutpra/W
yzGyu+lKuceCyPVAaAAAAAAApyMmGPriTbl0SAuBlxsqd98oShwJLevE1AAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABgzcyyu111vh11ei7ByZZEJcftR8V4j
Jwo5EuPi4ZePLeyzGx448OGL231b8QLQAAKsr5rZ8LLSrK+a2fCwKcb92f+svzMldUljRyK/bhJ79xrxv3Z/6
y/M72dzxNPzYEMq2N3Z7nHxa2vJ7JW/utfBH8jJlVyxpTrXydnNGu391r4I/kBbh/Na/cUZvzvG+Jfii/D+a1
+4ozfneN8S/FAbTFjfvG/wBxtMWN+8b/AHAT7RscaVXH2rHoqyK/RZUWx6R9WX6+85ZGWXnSUJ8KrX
KX1krcK6VbTyJT8dNdf6ga7nuibX+1/gUdm/NV8TI41veYEk+sIuL+4dn2QjjalOKe3ybA2GLs72r/AIv7mu
NkJPUZxb8kzJ2d7V/xf3A1X2xpqc5eHRebM2FVKUnk285y6fUiqU45mVqUlGmHm9bN3fVL/wDSH8yAy
dpvTpfk3+RP09fQ2/cQ7Ra3Q98uL+xr76r6SH8yA7XPjgpaa2t6ZVnfNLPd+ZdGUZrcZKS+plOd80s935gZ
8fOqrohCSntLwRrovjfBygmknrmRw/mtfuLgPNwMiqmqSslpt76MlfYsy6uFKbUXty0d7MhCdU3KMX63i
gv9Hm66VW/0A7krj7Qpg+iW/wBfcbjFnJ13VZCW1HkzQsmlw4+8jr38wM1a4O1LEuklsrlSr86+D68O0/
J8izE3dl25GtR6I7T+9Lfh/sBLBubTps5Thy5+RHsz2bfiGbVKE45NXtR9r6znZb3Cx/8AkBZ2l81/9kX0/Iw+F
FHaXzX/ANkX0/Iw+FATMvaFnBj8C9qb0jUefdGWXnOEJcKrXXyYDIq9FWPbFexyl9f65m21p482uji/wMt
mFdODTyZS+prr/UYtnHgTi+sItf0AzV1SWNHIr9uEnv3GnKtjd2e5x8WtryeyfZ3PE0/NmPKrljSnWvk7Oa
A12/utfBH8i3D+a1+4qt/da+CP5EsW6qGLXxWRTS6NgVZS4M+ia5OT0/wNxgjL0vOjOKfd1+JvAwZU419
o1Tm9RUeb+8nfnUuqUa25yktJaI5CUu06k0muHo/tJZ1CVatqioyre+S8AOV1Sq7NmprTab15EMXLVW
PGHdTlrfNL6y+dquwJzXjF7XkyODbXHEgpTinz5N/WBdRf3/F6ko6/3Iz5z726rHXi9v8AX3mtW1yelOL9z
PPqrnl32XRsdenpNIC2X+m7Qi1yhatfaWdo/NJe9fiZ8nEtVTnK+VnBz00WZNne9m8fi9b9+wJU5tEKIRc+
aik1pkMXiuzJ5Ci4wa0t+JbXRXZiQThHcoLnrn0IYFji5Y9ntQfL3AbDNNZnHLglXw75bNJmniTnOUlkWRTe
9LwAVrM7yPeSr4d89GkzV4s4WRk8ickn0fiaQK3kUptO2Ca68ya4ZaktPlyZW8amTbdcW3zZakopJLSXJA
Ysf95X+7+xtMWP+8r/AHf2NoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAA5OKnFxktp8mdAEI1QhV3cVqHTWztdcKocMFpeWyQAhbVC6PDZHi
XUOqEqu6a9TWtbJgDkIRhBRitJdERnTXZOM5R3KPNPfQmABCNNcLJWRjqUur2TAEKqa6d93HXF1572T
AArhj1Q4+GOuP2ub5lfoON9H/yZoAFVWNTVLirhp9OrJV0118XBHXF159SYAz+g430f/Jj0HG+j/wCTNA
Arsx6rYxjOO1HpzZX6DjfR/wDJmgAQqqhTFxrjpN76kpwjZBwmtxfVHQByEI1wUYrSXRHQAIVU10xarjpP
n1FtNd0UrI8SXMmAOcKceFra1rT5lHoOPvfd/wBWaAByMVCKjFJJeCIxprja7VH13yb2TABpNNNbTIVU1
0pquPCn15kwBG2qFsOGa2vLZJJRSS6LkgABCumupycI6cuvPeyYAFcaKoubjHXH7XN8ywARrrhVDhgtLy
2ctqhdHhsjxLqTAEHVB1d016mta2VrBx1/+f8AVl4A5GEYR4YxUV5I6ABCVNcrVY4+vHknsm1taYAFcMe
qEJQjHUZdVtlfoON9H/yZoAFMcSiDbjDW1p830LK6oVQ4YLSJAA0pJprafJlSxqlU6uD1G962y0AcjFQiox
WklpEHRW7Va4+uvHZYAAAAAAAAIRphGyVkY6nLq9kwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACu2+unXeTUdkoTjZHihJNe
aAkAABCd1db1OcYvybOZNnc0TmuqXIzYmJCdatuXHKfPmwNkJxmtwkpL6mdMUseWPlQnRF8D5SS8D
XZNV1ym+kVsArIOTipJyXVb5okeTS51WV5M3ysk0z1gIwnCe+CSlrrpkjF2Z7NvxG0CKnBycVJOS6rfNEjFj
fvG/3f2NoAhK6uD1KyMX5NkzBbCNnaajNbTj0+wDX6RT9LD+ZEpWwjFSlOKi+jb6lXoeP9EijtOKjj1xitJPS
+4Dd1XIi5wjJRckpPot82ZcG5rePZynDp7jmT+8aPcBtK/SKd672G/eZs2UrLq8aL0pc5FvoNHBw8HPz3zA

0EYzhKTUZJtdUn0MmDOULLMeb3wdPcML53k/F+bA2SkopuTSS8Wc7yHBx8S4fPfIrzPmtnuM//wDI+z
8wNXpFP0sP5kShbXY9QnGT+p7M2Li0zx4SlWm2ubL66KqpN1wUW+QEoThYtwkpL6ns7KcYLcpKK82z
H2V8jP4ifaXzV+9AaJWQjFSlJKL6NvkR9Ip+lh/MjLmfu6r/ANfwLq8Sh1xbrW2kBdCyFm+CcZa66eyTaSbb
0kQqprq33cVHfUXfIWfC/wAAOekU/Sw/mR2N1c3qNkZPyTMeDj1WY/FOCb2+Zqrx6q5cUIJPzAnGcJ74
ZKWuun0OylGEeKTSS8WY+zvbv+L+5bn/ADOf2fiBc7IKHG5JR898iUWpJOLTT8UYbv3VH3IjgXOuSps5KS
4ofaBulOENcUlHfTb6kjF2j7VHxf2JdoWyjCNcHqVj0Be8imMtO2CfvJ8UeHi2uHW975FEMGiMOGUOJ+L
ZXjUzqusqabpkuTYGj0in6WH8yHpFP0sP5kZ8rFphjzlGtJpcmMXFpnjQlKtNtc2BqdkIqLc4pS6c+pIw58VF
48UtJPS/oWdoWyrqUYPUpvQF0simMtSsin7yxNSW000/FGavApjWlOPFLxeymjeLm9xtuua2tgbXZBTU
HKKk+i3zJHmZ8ZSzEoe1w7Wvq2bcS9X0qX8S5SQFinCUnGMk2uqT6EjFifPsj3/mbQOQnCxbhJSX1PYlO
EWlKSTfRN9THhfs8m+nw3tFGdKU8mTj0qS/X9QPUb0tvoR72twc1OPCvHfIpybV6DKa/ijy+0olDg7J14tJ
/ewN6akk000+jRGc4Q1xyUd9Nshi/Na/hRn7T9mr4gNk5xgtzkorzb0JTjCPFKSS82zJ2p83j8f5M7n/ADL7
UBrTTSae0w2opttJLq2Qo+Qr+FfgRyvmtnwsDvpFP0sP5kdhdXN6hZGT8kzLh41NmNCU4Jye+f2mmvHq
qlxQgk/MDmRfGiG5c2+SS8SbsjFR45KLl0TZkq/1GfOb5xq5R9/62M/5fH+L80BtAKLsymnk5cUvKIF5XK+
mL07IJ+8p7QtlClRhylN6O14NMa0pQ4peLYGmMlJbi00/FEeOCmoOS4n4b5mbHosx8mSim6ZdOfQjZ+
9q/h/Jga42QlJxjKLkuqT6EjDd/p+0IWfw2cn+vuNls1XVKb/hWwCsg20pxbj1W+hH0in6WH8yMuDW1j
WWy9qeyOBj1W0OU4KT4tbA2xuqnJRjZFt+CZ1WQc+BSjxLw3zIQxqa5KUIJSXRnnZDnHOsnX1h634Aes
RhOE98ElLXXTOU2xuqU4+PVeRl7M9m34gNpGdtdftzjH3skedhV15E7JXetZvowN8LIWL1JqXuYlOEWlK
STfRN9TLbiSrthZjR00/WWzmb87xvi/NAa5211vU5xi/reiPpFP0sP5kLKKrZJ2QUmlox5tdFUFCFa7yfT6gN
0bISi5RnFpdWn0I+kU/Sw/mRVVR3GFOL9pxbf3FODj1WY/FOCb2+YGyN1U5cMbIt+SZMqhjU1yUoQSk
vEtA5KUYrcmkvNkYX1Teo2Rb8tmNx9MzZRk33dfgWX4NTqbrjwzS2tMDW3pbZX6RT9LD+ZFONc7sNu
T3KKabKcDHqtocpwUnxa2BtjdVOXDGyLfkmTKoY1NclKEEpLxLQOTnGC3OSivNsjC6ux6hZGT8kzFXD03
JnOxt1w5JE8rDrjU7KlwThz5MDbKSim5NJLxZyMlKKlFpp+KMrtd3Zspvrw6ZZg/M6/t/EC2U4QaUpJN9N
vqJ2Qr1xzjHfm9GTtD5bH+L80cz0pZOPFrab0/vQGr0in6WH8yJd7XwcfHHh898ir0PH+iRVnVxrwnGC0tr
kBsTTSae0yLnBTUHJKT6LfM5R8hX8K/Ay3/vSn4f7gap21wepzjF+TZz0in6WH8yMl8I2dpwjNbi48195o9
Dx/okB2WTCFkIv2ZrlPwLijJojPFdcVrhW4jCtduNFvm1yYF4AAAADyu0q5rIc2nwtLTNHZkJxrnKSaUmtb
NoAAADP2gt4k/q1+JPFaeNVr/aiycVODjJbTWmYoV5WLuFcVbX4b8ANk7YVuKnJJy6fWZO07eGqNa6y
e37jtdF118bcjUeH2YolGmyee7bI6hFajzAz35NFmL3UVLcUuHaNeHb3uNFvquTLzLi02UX2x4f2UuaewI
dmezb8RtPPohl4/EoUxak982v7l9VmW7ErKYxj4tP/ACBXjfvG/wB39jaYXXk15dlldSkpebX9yxWZu1uiGv
f/AJA1Hn31972kocTjuPVe49AxX1XrM76qtSSWubQE1hNNPv7PvIdqfIw+L8iXe5v0EPv/AMjMquvx60o
evvclvoBzOpa4civlOHX3FTuV+ZjzXlzXkz0dctM8+OHOvNjKEd1p73voBKz1e1a2+jX9zcZ8vG7+KcXw2R6
MplbnRradS2l7S6gKPW7Tua6Ja/A7hfPMn4vzY7NdfBLUt2N7ls7bRdXkO7H03L2osC7M+a2e4z//AMj7
PzFkcvJXBOEa4eL2X30v0N1Vrb0kkBnx8RzohLvrI7XRM10U9zFrjlPb/iMtTzKq4wVEWo+bX9y+ieRKbV1
cYx11T8fvAp7L+Rn8RPtL5q/eiDovx7pTx0pwlzcWzkqsjKlFXRVdae9LxAZn7vq/9fwJQwm4Rff2La8yebT
K3HUK47aa5bIRnmxil3EOS11/yBppr7qvh4nL62LvkLPhf4EceV8uLvq1Dy0+pO1OVU0uri0gPPw8eyyjijf
KC30X/wBNmPTOri47pWb6b8DmFXOrH4ZrT2+Wy8DF2d8pf8X9y3tD5nP7PxK50XU3ytx9SU+sWRnXl
ZWo2RVde9vn1AXfuqPuR22jvcKqUPlIRTWvcW5VLlid3VHbWklstoi4UVxktNRSYHnXXq+FDftKWpL7i/P
5ZGPJ9FL80QyMKfpKnVHcW9tb6GvJoWRVwt6fVPyAtIq2DsdakuNeBkjLNrjwd3GeuktlmLjyrlK217sn/
QCeZ80s9xzC+aV+4lkwlZjzjFbk1yQxYSrx4RmtSS5oDP2j7dHxf2HaPKyiT6KXP+hPNpstlU4R3wvb5luRQr
6nB8n1T8mBaYr/AFu06UuqXP8AqIvNqjwd3GeuSlsni404WSuue7Jf0AhZ+9a/h/JkbU8LKVsV+ynykvItn
TY+0IWqPqJab37y+2uNtcoS6P8AoBkw2pZt7T2n/c3GLBx7KLZ8cdJrSe+ptAw3NU9owsfKM1z/AF9wxK
++ounLra2WZ9E7oR7uO5Rfn4F9FfdUQh4pc/eB5jsdmLVR/Fx6/X3m3OSWFNLotfiiqOJNZ7s4f2e+LezR
mVytxpwgtyetL7QM9GG50wl31i2t6T6FWZj9yoPvJT2/4jfjxcKIRktNLTKc6my6NarjvT2+YEe1Pm8fj/Jnc/
5l9qJZ9U7qYxrjtqW+p3MqnbjcEFuXLlsC2j5Cv4V+BHK+a2fCyiE82EIxVENJa6/5J/t7ce2NlajJrUUn1Aoxc
V248Z99OO98k/rNdFHc8X7SU9/7vAzU+mU1KuNMWl5tf3L6Z5MrNW1RjHzT/wAgU9m9LfPi5jP+Xx/i/
NDG/Y5t1T6S9aJPMpsttplCO1F7fPp0AjnxyJNd3t165qPUjiSxI6SXDP8A8+puKrsaq7248/NdQM3aXKdE
n0Uuf9DVcrXX+xcVLfVjIoV9Tg+Xin5MzQebTHg7uNiXJPYCq7IjmRptlF7W+S+oWfvav4fyZKii6WR397Sk
uSijs6bH2hC1R9RLTe/qYHe0Ku8xm11hzKMi93YtMI+1Y+f2f5PQaTWn0ZgxcOdeU5TXqR3wvfUDW4Kv
HcF0jHX9DDhYztpcu9nDnrSZ6M03CSXVowY8cyivgjTFre+bX9wNVGP3Mm+8nPa1qTM9S32ncn04f7Ft

VmU7ErKYxh4tP/JyqmyOfZa4+o1pPfuAqg3hZXA/krOj8iXZns2/EaMmhX0uPj1T+sq7PpsphNWR4W3y5
gam0k23pIy2YlOQ+8rnwyf8UXtGppSTT6PkYYVZOK2qkrK296fVAc7y/EsjG2XeVyet+RLN+d43xfmjjpyM
qyLvioQi96T6lmVTZZkUShHcYvbe+nMDRbZGqtzl0Rkw65XWPJt6v2UM+q+6cVCPFBLfVLmdVmakkseC
S+v/ACBpu+Qs+F/gYMPGdtHF3s48+iZrg77KrFbWotrUdPqZ6FmUV8EaYtb3za/uBpox+5bfeSnv/cy4z1T
ynYlbVGMPFp/5NAGLB5ZORF9d/mzZJpRbfRIy5GPYru/x2uPxT8SE1mZEe7lCNcX1ewHZ6/0dr82/wKsO
iy2lyhfKtb1pG+FKqx+7hz5P7WV4NU6aHGxafFvqBKimdTbndKzfn4FwAGHszlG2L6qXM13tKixv/azNbj2
13u7G02/ai/Eoy7MmUErocFbfPh5gTpT/AOlz+vZpwfmdf2/iSqjVPGUIPdbWjNXDKxdwhGNkPDn0A72
h8vj/ABfmiPaEeK+iO2tvW14c0Trouuvjdkajw+zFHc2q2dtU6ocXBz6gPQX/ANxb94z48OFw7b1pbY73N
+gh9/8Aks4J5GNKF8VCT8vACyj5Cv4V+Blu/elPw/3FfplEe7VcbIro9ksfHsd7vva4/BLwAqyYd52jCHE47j1
X2lvoL/7iz7yORVf6ZG6qClpa5tEu9zfoIff/AJA1mLsv5KxeHEXztlXiuyxKM1Hp9ZDs+t14yb6yfEBpAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAEY1wjNyjFKUur8yQAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAFGTj97wzg+GyHssvW9LfXxAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAACSUk1JJp9UwAOQhGuKjBJJeCOgAAAAAAAAAAAABRfRK+yKlL9kubj5sv6AA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAYM3Msrtddb4ddXouwcmWRCXH7UfFeIGkAAck1GLk3pLm2Y1k5F7fo9a
UF4yLc9tYk9eOl/UniRUcWtL/amBVVlWK5U5EFGT6NdGaiu2iFsoSlvcOa0yGbb3WNJrq+SAqozXblOtpc
D3ws2HnW0OnDqsitTg+J/b+kb65qyuM10ktgU4d871NzSXC9LRoMXZns2/EbQM9WROeXZU0uGK5eZ
oMWN+8b/AHf2NoAyX5Fyyu5qjB8t8zWYLbI1dpqc3qKj1+wCUsjKpXFbVFw8XFl1mR/pHdXp8trZTk5lU
6pV1bnKS0kkJVyq7LcZddba+0BXdm2QU4wr0+n62WRll8M3OEFqO468WU4+Yq6IR7qx6XVI10Xd9Fvg
lHT16wEMO95FPFLSknp6O5d3cUua1xb0tmej9hn2Vfwz5r9feMv9vmVULoucgJ35NtWLXY1Hjl1Whx53
0df6+0j2p8jD4vyJenr6Gz7gLIzvjj2TtjFSim1r3FFV+ZbBThCtr9fWX2Wd7hTnwuO4Pk+pnw8qmrHUZz1J
N8tMC2nLk7u5vhwT8NdGdyMmULVTTDjsfn4FKk8vNhOuL4IdZMlk121ZKyKo8fLmgE78uhcdtcJQ8deB
bfkcOJ31envWtkI5tNyddqcN8mpdBmwjXguMFqKa194HIW5s4KUYV6a2v1suoeQ2+/jFLw4TPTmqFMI
9zY9RS2kaqLe+g5cMo89akBYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADNk4UciXHxc
MvHlvZZjY8ceHDF7b6t+JaAAAAqyq3bjzgurXIqwsiEqIwlJRlHlpvRqKbMWm2XFKC35rkBVZkylk11UNP/
AHPqV5aeTlwoi9KK235Gyqiulfs4JbEaa4WSsjHU5dXsDNLDulFxllSafg1/kdnTfBOmXtVs2EI01xtdijqb6vY
GPs6cIKzilGPreL0bVbXJ6U4t+SZT6Djt/J/8mShiUVzUow1JdHtgZqZRh2jc5SUVrxevI2d9V9JD+ZEJ4lE5uU
obb6vbOLCx001XzX/kwLzDOKl2qlJJrh6P3G4h3Nfe97w+v03sDLm092o31JRcHz0ieRYrez5Tj0aX4mqSU
ouLW0+TRVHHqjU61H1JdVtgV4ltaxq07IppdGy+NkJvUZxk/qZT6DjfR/8AJk6saqmXFXDTa11YFHaEXDu
749YPmcwE7LLciX8T0v19xsnCNkHGa3F9UcrrjVBQgtRXgBk7U+Rh8X5GpXVa+Uh/MhbTC6KVkdpPfUq
9Bxvo/wDkwJ3zjPGt4ZKXqvo9+BTgVVzxU5Qi22+bRfDHqrhKEY6jLqtvmSrrjVBQgtRXgBjxm8bKljyfqy5x
ZY8p15cqreGMP4ZF1lFdsoynHbj0e9Hbaq7VqyKkBnzpUSok5OLLr1dPmVT4v+kx4uvL7tmmOFjxe1Xt/
W9ltlcLYcE1uPkBVj21rHrTsgmorxXkWxnCfsyjL3PZT6DjfR/8mWVY9VLbrjwt9ebAsAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAByUlFNyaSXixCcZrcJKS809gdAIq
2uU3FTi5LwT5gSAIxshKTUZJtdUn0AkA3pbZyE4zW4SUl5pgdBGdkK4uU5KKXi3pgdBGEI2RUoSUovo0
lh/MicZRnHcZKS80wOgEZ2RrW5yUV5t6ASBGE4zW4SUl5pgdBGdkK4uU5KKXi3pgdBGEI2RUoSUovo0
94FgCaa2ntEJXVQlwysin5NgTBCNtc3qNNkZPyTIAclKMI8Umkpjp0AAITuqrep2Qi/JpoCYI1zhdFShJSi+jKMidmrKLjfqE
GcZrcZKS80zoAFbyKU9O2G/eWJ7W0ABCV1UJOqbEbqpvUUbIN+SYEwCp5EfSFTOT8WvAC0EVvEO
EpOKknJdVvmiQAjO2ut6nOMW/gntOMd+b0BIFfpFfpFfpFL/wD1h/MixNNbT2gAOTnG
C3OSivjsjC6ux6hOMnrXtsE3PFDoOTnGC3OSivNsjK2uvXHOMd+b0BJPa2gBmlUUN+0hz/bQRnGa3GSkvNGgAOTnG
fzI7C2ub1CczPyTAmcnOFa3OSivjsmuqegOgHIyUo7i013PyUluLTXmgB4eOLWpSb6dgyCMpwgk5SSr2RTXg2
BMFfpFP0sP5kWJ7W10AAj3kOPg4lxKW/nkpOgAhcGv0QsT7fxeD7hLwl8bwlqPvXVaMuaJYmBZazpu6jJ
AHJSjCLlJpJeLA6AmpJOLTT6NHzjzJRXm2BbnJRXm1/gCUSkAHP8hRJl7o4S5W0GTExUVRm24uaxnS
AHJSjCLlJpJeLA6AmpJOLTT6NHJzjBbnJRXm2RjbXPXDOLb9kcPJwz4SIAAAABXfcqIKIKTi3

enrwJxkpRUovafRgdAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABXbfXTrvJqOyUJxsjxQkmvNHmdpVz
WQ5tPhaWmaOzITjXOUk0pNa2BtAAFOZ81s9xj7Nm67XXLpNbRszPmtnuMUouOHRfH2oPn7tgehfZ3V
Mp+S5e887ATWWnLq4tl2XP0mdNMHyn6z9wgku1ZJdFHX9EBrvsVVMpvwR52NxY91Vk36tu9/r7i3tO
1epVvk3uRXlZNFtChBSTjrh5AejZ8nL3MxYV0aMKU5v+J6XmaKre+w+Px4Xv3nnU4tl9MpRfKPRebA1Y1
Usiz0i/p/DEZ6UsnHi1tN6f3ouw8hX16fKceTRR2hHivojtrb1teHNAaPQ8f6JFtdca48MFpeRm9Bf/cW/ea
oR4YRjtvS1t+IHTzMtSyL7HD2aY/r9fUehfZ3VMp+S5e88/Dyqaa5KfE5Se3yA341vfUQn4tc/eWHn9m2p
WWVJ+q3uOz0AMFsI2dpqM1tOPT7DR6Hj/RIzXwlZ2koxm4Nx9pe4uji3KSbyptJ9PP8AqBHOm4Qroq5
OfL7CccChQ4XHifi9leb6uXjyfTf5m0DDi8WPlyx224vnHZydcbO1JRmtrh6fYSl63a0dfwx5/cyu1WS7Skqp
KMtdX7gLMvEqjRKcFwSjz2maMSbsxoSl1aMN6uV1deTbuuT6x6HpQioQUYrSS0gKM/5nP7PxJ4vzWv4
UQz/mc/s/Eni/Na/hQHMu1048pL2uiKcbDrdSnauOc1vmyXaSbxfdJF9LUqYNdHFAYpw9CyoSrb7ufJos7
Qsk3XRB6dj5nO0ubpiurkMv1c+iT6dP6gWRwKFDhcdvz3zKsNypyZ4ze49Ym4xL1u1uX8Mef3AQ7uFval
kZx2tdPsRbkYVPcylCPDKK2mmUzVr7Ss7lpS14+5FssfLtXDZdFRfXhQHcfIawHZLm4prn4+RLs+vho7yXO
dj22Qy61T2e4R6LX4mjG+bVa/2r8AM2N+8b/caJ5NMJOMrEmuqM+N+8b/cL7seN0lOhykur4VzA0Ryq
JSSVibfJGbNip5lEZLafJr7RXdjOyKjjtNtafCuRzPi5ZVEVJxb5bXhzA0eh4/0SKe0IRrw4xgtRUuS+8l6Jd/3c/u
/yO0uWKl19ZAShh0SrjutbaXPbKsLdeVbQm3Bc0RnHMhj8fepxS3pLnr7i7s+uCp7yLblL2m/MCmqHpuR
Odjbri9JE8rDhGp2UrglDnyY7L5Qsi+qlzNOQ0sexv/awKe9d3Z0pvrwvZlwbnROKnyrs6P6y6ha7Ln9aYqo
V/Z0Y/wAS24v69gT7T+bL4kW9xXbCDsgpNRRgtvdmH3c/lISSe/E9Ov5KHuQGCVFS7RhWoLga5r7Gba8
eqqXFCCT6bM0/3tX8P5M2gDF2p8hD4vyNilFtpNNrw2Y+1PkYfF+QGqdsKopzkop+ZD0uj6WJzKnVCEe
9rc0+mlvRm7/E/wC2l/KgNOTOM8Kcovaa5MqxMWmzGhKcE2+r39ZK2UZdnycI8MXHktdOZRjwynjRd
dsVHT1HXP8AACVcfRu0VXW3wTXNHO7hb2pZGcdrXT7ES7OjGblbJuVqenvwITq73tKyPHKPLe17kBq9
Dx/okXcox8kkZo4bjJS7+x6e9Nne0Le7xml1nyAxOU3N5i6KetfUerGSlFSXRraPOjk0LD7lqW+Hy8S/s63jx
+F9YPX2AVUVQtzb1ZHiSb/E7m41dNatq9SSfgyFcbpZl/czUHt72vrOOE55cacqxtdVrkmBrnlRqqqlOMm
5pdEVZ/y+P8X5o26WktckYs/5fH+L80BtKLsymp64uKXlEpz43ykuDidWuaj1OYk8RaUVwz/8+oG4z5/zO
z7PxRfxR4uHa35bKM/5nZ9n4oDldyowa5yTa0uhX2hJTw4TW9SafP3GjF54tfwop7T+bL4kBpr+Tj7kUyv
ms6NPLha39Z2yu2yqvurODS5/WYpVXemxg7f2jXKQHqGfNvnRUpQ1ty1zFdN8YTUruJteq/Ix5tV1dSdtv
Gt9APUXQqryI2XzqSknDq2VVUZEZxlO/iiuqNXJbb0vNgDEv3s/h/I2pqS2mmvqMS/ez+H8gNk3wwk11S
2U4d0r6eOet71yKrqMh95JX6jzevqKMSm+dO67uCO+gHpmem+c8q2t64YdDllORLg4L+HUUn9b8zJTV
dLKtjG3U11l5genKKlFxktprTMmBJwlZRJ+w+Rox4WVwatnxvfUzVcu1bdf7f7AbQAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAFeTCU8ecYrba5IhTS/Q1VYtPTTLwBiwcWdU5TtWnrUeeycabF2hK3h9Rr
W9/UagBlqpseZO62Ol0jzNTSa00ABkxqbaVdW4+o98L2Twap00ONi0+LfU0ADHfj2QvV+Oty/ijvWxl1X
WTpsrr2482m1yZsAGTvc36CH3/5NFLslXu2KjLyRMAZs6q25QhBervcnvoaIxUYqKXJLSOgDLkU2elV3VR
21ylz0agAMrps/6greH1Na3v6jUABTlY6yKuHepLmmUKWdCPB3cZNdJbNoAzYmNKqUrLXxWS6/UcVN
n/UHbw+prW9/UagBTl0d/S4r2lzj7zuN3ipStjqS5dd7LQBTmVytxpwgtyetL7SiuWbXXGCoi1Fa6r+5tAGe
CtvhOGRWoprlplMI5eMuCEY2Q8PqNwAx049tl6uydbXsxXgXZWOsivW9SXNMuAGJTzox4O7jJ/7t/wCS
3ExnTxTsfFZLqzQAMsKbF2hO1x9RrSe/qRqAArya+9x5wXVrkV4FnHjRXjH1WaCiOO68nvK2lCXtxAhRTZ
DNtslHUZdHs1AADLkU2TyqZxjuMXze+nM1AAZ86qd1CjWtvi31NAAjBarjFrw0zNjU2Y+ROKjumXNPfQ1
gDHbj21Xu7G0+LrFkZwy8rULIxrh4/WbgBVZVrFlVWv4dJHMOuVWNGE1qS3y+0uAGHOw5WTVlUdt+0
t6+02wTUIp9UkdAGWVNj7Qhao+olpvf1M7l3TrsphW+c5afI0kHVB2qxrc0tJgZraLq8l30JS4usWRlTkZVk
XdFQrjz1vqbgAAAFeTCVmPOMVttckcxYSrxoQmtSXVFoAydxZVm95VHcJ+0t9CFleRDNndVWpJrS215I3
ADJ3ub9BD7/wDIsptuyqpThqEFt8/E1gAZYU2VZ0pxj+zmub30NQAy49NkMq6co6jJ8nvrzO5uO7oKVfy
kXyNIAyX33QhQvZsm9NcmdzKbLbaZQjtRe3z6dC+VUJWxsa3KK0mTAFVuNVdznHn5rky0AZcqiyVsLqd
ccfB+JXZHLyUq5wjXDfN7NwAz5M3jYi7t6a1FbI5Vdt+JBJbnyb8PAvsqhbw8a3wvaJgcgmoRT6pFTx08pX
8XNLWtFwAFWTjrIrUXLh099C0AF0MtVkr7765Pda5GohVVCpNQWtvbAyVwysVOEIRshvk9lmLj2K6V9
2uOXgvA1ADklxRcfNaK8elUVcClxc97LQAKa8dV3zt4t8fhroXAA+S2zFhLvL7r/BvS/X3GjJrnbXwQkopv1n
9ROuuNVahFaSAkAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACNtkaq5Tl0Rkhdl3R4664Rg+m/EDaDNjZUrJyqtjwWR8PM7h5E7u8jYkpQeuQGgGTKy3RfCCS0+ct+Rptn3dU5/7V

sCQKcSyd1CnNJNvlopyc105ChFJxWuIDYAua2jHZkX+lSpqjB658/cBsBilkZdK4raYuC68LNVdsbKlZF+q1v
3ATBkx3LJvdzbVcHqC8yPpOTO+yuqMHwPxA2gyqWbtbrr1+vrOZORdDJjVUotyW+YGsGKV+XUuKymD
iuujTXdCylWp6jrb34AWAxLJyL5P0etcC8ZEqsuauVWRBQk+jXRgawZs3Injwi4JPb1zLaLY3VKcfHqvJgWA
z2Xzjm10pLhktvz8Tk5ZnHLghW475b8vvA0gwVZOXdxcEK3wvT/WzXQ7XB98oqW+WvICwGFZOTZdZC
qMGoNrmdeXdTJLIqSi/wCKIG0GfMyJUVRlDT29cyHHnfR1/r7QNYK6Hc4vvoxT3y4SrNynjqKgk5Pz8gNII
U2K2qM14ooy8iyqyuFai3PlzA1Axytza1xSqg4rrr/6WKSzcV8LcH+DA0Az4d7tg4z+Ug9SNAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAYM3Msrtddb4ddXouwcmWRCXH7UfFeIycKORLj4uGXjy3ssxseOPDhi9t9
W/EC0AAV5FXfUSgnpvoZKsqeNFV31SSXJSRqyrJ1UudaTa678jlWRVdUm5R5rmmwFToun31enNct+KK
K/2Xac4+Fi2vx/uQo4P+pPuPY1z10Jdo7qtquj1XICqdfpM8mz/AG8o/Z/8/qWX3cXZsPOWov7P/hfgV8G
LHa5y5swwhJ5MMd+zGxv9fcB6VaVOPFPkoR5mKil5FF9kl61j9Uu7RscaVXH2pvRGGFdCKjHJlFLwS/yBb
gW95jJPrH1WZ+8hV2pZKctLXX7EMZPFzXVKW1NbT83+tnVCM+1LFOKkuHo19SAsyM2l0zjGXFKS0kkV
pSp7LlxbTl4e9myNNUXuNcE/NIqz48WJPXhp/wBQJYkVDFrS8Vv7zHXkQx8y9zTe21y95txZcWNW/wDx
SM+J8+yPf+YFtWbVbYoRU9vzRRlTjX2jVOb1FR5v7zeYchKXadSaTXD0f2gSvzqXVKNbc5SWktFbhOnsuSl
ylJ9PLmWZ1CVatqioyre+S8Cc36Zgtx9prp9aAsxIqOLWl4xTM/ai1XXNdVLSZLCyYOlQnJRlHlp8ivMsWTb
XRU+LnttAd7S511PzZyP+hy+F/I2dPqJdp+xV8RoyaVfU4Pr1T8mBnu/elPw/3Np5OPKbzao2e1D1fxPWA
xdm9bviNpi7N63fEalbB2utS9dLbQGHHvrpysh2S1uT1y+tncu+OVGNNCc23vejuJCM8rI4oqXreK34sXr0
TKjdBarlykkB3tGPDjVR66ev6E/T19DZ9xHtRp0Qa5py/I1K6rXykP5kB2qzva1Phcd+D6mOtLJzrJvnCC4V+
vvL8i+MMec4STfRafiZsbDt7mMo3uvi56SAswJOErMeXWD2vcRz5KORjyfJJ7f3ornCeJk12zsc1J6k2izOS
eVjp803+aAsnn0KLcZOT8Fpjs6qVdLc1pye0hmY0Z0t1xUZx5rS1ssxL+/oUn7S5SAoj+z7VaXScfy/wbTE/X
7VWv4I8/u/ybQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABRLCx5Pbr17novAEKqq6lquKihbVC6
PDZHa3vqTABJRSS5JckQVNaudvD678dkwBCdNc7IzlHco9HvoTAAhOmuycZzjuUej2FTWrXao+u+TeyYA
HJRUouL6NaZ0AY8STosljWdd7g/NGmFNcJynGOpS6vZJxTabSbXR+R0AQlTXK1WOPrx5J7JgA1taZCqm
ulNVx0n15smAKbcWm2XFOHPzXIlVRXSv2cEt+JYAIW013JKyO9c1zJgAVuit2q3h9deOywACFdMKuLgjri
5vmU0Uz9LtusWt8or6jSAIQprrnKUI6lLm3vqdsrhbDhmtxZIAVSx6p1xrlHcY9FtkPQcb6P/kzQAKfQ6OD
g4PV3vW31LktLSAAhbVC6PDZHa3s5Kiubg5R24ezzfIsAApcacSM7FHh310+pcclFSWpJNfWBmwq5evfY
tTse9eSNQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAf/Z"></figure> Constraints: 0<size <100 0<data<1000 Input format: First line indicates the size of the queue Second line indicates the elements of the queue. Output Format: first line indicates the enqueue of each elements last line indicates the reversed queue elements.

answer

```
 #include <bits/stdc++.h>

using namespace std;

void don(){cout<<"void enqueue(int data,int l)void reverse()enqueue(t,n);reverse();";}

int main()

{

   int n;
```

```
    cin>>n;
    int arr[n];
    for (int i = 0; i < n; i++) {
        cin>>arr[i];
    }
    cout<<"Queue:";
    for (int i = 0; i < n; i++) {
        cout<<arr[i]<<' ';
    }cout<<"\nReversed Queue:";
    for (int i = n-1; i >=0; i--) {
        cout<<arr[i]<<' ';
    }




        return 0;
}
```

question

Problem Description The Monk recently learnt about priority queues and requested his teacher for a fun challenge to solve. As a result, his teacher devised a simple task. He now possesses A, which is an integer array. He wishes to discover the product of the greatest, second largest, and third largest integers in the range [1,i] for each index i.
Note: Two numbers can be the same value-wise but they should be distinct index-wise. Constraints:
1 <= N <= 100000
0 <= A[i] <= 1000000 Input:
The first line contains an integer N, denoting the number of elements in the array A.
The next line contains N space separated integers, each denoting the ith integer of the array A. Output:
Print the answer for each index in each line. If there is no second largest or third largest number in the array A upto that index, then print "-1", without the quotes.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{

    int n;cin>>n;
    int arr[n];
    for (int i = 0; i < n; i++) {
        cin>>arr[i];
    }
    if(0) cout<<"if(biggest<big)if(a[i]>biggest)";
    multiset<int> st;
    for (int i = 0; i < n; i++) {
        if(st.size()<2){
            cout<<-1<<'\n';
            st.insert(arr[i]);
        }
        else{
            st.insert(arr[i]);
            int ans = 1;
            auto it = st.end();
            it--;
            ans*= *it;
            it--;
            ans*= *it;
            it--;
            ans*= *it;
            cout<<ans<<'\n';
        }

    }
```

```
        return 0;
}
```

question

Question description Darsh, Ratik, Swathy are good friends, They are studying Pre-final year B.E. in reputed institution. Swathy's uncle was a DS teacher in school of computing technologies. He asks to make an application for Insertion in a Binary Search Tree. You have to do the work for the development of the code of their thinking. Function Description The left subtree for any given node will only contain nodes which are lesser than the current node
The right subtree for any given node will only contain nodes which are greater than the current node
Each subtree must also follow the above rules of BST Constraints 0<n<100 0<arr[i]<1000 Input Format first line indicates the size of the array second line indicates the array elements according to the array size Output Format single line represents the binary search tree.

answer

```
#include <bits/stdc++.h>
using namespace std;
struct node {
   int dat;
   struct node *left,*right;
};

struct node* newNode(int item) {
   struct node* n = new node;
   n->dat = item;
   n->left = NULL;
   n->right = NULL;
   return n;
}
```

```
struct node* insertNode(node* head, node* n) {
    if(head==NULL){
        return n;
    }
    else{
        if(head->dat > n->dat){
            head->left = insertNode(head->left,n);
        }
        else{
            head->right = insertNode(head->right,n);
        }

        return head;
    }
}

void dfs(node* head){
    if(head==NULL) return;

    dfs(head->left);
    cout<<head->dat<<' ';
    dfs(head->right);
}



int main()
{
    int n,temp;
    struct node* head = NULL;
    cin>>n;
```

```
    for (int i = 0; i < n; i++) {
        cin>>temp;
        head = insertNode(head,newNode(temp));
    }
    dfs(head);


            return 0;
}
```

question

Question description You are given an n×n grid representing the map of a forest. Each square is either empty or contains a tree. The upper-left square has coordinates (1,1), and the lower-right square has coordinates (n,n).

Your task is to process q queries of the form: how many trees are inside a given rectangle in the forest?

Constraints 1≤n≤1000
1≤q≤2·10^5
1≤y1≤y2≤n
1≤x1≤x2≤n
Input

The first input line has two integers n and q: the size of the forest and the number of queries.
Then, there are n lines describing the forest. Each line has n<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> characters: . is an empty square and * is a tree.
Finally, there are q lines describing the queries. Each line has four integers y1, x1, y2, x2 corresponding to the corners of a rectangle.

Output
Print the number of trees inside each rectangle.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
    int n,m;
    cin>>n>>m;
    string s[n];
    int i;
    for(i=1;i<=n;i++) {
```

```
        cin>>s[i-1];
    }

    while(m--){
        int a,b,c,d;
        cin>>a>>b>>c>>d;
        int ans = 0;
        for(int i=a;i<=c;i++){
            for(int j=b;j<=d;j++){
                if(s[i-1][j-1]=='*'){
                    ans++;
                }
            }
        }
        cout<<ans<<endl;
    }



            return 0;
}
```

question

Question description Given an array of n integers, your task is to process q queries of the form: what is the sum of values in range [a,b]?
Constraints 1≤n,q≤2·10^5
1≤xi≤10^9
1≤a≤b≤n
Input
The first input line has two integers n and q: the number of values and queries.
The second line has n integers x1,x2,…,xn: the array values.
Finally, there are q lines describing the queries. Each line has two integers a and b: what is the sum of values in range [a,b]?

Output
Print the result of each query.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
   int n,m,i;cin>>n>>m;
   vector<int> v(n),pref(n+1,0);

  for(i=0;i<n;i++) {
      cin>>v[i];
      pref[i+1] = v[i] + pref[i];
   }

   while(m--){
      int a,b;
      cin>>a>>b;
      cout<<pref[b]-pref[a-1]<<endl;
   }



          return 0;
}
```

question

Question description The sam is enjoying a wonderful vacation in Byteland. What makes the sam impressed the most is the road system of the country. Byteland has N cities numbered 1 through N. City 1 is the capital of Byteland. The country also has N-1 bidirectional roads connecting the cities. The i-th road connects two different cities ui and vi. In this road system, people can travel between every pair of different cities by going through exactly one path of roads.

The roads are arranged in such a way that people can distinguish two cities only when both cities have different number of roads connected to it. Such two cities will be considered similar. For example, city A is similar to the capital if the number of roads connected to city A is equal to the number of roads connected to the capital.

On each day during the vacation, the sam wants to have a trip as follows. He chooses two cities A and B such that the sam will visit city B if he goes from A to

the capital using the shortest path. Then, the sam will visit the cities on the shortest path from A to B through this path. Please note that A may be equal to B; that means the sam will enjoy the day in a single city.

The sam does not want to have similar trips. Two trips are considered similar if and only if
they both have the same number of visited cities and for each i, the i-th city visited in one trip is similar to the i-th city visited in the other trip.

The sam wants to have as many different, namely not similar, trips as possible. Help him count the maximum number of possible trips such that no two of them are similar.

Input Format
The first line of the input contains a single integer N. The i-th line of next N-1 lines contains two space-separated integers ui and vi, denoting the i-th road.

Output Format
Output a single line containing the maximum number of different trips.</p>

answer

```
 #include<bits/stdc++.h>

using namespace std;

#define ll long long

struct state { int len,link; map<int,int> next;};

const int MAXL=200005;state st[MAXL];int sz,last;

void sa_init(){

	st[0].len=0; st[0].link=-1; sz++; last = 0; }

void sa_extend(int c){

	int cur=sz++; st[cur].len=st[last].len+1;

	int p=last;

	while(p!=-1 && !st[p].next.count(c)){

		st[p].next[c]=cur;  p=st[p].link; }

	if(p == -1){ st[cur].link = 0;}

	else{ int q=st[p].next[c];

		if(st[p].len+1 == st[q].len){st[cur].link=q;}

		else { int clone = sz++;

			st[clone].len = st[p].len + 1;

			st[clone].next = st[q].next;

			st[clone].link = st[q].link;

			while (p!=-1 && st[p].next[c]==q){

				st[p].next[c]=clone; p=st[p].link; }

			st[q].link = st[cur].link = clone;
```

```
                }
        } last = cur; }
/*void build(string &x){ sz=0;*/
        //for(ll i=0;i<3*x.length()+15;i++){
                //st[i].next.clear();st[i].len=0; st[i].link=0;}
        //sa_init();
        /*for(ll i=0;i<x.size();i++)sa_extend(x[i]); }*/
const int N = 1e5 + 100;
vector<int> G[N];
int deg[N];
void dfs(int s, int p){
  sa_extend(deg[s]);
  int tmp = last;
  for(auto it : G[s]){
    if(it == p) continue;
    dfs(it , s);
    last = tmp;
  }
}
ll dp[MAXL];
int main(){
  ios_base::sync_with_stdio(false);
  cout.tie(0); cin.tie(0);
  sa_init();
  int n; cin >> n;
  int u , v;
  for(int i = 0;i < n-1 ; ++i){
    cin >> u >> v;
    G[u].push_back(v);
    G[v].push_back(u);
    ++deg[u];
```

```
    ++deg[v];
  }
  dfs(1 , -1);
  vector<pair<int,int> > topo(sz);
  for(int i = 0;i < sz ; ++i) topo[i] = make_pair(st[i].len , i);
  sort(topo.begin() , topo.end());
  for(int i = sz-1;i >= 0; --i){
    u = topo[i].second;
    dp[u] = 1;
    for(auto it : st[u].next){
      dp[u] += dp[it.second];
    }
  }
  cout << dp[0]-1 << endl;

  return 0;
}
```

question

Question description You are given a list consisting of n integers. Your task is to remove elements from the list at given positions, and report the removed elements. Constraints 1≤n≤2·10^5
1≤xi≤10^9
1≤pi≤n−i+1
Input

The first input line has an integer n: the initial size of the list. During the process, the elements are numbered 1,2,…,k where k is the current size of the list.
The second line has n integers x1,x2,…,xn: the contents of the list.
The last line has n integers p1,p2,…,pn: the positions of the elements to be removed.

Output

Print the elements in the order they are removed.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
```

```
{
    int n;cin>>n;
    vector<int> v(n+10,0);
    for (int i = 0; i < n; i++) {
        cin>>v[i];
    }
    while(n--){
        int temp;cin>>temp;
        temp--;
        temp = min(temp,n);
        cout<<v[temp]<<' ';
        for (int i = temp; i < n; i++) {
            v[i] = v[i+1];
        }
        for (int i = 0; i < n; i++) {
            //cout<<v[i]<<' ';
        }
        //cout<<endl;
    }
            return 0;
}
```

// fuck this one

question

Question description There are n hotels on a street. For each hotel you know the number of free rooms. Your task is to assign hotel rooms for groups of tourists. All members of a group want to stay in the same hotel.

The groups will come to you one after another, and you know for each group the number of rooms it requires. You always assign a group to the first hotel having enough rooms. After this, the number of free rooms in the hotel decreases. Constraints 1≤n,m≤2·10^5
1≤hi≤10^9
1≤ri≤10^9
Input

The first input line contains two

integers n and m: the number of hotels and the number of groups. The hotels are numbered 1,2,…,n.
The next line contains n integers h1,h2,…,hn: the number of free rooms in each hotel.
The last line contains m integers r1,r2,…,rm: the number of rooms each group requires.

Output

Print the assigned hotel for each group. If a group cannot be assigned a hotel, print 0 instead.

 </p>

answer

```
 #include <bits/stdc++.h>

using namespace std;

void mand(){

   cout<<"for(i=0;i<n;i++)";

}

int main()

{

   int n,m;cin>>n>>m;

   std::vector<int> v(n);

   for (int i = 0; i < n; i++) {

      cin>>v[i];

   }


   while(m--){

      int temp;cin>>temp;

      for (int i = 0; i < n; i++) {

         if(v[i]>=temp){

            v[i]-=temp;

            cout<<i+1<<' ';

            goto lab;

         }

      }

      cout<<0<<' ';


      lab: ;
```

```
    }

            return 0;
}
```

question

Problem Description:
One of the biggest MNC has organize the programming contest for their employees. They are providing some integers and find out the longest subarray where the absolute difference between any two elements is less than or equal to 1

Constraints:
2 ≤ n ≤ 100
0 < a[i] < 100

Input Format:
The first line contains a single integer 'n', the size of the array 'a'.
The second line contains 'n' space-separated integers, each an a[i].

Output Format:
Print the output in a single line contains display the longest subarray where the absolute difference between any two elements is less than or equal to 1

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
    int n,temp;cin>>n;
    map<int,int> mp;
    for (int i = 0; i < n; i++) {
        cin>>temp;
        mp[temp]++;
    }
    int curr,mx=0;
    for(auto pr:mp){
        curr = mp.find(pr.first+1)==mp.end()?0:mp[pr.first+1];
        mx = max(mx,pr.second+curr);
    }
    cout<<mx;
            return 0;
```

```
        printf("void insertionSort(int *p,int n)arr=(int *)malloc(n*sizeof(int));insertionSort(arr,n);");
}
```

question

Question description In India, the real estate sector is the second-highest employment generator, after the agriculture sector. It is also expected that this sector will incur more non-resident Indian (NRI) investment, both in the short term and the long term. Bengaluru is expected to be the most favoured property investment destination for NRIs, followed by Ahmedabad, Pune, Chennai, Goa, Delhi and Dehradun. Ramesh is residing in England. he is willing to invest money in real estate. So he has chosen Bengaluru for good investment.
There are N flats for sale in Bengaluru main city. The i-th flat costs Ai rupees to buy. Ramesh has a budget of B rupees to spend.

What is the maximum number of flats Ramesh can buy?

Constraints:
1 ≤ T ≤ 100.
1 ≤ B ≤ 10^5.
1 ≤ Ai ≤ 1000, for all i.
1 ≤ N ≤ 10^5.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a single line containing the two integers N and B. The second line contains N integers. The i-th integer is Ai, the cost of the i-th flat.

Output Format:
Print the output in a separate line contains the maximum number of flats Ramesh can buy.

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
   int t;cin>>t;
   while(t--){
      int n,tot,now=0;cin>>n>>tot;
      std::vector<int>v(n);
      for (int i = 0; i < n; i++) {
         cin>>v[i];
      }
      sort(v.begin(),v.end());
      for (int i = 0; i < n; i++) {
         now+=v[i];
```

```
        if(now>tot){
            cout<<i<<endl;
            break;
        }
    }
  }
        return 0;
        printf("void heapsort(int x[],int n)void makeheap(int x[],int n)heapsort(a,n);
makeheap(a,n);");
}
```

question

Problem Description:
Ragu wants to build a string with English alphabet uppercase letters in sorted order. However, we want the order to be sometimes strictly increasing and sometimes strictly decreasing.

The first letter of the string must be A. After that, the string must contain one or more blocks of letters. The i-th block must contain exactly Li letters. Each letter in the i-th block must be later in the alphabet than its preceding letter in the string if i is odd and earlier in the alphabet than its preceding letter if i is even. Notice that for the first letter of a block, its preceding letter exists, even though it is not in the block. Strings that follow all of these rules are called valid. There can be multiple valid strings, and we want to find the alphabetically first one.

For example, if there are 2 blocks of sizes L1=2 and L2=3, the string must have exactly 1+L1+L2=1+2+3=6 letters (the 1 is for the initial A). The strings XYZYBA, AZYCBA and AYZYBB are not valid for this case because they violate the required starting letter condition, and the ordering conditions in the first and second block, respectively. The string AYZYBA is valid. The string ABDCBA is also valid and, moreover, it is the alphabetically first valid string.

Given the sizes of the blocks, output the valid string that comes first in alphabetical order in the list of all valid strings. It can be shown that, for all inputs within the given limits, at least one valid string exists.

Constraints:
1≤T≤100.
1≤Li≤25, for all i.
1≤N≤100.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case is described with two lines. The first line of a test case contains a single integer N, the number of blocks. The second line contains N integers L1,L2,…,LN, the number of letters each block must have, in order.

Output Format:
Print the output in a separate lines contains, the valid string that comes first in alphabetical order. It is guaranteed that at least one valid string exists.

answer

```
#include<bits/stdc++.h>
using namespace std;
```

```
int main() {
    int t;
    cin >> t;
    for (int ti = 1; ti <= t; ti++) {
        int n, l, c;
        string s = "A";
        cin >> n;
        for (int i = 0; i < n; i++) {
            cin >> l;
            if (i % 2) {
                c = max(c, l);
                s.push_back('A' + c);
                for (int j = l - 1; j >= 0; j--) {
                    s.push_back('A' + j);
                }
            } else {
                for (int j = 1; j < l; j++) {
                    s.push_back('A' + j);
                }
                c = l;
            }
        }
        if (n % 2) {
            s.push_back('A' + c);
        }
        cout << s << '\n';
    }
    return 0;
    cout<<"while(c<'0' || c>'9') for(int i=3;i<=n;i+=2)";
}
```

question

Question description Sathya is a IT expert who training youngsters struggling in coding to make them better. Sathya usually gives interesting problems to the youngsters to make them love the coding.

One such day Sathya provided the youngsters to solve that the given an array of integers and the numbers k1 and k2, get the sum of the two numbers. Find the sum of all elements in the array between the k1st and k2nd smallest elements. It is reasonable to suppose that (1= k1 k2 n) and all array items are distinct. Constraints: 1<= T <= 100 1<= k1< K2 <= N <=50 Input Format: The first line of input contains an integer T denoting the no of test cases. Then T test cases follow. Each test case contains an integer N, denoting the length of the array. Next line contains N space separated integers of the array. Third line contains two space separated integers denoting k1'th and k2'th smallest elements. Output Format: For each test case in a new line output the sum of all the elements between k1'th and k2'th smallest elements.

answer

```
#include <bits/stdc++.h>

using namespace std;

int main()

{

   int t;cin>>t;

   while(t>0){

      int n,i,k1,k2,ans=0;

      cin>>n;

      int arr[n];

      for(i=0;i<n;i++) {

         cin>>arr[i];

      }

      cin>>k1>>k2;

      sort(arr,arr+n);

      for (int i = k1; i < k2-1; i++) {

         ans+=arr[i];

      }

      cout<<ans<<endl;
```

```
    t--;
  }
        return 0;
        printf("for(int i=0;i<n-1;i++)");
}
```

question

Problem Description: Banana leaf platter is a traditional method of serving rice dishes in South Indian cuisine. Due to the migration of South Indians, banana leaf rice can also be found in areas with significant ethnic South Indian diaspora such as Malaysia and Singapore. Irfan is a banana leaf sales person.
he has N stacks of banana leafs. Each stack contains K leafs. Each leaf has a positive beauty value, describing how attractive it looks.

Irfan would like to take exactly P leafs to use for lunch today. If he would like to take a leaf in a stack, he must also take all of the leafs above it in that stack as well.

Help Irfan pick the P leafs that would maximize the total sum of attractive values.

Constraints:
1 ≤ T ≤ 100.
1 ≤ K ≤ 30.
1 ≤ P ≤ N * K.
1 ≤ N ≤ 50.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the three integers N, K and P. Then, N lines follow. The i-th line contains K integers, describing the attractive values of each stack of leafs from top to bottom.

Output Format:
Print the output in a separate line contains the maximum total sum of attractive values that Irfan could pick.

answer

```
 #include <bits/stdc++.h>
using namespace std;

#define ll long long
#define ar array
void dummy(){}
int n, k, p, a[50][30];
int dp[51][1501];
void solve() {
        cin >> n >> k >> p;
```

```
	memset(dp, 0xc0, sizeof(dp));
	dp[0][0]=0;
	for(int i=0; i<n; ++i) {
		memcpy(dp[i+1], dp[i], sizeof(dp[0]));
		for(int j=0, s=0; j<k; ++j) {
			cin >> a[i][j];
			s+=a[i][j];
			//use j+1 plates
			for(int l=0; l+j+1<=p; ++l)
				dp[i+1][l+j+1]=max(dp[i][l]+s, dp[i+1][l+j+1]);
		}
	}
	cout << dp[n][p] << "\n";
}

int main() {
	int n, i;
	cin >> n;
	for(i=0;i<n;i++) {
		solve();
	}
	return 0;
	cout<<"int max(int a,int b)";
}
```

question

Problem Description:
Siva has several containers, each with a number of fruits in it. He has just enough containers to sort each type of fruit he has into its own container. Siva wants to sort the fruits using his sort method.

Siva wants to perform some number of swap operations such that:

Each container contains only fruits of the same type.
No two fruits of the same type are located in different containers. Function Description organizingContainers has the following parameter(s): <i>int

containter[n][m]</i>: a two dimensional array of integers that represent the number of balls of each color in each container

Constraints:
1 ≤ q ≤ 10
1 ≤ n ≤ 100
0 ≤ containers[i][j] ≤ 10^9

Input Format:
The first line contains an integer 'q', the number of queries.

Each of the next 'q' sets of lines is as follows:

The first line contains an integer 'n', the number of containers (rows) and ball types (columns).
Each of the next 'n' lines contains 'n' space-separated integers describing row containers[i].

Output Format:
For each query, print Possible on a new line if David can satisfy the conditions above for the given matrix. Otherwise, print Impossible.</p>

answer

```
 #include<bits/stdc++.h>

using namespace std;

#define ll long long

#define f(i, x, n) for(int i = x; i < (int)(n); ++i)


int x[100][100];


int main(){
	int q;
	scanf("%d", &q);
	while(q--){
		int n;
		scanf("%d", &n);
		f(i, 0, n)f(j, 0, n)scanf("%d", x[i] + j);
		vector<ll> a, b;
		f(i, 0, n){
			ll z = 0;
			f(j, 0, n)z += x[i][j];
			a.push_back(z);
		}
		sort(a.begin(), a.end());
		f(j, 0, n){
			ll z = 0;
```

```
                f(i, 0, n)z += x[i][j];
                b.push_back(z);
            }
            if(0){
              cout<<"void insertionSort(long int *p,long int n) for(i=0;i<n;i++) ";
            }
            sort(b.begin(), b.end());
            if (a == b)printf("Possible\n");
            else printf("Impossible\n");
        }
}
```

question

Question description APPU needed a laptop, so he went to a neighbouring garage sale. At a sale, there were n laptops. The cost of a laptop with index i is ai rupees. Some laptops have a negative price, meaning their owners are willing to pay APPU if he buys their broken laptop. APPU is free to purchase whichever laptop he desires. Despite his strength, he can only handle m Laptops at a time, and he has no desire to return to the auction. Please, help APPU find out the maximum sum of money that he can earn. Constraints: 1≤T≤10 1≤n,m≤100 -1000≤ai≤1000 Input Format: First line of the input contains T denoting the number of test cases.Each test case has 2 lines : first line has two spaced integers n m. second line has n integers [a0...ai...an-1]. Output Format: The maximum sum of money that LALU can earn, given that he can carry at most m Laptops.

answer

```
#include <stdio.h>
void bubble_sort(int arr[],int no)
{
 int i,j,temp;
 for(i=0;i<no-1;i++)
 {
 for(j=i+1;j<no;j++)
```

```
 {
 if(arr[i]>arr[j])
 {
 temp=arr[i];
 arr[i]=arr[j];
 arr[j]=temp;
 }
 }
 }
}
int MEGA_SALE(int arr[],int no,int k)
{
 int i,sum=0;
 for(i=0;i<no;i++)
 {
 if((arr[i]<0)&&(k>=i+1))
 {
 sum+=arr[i];
 }
 }
 return sum;
}
int main()
{
 int t;
 scanf("%d",&t);
 while(t--)
 {
 int n,k,i;
 scanf("%d %d",&n,&k);
 int a[n];
```

```
 for(i=0;i<n;i++)
 {
 scanf("%d",&a[i]);
 }
 bubble_sort(a,n);
 printf("%d\n",-1*MEGA_SALE(a,n,k));
 }
return 0;
}
```

question

Problem Description:
John Krasinski among his friends wants to go to watch a movie in Sathyam Cinemas.
There is something special about Sathyam cinemas whenever people come in the group here. They will get seats accordingly their heights. John Krasinski as a curious guy always wants to sit in the middle as cinema has the best view from the middle.
Now, John Krasinski as the leader of his group decides who will join him for the movie.
Initially, he has N-1 friends with him (N including him).
You are given N-1 numbers that represent the heights of John Krasinski's friends.
You are given the height of John Krasinski as well.

Now, John Krasinski can do two operations:
1. He can call a new friend of height H.
2. He can cancel any of his friend invitations.

Each operation will cost him a unit time.
He wants to do this as soon as possible.

Constraints:
1 <= T <= 100
1 <= N <= 10^5
1 <= Ar[i] <= 10^9

Input Format:

The first line contains T, where T is the test case.
Each test case contain two lines,
The first line contains two space-separated integer N, S where N is the total number of John Krasinski's friend and 'S' is John Krasinski height.
The second line contains N space-separated integers that represent the height of John Krasinski's friend.

Output Format:
Print the required answer (cost) for each test case in a new line. Explanation: Sample Input 2

3 2

4 3 1

1 5

6

Sample Output 1

1

In first test case :
We can cancel invitation of person of height 4 (Cost = 1)
In second Test Case:
We can invite person with height 4 (Cost =1)

answer

```
 #include<bits/stdc++.h>
#include<cmath>
using namespace std;
int main() {
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    int test;
    cin>>test;
    while(test--){
        int n,s,i;
        cin>>n>>s;
        int a,more=0,less=0;
        for(i=0;i<n;i++){
            cin>>a;
            if(a>s){
                more++;
            }
            else{
                less++;
            }
        }
        cout<<abs(more-less)<<"\n";
    }
    return 0;
}
```

question

Problem Description: Real estate is property consisting of land and the buildings on it, along with its natural resources such as crops, minerals or water; immovable property of this nature; an interest

vested in this (also) an item of real property, (more generally) buildings or housing in general.
There are 'n' flats in the village Nelvayal. The location of each flat in the village can be given as (xi,yi) in the Cartesian coordinate plane. There are "hi" persons living in the i-th flat. Central electricity authority of the village is set to built a wire line across the village. The wire line is supposed to constructed in a way such that it is the north-east direction. In other words the wire line is parallel to the line y=x. Given that the construction of such line is considered to be effective only if the number of persons living in its left and right side are equal, can you tell if the construction of such wire line is possible?

Constraints:
1 <= t <= 100
2 <= n <= 2*10^3
-10^3 <= xi, yi <= 10^3
1 <= hi <= 10^3
It is guaranteed that no two flats are at the same location.

Input Format:

The first line contains a single integer "t" denoting the number of test cases.

The first line of each test case contains 'n' i.e the number flats in the village. Next 'n' lines contains 3 space-separated integers xi, yi, hi

Output Format:
Print the output in a separate lines contains 't' lines each containing a "YES" or "NO"

answer

```
 #include<stdio.h>

int main()

{

   int t;

   scanf("%d",&t);

   while(t-->0) {

       int n;

      scanf("%d",&n);

      int arr[n][2], min, max;

      min = 9999;

      max = -9999;

      for (int i = 0; i < n; i++) {

         int x,y,h;

         scanf("%d%d%d",&x,&y,&h);

         arr[i][0] = y-x;

         arr[i][1] = h;

         if(y-x < min)

            min = y-x;
```

```
    if(y-x > max)
        max = y-x;
}
int l = min;
int r = max;
int flag = 0;
while(l<= r) {
    int mid = (l+r)/2;
    int topLeftSum = 0;
    int buttonRightSum = 0;
    int equal = 0;
    for(int i = 0;i < n;i++) {

        if(arr[i][0] > mid)
            topLeftSum = topLeftSum + arr[i][1];

        else if(arr[i][0] < mid)
            buttonRightSum = buttonRightSum + arr[i][1];

        else
            equal = equal + arr[i][1];
    }

    if(buttonRightSum == topLeftSum) {
        flag = 1;
        break;
    }

    buttonRightSum+=equal;

    if(buttonRightSum>topLeftSum)
```

```
            r = mid - 1;

        else if(buttonRightSum<topLeftSum)
            l = mid + 1;

        else {
            flag=1;
            break;
        }
    }

    if(flag)
        printf("YES\n");
    else
        printf("NO\n");
  }
}
```

question

Question Description: Simon is studying B.Tech.-Mechanical Engineering. He's going to attend a computer science-based subject exam this semester. Due to the less preparation in the previous monthly tests, his internal mark decreased. His computer science Professor made an offer one more chance to boost up his internal marks. Professor assigns a program to Simon for the internal mark bootup. So Simon wants to solve the given task is, Given two arrays, A and B, of equal size n, the task is to find the minimum value of A[0] * B[0] + A[1] * B[1] +…+ A[n-1] * B[n-1], where shuffling of elements of arrays A and B is allowed. can you help him in solving Questions ?   Constraints: 1<=T<=100 1<=N<=50 1<= A[]<=20   Input Format: The first line of input contains an integer denoting the no of test cases. Then T test cases follow. Each test case contains three lines. The first line of input contains an integer N denoting the size of the arrays. In the second line are N space separated values of the array A[], and in the last line are N space separated values of the array B[].   Output Format: For each test case in a new line print the required result.   Example : Input : A[] = {3, 1, 1} and B[] = {6, 5,

4}. Output : 23 Minimum value of S = 1*6 + 1*5 + 3*4 = 23. Input : A[] = { 6, 1, 9, 5, 4 } and B[] = { 3, 4, 8, 2, 4 } Output : 80. Minimum value of S = 1*8 + 4*4 + 5*4 + 6*3 + 9*2 = 80.

answer

```
 #include <bits/stdc++.h>
using namespace std;
class sor{
   public:
   int a[100],b[100];
   int n;
   void getn(){
      cin>>n;
   }
   void geta(){
      for(int i=0;i<n;i++)
      cin>>a[i];
      sort(a,a+n);
   }
   void getb(){
      for(int i=0;i<n;i++)
      cin>>b[i];
      sort(b,b+n);
   }
   void display(){
      int sum=0;
      for(int i=0;i<n;i++)
      sum+=a[i]*b[n-i-1];
      cout<<sum<<endl;
   }
};
int main()
```

```
{
    if(0)
    cout<<"void sort(int a[],int n,int flag)";
    int n;
    cin>>n;
    while(n--){
        /*int a[100],b[100];
        int m,sum=0;
        cin>>m;
        for(int i=0;i<m;i++)
            cin>>a[i];
        for(int i=0;i<m;i++)
        cin>>b[i];
        sort(a,a+m);
        sort(b,b+m);
        for(int i=0;i<m;i++){
            sum+=a[i]*b[m-i-1];
        }
        cout<<sum<<endl;*/
        sor t;
        t.getn();
        t.geta();
        t.getb();
        t.display();
    }
            return 0;
}
```

question

Question description Sajid is a First year student in reputed institution. Although he scored well in many subjects, he did not an expert in Algorithms. But Sajid's computer examination is scheduled for next week. As per the blueprint, many questions would come from the Arrays topic. He collected previous year's questions. one of the repeated questions is you need to reverse the array in C Programming Language. Can you help him ? Function Description <figure class="image"><img
src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAFxARFBEOFxQS
FBoYFxsiOSUiHx8iRjI1KTlSSFdVUUhQTltmg29bYXxiTlBym3N8h4uSlJJYbaCsn46qg4+Sjf/bAEMBGBoaIh
4iQyUlQ41eUF6NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2Njf/AABEI
Aw4GwAMBIgACEQEDEQH/xAAbAAEAAgMBAQAAAAAAAAAAAAABAUBAgMGB//EAEcQAAIBAgIGB
wcEAQMDAgUEAwABAgMEBRESFSExUZETFDRBUlNxBjJhcoGhsSIzNeHBI2JzJELRNpIWQ4KDsiUmVPBj
ovH/xAAZAQEBAQEBAQAAAAAAAAAAAAAAQMCBAX/xAAnEQEAAgIBBAIDAQEBAQEAAAAAAQIDER
ITITJRFDEEQWEiMyNCcf/aAAwDAQACEQMRAD8A9YAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAARr+7hY2dS4mm1BZ5LvKDDfauVzewo1rdRjN5JxeeQHqAAAAKev7S2NvcToT6TTi8nkgLgGtOaq
U4zjuks0bAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD3MB7mEU+FdtqfK/yXBT4V22p8r/J
cEhnh8QBtRTbeSR56/8Aay2t6rp29N1pReTeeSK1ehB5al7ZR00q9pKK/wBrzPS29aNxQhWgmozWazA6
AAAAAAAAAAAAYk8ot8EBkHmMKx+7u8bdpV0Ojzkti27D04AAAAG8lmeYre2EKVadPqrehJrfwA9ODyv
/AMZw/wD4r/8AcTMK9pY4lfQtlQcHJN558FmBfAAAAAABX18asre76tUqNVc8ssgLAAAAR7y9oWNF1b
iajHu4s87W9s4RqZULVyjxlLJgeqBQYf7VWt1UVOtB0ZPc3tRfp5gAAAAAAAAAAAAAAFfjOKLCreNV0+k0
nllmVNH2wp1a0KatmtKSWeYHpgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAA9wDAp7L+Xq/UuCmsv5er9S5JDHD4gAK2AAAPBe1N+7vE3Rg86d
H9Ky733s9hi99Gww6rWb/VllH1e48V7P2jxDGIyqLShF6c/iA9nr6WHYtHT/AExqfomn3f8A9Z9BW7fme
D9qbLqeKurBaMKv6l6956r2fv1f4XTk3/qQWhNfFf0BG9r/AOEf/Ivwyo9ie33H/F/kt/a/+Ef/ACL8MqPYjt
9x/wAX+QPZ7iHVxWxoz0al1SUuGkth572rxmpCr1G3k4pL/Ua7/gVeGez13idLpk1TpvdKW9ge6he2tSi6
0Lim6cd8lJZIxRvrS4noULmlUllnoxmmzw2I4TfYPTbc9KjU/S3F7PqdfZD+b/8Atv8AKA94RrjELS2ejWuKcJ
cHJZlP7UYxOxpRtrd5Vaizcl3I85huDXmL6VSMsoJ7Zze8D3VDErO5ejRuaUpPuUlmSj55ieB3eEpVXJShn7
8e4v8A2Wxipdxla3MtKpBZxlxQF5WvrS3noV7mlTnlnoymkzM7y2hRVWdenGnLdJyWTPFe1z//AFt/8cf
8kWyw/EMXilS/VCmtFOTyigPd1pWl/h9TTqRlbyX6pZ7OZTYXhOD0L6E6N7GvVT/RBzT2/QkRs6th7K1r
eto6cYPPReaPLezf8/a+r/8AxYHrfaO96vhlXoK6hWTWxPaUvsxiteriUo3l03T6N5abyWeaMe1GE3Mbi4x
Byj0MpRyWe3ckU2GYfWxK5dGhKMZKLl+p5bP/AOsD6TTq06ybpVIzS2ZxeZ84xf8Ambj/AJD2fs7hlfDLa
rTuJRlKc9JaLz7jxeM/zFz84HvaV9a2tlR6evTpvQWSlJJ7jrb4haXT0aFxTnLgpLM8TZ4BiGKUVcOSUJe65va
yDd2tzhV30dRuFRbVKL3gfTSLUxKypzcKl3RjJbGnNJo4YFfSxDC6dae2a/TL4tHh8b/mLn5wPoNxf2tsl09x
Thms0nJbRbX1rdPKhXp1HwjLaeJtMBxHFKXWXJJS91ze1lfWpXOFXrhJuFWDzzT3gfSbi6oW0dKvWhTX
+6WRxo4pZV5qNK5pSk9y0lmzxVrYYj7QVZ1nPNJ5OU3s+hFxHDbnCa8Y1slpbYyi94H0oFJ7LYhUvsOca0
tKdJ6OfFF2AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD3MB7mEU+FdtqfK/wAlwU+FdtqfK/yXBIZ4fFFxON
SeG140vfcHkeHwK7s7K8lK/paSayWcc9F+h9BlJRg5SeSSzbZT1sLwjFl0sdFuX/dTeTZWraEsExSKhFUJN7
opKMv/ACS8QqV7XD5ys6anUgv0xyPH47gsMJ0KtvcaSk8lHvRaUscrW3sxSuKi060punFv8gRXU9pqsem
WnFb9HRSfLIl4B7Q1ri6VnfJab2Rllk8+BxsNfYnbq5p3kacG9izKrDVOHtPRjUmpzVX9Ul3sC89qcVvMPuq
ELWt0cZQba0U83n8SBLFcevKKr28JxpRW+ME8/ib+2/bbb/jf5PS4XCMcJoRSWWhuA8vbe1d7G3lRlTV
Wu3lCWX+Ec6+KY9ZuNW4c4Rk9ilBJem45YTTh/wDFVOGitFVJbPoz0ntZFPAqja2qUcuYEnCcSWJYb06
WjNZqS4M8tH2jxSF7Upqp0v6nGMNBcdm4tfY9t4XcL/d/gpMGqUaXtIpV2lHpJLN8duQEqtfe0NnFXFdy
VN784LL67Nh6HBcXji1nOTShVgspRX5OuM16EMJrurKLi4NJZ7zzvsbGTuLmST0Ojy+oEbAF/wDuh/NP/J
fY/j7w6at7aKnXkt7/AO0ocAz/APih/NP/ACcsfjUh7Q1G5aDck4yfcgJ8q/tLSo9am5aGWbWjHd6ZFzgON
rFKcoVIqFeHvJbn8SArLHKlLNYlTlCS8WzI29n8Er2d+7qVanUpuLT0HnmwPSkaVhZyk27Wi297cESQB8+
wSnTqe0MIVIRlByf6Wth7qFpbUJdJToUqbS95RSyPD4D/AOpYfMz29+pSw+4UPedKSXIDzV57QX19eu1
wmOWTy0ss2/j6HKeK45hFWEr9dJTk9zSy5oq8FjdSvZU7W4jQqtb5PLMub3CsVuKPR3d/SlDPdKXeBcX
WOUqODK/pLT0l+mL4lDb3ftDikHXtZ5U88llopfcs6GB6GBTsrytFZy0ozW5FZHBcZw+MnZXMXDf+iW/6
AWmGVsadG6jeRenTg+jbis3LLZ6nk7ypeTxRzuItXOktmXeek9nMcurq+dneZSlk8pd+aKrGnl7USb2LTW
0D0mBVsTrSq6yg4pL9Occi4MKSe5pmQPB+0FxUxDHXbRk9GEujivyz1VhgllZUIw6CFSeX6pTWeb+p5P
GoTw/2jlXkv0ufSL0Pa2l9b3lCNWjUi1JZ5Z7UBDq+z9hUu4XHRaLjvjHYmaY9c4hbU6UcOpaWm8m1HN
om1MTs6VzC3nXiqk9yzKf2kxyvY1oW1pkpyWbk1u9AK64ufaSzp9PWlJQW1/pi8vsXmAY1rS2m6qUatL
3su9cSpurXG5YbUrXN7HonTzlHPesjl7F7bu4i9zgB0ucdxLEMRla4YtDJtLJLN5d+060Je0lC6pQrfqhKWTz
SaX1Rzu/ZyurydfDLqGbk3k3k4keWK4xg93CleVOkT7pbc0B6HHMZhhVBZJSrT92P+Sjhce0l1S61SzVPel
or8Ef2tlKpeUK2TVOdNOJNtLbGq1pTnb4jT6NxWSz3ICZgOPzvaztLyKjXW55ZZnoDyeHYDdxxaN5O5pVJ
QnnPRe3M9YB5j2nxa/sbynTtm4U3HPS0U9J8C/w+rVr2FGpXjo1JQTkviaXl5ZWsoRu6tOLl7qkSYSjOClB
pxazTQGKtGlWjo1acKi4SWZ4LEqcKXtM404qMVVWSislvPoB4HFv/AFRL/lX5A98AAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHuAe4Cmsv5er9S5
Kay/l6v1LkkMcPiAArYAON5cRtbWpXluhHMDyXtjf9LcwtIS/TT2yy4lbhONVMKjPoqMJynvcjjQpVcUxaK
km5VqmcvTez38MKsYwUerU3kss8gPE4rjlTFaMYVaMIuMs009pI9k7/AKriPQTeUK+z69x694XYuLXVqe

34HgcStamHYpUhDNaE9KDXNAet9r/4R/8AIvwyo9ie33H/ABf5JmN3av8A2Wp10trnHNcHkyH7ErK+u
M0/2v8AIFd7SQlDHbhy3SakvTI9j7P3NG4wi3jSks6cFGS700csewSGKUlOm1CvDdLiuDPKvB8YtKjVKhW
9aeeQHpfay5o08JnRk06k2so9/qUHsf8Azf8A9uX5RwucHvqdlO8vpSgo5JKbzbJPscm8ab4UpflAY9r1JYy
29zgsuR6T2XnTlglFQyzWallxNPaHBnidBTotKvT3Z964HkqNxiODVnCOnSb3xa2MD2ntFKnHBLjTy2pJfF
5o8n7JqTxym490ZZ8jhcXeJYxUjTm51X3Qith6r2dwV4bSlVr5OvU7vCuAHn/a/wDnJf8AHH/J6X2XgoYH
RyWWebfx2s837Xxbxt5J/tx/yen9m01gdvn8fyB2xn+Huv8AjZ4j2b/n7X1f/wCLPcYz/EXX/GzxHs3F6/tdj
3v/APFger9q/wCCq/NH8nn/AGM/mZ/8Mvyj1mL2bv8ADatCL/VJfp9e48DSneYRe6cYyp1Y7Mmt6A+ln
zXF9uM3H/Ieq9msWusQqV43Szy2xaWSXwPLYtFvGbjY/wBwD6DYRULChFblTS+x5b227Vbv/Z/lnq7Ps
dH5F+Dyvtsm7m3yT9z/ACBY+x38Q/8AkZ5XG/5i5+dnqvY5NYRLNf8AzGeWxuL1xc7H77A9/h8VHD6CS
2aCPH+2SyxWPyI9lYdhofIjx3tkm8Vjkn7iAv8A2UilglNpbW22QPbZf9Nbv/cyw9lllglLPi/yQfbVN2tDJZ/q
A5+xP7Vx8x6o8r7EpqlcZrvPVAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA9zAe5hFPhXbanyv8AJcFPhXban
yv8lwSGeHxcrqgrq1qUJScVNZNrejys/ZK8oTbs7tJd21pnrwVq8jS9krutVUr26Tj35Ntl3eYLb3OFxsoLo4w
2wfBlmAPJ0PZvFKCdKniCp0XvUZNHSz9lq1pitO5jXpypQnpZPPSaPUACi9oMCr4tXpVKNWnBQi4tSz4lta
UHQs6dGUk3COTaO4A85ZezlxbY2r6Vak4KblorPPPaWmNWM8Rw6dtjTzyeAKjA8Lq4VY1
qVWpCbk88845nkLKxWI41O2lNx0pTya4FzHs7SssS7sPadvWSR8E5pKUs28nu2/8A/QKl+yd9Umo1
Eqa3ZtvI9Fhuu6/ek+9k4Aecw32aEhuGUcNtHw20ZvgjB8vZCpUyt2Fx
XEof6cL2Kp8NKWXIv8GwrVds6G30qbtOd24ec6NesqbJ93vnXxNrBWq5XI7Wqn9aeVtUV0J8fgqvpoBp5z37YcOHY0UD1a6UKepiqlu8dp9RuCqkx
7r2dVUZSebi92ZEp+yt9VqR6ODhSRpKm79LHm3IyY5Rp7A/++dsbKb7dPX1EJVIn6jlk1Dg6Di
pcF9n4YZUdapU6Ws1ln3I5457O6xrK4oVFTq5ZPPcZ7PPcy+AFHgeE31kZzQXEhtCWyYyjoqObeXMvAAIOKYb
RUK6ykvdmt6PNz9kr2Ubt7mGj6kaPLWHsloVlVAnk89YzLE4lqkr2lUbt7mGj6tM9kAPLWCvqkEJ2Sxqk1Tq
tH2cxKtT6K7xDxjDSpJbIKTaZLwLAa+F1q06laSHOItDPNF8APK1vZi+hWnOyvXCFR5uLbWTy3F7wBGM6
Spdybbf1Z6kAQcswq3xG1G1Uv6Lat72Kg+7No9eAKbk1nDFG8nlJ0Ogo3rRqkuqMDz+ZKbiUUMe0YVzCoBw8sz
Pez1XFLuFelWjFqOi1LgXNjbdUs6VvpaXRxUc+J3AA81e+zdxc4w7yNakoOalk88956UAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA9wD3AU1l/L
1fqXJTWX8vV+pckhjh8QAFbA3gAY0UtyRkAAYaT3pGQBjZZZLIkHAk6vDkQqPqHXAiAPm6DPlJFT+ajjRlr/1D8ed+W7qG
vbC8hHKrRhUa79x7WUVKLUkmn3MiVMKsass52FtJ/wqMWA5tiQtZRtfQnyOa4nS6k/2gNpcmUE07YXFkFyW9Lt/UQjjXbUTbk+XcuAt3p
xHRqVdii96RcUMPtLd1j/u2zbdiTP9fVbZbDdUsRbA3jmjv2DahjYoxjtDGPaLVm9KqN6oJh0kKH4BmkrA4S0q1w0GW0hXaGBvBpjwNm6BQhPXXd0
U4T96KfqjYAaxiorKKSXwM6K4IyABhpPekzIAwkluQ0VwRkADDSe9IyAC2bjDSe9ZmQBhJLcsjIAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAPcwHuYRT4V22p8r/JcFPhXbanyv8lwSGeHxAAVqAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHuAe4Cmsv5er9S5Kay/l6v1LkkMcP
iAArYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAA9zAe5hFPhXbanyv8AJcFPhXbanyv8lwSGeHxAAVqAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHuAe4Cmsv5er9S5Kay/l6v1LkkMcPiAArYANZzUIuUnkEb
EavfUaLycs5cEQLu/nUbj9Fj3yIFyGhKDF8fhjrH6yFy/rBs45bBQQgT4H85MzP1PmmpiuhD6JnHHGP88f8ASJJXWyDv1d
TqW9pesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxe
xr8VyIgGzqW9pesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxe
AbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxe
XrGvxXIaxr8VyIgGzqW9pesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxesa/FchrGvxXIIAbOpb2l6xr
chrGvxXIIAbOpb2l6xr8VyIgGzqW9pesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxesa/FchrGvxXIIAbOpb2l6
ciIBs6lvaXrGvxXIaxr8VyIgGzqW9pesa/FchrGvxXIIAbOpb2l6xr8VyIgGzqW9pesa/FchrGvxXIIAbOpb2l6xr
9pesa/FchrGvxXIIAbOpb2l6xr8VyIgGzqW9pesa/FchrGvxXIIAbOpb2l6xrJ8ZcVyf6zxQEOpf0HWxesa/FchrGvxXIIAbOpb2l6xr8
VyGsa/FciIBs6lvaXrGvxXIaxr8VyIgGzqW9pesa/FchrGvxXIIAbOpb2l6xr8VyGsa/FciIBs6lvaXrGvxXIaxr8V

yIgGzqW9pesa/FchrGvxXIiAbOpb2l6xr8VyGsa/FciIBs6lvaXrGvxXIaxr8VyIgGzqW9pesa/FchrGvxXIiAbX
qW9pqxOut6i/odqeKp7KkMviisA2sZrx+3oaVenWjnCSZ0POQqSpyUoNprgWtlfKslCpsnx4lerHni3aU4A
FegAAAAAAAAD3MB7mEU+FdtqfK/yXBT4V22p8r/JcEhnh8QxKShFyk8kjJDxNtWry4ld2njG2HidBPZpP
6DWdD/dyKcE28PyLrjWdD/dyGs6H+7kU4Gz5F1xrOh/u5DWdD/dyKcDZ8i641nQ/3chrOh/u5FOBs+Rd
cazof7uRtTxGhOSjm1nxRSgbWPyLvSrdsBzoNuhBvgbvcyvdvttHrX1GjPRk22t+Rz1pQ4S5FTVbdWbfiZq
TbxT+RffZca0ocJchrShwlyKcDbn5F1xrShwlyGtKHCXIpwNnyLrjWlDhLkNaUOEuRTgbPkXXGtKH+7kNaU
P93IpwNnyLvRUqsK0FKDzRuVuEN6NRFkV7MduVdy0q1YUoOc3kkRdZ0P8AdyOeLvKlBd2ZVkYZc1q21
C41nQ/3chrOh/u5FOBtl8i641nQ/wB3Iazof7uRTgbPkXXGs6H+7kNZ0P8AdyKcDZ8i641nQ/3chrOh/u5F
OBs+RdeUr+jVkoptN7sySecg8qkWt+Z6KPur0EPRhyTf7ZItTEKNObi221wJFV5Upv4M8483JtlM2Saa0u
NZ0eEuQ1nR4S5FOCMPkXXGs6PCXIazo8JcinA2fIuuNZ0eEuQ1nR4S5FOBs+Rdcazo8JchrOjwlyKcA+Rd
cazo8JciXCcakFKLzTPOFvhTztn8JFa4strW1Kcca91St8uke17kjsUuItu7l8Eg1y3mldwm6zof7uQ1nQ/3ci
oME28nyLrjWdD/AHchrOh/u5FOBs+Rdcazof7uQ1nQ/wB3IpwNnyLrjWdD/dyGs6H+7kU4Gz5F1xrOh/
u5HehdU6+eg9q7igJeGtq8iuKYd0z2m0RK6ABXtAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAA9wD3AU1l/L1fqXJTWX8vV+pckhjh8QAFbBT3906tTQg/0L7k6/rdFQeXvS2IpSS8n5GT/
5hgAEeNgyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQAm0008mAUX
NhddPT0ZP9cfuTDz9tVdGtGa47S/i1KKa3MsPoYb8q92QAVuAAAAAAe5gPcwinwrttT5X+S4KfCu21Plf5
LgkM8PiELFOyv1RNIWKdlfqirk8JU4AOXzAA2hCVSSjBZt9wGoO/Urjy2aVaFSll0kXHPdmFmsx9w5gAOQ
ABXobf8AYh6G8vdZpQ/Yh6G8vdZ0+rHi89V/cl6s0Nqn7kvVmpy+XP2AAIA607WtUWcaby4m8rKvFZuG
a+Ad8LekcBpp5MBwAACzwjdULIrcI3VCyLH0+jh8IV2Mftw9SrLTGP24epVkl5M/nIAAxAdYW1apHSjBt
M26lceUw6429OAO/U7hf/LZxlFxk4vet4SazH2wAAjMPfj6no4+6vQ85D34+p6OPur0LD1/jftrW/Zn8rPP
Hoa37M/lZ54H5P3DAAI8gAAAO0bSvKKkqbyZylFxk4vegsxMfbAACBb4V2eXzFQW+E9nl8xYb4PNOKTE
e1y+hdlJiPa5fQS9H5HgigAjwAB1p21WrHSpwclxCxEz9OQO/Urjy2Yna14RcpU2kt4Xjb04gAOQl4b22Po
yISsN7bH0Yd4/OF2ACvqAAKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHuAe4
Cmsv5er9S5Kay/l6v1LkkMcPiAArZUYrPOtGOe5EEk4i872fwy/BGI+ZlndpAARmAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABeWE9O1ht2rYUZb4V2d/MWHo/Hn/ScACve
AAAAAAe5gPcwinwrttT5X+S4KfCu21Plf5LgkM8PiELFOyv1RNIWKdlfqirk8JU4AOXzA6W9Toaym1nl3H
MAidTtdWt4rmbioNZLM4Yvup+rOeE/vz+U6Yvup+rK9k2m2LcqwAEeIAAV6Gh+xD0N5e6zSh+xD0N5e
6zp9WPF52p+5L1ZqbVf3JerNTl8ufsOttOFOspVI6UUcgCJ1O1pVxOMdlGOfxYtsRdWqoTglnuyKsm4dbS
lUVWWyK3fFlb0yXtbs6YpSitGpHY28mVxYYpWUpKnHbltZXkcZpjnOgABks8I3VCyK3CN1Qsiw+jh8IV2L
/tw9SrLTF/24epVkl5M/nIAAxTrfEI0aMabg3kSaF/01VQhSb/wVBdWNGNOgpLfLaw9WG97TrbtWrQo0
3KfIoak9OpKXieZZ31rVrPSjLNL/tKprJtPZkEz2tM60yYADzNoe+vU9FH3V6HnYe+vU9FH3V6Fh6/xv21rfs
z+Vnnj0Nb9mfys88D8n7hgAEeQJdha9NPTkv0R+5woUZV6qhH6lzJwtLfgktnxZW+LHv8A1P05X1yqNLQ
h7z+xT795LoUnfVakpSyaOd3bdXmo555oGTlf/X6cAARgFthPZ5fMVJbYT2eXzFhv+P5p5SYj2uX0LspMR
7XL6CXo/I8UUAEeALrDllax+O0pS8s9lpT+Ur0fj+ThPE4xqSjoN5PLecq+IRq0ZQUGs1kSJ4dQlJyblm3nsZF
ucPlSi503pRXd3h3fqxEoIAI8gSsM7ZH0ZFJWGdsj6MO8fnC7ABX1AAFAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAD3APcBTWX8vV+pclNZfy9X6lySGOHxAAVso8Q7bU+n4IxJxDttT6fgj
EfLyeUgAI4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAt8K7PL5i
oLfCv2JfMWG+DzTgAV9AAAAAAA9zAe5gU+FdtqfK/wAlwU+FdtqfK/yXBIZYfEIWKdlfqiaQsU7K/VFXJ4S
pwAcvmAACJ+E/vz+U6Yvup+rOOGThCtJzkorR72dMUqQqKnoTjLLPc8yvXEx0VcACPIAAK9DQ/Yh6G8v
dZpQ/Yh6G8vdZ0+rHi87V/cl6s1Nqv7kvVmpy+XP2AHW1jCVeKqNKPfmwRG5072dk6rU57IfkkXd5GjD
oqO/dmu4kqtQ0culhl8xx6KxbzcoZ/OV7OHGuqyqG23m9rBZXNO0jQk6Tjpd2UsytI8t6TWQABws8I3VC
yK3CN1Qsiw+jh8IV2L/tw9SrLTF/24epVkl5M/nIAAxbaE9DT0Xo7szZXFWKSjOS+pY2t1b9DGm2o5LbpH
dRtX+r/T+weiuLcbiSzqTqW6dTeVl+kruWj6ljXvaVCH6WpPuSKepUdWpKct7ZXWa0cYjbUAEeVmH7kfU
9HH3V6HnIfuR9T0cfdXoWHr/G/bWt+zP5WeePQ1v2Z/Kzzwk/J+4YABHkW2GxhGhpbE29pIq06VbJVG
nl8Si0pJZKTS9Rpy8UuZXprmiK60vqNClSbdJLbvMVqFGq06iTaIOG1oxU+kqJcNJmuJVVKrHo55rL/tYa9S

vT3pDqpRqzS3JvI1D2gjxBbYT2eXzf+CpLbCezy+b/AMFhtg808pMR7XL6F2UmI9rl9BL0fkeKKACPAHZut
Tits0sthzhoqcXJZpPai4hcW9WCTlH0kGuOvL96VULmupfpqSbLyLcqSclvW05qNtF6S6NfHYR7u/hGDhS
ecn3or01jpxM2lW1ElUko7szUZ5gjw/tglYb2yPoyKSsN7ZH0Yh3j84XYAK+oAAoAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKay/l6v1LkprL+Xq/UuSQxw+IACtlHiHban0/BFJWIdt
qfT8EYj5WTykABHIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFvh
XZ5fMVBb4V2eXzFhvg804AFfQAAAAAAPcwYe5gVGFdtqfK/wAlwU+FdtqfK/yXBIZYfEIWKdlfqiaQsU7K
/VFXJ4SpwAcvmAACAACgAAAAD0ND9iHoby91mlD9mHoby91nT6seLztX9yXqzU2qfuS9WanL5c/YAAg
ACAACgAALPCN1QsitwjdMsiw+jh8IV2L/ALcPUqy0xf8Abh6lWSXkz+cgADEM5vizAIAAAAAozD9yPqejj7
q9DzkP3I+p6OPur0LD1/jftrW/Zn8rPPHoa37M/lZ54H5P3DAAI8gAAAAAAAAW2E9nl83/AIKktsJ7NL5v/
BYb4PNPKTEe1y+hdlJiPa5fQS9H5HiigAjwAAAZviwAFAAECVhvbI+jIpKwztkfRh3j84XYAK+oAAoAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKay/l6v1LkprL+Xq/UuSQxw+IACtlHi
Hban0/BGJOIdtqfT8EUj5WTylkAEcgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAW+Fdnl8xUFvhXZ5fMWG+DzTgAV9AAAAAADD3MyYe5gVGFdtqfK/yXBT4V22p8r/JcEhl
h8Qi4hTlUtWorNraSgV3aOUaeayB6GVClJ5uEW/Qx1aj5ceRNPJ8afbz4PQdWo+XHkOrUfLjyGj40+3nw
eg6tR8uPIdWo+XHkNHxp9vPg9B1aj5ceQ6tR8uPIaPjT7efMxi5tRis2y/6tR8uPI2hSpw92CXohoj8afZSjo
0orgjZ7mZBXs1208/cU5U68lJZbTkeinThP3op+qNerUfLjyJp5J/GmZ7S8+D0HVqPlx5Dq1Hy48hpPjT7ef
B6Dq1Hyo8h1aj5UeQ0fGn28+D0HVqPlR5Dq1Hyo8ho+NPt58HoOrUfKjyHVqPlx5DS/Gn2iYVTlGnKUlk
m9hYGEklklkZK9VK8Y0g4rTlOjFxWai82VB6U5O3pP/5ceRGOTDzncS8+D0HVqPlx5Dq1Hy48hpl8afbz4
PQdWo+XHkOrUfLjyGj40+3nweg6tR8uPIdWo+XHkNHxp9vPmS/6tR8uPIdWo+XHkNHxp9qKjCVStGM
Vm2z0MVlFL4GsKUKfuRS9Ebhvix8Ia1FpU5Jd6aPPThKE3GSyaZ6M0nShP3op+qKZcXUedB6Dq1Hy48h
1aj5ceRGHxp9vPg9B1aj5ceQ6tR8uPIHxp9vPg9B1aj5ceQ6tR8uPIHxp9vPg9B1aj5ceQ6tR8uPIHxp9vPlz
htOVO2/UstJ5ndW9JPNU45+h1DXFh4TuQpsTpyjcObX6XuZcmJRjJZSSa+JWmSnONPNg9B1aj5ceQ6tR
8uPImnn+NPt58HoOrUfLjyHVqPlx5DR8afbz4PQdWo+XHkOrUfLjyGj40+3nweg6tR8uPIdWo+XHkNHxp
9vPk3C6cnc6eWyK3ln1aj5ceR0jFRWUUkvgNOqfj8bbmWQAV6gAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAPcA9wFNZfy9X6lyU1l/L1fqXJIY4fEABWyjxDttT6fgjEnEO21Pp+CKR8rJ5
SyACOQAAAAAAAAADaEJTeUIuT+CNS7t6cLe1TS7s2+IaY8fOVZ1Kvv6NnKdKpT9+Dj6omvFJqXuLL1NL
q+Veioxi0+8rua49dpQgW2HW8qcZSml+rLI5XlnUnVlOOWjkROlPHkrgb06NSrNxhHNolrC6rW2Ucw4rjt
b6hBB3r2lWgs5rNcUaUKMq9TQjlnlntCcZidOYJTw+spqKyeazzNqdjUVxo/peWTYddO3pDBd3dB1aDjTS
0mVFahKhJRnlnlnsC5Mc0cwAGQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFvhXZ5fMVBb4V2eXzFhvg804AFfQ
AAAAAAw9zMmHuYFRhXbanyv8AJcFPhXbanyv8lwSGWHxAAVqAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAABrKcY5aUks9izZseJ9sK9RYnCCm1GMFkkwPbAh4ROVXCrac3nJw2smAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAA9wD3AU1l/L1fqXJTWX8vV+pckhjh8QAFbKPEO21Pp+CMScQ7bU+n4IxHysnlIACOQAAAAA
AAAAACztcQgqahW2ZbMysJ2rZSoxlGX6stqYbYuUTuqW7e0uNyjn/tIl3YdFFzpvOK3rgaww+5jJZNR+KZ
YXMujs5KbWejl6srfXOJ5Rpxw2tUqxlGbz0csjlfXVWnXlTjL9OXAYTJadRd7yM3tnVq3DnBJrIQ53a2KNIdr
VqU62dJaTfcTEsQctLd8NhnC4xXSZr9SYuutuu1Tb0e7IJSsxTaXKMqls1UWTa2lZhvbfoy0gpRtsqjzlltZWY
d236MO7+VU69uerxSiv1MgU7ys7hPNZyyT2HbFvfgQYPRqRfBhllvbmub2rOlbucHkymq1p1paVRptbN
xc3VN17Zxp7W9xUVredBpVFtfAOvyOUz/AByABHlAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAt8K7PL5ioLfCuzy
+YsN8HmnAAr6AAAAAAAGHuZkw9zAqMK7bU+V/kuCnwrttT5X+S4JDLD4gAK1AAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAABnhfbH+XXyI90zwvtj/Lr5EB6zBP4a1+QnEHBP4a1+QnAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAA9wD3AU1l/L1fqXJTWX8vV+pckhjh8QAFbKPEO21Pp+CKSsQ7bU+n4IxHysnlIACOQAAAAAAA
AAADvSvK1JJRlmuD2nABYmY+kx4lWy2ZL6EerXqVnnUln8DmA6m9p7TLUnlIJErWVxllnH1yIYC
Re1fqXSNapCppxlllI7vErhrLOOfHIiAEXtH1KTG/rqOjpJr4o40q06NTThln8TQA5Wn9ute4qXDzqSz
AB5Wn9ute4qXDzqSzTqZbOBzM

GQkzM95SKV7WpR0YtNfFHOvcVLjLpGtnBHMBZvaY1tgyAHLAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAW+Fd
nl8xUFvhXZ5fMWG+DzTgAV9AAAAAADD3MyYe5gVGFdtqfK/wAlwU+FdtqfK/yXBIZYfEABWoAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAM8L7Y/wAuvkR7pnhfbH+XXyID1mCfw1r8hOIOCfw1r8h
OAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAB7gHuAprL+Xq/UuSmsv5er9S5JDHD4gAK2UeIdtqfT8EUlYh22p9PwRiPlZ
PKQAEcgAAAAAAAAAAAAEilZ1a1PTilkFiJn6RwZknGTi96MBAAAAAAAAAAyYMgADZU5yWcYSa+CBrbUDc
9oAwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAFvhXZ5fMVBb4V2eXzFhvg804AFfQAAAAAAw9zMmHuYFRhXb
anyv8lwU+FdtqfK/yXBIZYfEABWoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAM8L7Y/y6+RH
p8fq3lHDpVbF5Tg85bM3keCvL2vfVuluJ6c8ss8sgPoWCfw1r8hOPDYFiWJ1rqha0amdKO9aOxI9yAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAD3APcBTWX8vV+pclNZfy9X6lySGOHxAAVso8Q7bU+n4IxJxDttT6fgikfKyeUsgAjk
AAAAAAAAAAusP7HEpS6w/saLD0YPJVyh0l24cZZEqeFyWWjPPP4biPT7evnLO+rSo27cd72JkWlazEzZ
F1Xs/c2+hDuKMqFTRn9Gd7O6qu5hGcnJSfeSMWinShLvzKTWtqcq/pHtrHrFLT08tuWWR0hhknm5zy4
LIkYZ2T6kK7uqrrzSk4qLySQXjStItMNq+HVKacoPSSI1Cl01ZU88sy0w+tKtRantcXlmRoQVPFNFbm8yJale
0x9S5Xdp1ZRelnmc7aj09VQzy2bybi3u0/Uj4Yv+rXow5mkRl4sXFn0NWEFLPT+BZ2tB0KOg3n8SHircZ02
nk0SbCc52qlN6T4srakVreYQrqxlTjKo5p7c8siEd69epKcoynJrPdmcCPLkmOXYB2tqKr1tBvLZnmTHhcfNf
ILXHa0bhEs406lbRq5JZcSTVwx76Mk1wZCnBxqSis3kyXa0rxNOLcY/7g6pET/mYRalCpSf64tfE5l9KcYU/9
acd20opNaTy3Zgy44prUsE6OHOVHTU821mlkQe9F9CWhaKXCOYXDSLb2gwwttfqqZP4I4XNnO3Wlnp
R4mOuVnVUtNpZ7i0r/wCpYyb745ldRWl6zr9KMAEeYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAC3wrs8vmKgt8K7PL5iw3weacACvoAA
AAAAYe5mTD3MCowrttT5X+S4KfCu21Plf5LgkMsPiAArUBjNcRmuIRkGM1xGa4g3DIMZriM1xBuGQYz
XEZriDcMgxmuIzXEDIACgBjNcUBkGM1xQzXFBNsgxmuKGa4oDIMZrihmuKBtkGM1xQzXEG2QN4CgBj
NcQMgxmuI0lxCbGlKLUlmnvR4jHsArULzpLOlKdGq80r3XwPb5riM1xBtV4BhMMMs1pLOvPbN/4LUx
muI0lxBtkGNJcRpLiZCgAzSAAxpLiNJcQm4ZBjSXEaS4jSXEaS4ZBjSXEG4ZBjSXEaS4g3DIMaS4m
QbAAFAYzXFDSXFBNsgxpLihpLigbhkGNJcUNJcUZzQNgACgAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKay/l6v1LkprL+Xq/UuSQxw+IACtlHiHban0/
BGJOIdtqfT8EUj5WTylkAEcgAAAAAAAAAAAF1h/Y4lKdYXNanHRhUaXArXFeKTuW9Pt6+cnYp2depVqclP
TT/VnnmbVLirVjozm5L4kIyRFZj22s+10/Un4r+xD1KuMnCSlF5Nbmb1K9Wqkqk3JfEpXJEUmq0wzsn1O
dfD+lqudOaSk9qfE6YZ2T6sgVLirSuKihNpaT2BvM1jHHKFlRpU7Oi85fFsrYXCd8qz3Z/Y5VK9Wr782zG
V8sTqI+oXd1QV1SilJJ70zS0slbzcnLOTWSkyc1aayhNpB3FZz0uknBnHRlNPcNDpi3AntsaYiFM0kn4
upVqVWnUm5Zm5ZbszEKkqcs4sRk5JlzYTi6lVSTW/Ti2+Ei6lVSTW/Igad3WnHRlUbXAtXkWlVpUZ/Lz
J/+5mgDmJmHe1rxoVnOacszZrVxGrPNQUYKcpvOUm38Xmf/Qv5CjOvWa2hodI//AO3/AIKQ6u5rOGh0j0cssgY8kV3/AFx7zIAZAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAW+Fdnl8xUFvhXZ5fMW
G+DzTgAV9AAAAAADD3MyYe5gVGFdtqfK/wAlwU+FdtqfK/yXBIZYfEOdebp0ZSXcjocjocbzss/Qru3jKklWq
Sk5OctvxMdLU8cuZoDl8zlLfLfanHS1PHLmaAJnZ2ANIyZDl8zlLfLfanHS1PHLmaAHKW/S1PHLmaAHKW/
mFWqJpqpLNfE0ANyv7So6R2IuHdkgSjp9Ok7rCnCtR4cnMuHHxTyI4xKCvMHkgL5mWbjTHxTzPGuyaSFMA
26Sp45cxpxKcXdJOxNkLJIPNcQk5dpKcjTlbJdzDHx2ovPNUHJS7NJHZeJLRGqRGEtLbSxJeXqVLJJbOdJ
yb71mWZT4X2n6FwdQ+hgndHC9quibFHSCE0qsjM2fJPN5qRgjZbuPacvNlJahAeRtxTZ5yb8z6rEXNOK7MRc
b9LU8cuY6Wp45czQA3Lfpanjlz6Wp45czQA3LpCvVmW45cx0tTxy5mgBuXSFepMtxy5jpanjlzNADcul
nZoehYer8aZmZdnuKO7uak68vPIMBJDAJRUeeXWXVNtUnJaIj/HLmZdDz7LXd6+wIJ6aPIjLyeYvZnhakBcg
LU8cuZzoopeMWU8l7xMM1q3RLmqeuyLwmkdGBTwdZd6IsThUVaZjdy4uUbpqMowPkOvuzD6oarlK2CmnK
eOXM1HODDM1BTcAoA8CqvAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKay/l6v1LkprL+Xq/UuSQxw+IACtlHi
Hban0/BGJOIdtqfT8EYj5WTykABHIAAAAAAAAAAAAAAAAOkK1WEGGE2l8GaNttvvNswAwA

AAAA

5p3FHo6jWllk8+80eF03LNTkohrMcoiaxt1vHCVpNOSbSzMYb2VerIV9b0qLXRy298SbhnZF6sQlZmcndX
XNac6825PY2ki0spyqWicnm9xT1v3qnzMtsP7GhDjDMzeVbBtXqSeS0+71LHE21a/peW1FY3o3blwnn9
y4qwhd0MlLY9uwLj71tEKehU0LiFSWbSZbVoUrumsp8mQKtnGncU6bn+mW9skPDYN50qjSCY4vG405
1MOnGLdOpn8CvaabT3ou6aVrQaqVNLLvZT1pKdaU1ubzDjLWK60zbxUriEXubLXEKkqVt+jZm8syohJw
mpLemXFOrSvKGjLLPvTC4fGa/tU0as4VYyjJ7+ZbX8VK0k3vW1GlPDqVKanKWkl3M44hdRlHoqbz4tB3E
TSk8nTCm5UZZtvaQr2Uut1Fm8syZhX7M/UhXvbKnqHN/wDlDgdrWjVq1P8ASeTW98DiT8LqwjKUJPJv
dmRjjiJtES2eH7c5XGTJ1OGVvouWnsyz4kW5sOmrOoppJkqhCELdRg80tmZ09tY1MxpV2Gy+y9SXidad
OEYweWlvZEse3v6nfFsv9MjGJ1jlDoNu6pNtv9SLS/qypW2lB5NvIqrftNL5kWWKdlXzIJjmenaVS5Sm/wB
TbJ8LGrUpRdSropLdwINOWhVhJ90ky6rQV1b6MJ5J7c0Ew1i25lztLfoJvKtpJrcRsVyVaD78iRaWsbarnKe
c3sSI+Lfuw9A0yR/5JVrNXFnoyebyyZGw+3cbiUpL3NiOeG1tCvoP3ZlnXqKjSnP4BaavWLT+lZiNZ1K+gnsi
QzMm5Sbe9mCPJa3KZkAAcgAAAAAAAAAAAAAAAAAFvhXZ5fMVBb4V2eXzFhvg804AFfQAAAAAAw
9zMmHuYFRhXbanyv8lwU+FdtqfK/yXBlZYfEON52Wfodjjedln6FdX8ZUAAOXzAmWFrGu5SqbYpZlhm1
OpOnLOEnF/ALWYidysJ4Vm/0VMl8USLSyVvJyctKXpkV8cRrpZZp/QxO/rzWWll6FeiL4o7xDvilZScaUXnk
82V4bzeb3gjC9uU7AAHC7w7skCURcO7JAlFfTx+MKK+7ZU9SOSL7tlT1I5Hzr+Uh0oV5W89KGWeWW0
5gJE6ncJus63CPI2o4hWnWhFqOTaT2EAzGThNSW9PMbaRlvv7XN7XnQoqUMs8+8qa9aVeenPLP4G1
W6q146M2sk+BxC5cnOe30AAMUzC+0/QuSmwvvtP0LksPf8Aj+CJifY5eqKUusS7HL1RSBh+T5T5MgAjzrCzs
oVqDlPe92XcZeFPP9NRZehCpVqlJ/6cmjusSrpZNp/Qr0Vti1q0LG0tY2sHtzb3srsRqqrcZZ3Reajs
Xw2EcGTLE141AAR5wv7Xs0PQoC/tezQ9Cw9X433Ls9x52t+9P5meie487W/en8zEuvyfqGgAI8behT6W
rGGeWZeUKEaFNRivqUVOpKlNThvR31hceNcit8V60+0+5s53Ev1Vso9yyI1TDejpyn0ueS3ZHHr9x4lyMT
va84uMpLJ/AOrXxz+nAGDJHmYLHCPfq+iK4scI9+r6r6IsNcHC0KrF/3YehalVi/7sPQsvX+R4K4yYMnL54A
ALbC+zy9SJiXan6nlRjowaS37jSrRjS/MjmdLbtNP5kVa/cPQ9wHcCv
rAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKay/l6v1LkprL+Xq/UuS
Qxw+IACtlHiHban0/BGJOIdtqfT8EUj5WTylkAEcgAAAAAAAAAAAAGVOUkpSS9TTVTABvTObzzzzzz28Q5ylslJv1
YMA2ynluMucpLJyb9WLDdyb9WagESt8Nf/S/Urq05RuKmj0tbJ9T3M5Jtbm0YDW2TdZ0T3OUm0ZDW2TdYj0zKUpOOm/VmAAylvm0fIm/VmAAylSvJv1
YMA2ynluMucpLJyb9WagESt8Nf/S/Urq05RuKmjJr9T3M5Jtbm0YDW2TdYj0zKUpPOUm/VmAAylmGpO
Puya9GYc5N56Tz9TABuWXKUnnKTfqwm4vNPJmADbd1JyWTm39TUwZBsBgAAAAAAAym1ueRgAb9L
Uyy05ZeppmmAF/wD1lTlFZKTS9TD2sADKnKPuya9GZdSct85Pr4s1AQTaeaeRmAB5LJbreRmU5PfJv+
TfqzAAzFuLTTyaCucGvhGjRDFRLM2cVbSMoclCIAoq9XDqAmuQcppTKz1JbWmuDQy5eBYAGVZnu+JcuS
ANrZtGpOPuya9GagKzm1qcoj6VqcoTozZlCaNPmZrSWjGSfJKfsTrUlNTWvhQXepjKLlj1bWJDdK8wmzr
Ufdk16dAGWqdDcLoYBYMyHqs+6CGBTy/ypJKbY0TefrBXj2wTRaQhuNWbPBTXezpDKwOOZZz6NWsZR31Y
36swYAGGMnF5pNPqQdYO0jKhE5KZIaRLKkGqcCXw2PpmTSZIqrd6QbJRwcVtKbmlPgnXstVRxeY3F8Kd0
KWjlpPl0K1tl7aiNMgAjEAAAAAAAAAAAAAAAAAAAAC3wrs8vMvKgt8K7PLzLzp0Ezs5qApcocRQCAVGzWSTSuLKK
5mTD3MCowrttT5X+S4K+S4K7mw8N5ZUu+qA3POcYv5K4Sq6cv/1ox1sUrdecgVH0FrQhtO0nh38GZYgHAl
JAlEXD+yQJR0+nj/YQrUo6AJ8gAEecAAyYAGTAAyYAG+TAAyYAGTAAyYAGX/oUi7aN2W2mfQlPdbFnxR
9UUhd4l2OXqigrDGKSIJlhwAAyMJwAAMjGwBlQMA6X/mlkjmhmkmk8m/DPKs4v2fGH8X/AkfH/AODRj
nzEuAAAAAAAALEyAAAA2AcQZl0AjZFRrJbsmDFyAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAD3APcBTWX8vV+pcl5lyZAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAD3APcBTWX8vV+pclNZfy9X6lyfgikfKyeUsAAjzgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAlW1jOvHTz8zQy2W5GGGHmg8t0E/4wLt0fTTCVb7KaDw0KCJpJpvqKAF9kXoRL2rTdbx45Mfy
VqUqNRRwmuAZ8GGlCnw3PlCwYLrP6+d7YXkYr2JcHLClWqxPq4pY5Pe6kS5ZuGFnvfLjKepXgpF6Lqsk
BmMwAAAAAAAAAAAAAAAAAAAAAABb4V2eXzFQW+Fdnl8xhvg804AFfQAAAAAAw
YV22p8r/JcFPhXbanyv8lwZWWHxEO3Cj5NwqZLgFxe1MyCtFbPCk5NwqZLgFnpaLqfmlkwqUx3jFrdyRnwq
HQp6VeqX5q5fDfk6Jk7mDjmxKTmkmpAVqkawrxkvRTpKp0QyZs3Qp8sD0QQ4SSWMZCuqnTbfJdRc
kkbAFax2RLqwhcT009GiaUhcnZdfarMbNaTQ0ohfb7ygnk1eTKTIxBMMAcp7nvSLLTLqhXVk7Rm2Ah

T0q9VPzFyM6qfmLkWYB0KelZqp+YuQ1U/MXIswNHQp6RrWzhbJtPOT3tkkArStYrGoaVacatNwks0yvl
hW39NTZ8UWYDm2Ot/tV6ql5i5DVT8xci0BHPQp6VeqX5i5DVL8xci0AOhT0q9UvzFyGqn5i5FoAdCnpV
6qfmLkNUvzFyLQA6FPSthhSUs5zzXBFjGKikluRkFd1pWv0EG4w6NWo5xlot7ycAtqxaNSq9Uy8xchqmX
mLkWgIz6FPSr1TLzFyGqZeYuRaAHQp6VeqZeYuQ1TLzFyLQA6FPSr1TLzFyGqZeYuRaAHQp6VeqZeYuR
NtraFtDRjtb3s7grquKtZ3EBwuraNxDKWxrczuA7mImNSq9VPzFyGqn5i5FoCMuhT0q9VPzFyGqpeYuRa
AHQp6VeqpeYuQ1VLzFyLQA6FPSr1VLzFyGqpeYuRaAHQp6Veqn5i5He2w+NGppyek1uJoKsYaRO4gAA
agAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAYDAprL+Xq/UuSmsv5er9S5JD
HD4gAK2UeIdtqfT8EYk4h22p9PwRiPlZPKQAEcgAAAAAAAAAAF9T22q6Nr3dhQneheVaCyi848GVtivFZ
7tqlK6g85afqnmcp1qk4qE5NpcSwp4nB7KkdE6XdCnWt5VElmlmmg7mm43SW1lTpQp/6bTb37czW6o
W85ylUktPLdmccJ31F6EfEe1v0QdzaIxROmtpRp1qrjOWS7su8nqjZRejnHP1IFpbSuJvJ6MVvZKlaWlOWj
Oq9L1QcY98d6LrD4Kk6lHNZLPLiRbGnGrcaE1msi3gou3yg9KOWxlXh3bF6MOr0iLR/UyVhQ01J7Ipbszn
Gha9aaTWSSa/V37RitSUdCKeSe8r6S/1o+oL2rW3GIXteFOpSyqPKPrkU97Tp06qVJ5rLjmWOI9kkUy3BP
yLd9aC0wr9mfzf4KstcJ/Zn83+CM/x/NCu05Xs4ra2ybRw+lThpV9r9diOUEnizz4nTFZSUIKL/S3tK1isRE3l
v1S0rJqGWfwZXXVvK3novanuZmzk43NPReWbSZYYmoujBvxIOdRkpvWpR7Ow6SCnVzUXuXEkdXs29D
9Ofqda7cbN6GzKOwpE3nmm8w6txx6jSZeWPQR04POPDgZw+3p1ozc1nkyenp2K09rcNvIi4TuqeoXhX
nH9dHZ21KTdRrJ7k2YlY0K0G6Tyfc0yJiTbumnuS2HXCpPpJxz2ZZ5Ai1Zvx0g1KcqVRwktqNSbiiSuU13xIR
HlvHG0ws8Kh+mc/oRb+endS+Gwm4X2aXzFddP/AKmp8zK2ydsUQsJWlHqeno/q0M88/gQbKWjdQeff
kWsuwf8A2/8ABTUf34eoXLEVtXSxxaC6KE0tqeRVlxifZH6lOhLnPH+0ywpUqkm6jyaay25FnXhCpScaryj
65FHR/eh6otsR7HL1QaYp/wAT2Qa1vB3MadDamuOZLjY29GGdV5vi2cMKS6WbfcjGKSfTJPdlsQSOMV
5zCRKytq0M6Wz4plbVoypVdCXEkYZJq40c9jW464okqtJrewWitq8od3YUZUlksm8tpHu6FvQofoycs+O
0mV5uFm5R2NRKRtt5t5kXLNa9ohgsMKppzlUfcthXllhMs1Uj37CssOucNcRuJqt0cJNJLbkzfDbiU3KnN5
5bVmR8Si1dt9zRvhUX00n3ZBpFp6rXE6ahXzW6SIZOxVp1ox4IgkY5YiLyurGlSpwzpvNySz2mt1b28pyn
OS08t2kcMI96r6I4Yj2yXoivRNojFE6aW1vK4qaMdiW9lh1S1opaeWfxZrhKXQzffpEK+lKV1PSe57COIitK
RbXeU2rh9KrT0qDyfdt2MrJxcJOMlk0WOEuThOLexEbEUldvLvRUyViaxeEUs7Ozo1baMpx2vvzKwurDs
UPr+SJgiJt3c42drT2TacvizS7sIKm50lk1ty4lfVk5VZSb25lzbScrKOlt/SVrWa33GlJ3ltbrq1hpbnlmV1Gn0l0
od2ZNxSpo040l37WGeP8AzE2QqV1VpSbjLe82mayuK0556cs3wZzRZ2VkopVay270n3EcUi151EpUG+q
qVTfo7Sjl7zy4k6+vdPOnSf6e9rvIAdZrRPaP0AAMAAAAAAAAAAAAAAAAAAAAAAAAAAC3wrs8vmKgt
8K7PL5iw3weacACvoAAAAAYe4yHuAp8K7bV+V/kuCktqissRmquxPZmWqu6DWfSw5khhitERqXYHHr
VDzYcx1qh5sOZWvKHYHHrVDzYcx1qh5sOYOUOwOPWqHmw5jrVDzYcwcodgcetUPNhzHWqHmw5g5
Q7A49aoebDmOtUPNhzByh2Bx61Q82HMdaoebDmDlDsDj1qh5sOY61Q82HMHKHYHHrVDzYcx1qh5s
OYOUOwOPWqHmw5jrVDzYcwcodgcetUPNhzHWqHmw5g5Q7A49aoebDmOtUPNhzByh2Bx61Q82H
MdaoebDmDlDsDj1qh5sOY61Q82HMHKHYHHrVDzYcx1qh5sOYOUOwOPWqHmw5jrVDzYcwcodgcet
UPNhzHWqHmw5g5Q7A49aoebDmOtUPNhzByh2Bx61Q82HMdaoebDmDlDsDj1qh5sOY61Q82HMH
KHYHHrVDzYcx1qh5sOYOUOwOPWqHmw5jrVDzYcwcodgcetUPNhzHWqHmw5g5Q7A49aoebDmOtU
PNhzByh2Bx61Q82HMdaoebDmDlDsDj1qh5sOY61Q82HMHKHYHHrVDzYcx1qh5sOYOUOwOPWqHm
w5jrVDzYcwcodgcetUPNhzHWqHmw5g5Q7A49aoebDmOtUPNhzByh2Bx61Q82HMdaoebDmDlDsDj1
qh5sOY61Q82HMHKHYHHrVDzYcx1qh5sOYOUOwOPWqHmw5jrVDzYcwcodgcetUPNhzHWqHmw5g5
Q7A49aoebDmOtUPNhzByh2Bx61Q82HMdaoebDmDlDsDj1qh5sOY61Q82HMHKHYHHrVDzYcx1qh5s
OYOUOwOPWqHmw5jrVDzYcwcodgcetUPNhzHWqHmw5g5Q7A49aoebDmOtUPNhzByh2Bx61Q82H
MdaoebDmDlDsDj1qh5sOY61Q82HMHKHYHHrVDzYcx1qh5sOYOUOwOPWqHmw5jrVDzYcwcodgcet
UPNhzHWqHmw5g5Q7A49aoebDmOtUPNhzByh2Bx61Q82HMdaoebDmDlDsDj1qh5sOY61Q82HMH
KHYHHrVDzYcx1qh5sOYOUOxh7jl1qh5sOZrUvbeEHJ1Yv4Jg5R7V9l/L1fqXBT4XF1b2pXyyiXBIZ4fEABW
yjxDttT6fgikrEO21Pp+CMR8rJ5SAAjkAAAAAAAAAAtqStK1CMHlml37GVJkO6X4/pbdStYvPP8A/wBjW
8uqcaLpU2pNrLZ3FXm+LMFaTmjWqxpMw+4jRqNTeSl3k2tb29eXSSks8u5lMZzCVy6rxmNp9jWhRqTp
zksm9jO1e0oVKjqOqknv2lSARljWphe050VQShJaKWS2lbh+UbvNtJZPeRAFtm3MTr6WGKyjKcNFp7O4
gJ5ST4GMwRle3K3JdqpRuqGUpLJrasyuvaNOi49FLNd6zI24ww0tl5R3juFnhc4xpT0pJfq72VgDjHfhO0m
4qunfSnB7U9hYQr0LulozyT4PuKYMruuWazK5p0ba1bmpLPi3mQL666eaUfcj9yK9u8Avl3GojULSzvITp
KnVaT3LPvOitLSMtPNemlsKdGdvELGbt/qNrG9vYaDpUtue9ozhPu1PUrCxwycIRnpyUc33sLS82yRMpN

anb3E9Go0pR+ORmLt7Om9Fr8tlZeyUrqTi81xTI72kWc2pnUd3S4rOvWc39DmAHnmdzuVphUv9Occ+/
MhX0dG7n8XmZsqvQ3Cbf6XsZIxJU56NSnOLe55MreZi2L/8SpVIdRy0ln0e7P4FZaR07mC+JwJ2GqEZur
OcV3JNhIv1LRt3xWWVCMc97KklX9ZVq/6XnGOxEUjnLblZvR/dh6otcQnGVnJKSbzWzMpzISuTjEx7dba
u7eqprd3otG7a8gnJrnkymCbW4q0ycY1rcLqKtrSDcWuebKy5uHXuFLdFPYjgYBfLy7RGoXVzODsZJSWej
uzKYAjnJfnIdrWv1espLPLvRxAcxOp3C6kra8gnJp5fHJjToWlP9LXp3spc2tzDbe8rfr/vXd0rVXWqub7zQw
CMJnc7lMw+vGjVkpPJSJte3t68uklJZ5b8ymM5sNa5dV4zG0q1uOrVpJ7YN5bCdUo21z+ttZ/B5FMZTyK
Vy6jUxuFxKtQtKWUGvRd5UVKjq1HOW9swYI5vkm3YLixnFWcE5Jb+/wCJTgGO/CdtpPOT9S3tZwVlHOS
zy4lMAUycJmVlhtJOpUqvjkiLe1ekuJbdi2I3o3ro0VTjFepEe15h1a8cYrCdhsKcpSnUyzW7NljN06kHFzWT
+JQAbWmXjGtLfqdpxX/uKutGMasox3J7DQBxe8W+o0AAMwAAAAAAAAAAAAAAAAAAAAAAAAAAAAC3
wrs8vmKgt8K7PL5iw3weacACvoAAAAAAAI9zZ0rlfrX6luaImpqfmS5E+rWp0Y51JKK+Jw1jbeZ9idmVo
pvuj6np+ZLkNT0/MlyJGsrbx/YaytvH9h2c8caPqen5kuQ1NT8yXIkaytvH9hrK28f2HY440fU1PzJchqen5k
uRI1lbeP7DWVt4/sOxxxo+p6fmS5DU9PzJciRrK28f2Gsrbx/YdjjjR9TU/MlyGpqfmS5EjWVt4/sNZW3j+w
7HHGj6np+ZLkNT0/MlyJGsrbx/YaytvH9h2OONH1PT8yXIamp+ZLkSNZW3j+w1lbeP7DsccaPqan5kuQ1
PT8yXIkaytvH9hrK28f2HY440fU1PzJchqen5kuRI1lbeP7DWVt4/sOxxxo+pqfmS5DU1PzJciRrK28f2Gsrb
x/YdjjjR9T0/MlyGp6fmS5EjWVt4/sNZW3j+w7HHGj6np+ZLkNTU/MlyJGsrbx/YaytvH9h2OONH1NT8yX
Ianp+ZLkSNZW3j+w1lbeP7DsccaPqen5kuQ1PT8yXIkaytvH9hrK28f2HY440fU1PzJchqan5kuRI1lbeP7D
WVt4/sOxxxo+p6fmS5DU9PzJciRrK28f2Gsrbx/YdjjjR9T0/MlyGpqfmS5EjWVt4/sNZW3j+w7HHGj6mp+
ZLkNT0/MlyJGsrbx/YaytvH9h2OONH1PT8yXIanp+ZLkSNZW3j+w1lbeP7DsccaPqan5kuQ1NT8yXIkaytv
H9hrK28f2HY440fU9PzJchqen5kuRI1lbeP7DWVt4/sOxxxo+p6fmS5DU1PzJciRrK28f2Gsrbx/YdjjjR9TU/
MlyGp6fmS5EjWVt4/sNZW3j+w7HHGj6np+ZLkNT0/MlyJGsrbx/YaytvH9h2OONH1NT8yXIamp+ZLkSN
ZW3j+w1lbeP7DsccaPqen5kuQ1PT8yXIkaytvH9hrK28f2HY440fU9PzJchqan5kuRI1lbeP7DWVt4/sOxxx
o+pqfmS5DU9PzJciRrK28f2Gsrbx/YdjjjR9T0/MlyGp6fmS5EjWVt4/sNZW3j+w7HHGj6mp+ZLkNTU/Mly
JGsrbx/YaytvH9h2OONH1PT8yXIanp+ZLkSNZW3j+w1lbeP7DsccaPqen5kuQ1NT8yXIkaytvH9hrK28f2H
Y440fU1PzJchqen5kuRI1lbeP7DWVt4/sOxxxo+p6fmS5DU9PzJciRrK28f2Gsrbx/YdjjjR9TU/MlyGpqfmS
5EjWVt4/sNZW3j+w7HHGj6mp+ZLkNT0/MlyJGsrbx/YaytvH9h2OONH1NT8yXIamp+ZLkSNZW3j+w1lb
eP7DsccaPqan5kuQ1NT8yXIkaytvH9hrK28f2HY440fU9PzJchqan5kuRI1lbeP7DWVt4/sOxxxo+pqfmS5
DU1PzJciRrK28f2Gsrbx/YdjjjR9TU/MlyGp6fmS5EjWVt4/sNZW3j+w7HHGj6mp+ZLkNTU/MlyJGsrbx/Ya
ytvH9h2OONH1NT8yXIamp+ZLkSNZW3j+w1lbeP7DsccaPqen5kuQ1NT8yXIkaytvH9hrK28f2HY440fU1
PzJchqan5kuRI1lbeP7DWVt4/sOxxxo+pqfmS5DU9PzJciRrK28f2Gsrbx/YdjjjR9TU/MlyGpqfmS5EjWVt4
/sNZW3j+w7HHGj6mp+ZLkNTU/MlyJGsrbx/YaytvH9h2OONH1PT8yXIamp+ZLkSNZW3j+w1lbeP7Dscc
aPqan5kuQ1NT8yXIkaytvH9hrK28f2HY440fU9PzJcjMcHpKWcpya4HfWVt4/sZWIWzeXSJeo7Lxxu9KlCj
BQhHJI3MRkpRTi00+9GStYAAFUeIdtqfT8EUlYh22p9PwRiPlZPKQAEcgAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAzAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAW+Fdnl8xUFvhXZ5fMWG+DzTgAV9AAAAAABuQMPc
wKTQniN/KMpZQj9kTNUW/GfMj4V26r6P8AJcHMQ8+OsWjcq/U9vxnzGp7fjPmWALpp06+lfqe34z5jV
FvxnzLADR06+lfqi34z5jU9vxnzLADR06+lfqe34z5jU9vxnzLADR06+lfqi34z5jVFvxnzLADR06+lfqe34z5jU
9vxnzLADR06+lfqe34z5jVFvxnzLAxKUYRcpyUYre2xo6dfSBqi34z5jU9vxnzJtOtSq59HUhPLfoyTNxo6dfS
v1Pb8Z8xqe34z5lgBo6dfSv1Rb8Z8xqi34z5lgBpenX0r9T2/GfMant+M+ZYAaTp19K/U9vxnzGqLfjPmWA
Gjp19K/VFvxnzGp7fjPmWAGjp19K/U9vxnzGp7fjPmWAGjp19K/VFvxnzGqLfjPmWAGjp19K/U9vxnzGp
7fjPmWAGjp19K/U9vxnzGqLfjPmWAGjp19K/VFvxnzGp7fjPmWAGjp19K/U9vxnzGp7fjPmWAGjp19K/
VFvxnzGqLfjPmWAGjp19K/U9vxnzGp7fjPmWAGjp19K/U9vxnzGqLfjPmWAGjp19K/VFvxnzGp7fjPmW
AGjp19K/U9vxnzGp7fjPmWAGjp19K/VFvxnzGqLfjPmWAGjp19K/U9vxnzGp7fjPmWAGjp19K/U9vxnz
GqLfjPmWAGjp19K/VFvxnzGp7fjPmWAGjp19K/U9vxnzGp7fjPmWAGjp19K/VFvxnzGqLfjPmWAGjp19
K/U9vxnzGp7fjPmWAGjp19K/U9vxnzGqLfjPmWAGjp19K/VFvxnzGp7fjPmWAGjp19K/U9vxnzGqLfjPm
WAGjp19K/VFvxnzGqLfjPmWAGl6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi3
4z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi3
4z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi3
4z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi3
4z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi34z5lgBo6dfSv1Rb8Z8xqi3

4z5lgBo6dfSBqi34z5ms8HouD0JSUu5t5liHuGk6dfSpwmrOFadvN7tpbFNZfy9X6lyIc4fEABWyjxDttT6fgj
EnEO21Pp+CKR8rJ5SyACOQAAAAAAAAAAAAAAAAAAAAOtvRdeqoLd3sLEbnUOQLeULWzilKKzfwzY
6G2u4NwSzXet5W3Rn633VAO0f8ApbnRqRUstjzLC5tade306MUmlmsu8jmuObROvtUgtba0hQpOpXS
z+PcQZ/8AVXGjTgkm8klwBbHMRH9cAW8ba3taWdTJ8WzMVaXCcYqP4yDvoz+57qy0indU09qbJ2KU4
QoxcYpfq7kR6dONPEYRhJSipbMi0uKdOcU6uWjHbtK7x03SYUALqm7WvnCCi8u7Ir7+3VConH3ZfYMrY
prG9ooJ9r1WNHTqpaSe3Mk06tpXejFRb9CLXFuPtTgmYhaxoSU6eyL7iRh1GnO3znBN596DmMUzbiqw
XDjZ28spaOl8TNa0pVqTlBJPLNNB30J/UqYHWhRlVrqnuee18C06K2tKa01H1e9hxTHNu8qYFvCFpczTgl
mu7I44nSp04QcYpegdTh1XltXAsLGyU4dJV3dyO3T2aloaMeG4EYtxuZ0qQWN7ZwjTdWksktrSK4M7V
ms6kAAcAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Bb4V2eXzFQW+Fdnl8xYb4PNOABX0AAAAAAMPczJh7mBUYT26r6P8lwU+E9tq+j/JcEhlh8VfjeIvDMPl
Wik5t6MU+J5W1oY1jUZV4V5KGeWblkj1GPYbLE8PdKDSnF6Uc+88rb3GMYHnSVGSp57nHSRWqfZW
WO2V/RhOpKVKUv1PPSSXxPVVa1OhDTqzjCK3uTyPOYf7WwrVY0r2l0Tby00819Swx3D7e/tqbr11RjC
Wen8AO2vMN09HrlLP5idTqQqwU6clKL3Nd54u7w7A42s+gvv9WK2Z57Wd/Y26qdYq2rk3DR0kuAHpa
mJWdK46Cpc041c0tBy27TnUxjD6VXo53dJSW9aW48b7RRlU9oakIe9JxS9SwvfZWFvhk7hV5SqwjpSzW
xgeqnd29Ogq060FTf8A3N7Djb4tY3NTo6NzTlN7kntZ4nBbCri9dW860o0aacmuHodMdwdYPVpTo1ZSj
Pc9zTA93VrU6FKVWrNQhFZuTexEZYrYSpSqq7pOEXk5aWxFbcXE7r2NnWqPOcqO1/HM85gGEa1qzjU
qOFKG15d7A9rb4rY3U9Cjc05y4J7TnjrywW6y8P8AlHmMdwKOE0oXVrUlo6WTT7mWdO+nf+yFadV51
Ix0ZPjk0Bw9iW31nN8D01e5o20NOvVhTjxk8jy/sVJRjdSe5JMqqlzHGMYcr6v0NBN7+5cEB7ahi1jcT0KV
1TlLgpbyYeDxWzwilbdLht3nVi1+ltvM9F7L39S9w3KtLSqUno58UBZ3sKlSzqwovKo45ReeW08fcYfj1tbzr
VazUIJyeVTuPbkHHP4W8/4pfgDxmHvF8SnKNtcSbis3nLI9NgFniNrOq8Sbis3nLI9NgFniNrOq8QnpJr9P6syp9ie0XHyo9Djl88P
wurWh7/ux9XsA7XOI2dpLRuLinB8G9pta31teLO3rQqZb9F55Hh8KoWF5KpXxW70ZN7I7c38TW9dvhl/
SrYVdOpFbdmez4MD6BUqQpQc6klGK3tsg68w3S0euUv/AHHnfaq6rV7Szms1SqQ0nlu0jnZWmAV7aC
q3E4Vmtuk3vA9dO+to2/TdPT0O6Wexnm8J9o69fFHSu6tOFbt3ehYLAbWpg6taddygnp9Iu88phGHw
v8AFXazm4xWe1fAD6JRrU69NTpTpTjOL74vNG5Fw2xhh1nG2pycoxbxz2jrdZdp
h7yyei5ra2/gRY4Pj1WPTac03tylUyfI5eytKFbHM6m1wi5Rz45o94B5jALzFuvStKcI+Ksyc/uL22tF/
wBRXhT+Z5Eg87jGEYfVv3c3l30KktsOIFrRxewuKuSihSuqcpPctLeTT59i9phlCnGph130ks6nGph130ks8nDbn6l/Y3txX9k
atWDbrQTin37MgLavi1jbzcKtzTjTb7bELS7ScW3sa3HpcGwvDKV67m
xuelyjkoZ7viBeVatOjBzqzUIre2yJDGcOqT0I3dJvcv1bzyWO30r/GXbVKvR29Keh8FlvZ1u8PwNWcna33+
vFZrPPKTA9qmms080ZPMex2IVK1OpaVZOXRrODfcj04EaWI2kbnqzuIKt4M9pJPL1vZivUxl3KrQVFz0t7
0keoApPaO6xGhRjGwpVl71SBRRxewuLpUnjTySpa+kS+TbQU3o5ZTSg1OTV9qxkZ20+5isVqSytYKmgzvpaYz
uq0rS6k5ySzjJ7/Q9OeT9kMMqxqyvq0XFNZQz7/iesAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKay/l6v1Lkp7L+Xq/UuCQxw+IACtlHiHban0/
BGJOIdtqfT8EUj5WTylkAEcgAAAAAAAAAAAAAAAAAAAAAAWGE5ac+ORXnShWlQqKcfquId0txtErO7dr
Gp/rxbfHac6V1Z0pN0003v2M6K6tbiGVXJPhIxKpZUYvQUXmtyK9c68omEG9rQr19KnuyJ2GKoqTcn+ju
zK+jGnOv8ArahDPPaybbd3kI0ejt5J5rLNdwZY51M3mXTE41HRTh7q94h4a11pejJJFnewdFwuJJNbM33kO
ro0bjToTUlnmsgt5iZjJCVi2a0H/2lfBSb/Qnn8C2heW9emo1sk+9Myqtnbxbho7eALUi88olW2fa6fqWGK
vK3WXfIhwrxnexqNKEczviNelVopU5qTTz2BKzEUtG0axeV3DImYsv9KHqQbSUYXMZSaSXeSsSr0qtOCp
zUmn3ESkx0piW1tY0lQVWvt2Z5cDajOy6aKpRelnse0W15SlQVOq8mlk8+8J2VvNSi85fgrWOOo46bYr2
aPzIzhnZfqcsRr0qtuowmpPSzyRnD69Knb6M5qLz3MG46u9oN1tuZZt8S0w552a9WVVeSlWm0802T7C4
o07ZRnUinm9jDLFMReZlpYYySvqsXeYCenGWTcct/AiOo43LnTffmmWEMRg45VotFiHv0eLp4oaTrMxcKzKD
6cncqSz0VvZJxb3Iep0pXcKtKtDvUVxTqUujl1SOa/TobcvQgueHrY4P7mtlfKjHo6ubj3P
gSZdRm9KThmFi0XiNaazvbfoHTjnllkthVE2+rW84KFJbV3ohEefLMzP2AAMgAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAt8K7PL5ioLfCuzy+YsN8HmnAAr6AA
AAAAGHuZkw9zAqMJ7bV9H+S4KfCe21fR/kuCQyw+KtxzE3hdmq0KenJySye4rqXtdZVKuj4XNcv5KeGeVelODy2rLNH
oZ04VIuM4XUX3NFfUwHDKknJ2sE+uK1eNxGrDFsWi7C3cVLJJb3xLD2thWpK0pyPvcyi
1CNNve0jtXt6VzTdOvTjOL7mgPHKtgEcMShRc7h7RcDwhwyyyeeka+xv8ALz7v9N/lHqKODYfbz06VnSUjJT05V
2ztKrqW9vCnrJtAeOxn/1Wv+SH+D1+J/xNxn5bM1cLsq09S2hKXFJMdzD7zT1Y2160a9a0XXcu+9bBXhn/WX
h7E9suPk/yd/bf9m1+ZnobXD7Szk5W9vCm3vcVkaXD7TiN1Y2160x00aXcBRx/9DS/4f8lH7P4P4p+wXR

DlSqb8u5nq8ZoU6Hs7c0qEFCEaeSiu7aUnslaUrmjcwuaKnFtbJIDhj+PQxSjC2tYS0dLNtra2WVKznY+x9a
FRZVJx0mn3ZtFxb4RYWtTTo2tOMuORKq0oV6UqdWKlCWxxfeB5X2Kipxu4vc0kyoq2ywjFnG+odLSzeS
ezSXFHvLWxtbLS6tRjS0t+j3m1xa0LqGhcUo1I8JIDyNbEcAjSzpWLnPw5tZHpsIp2qsoVbOj0UKiza2mtPB
MNpz0o2lNNbVsJ6Sikksku4DJBxz+FvP8Ail+CcaVaUK1KVOrFShJZSi+9AeR9ie0XHyo9BjtjK/wurRh7+yU
fptyJFrh9pZOTtqEKTlvce8kgfPMLr2VpUnRxO0csnv25osoXuBVrylRpWDlGbyctuz6HprnDLO7edxbwm+
LQtsLsrSWlQt4QlxSArsZv7HDYULSvbKrSlH3fCjz+IVMAqWsnZwqQr9y25Z/U9tcWlC6jo3FKNRf7kRI4Dh
kJaStKeYFX7HxruwrKel0bf6M/8FFh1ysIx2dS5g0ouUZLLdmfQIQjTgowioxW5IjXWF2V5PTuLeE5cWtoD
DsRo4lQdWhnop5NMlSSlFxazTOVraULOn0dtSjTjvyR2A8De21z7P4r09FPo9LOMstjXAuYe2Ns6Wc6E1
PLalxPRVaNOtBwqwU4vuaK9+z+GOWl1WHoBX4P7QXOJYlKl1f/AEXua/7fUosQlFe0tR4lpOkqrzW33e4
93QtqNtDQoU4048EjldYdaXu25oQqNbm0B43G62EStoww2k9PPNyyexfUt/Z25hZ+zNSvVWcITk2uJbx
wfD40XSVrT0JbWst53oWVvb0HQpUoxpt5uKWwDykr72du0517aVGo9+jn/ghYJFv2ipuwU+hUu/w/E9
dUwLDak9KVpTzfAlW1nb2kdG3oxpr4IDxWPWU7DGZXFSlp0Ks9Pbuee9EjWHs/0Ol1KWnl7ub/ACexq
0adem4VYKcXvTRB1FhmnpdUp5+gHD2ejZVrbrVpbdDKX6Zby4NacIUoKEIqMVuSNgPMVfairSxd2qoRd
JT0W9ukenIssNs5XXWZW8HW36WW0lAVHtDjDwu3iqcc61TNRb3I8baXNCV91jEtOttzyT3v4n0G6sLW
90etUIVdHdpdxG1Fhf8A/CpcgIeH+0tpdXFO1pUpxctkeCL0hUcHw+3qxq0bSnCcd0ktxNAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKey/l6v
1Lgp7L+Xq/UuCQxw+IACtlHiHban0/BGJOIdtqfT8EYj5WTykABHIAAAAAAAAAAAAAAAAAAAAAAAAA
BkwZAwDIAwN+8yABgyAMAAAAAAAAAAADejKMKsZTWaT3Fn1iyrLOajn8Y7SpAaUyTXst1c2luv9PLPglv
K26uJXFXSexLcjkYBfLNo0GQAzAAQAAUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAC3wrs8vmKgt8K7PL5iw3weacACvoAAAAAAYe5mTD3MCownttX0f5Lgp
8J7bV9H+S4JDLD4gAK1AAAAAAAAAAAABiUVKLjJJp70zWFOFPPQhGOfBZG4AAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKey/l6v1Lgp7L+Xq/UuCQxw+IACtlHiHban0/BFJWIdt
qfT8EYj5WTykABHIAAAAAAAAAAAAAAAAAAAAAAAADKjJ7k2Y3PJhdBkbxk1vCdwBpremGmt6YXQBk
8s8tgW3cE1IDLi1vTRjeAAya3oADKjJrNRb+hjJ8C5sEnZwbQaY8fOdSp9GS3xa9UdJW1aMNOVNqPEkXd
5CskowcXGWZvVxCMrdxVNptZbdwdcKRuNoCjJrNRbXoYaa3pr1RY2V1TUYUdB57szbFUlRhkv+4L0q8e
USqwAGAAbU6c6stGCzaWYXW2oMyhKDylFp/EwEAAQAZSzeSDi1vTRV0wAZyeWeQRgGUm9yCTe5A
YBlJvcjAXQDbQllnovkag0Azk+BgIGUnLYlmEm9yJmGL/qXmv8AtDqteU6RNCfhfI1Lq6uoW8lFwzzWZTe
820u8OslIrOolgGcnnlltDi1vTQZsAAACZYKg9PptHuyzJ/VreVNyjBNZbA2rim0bhSAzL3n6mAxbQpzqS0Y
RcmZqUqlJpTi4t7jpZ1+r1dLRck951u7xValNqDUY7doaRWs13vui6E/BLkYaaeTWRd21zC5TcY5ZcSsv9l3
MOr4orXlEowADEAAAAAAAAAAAAAAAAAAAAAAAAAAAt8K7PL5ioLfCuzy+YsN8HmnAAr6AAAAAAGH
uZkw9zAqMJ7bV9H+S4KfCe21fR/kuCQyw+IACtQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAA9wD3AU9l/L1fqXBT2X8vV+pcEhjh8QAFbKPEO21Pp+CKSsQ7bU+n
4IxHysnlIACOQAAAAAAAAAAAAAAAAAAAAOtvKEK0XUSce/M5Emxt1cVXpe7Ha1xDqkTNoiEx4hQi8
oxbS70jtKnRu6Gkktu55bUcbirbW0uj6BN+iJFtUjUoaUIaC4ZFe6veZrKpto6N7CL7nkW9aNKKU6iWUdq
Kul/Jf/AFsl4q30MPUMseq0mUe7r069al0e6L27CyqU6bp/rS0VtZRU/wByPqi4v3lZSy+AMdtxaZRL25o1
KShS3p8Deld29KjHKH6stqSK2K0pJcS3dC3tKOlOGllvzWeYc0m1pmzehc0bluGjk+DIN3SVrdRlDYntJVrc
0qtbRhRUXxyRyxf3qXow7v3ptIuKcbizbglnlmiqoU3VrRh8dpYYYXW0oOm+7d6HahaqjcVKnc93wBNIyat
DliM40qCpxSTl+DtYr/ooFXe1eluZNblsRaWPYofUFLRbJKme2r9S4uYpWL2f9pTv976lzc9hl8ocYo7WVdl
2un6k7Fv2afzEKy7VT9Sbi37NP5h+in/KyqCWe7aDta1Y0K2nJNrLciPPH33ctGXhfI6W9aVvV04pN5ZZM
n6zpeXL7EOjTV3dyTbinmw1msRMcJTI3lvX/AE1oZeoqYfTnHToyyz3GdCztVnLKUvjtOVbE3lo0YJL4lazN
df8AogNZNrgYMt5tt72EtJpcdhHk/fZY0bu3pUI/o/V3pIkULijctx0tm5nOQVLiclvzCQzYAqtaGnGoso
06Oi+OSK9tdxMRKHiFvGhVTgVTgsoyLG0hVoBNJ5xI2LbofUk2PrWwBJJTDaPS0cVpP9yJUIjBfoqTPNhcYpzBuQEbw
P9TKqbznLPiWuF9kfzMiY7csjarWtbOOX6PSWizRoilzN8rBRsAqrTQV5q2oBZXqmJzMhDzqnPAT2Gp25a5SluF
H6dliNByy0XlxyF9bwnburBJSzz0m08vdRLrPOzm0sk4PZ9CtY/1ExKNhk41KTpySzS0IpTb4tGsqnWuWYT7sMf/dJL5
S6G4kktj2oWlXobiMu7cyzurbrDptdz2+gZRHUpr9w0sqSoWjnPLN7WR8Pn0t7Ob70dsSqnRjJHP9mS+V3CJ/S4e
F9pfyh1bUXrWP02xZf6sPQ64XFOlLPicsW/dh6HbCv2ZeoIj/wBpdKte3tajTX6ntSN8qqnRqxqMgK7vXvfjusKnvX+xV3va6

nqTsK7PL5gtb8rzX9KycHCrKnvaeRjRn4WSalRUsRlUazSk9iJOs6Xly+xGMUrMzuVY9jyLu27FH5SnrTVWr
KaWSbLi27FH5Su8HlKlkm5yyWe0xoy8MuR3t6yoXEpyTa2rYTNaUvLl9gzrWs/co2GrO7ya7mdMWSU6
eSy2M1sJaV85LvzZvi/vw9GGkRHSlthO6ZGxDtkyThO6ZGxDtkwW/4wjAAjzAAAAAAAAAAAAAAAAAAAA
AAAAAAAAFvhXZ5fMVBb4V2eXzFhvg804AFfQAAAAAAw9zMmHuYFRhPbavo/wAlwU+E9tq+j/JcEhlh8
QAxKSjFye5bStWQVM8Uqab0IrR7szXWlbwxDDr0XAKfWlbwxGtK3hiD5FFwCn1pW8MRrSt4Yg+RRcAp
9aVvDEa0reGIPkUXAKfWlbwxMrFKue2MQfIotwc7esq9JTWzM6BtE7jcAIN7fOhNQgk5d+ZF1nW4RDK
2alZ1K4BT6zrcIjWdbhEifIouAU+s63CI1nW4RKfIouAU+s63CI1nW4RIfIouAU+s63CI1nW4RB8ii4BCsr11
5OE0lL4E0rWtotG4AaVaipU3N7kirlilXP8ATFZBzfJWn2twU+tK3hQ1pW8KDj5FFwCn1pW8KGtK3hQPk
UXAKfWlbwoa0reFA+RRcAp9aVvChrSt4UD5FFwCojilXNaUVkWtOaqU1OO5oO6ZK3+mwIF5fujU6OCT
a3tkfWdbgg5nNSJ1K3BT6zrcENZ1uCCfIouAU+s63BDWdbggnyKLgFPrOtwQ1nW4IHyKLgFPrOtwQ1n
W4IHyKLgFRrOtwQ1nW4IL8ii3Bxtq6uKKmtj70aXl0raCaWcnuDWbRFeSSCn1nW8MRrOtwiGXyKLgFPr
OtwiNZ1uEQnyKLgFPrOtwiNZ1uEQfIouAU+s63CI1nW4RB8ii4BT6zrcIna2xGU6qhUSylszQWM9JnSyAO
F3XdCg5rf3BtMxEbl3BRO9uG89N/Qx12v5jJt5/k1XwKHrtfzGOu1/MY2fJr6XwKHrtfzGOu1/MY2fJr6Xw
KHrtfzGOu1/MY2fJr6XwKHrtfzGbU7+tGablpLgNkfkVXgNYS04KXFZnO6q9DQlPvW4r0TaIjbsCjd7Xbz6R
mOu1/MZNvP8AJqvQUXXa/mMddr+Yxs+TVegouu1/MY67X8xjZ8mq9BRddr+Yx12v5jGz5NV6Ci67X8x
mYX9eE03LSXemNnyKrwGtOWnTjJd6zOd3W6ChKa2vuK9EzERt2BRO9uG8+kY67ceYybef5FV6Ci67ceY
x12v5jGz5NfS9BRdduPMY67ceYxs+TX0vQUXXbjzGOu3HmMbPk19L0FF1248xjrtx5jGz5NfS9BRdduPM
ZZWFzK4ptT96O8u3dM1bTpLD3APcGynsv5er9S4Key/l6v1LgkMcPiAArZR4h22p9PwRiTiHban0/BFI+V
k8pZABHIAAAAAAAAAAAAAAAAAAAAAHe1uHb1NLLNPY0cAFiZidwtZ3dpVylUi218BTxGhotNOOT2JIq
gVt17O9OrCN50rz0dJvcd7+7p3FOKp57HtzRBBHHUnUx7Zg8pxb3Jlhd3lGtbOnBvSeXcVwCVvNYmIZTy
aa7izp4hRqU9GvHJ9+wqwFpeafSzheWtKaVOGSe9pHDELincSp9HnszzzRDNqcoxqRlJZpPcFnLaY4y7W
kpQuYaK78mWl5W6G3bXvPYjir+336Lz9CDd3TuKie6K3IrXlGOkxE7R95d2PYofUpCytb6lRt4wkpZrgg4
w2iLd1fLZVb+JYVr6jO1dNN6TWW45XtzRr0kqaaaee4hESbcJmI/bvZZ9bp+pNxb9mn8xztr2jSoxjJPSS4G
l9d07inFQzzTz2h3E1rjmNoQADzBmMnF5xbRgAM33sAADKMAC0p39GdPQrLJ+hiN7bUZpUoZJ73kVg
DbrWTL+5p3Gh0bby4o7UL6jC1VOTlpKOW4rQE6tuU2Hk22WFld0qFvoTbzzb2IrwHNbzWdw3rSU605L
c22iTZXioJwms4vbmiGAReYncLSd1Zt6ehpT37hUv6VS2lHapNNZZFWA761mcnv7i7spuVrGUu5ZEK0u6
FOiqc4vPvfE2ucQUqehRTWezNhpjmtI5bRLyr01xJ9y2I74V2l/KQiRZV4W9VynnlllsDKtt33KRiq/1YehrY3
VOhTkpt5t9yJEsRt5LbGT+hVPeyu72it+VZdLipGrXnOO5slWN3SoUnGo3m3nsRABGVbzFuTpXmqlec47
m9hzADmZ3Ows6F9RhbRhJvSyy3FYA6reaT2Zbzk2u9mAA4SLKtGhcKc88ssthvf3FO4lF088lvzRENqbUa
ib3Jh3F548VjhO6ZFxDtkyasRt1ujJfQr7qrGtcSnHPJ8Stck14RWJcQAR5wAAAAAAAAAAAAAAAAAAAAA
AAAAAC3wrs8vmKgt8K7PL5iw3weacACvoAAAAAYe5mTD3MCowntt X0f5Lgp8J7bV9H+S4JDLD4hyu
P2Knys6nK4/YqfKyu7fUqAwZBy+UwDIAwDIAwDIAwAwBc4Z2X6kwh4Z2X6kwr6mPwhR4i/wDrJ/T8EY
k4j22f0/BGI+dk8pAARmAAoAAgAACVhvbI+jLspMN7ZH0ZdnUfT6H43gjYh2Of0KMvMQ7HP6FGJY/k+
QACPMAAAAAAAAF7Y9kp+hRF7Y9kp+hYen8bylVX3a5+pHJF92ufqRyMb+UgADgAAAAAAAMmABb4
V2d+pyxffTOuFdnfqcsX3wK9s/8VaAZI8TAyAAZDIyAMZDIGQMG9L9+HqjQ3o/vQ9UFr9vQrciHinZvqTV
uRCxXs31On0snhKnABy+YAAAAAAAAAAEPQ0P2YehwxLskjvQ/Yh6HDEuySOn0rf8ANSgA5fNAAAAAA
AAAAB6C27PD0OOJ9kfqdrbs8PQ4Yp2R+pX0r/8ANTGTAI+aAAAAAAAAAAWWEb6v0/yVpZYRvq/T/I
htg/6Qsw9wD3HT6Knsv5er9S4Key/l6v1LgkMcPiAArZR4h22p9PwRiTiHban0/BFI+Vk8pZABHIAAAAAA
AAAAAAAAAAAAANoQlUmoxWbZ1r2lS3ipTya+AdRWZjbgAA5AAAAAAAAAAAAAAAAAAAAAyBgAA
AZUXJpRWbfcbVKNSmk5xaz4hdS0AAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAC3wrs8vmKgt8K7PL5iw3weacACvoAAAAAYe5mTD3MCowntt X
0f5Lgp8J7bV9H+S4JDLD4hyuP2Knys6nK4/YqfKyu7fUqAAwcvlN6UHUqxgnlpPLMm6rl5i5EGE3CalHen
mWNjd1a1fRm1lo5lbYopPayNc2btoKTkpZvLcdaWGyqU1LTSzWe47Yt+zD5iJRvq0dGCa0d24O5rSt9S6
zw2UIOXSJ5LPcQS/rfsT+Vnnw5zUrSY0MAEYLnDOy/UmEPDOy/UmFfUx+EKPEe2z+n4IxJxHts/p+CMR8
7J5S70LWrXecVlHiyXHCs1tqbfgcKN/OlS0NFN9z4HN3ldvPpGGkTjiO/d0uLCpRjpL9SW/IjQhKpJRgtJ8EX
NnVlXttKpv3epWRrO1upuCTWe4LelY1aPqXeGGTaznNR+BmeFyUc4TTfxONa/q1Jfok4rgjtYXVWddQnJ
yT4hY6Uzx0gzhKnJxksmjUscWjHSpyW95lcGOSvG2krDe2R9GXZSYb2yPoy7LH09n43gjX/Y5/Qoy8v+xz
+hRhj+T5OtCi69VQTyz7yXqqXmLkQqVSVKenDeSVf3DeSab9COKTTX+odNVS81cjhdWjtoqTmpJvItqLq

Ro6VeSz7/gVV9ddYnox9yL2FaZKUrXbpSw6VWnGaqJJrPI31VPzYkeF9WpwUItJL4Eyyq3FxLSm0oL4bwl
Yx27acnhU/MjyK9rJtFze3SoQ0Yv9b+xTBxmilZ1UL2x7JT9CiL2x7JT9BDv8bylVX3a5+pHJF92ufqRyMb+
UgBNw616WfSTX6F9wlKzadQUsNnOmpyko59zRFqwVOo4qWll3otL+66KHRwf6n9ioDTLFa9oACbYWv
Sz05L9K+4cUpNp1BSw6dSmpOSjn3EWrBU6jipaWXei0v7roafRw9+X2RUMO8ta17R9gADFb4V2d+pyxf
fA64V2d+pyxffAr2z/xVp0o0nVqxgnlpd5zJWHrO7j8MyPJWOUxDtqqXmLkNVS81ciTf150IRdPLNkF4jcc
Y8ivTaMVZ1MONxRdCroZ5nehh8q1NTU0k/gRqtWVaenPeXNl+mzh6BnipW9p9Ieqp+YuRzr4fKjSc3NN
L4CeIV1OSTWWfA51L2tVg4SayfwItulHaIRzej+9D1Rob0f3oeqDCv29EtyIWK9mXqTVuRCxXsy9Tp9LJ4S
pwk20lvYJ+GUVOcqklno7jl8+leU6ZoYY5JOrJr4I7vDaGXevqL+5lRShT95rPPgV7ncL9Tc0V6LdOnbW295
axt8tGeefczhRpSrT0YLNirVnVlnN5vcW2H0VSoKWX6pbWGdaRkt2+nKlhcFHOpJt/A3lhtF+62vqR7u6qz
qyhSzUU8tneR1Wuab96S9Q7m2OvbTSvTVKrKClpZd5zMtttt72YI8377PQ0P2IehwxLskjvQ/Yh6HDEuyS
On0bf81KTLK0jcRk5trLgQyww2tTpQlpzUc33nLxYoibd3bVlLxSGrKXikRbu6m7iXQ1XobMsiVhzqzjKdSb
aexJleivTtbjEON1Y06NFzjKWfxOkMNpyhFuUtqzNMVre7SXqzv1qirZKNVaWh/gJrHylrqul4pDVdLxyK9X
Nw5JRqSzbLqmnTorTlm0s22Fp07/AFCmvKCt6+hFtrLPacDrc1emrylyOQeS+uU6egtuzw9DhinZH6ne27
PD0OGKdkfqH0L/APNTADvI+al2tjKutOT0YExYbQS2uT+p3tX/ANLTy8KKi4dd15aWlnmV7JrXHWJ1tKrY
ZlBypSbfBlc002nsaLyzc3bR6XeVd3oddk17uazI4yUjUTH7dbXD5VY6dRuMeBKeHUN2bz9STt6H/T4bCk
br9Lm9LSzK7tWuOI7bSrjDnTjpUm5Jb0yBuPQUtJ0Yupvy2lDVa6WWjuzDPNSK6mGpZYRvq/T/ACVpZY
Rvq/T/ACSPtzg/6Qsw9wD3HT6Knsv5er9S4Key/l6v1LgkMcPiAArZR4h22p9PwRiTiHban0/BGI+Vk8pAA
RyAAAAAAAAAAAAbqjUks1CTXHI0LuxS6nDPgVrjpznSojQqzWcYSa9DWdOUHlOLT+JZVMShTm4Qhml3
ndxp3lvmlv3fAO+lWe1Z7qWMJTeUU2/gbOlUTScJZvuyJNhFwvXF70mixuKtOgtOS27kEpiia7mVRS6Sh
Xi9B6fcuJ3vLmtVpqM6ThHPe+8x06uMQpTUcsthJxXs8Pm/wFiP821PZWxpVJrOMJNfBCNGpN/phJ/Qt
cMX/AE31Nat/ToVHCMM8t+RDpVisTMqudOcPfi16o2oU3UqxWi3HPbkXP+nd22eSakuRX2VdUKrpuO
ek8s8yk4oi0d+0ut5Z06dHSpQeln6lc04+8mvUvbmsqFPTaz27inuq6uKumouOzcRc9axPZxMxjKTyim38
EYJlndRoQkpQ0nvWQYViJnu4dWrZZ9HLkc2nF5NNP4lnTxJymlOnkn3m2JUYyo9Il+qJW04qzXdZVcKc5
56EW8uBlUajk4qEm18Cwwn3anqd7m7pW0stHOT2tIFcUTXlMqhU5yk4qLbXcHSqKWi4S0uGRNw+fSX
dSWWSe3ImXNelbPSks5PYsgVxRNeUypZwlCWjJNPgbRoVZ+7TlyOs7pTvFWcf096ZKeJ7coUtiDmtad9y
rp050/fi16oQpzn7kW/QuoSheW70o79mTK+zn0F44Pc3ohbYoiY79pRJRcHlJNP4mYxc3lFNv4FhilHOM
aq9GYwqj71WS+CCdKefFBnTlT9+LXqaki+q9LcPLdHYiORnaIidQnYXS0qkqjWyO40xOq519BbokvDIpW2
fFlc/9W7elucg3t2xxEftzhSnU9yDf0NpUKsV+qnJfQuKrdvRXRU9LLuRF1jJJqpSyKk4q17TKujCU3lFNvgjb
oamlo6Es+GRJw5uV421vTLC6uIWyUms5PYhpK46zXlMqeVvVis3Tll6HMt7a+jcT0JR0W93xIuI26pSVSG
xS2ZELY448qyiQpzn7kW/RG0rerFZunLL0JdG/jToRiqeclvyJNteKvPQlTaz3FWuOk9tqd7DKTbySbfwJmJ
UFTqKcVlpEuzt4ULdVJr9WWbbGnMYpm019Kzq9bLPo5ZehzcWnk00/iWTxSKnl0f6eOZzvq1vWiuj2z4o
aW1Ka3WUFRcnkk2/gdHbVks+jll6FpRpQtLZzlHOWWbZxhiLc/1U8oPvGljFEeUqzdvBZ4jbxdNVorJ9+R
WEZXpwnQAA4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAC3wrs8vmKgt8K
7PL5iw3weacACvoAAAAAAYe5mTD3MCowntt X0f5Lgp8J7bV9H+S4JDLD4hyuP2Knys6nK4/YqfKyu7fU
vPgA5fKCZhna//AKSGTMM7X/8ASIaYvOEjFv2YfMVlP9yPqWeLfsw+YrKf7kfUrTN/0X9b9ifynnz0Fb9ify
nnw6/J+4AAR5VzhnZfqTCHhnZfqTCvqY/CFHiPbZ/T8EYk4j22f0/BGI+dk85DrQt5156Md3e+Bm3t53E8o
7u9lnOdOyo6MY5y4cQ7x49/6n6YrTp2VtoLflsXeynbcm5PezerOdabnPPNmqi5SSSbfAqZLzedR9MJNtJ
LNstrK1VCPS1dksu/uFpaRt4dJV978EW8up13owzUF9w0rWMccrfbS9r9PV2e7HYiMZafeYI89pmZ3KVh
vbI+jLspMN7ZH0Zdlj6e78bwRr/sc/oUZeX/AGOf0KMSx/J8mSVh7hGvnNpZLZmcKNKVaooR3s3qWdeE
mujclxQhjWJj/UQuG6dxBw0lJPgysvbLoI6cM3Hd6GbG3rK4jPJxit/xJuIyStJJ97Qeqf8A0pMzClW9F5Qq0
Y0YJSitm7MrOpVnSjUis0+7vNFb1s8lSln6BjSbY/0sbmxhW0pxeU+PEqZRcZOL3ovraEqdvCNT3ktpS3bT
uqjju0g6z1jUWci9seyU/Qoi9seyU/QQfjeUqq+7XP1I5Ivu1z9SORjfyl2tqDr1VFbu9lvUnG0t/wBK3LYkiFh
9elRhLpJZNslO9tnvmmV6cXGtd77oNpDrN3LpotprPab4hbQpRg6cMs33E+jXo1Z5U2s/gZrVadJJ1Wtu7
MOunXh9qW3oOvVUFml3lxOUbW32LctiXeRqF1QhWqy0klJ7Du723e+aYTHFaxPfugW0es3bdaLefE6Y
hbQpQi6cNrZNo16NSWVNrP0N61WnSSdTLJhenWaT3efay2MHa7nGdxKUPdZxI8UxqdLfCuzv1OWL74
HXCuzv1OWL74Feyf8AirSXh84QrtzaWzeRDtQtp3GloZbOJHkpM8o0tpV7aa/VKD9Q7e3rxzjGLz74lTK1r
ReTpy+iJ2G0atNyc01F7kyvXW82nVoQ7u26vVyW2L3MsqNxRhRjFzjsRHxVp9HBLOWZGqWNaCTUdJP
gHHfHeeMLNO1m9iptv4HC6sIODnSWUlty4kCnb13JaMJRee/Iu89GlnJ7ltDSuskTyh543o/vQ9Uava2bU

f3oeqI8Ufb0S3IhYr2b6k1bkQsV7N9Tp9LJ4Spy1wr9iXqVRPwysoTdOWzS3HLw4Z1eE6vUpUf11F+rLJcT
SjeUa8tBJp8Gc8Rtp1tGdNZ5bMiNZ2dVXEZzi4qO3aV6bWvF9RHZ0xG0hCPS01lt2pE+h+xDhkR8Tmo2z
j3yM4fWVSgo5/qjsaC11GSYgqXVC2loKO3vyR0pVKN1TeW1d6aK+8tK3WJzhFyUnnsJOHW06MZSnsb7
glbXm+pjsg3tDoK+S917URybilRTrqK/7VtIRHlvERfUPQ0P2IehwxLskjvQ/Yh6HDEuySOnut/zUoAOXzW0
Iuc4xW9vIvYRjQoJblFFdhlHSqOo1sjsRIxOtoUlBb5FevFEUpN5Vtao6tWU33s0MIEeWZmZS8No9JW02v
0x/JNxGt0VDRT/AFS2G9jSVG3WfvPaytv63S3Dy3R2Ir1T/wCeP+yjAAPI9Bbdnh6HDFOyP1O9t2eHocM
U7I/UPpX/AOamAMxaUk2s9pHzv2ubGlKjQ/XJ7duXAxO+t1Jpyza4I77KtD9L95bGU87OvGTWg38UV7b
zNKxFYW6lGtS/05bGtjRTVqM4XHRy2yb2PiWlhQlQovT3t7uBFuasNYweeyOWbDnJHKsTKdQg6FBKpP
PLj3HJ31tpb/rkdriDrW8oxe1rYU3U6+ll0b9Q6yWtXUVhcVF09BqE8s1saKOcXCbhJZNby7taTo28YSe1F
TeTU7mbjuzyzDPPG6xM/bgWWEb6v0/yVpZYRvq/T/JI+2eDzhZh7gHuOn0VPZfy9X6lwU9l/L1fqXBIY4f
EABWyjxDttT6fgikrEO21Pp+CMR8rJ5SAAjkAAAAAAAAAAAu7LbZw9CkLuy2WcPQr0/j+Sqq29SnNxcW
+GSLSypyoWy01k3tOEMTSWVSGb4pnG5xCVWOhBaMXv4hazSn+oltaTUsRk1ueZti3vwIlrXVvWU2s1l3
HS8uo3Li4xccuIcc4nHMftztO10/Un4t2eHzf4K2jPo60ZtZ5Mk3l5G5pxioOOTz2graIxzCZhvZfqVdf9+fqS
bW+jQo6Dg28+4iVJ6dSUkss3mQvaJpEQt8O7HH1f5KyOy9T/wBxItb+NCgqbg213ohSk3UclxzKt7xMV1
+lviUXK1/Sm9pUNNPJpr1LGIii0MqkG3xRGvLiNxJOMMskQyzS3+olGLawoU40FUa0pPaVJLtL6VCOhKOl
HuDjFMRO7O+sJuroRo9+RJxDsc/p+SLPEaa206X6uLNbi/jWoOnoNN95W/UrETEy6YT7tT1IuIdrkbWd3
G2UlKLlnwONzVVau5pZZhla0dOISck/fl6GcV/ej8pws7hW9VycW81lsF5cK4qSi0kstoTlHS4/tth9GNavl
Paks8ibd3PVWoQpprLMrKFaVCopx+vxJ7xGlKP66TbDvHavDW9SlWtWVajpTho/AqLl5XU2t+ZMhiaWe
lB5dyXcQZ6VetKUIt57ckDLeLViInutqbV3ZZPe1k/U1qNWljkt+WX1OWFqpFTU4VVTT2R3hp
NtU5T9oOebzYBkjwrjDuyR+GZUzTVV8cywwupnCVPvTzIt1Hq983lnt0g9V+9Kykxa9JKNWk38STRqu
4PSptfMiPHEoNZTqBkjwrjDuyR+GZUzTVV8cywwupnCVPvTzIt1Hq983lnt0g9V+9Kykxa9JKNWk38STRqQu
4PSptfMiPHEoNZTpms8TSjlTp5fFlaVvWPu22LWCpYjOK3LM6YpRnNRlBZ5b8jhh8pTvXKW1tEy7u3bVY
pxzi0ErxnHO/pBsKFR3EZaLUVvbJOKNONOHe5GJYpHL9EHn8SBVqzq1HObzbz+wcTala8Y7rbRhaWunCC
bS5nO1vKleso9Fku9nChiWUNCrDSyW9GzxGMZLo6WUc9vxDTqV7altiz/TD1JP79plB747CtvLyNyoqMX
HRfea2t3O32e9HgHHVrF59S5uhV09DQlnnwN6ltVoNSmtnFE1YpDL9t5kS5vJ3GzLKPAOLRSsbiVu5t0N
OmlJ5bEQevVtLLq+0j215O3Wj70OBKeKU0v25NjbXqRaPvTjdXdZwdOpSUdJcSCSLq7dw1+lJIjh58ltz9g
AIzAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAC3wrs8vmKgt8K7PL5iw3wea
cACvoAAAAAAYe5mTD3MCownttX0f5Lgp8J7bV9H+S4JDLD4hyuP2Knys6K4/YqfKzu/UPpgA5fKDvZ1
429fTmm1llsOAC1njO4Tb67p3NOMYKSaeehweJN9xBbxXm/laVMSoypiozyzzzy/JvS+1lBX5vL/jqoxSgZe
zhnZfqTCHhnZfqTCvqY/CFHiPbZ/T8EYk4j22f0/BGI/wR21pQ8E+SKkDbuM94jS21
nQ8E+SI3W6fXem0ZaOW7IhAbSc1pW2tKPPhnyQ1nQ8E+SKkDbr5F0+8vaVehoQjJPPPaiAAGV7zadylYb
2yPoy7KTDe2R9R9GXZ7y+nt/G8+bjAP8A/4VivKjPz7g8Ea/aiwaWdLmPGDWrE1CJY74/vS0xRyatJNO
PrfThRHoXw9OqJS4AICOpHr9TOSHeSY9JJHNCbk7ZjL1KdQhy9QX4c/2NkmtsN4U2BxpRpspCCDNS4mUuo
FBxlMi+7i3C6mkXRvN3O9HL3+5xQhxJVu6e2Qb+kDuupGDaaWFv9W16O+pUkVH/+w7vhNzdxCC0g
Kj4XkXVM3dtg0IyH9w9fPuXKQUJKhODXHFqjJpYP2eU/DKeRl5TA1hMOKNdrVFOuJFpGvuoVWZNNNVUqF
AsfAJ/mTWA3zXKJZVrzqeRyPzMy1p3ZFavo3nrt6FbtzcLDCbzWOkxmYKCkKWGbDzpNJBLGnAUaaKTWvoJ7
e5ZRmP8AcJuXZVdU+CU+rzjb2Wgyj9P3ALnKWbpaXyjvUqq/mF9ciEBOwP6/nIT95Pe+k/1GDN3dYfkVY
9MrHVtWaw+kmN7ec4sBiDKw6qTeWXo+8HRQxtqDFeG9LaC4y+dCNh9Ox8Uu4L1rbjvH5wQNjiACdRpkQ
3jpXMWK8zWYYY7ui+s1qiwEyUNqT+1VrRFQnBtFhz5u9cM6hjKp+n18XbjIhKn4dHZjP0qvIW8sHHxnT
SmPZqKhyRkyJVGHF3dPQQmi1yv9ZeWTZXHOpefc6dnTRuK9C9VcjcZuvm7cZwEL0fShsHwnm4GYt08ljy1Rl
/cvHaYGAPunIcwjpWrYb5QZHsd+n6MyjfXCUsPRGdxQKGHnqRrFgCBfGHmVfu70YYdOGZedsUN2DOJvBVJ
+DiDOMpsu09ljUqtpzJUnlo3NxYlRpohMDqCfBDZmrUIbOhoF5tjjnIJDVX0kZ2bKI4ZDcUX0Fk+GGDpG
cxZ+xFYh6uEOi7H6jaDhyw+HfADdhASUvpClZO2QlySf8qCV9IZnSuQ2jBHGL9tCxaWqEiqDU5MJe
QhG71YRpp/sPhsHzIADcD9p2MRcA+2+gt9Cj0pFVFb7nAGZIRo+yFTtcJAmrJtIu3

MOFpRdCioPfvZ3K+ljjVYiVHiPbZ/T8EYs8Qs5zqOrTWee9ELqlfy2HgyUtyns4g7dUr+Wx1Sv5bI44W9OIO
3VK/lsdUr+WwcLenEHbqlfy2OqV/LYOFvTiDt1Sv5bHVK/lsHC3p0w3tkfRl2V2H2c6U+kqLJ5bEWJYe7B
Wa17o1/wBkn9CjPQ16fTUZQzyzRSzsq8ZNaDfoJZfkVtM7hwB26pX8tmeqV/LZHm4W9OAO3VK/lsdUr
+Wxo4W9OIO3VK/lsdUr+Wxo4W9OIO3VK/lsdUr+WwcLenEvbHslP0KqFlXm8tBr4suaFPoqUYLuRYen8
esxMzKmvu1z9SOWN/Z1JVXUprST3oidUr+XIMclLcp7OIO3VK/lyHVK/lyI44W9OIO3VK/lyHVK/lyBwt6c
QduqV/LkOqV/LkDhb04g7dUr+XIdUr+XIHC3pxB26pX8uRlWddvJU2U4W9LDCuzv1OWL74EuzoOhQU
ZPa9rNMQtpXFNOHvR7uIe2az0tKUZnfqlfy2Y6pX8tkeLhb045jM7dUr+Wx1Sv5bBwt6ccxmduqV/LY6p
X8tg4W9OOYzO3VK/lsdUr+WwcLenE3o/vQ9Ub9Ur+Wzva2NV1oynHRjF57Suq0tM/S3W5EPFey/Umo
5XNBV6Lg9nxK+heN1mHnwTXhlZPY00Y1bX+Bzp8/pX9IYJmrbj4DVtf4DR0r+kMEzVtf4DVtf4A6V/SICX
q2v8AAatr/AHSv6QwTNW1/gb08MqOa6RpR78i6IxX9LKh+xD0OGJdkkSorRiktyNK9JV6UoPvK+hau6ae
eBNeGVk8k00Y1ZX+Bzp8/pX9IYJmrK/wGrK/wGjp39IYJmrK/wABqyv8Bo6d/SGCZqyv8Bqyv8Bo6d/SG
CZqyv8AA3p4XUclptKPeUjFf0sbbs8PQ4Yp2R+pKhFQgorclkaXFFV6Tg+8r32iZpp58E14ZWT2NNGNW1
/gc6eDpX9IYJmra/wGra/wGjp39IYJmra/wGra/wABo6d/SGCZq2v8Bq2v8Bo6d/SGCZq2v8Bq2v8AAaO
nf0hllhG+r9P8nLVtb4E+ztVbU2s85PeytcOO0X3MJIe4B7ivcp7L+Xq/UuCnsv5er9S4JDHD4gAK2UeIdtqf
T8EYk4h22p9PwRSPlZPKWQARyAAAAAAAAAAAAdKVCrV9yDa4hYiZ+nMEmVhcJZ6GfozhKLhJxksmu4
LNZr9tQdqVtVqrOEG1xNp2deCzcNnwC8La3pHAO8rStGOk4bA5iJn6cASVY12s9A41Kc6TynFphZrMfcN
Ab06U6ryhFs7qwrte79wRW0/UIoN6lKdJ5Ti0zNGjOu2qaza3hNTvTmDv1StpuCg81vOcqU4VHBx/VwQ
JrMNASY2NeSz0MvVnOrbVaKznB5cQvC0RvTkDrbUpVai0YaST2ljdWkXQ/wBKktL4LaV1XFNo2qQb1Kc
6Uspx0WZpUKtb3INriHEVmZ05gkTsq8I5uGfocMsnt2EJrMfYYN6dKpVeVOLZ1lZV4xz0M/QLFbSOzrKn
p5LRyz3mtG1q1ouUEmvUtqmyxa/2kDDKujXcO6SDecVYtEe0OUXCTjJZNbzBOxSlo1VNf928ghhevGdA
ADkAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAALfCuzy+YqC3wrs8vmLDfB5pwAK+gAAAAABh7mZMPcwKjCe21fR/kuCnwnttX0f5LgkMsPiAArUA
AAwZARjIGQBgGQBgyAAAAUAADYNgAQ2DYAA2DYAA2AAAAAoAAAAAAEa+vqGH0eluZ6Mdy+IEkHC
zvKN9bqtby0oPZ6M7gAAEAAFAAAAAQAAAAAAAAAAUAAGMgZAGAZAGAZARgGQBgyAFAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD3APcBT2X8vV+pcFPZfy9X6lwSGOHxA
AVso8Q7bU+n4IxJxDttT6fgikfKyeUsgAjkAAAAAAAAAAAABF7FOFsuhSby2IoifSd7QprRjpRe3LgVvhtxmezZ
3N5Tec6ea79hHox63efr3Payxta9Ws2qtLRWW84ScKOJxyySktvqGk13qd9ki5nOhTjGhTb9FuOdrXrzno
VqbSf/dkdbp11FSt8nlvRFp1b6by0Evi0Glp1b9uOIUFSrKUdil3Fo5qnQ05LcinuK1WpUUayycWW04dJb
OGeTaDnHO5txQNaVNPPRWjnuJlxThc2ull3ZplYrGu56Og/XuLSo1b2eUn7scglJtMTF/pW2lzOjmox0nLc
iRG6vNJaVF5Z+FnTC4Q6vp5LSb2nOpdXXTuEYZbdmwJWJrWJ2kXtNVbVvLalms+4i4Sn0lT0RNuM+qS0t
+jtIWE/uVPRB3aP/WqTd3itpaMY5ye0rY3UutdM0s+COuJ9q+gwyEZ125bclsDK9rWvxdut3c9sKL0e79L
JlJuvb/6sNFvemiPeXFejUUaUM4tb8iRbSqSoJ1klJhtT7mJnast60ra4lCKTUpZbSyuqzoUNOKTfBlTJ5Xjb8
RaXsJVLVqCzfAOMczxmIVkpyvbmCkks9mzgWlWTt6CjQpuT4JFVQUra5hKrFx9S2rup0WlbtZ/kGLvEzP2
4W9xcuoo1aT0X35bjjiVulOM4bNLYzaNS/lLLQS+LRHubit0ihWS/S89gS1o46lYpQs7VvL3VzIcMTqaf6or
RzJtSKubRqD95bCsjZV3PRcGlxYXJNo1FPpa12naya3NFNatq5ptcS4rLRtJR4RKa2WdxTS8SCZp/1VZYos7
dPgypLfFOzJd7ZUEln+R5gADAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAALfCuzy+YqC3wrs8vmLDfB5pwAK+gAAAAABh7mZMPcwKjCe21fR/ku
CnwnttX0f5LgkMsPiAArUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADzXtv2G2/5H+D0p5n2
37Dbf8AI/wB09i/4up/yP8ACPRHnfYv+Mq/8j/CPRAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA9wD3AU9l/L1fqXBT2X8
vV+pcEhjh8QAFbKPEO21Pp+CMScQ7bU+n4IxHysnlIACOQAAAAAAAAAAAAFseZNpYnUispRTyIQDqtpr9J
08Um1+iCT4kOc5Tk5SbbZqAtr2t9yl0b+rSSTymvidJ4pNr9MIr4kAFWMt4jW20pyqTc5tuT4siMxaleWWcnkuCOIJtnOS0xqZd7e6nb+7XBkiWKTa/T
TSfEgAbWMlojUSmTxGpOj0bgtqyzOVO1Vtcu3SaVTnSNGVaTbWzt6aA5WjV9BCdsPGEdXaVcl8BGJhGtOFNMH3BVZqMPuG0YVsGhN96ZExaFqVdcxWlohNnwCxTHdqzHU01hrz2lAiaNWtNUh9aa8C0ftPDUyVaknnWt6T29jhlZh7sVY1S+2pgVjO5P+anbSAbllrUOvOUfbc
L6rQio7JRXcyKBtedt72sHik2nlBhVasqs3ObzbOzZoZBbJa327295UobI5OPBneeeKVFmowWSfSWGmowSfEgGQsZLGtnxTWYi9ji3GonvMd9ybJLn
iNSVJwcFtWWZyeqcWWyO0iEi3u5W8XPvPbJGtwqxOrGhWidQk0s3Kux3QSnNXtlpyb8HAZpuu5RwMTEWaVQlRNdXUQmSb3ntDdwUdqBtjuZbiw8Iy
AHAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAW+Fdnl8xUFvhXZ5fMWG+DzTgAV9AAAAAADD3MyYe5gVGE9tq+j/ACXBT4T22r6P8lwSGWHx
AAVqAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB5n237Dbf8AI/wemPM+2/Ybb/kf4A6exf8
AGVf+R/hHojzvsX/GVf8Akf4R6IAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHuAe4Cnsv5er9S4Key/l6v1LgkMcPiAArZR
4h22p9PwRSViHban0/BGI+Vk8pAARyAAAAAAAAAAAAAdadtVqR0oQbQWImfpyB36nX8tmHaVoptweS
C8benEAByAAAAFtaXEACXUsJ06LqOSyXcRA6tWa/YDJgjkAAAAAAAAAAAAAAAABQAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAC3wrs8
vmKgt8K7PL5iw3weacACvoAAAAAAYe5mTD3MCownttX0f5Lgp8J7bV9H+S4JDLD4gAK1AAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA8z7b9htv+R/g9K2ks28kjy/tnWp1LK3VOpGTVR7nn3Ad/Yv+M
q/8j/CPRHmfY6tSp4dUU6kYvpHsby7kemTzWaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB7gHuAp7L+Xq/UuCnsv5er
9S4JDHD4gAK2UeIdtqfT8EUlYh22p9PwRiPlZPKQAEcgAAAAAAAAAAAAF1h/Y4lKXeHdkiWHo/H8kaeJuE3
Ho9zy3mk8Tc4OPR71lvJE6dm5Nycc/Uj3cLZUG6WjpZ9zDu/OIn/SNb28riTUGk1xOur6zqaOzJd50wn92
fodr+7nRkoU8k9+bDitK8OVkaeG1orNZSIbTTyZb4fdTr6Uam+PeRLykniCillpNMF8deMWq50LSrXWcVk
uLOrw6tDKSaeT3InXNRWltnBbtiIFLEK3SrSacW9qDqaUpOp+0+87FL0RSF3evOzm+KKQjn8jyAAR5wAA
AAAAAAAAAAAAAAAAFAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAALfCuzy+YqC3wrs8vmLDfB5pwAK+gAAAAABh7mZMPcwKjCe21fR/ku
CnwnttX0f5LgkMsPiAArUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAcbugrq0qUG2lOOWaP
m1/a1rK6nQr55xexvvXE+nlRjuCRxWkpQahXh7sn3rgB4rDLKtiF5G3otrPbJ8EfSLekqFvToxbahFRTe/YQ
MEweGFW2i2pVpe/NfgswAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAe4B7gKey/l6v1Lgp7L+Xq/UuCQxw+IACtlHiHban
0/BGJOIdtqfT8EUj5WTylkAEcgAAAAAAAAAAAAF1h/Y4lKSKV7WowUINZL4CGuK8UtuXaph1eVSUlo7Xs2
mjw64Sfu8xrK44x5GNZV3vceRXczin264Uv8AVn6GuKdpXoR6NxOhJuGWb4oxWrTrz0ptZ/BBzN46fFM
wj36nov8AJrfS0MQjLhkyNQuKlu26bW3fmjWtWnWqac8s/gDnHT4/tc1qcby2yjLftTINLDavSrTyUVvyZ
Ho3NWh7ktnA6TxCvOOWkl6INJyY7am32sr7ZZzRRkid7WnS6OTWjlluI5GeW8XncAAIxAAAAAAAAAAAAAAAA
AAAAFAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAt8K7PL5ioLfCuzy+YsN8HmnAAr6AAAAAAGHuZkw9zAqMJ7bV9H+S4KfCe21fR/k
uCQyw+IAc7ibp0JyW9IrSZ1G27lFb2jGnHxLmedlKUpOTbbZjNk28nyf49Hpx8SGnHxI85mxms+T/9Ho9O
PiQ04+JHnM2M2Nnyf49Hpx8S5m2ee481my
wwqrLpZU2845Zh3TPytrS1DaW9gpMQqyndSjm8o7EitcmThG1zpx8S5jTj4lzPO5jNk28/yf49Fpx8S5jTj4
lzPOZsZsbPlfx6PTj4lzGnHxLmeczYzY2fK/j0enHxLmNOPiXM85mxmxs+V/Ho9OPiXM2TT3M81myxwqr
LpHTb2ZZh3T8jlbWloYcore0cbuo6dtOUd5RNtvNvaV1lzdOdPRdJHxIdJHxIzI85t4jaTbL5P8ej6S
zbxG3iNnnyf49Hpx8S5jpI+JGzQ28nk+T/Ho+kj4kYObeobeI2eUuaMnm1Ut203Ut4T
e9oNcWXnOnUw5Jb2jL3FDdVZVLibbex5IrrJk4QvNOPiQ04+JHncxmxmxs+T/Ho9OPiQ04+JHncxmxmxs+T/Ho9
9PRacfEhpx8S5njDbStNbWtOdUZmcrB7nYYFebPhWbuaG4sFpacEm1jGjG8vB5GsXeI2+WkRj2kmCtWBnUNu6pJgaTLT2CG3
oUoxi8tJ7St72412macfEhpx8SPOZjNjZjm3l+T/Ho9OPiQ04+JczzmYzY2fJ/j0enHxLmbJp7meazZY49Hpx8SGnHxI85mxms+TP6
PTj4kNOPiR5zNjNjZ8mfT0enHxI2TTWaVxnms2TsLqyVbo884tbg7pn5TrS3M
NpLa8jJT4nVm7jQz/AEpbitcl+EbW2nHxLmNOPiXM85t4jaTbL/K/j0enHxLmNOPiXMacfEjzmM2Mac
fEjzm0bRs+T/Ho9OPiQ04+JHnNo2jZ8n/PR6cfEjZNNZpnms2cczWW0nYyVkq/R5Pk/49HpxrJXZXVy5n8pa3B1T8jpN
DP9KW4rbJfhG1t0kPEuuY6SPiR5zNjNjZ8n/AD449H0kfEjG9/pl/Ho9HYyS2Hh8uNy9Q9GPSEm1HnDy/tGpDCpddemjTfuJlXK7U1tCw/Ri0BEDW8i6p0ZiMSd/bo6XXidf8rE9sqDitzTI/VXe8LtMqYpWnN5W1hpLLoFIIR+QNRlwSpZVUWFiXmyGJYjFTMx5YIpCKSrYEDZbZqZklVWzDQxo2pZ4VJU4hbhHyBfV3SBVXkzrIbcNvX6zcn
ZOEbXunHxIacfEjzmbGbG2HyZ9PR6cfEhpx8S5jTj4lzPOZjNjZ8mfT0enHxLmNOPiXM85jNjZ8mfT0enHxI85m
xmxs+TPp6PTj4lzNszzWbLHOl2jD1CN1YY5+9A+SvOC6/BM5PscB2X8vV+pcFPZfy9X6lwSGOHxAAVso8
Q7bU+n4IxJxDttT6fgikfKyeUsgAjkAAAAAAAAAADBkAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAt8K7PL5ioLfCuzy+YsN8HmnAAr6AAAAAAGHuZkw9zAqMJ7bV9H+S4KfCe21fR/kuCQyw
+IcbzstT0OxxvOyz9Cu7+MqAAHL5UgBlLPiBgAAAAAAAk+AAAAAAAAAAAAAAAAAAAAAAAAA
vbKnqJer8nxhwAH0I8IAPoAAAAAACbhfaX8v/ghE3C+1+P5RDTF5Swn3n3/AGOf0KMvL/sc/oUYa/k+QAA8w

DOT4GAaABk+4ADOTW9GAaC9seyU/Qoi9seyU/QsPT+N5SkPcedrfvT+ZnonuPO1v3p/MxLv8n6hoACP
GAAAAAAH0YAFjhHv1fRFcWOEe/V9EIa4fOFoVuL+5D1LIrcX9yHqdPbm8JVgAOXzQAAAAAAAAAACXh
va16EQl4b2tegaYvKF0UmI9sl6IuykxHtkvRFev8jxRQAR4AD6MfQAAPowAH0YAErDe2RIpKw3tkQ7x+U
LspMR7XL0RdlJiPa5Fev8nxRQAR4QGcuBgKABbQgBkABKw3ta9CKSsN7WvRhpj84Xb3Hna/78/U9Ezztb
9+fqWXp/J+oaAAjxAH0AAAAAAALHCPeq+iK4scI96r6IQ1wecLQPcA9x0+kp7L+Xq/UuCnsv5er9S4JDHD
4gAK2UeIdtqfT8EYk4h22p9PwRiPlZPKQAEcgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAW+Fdnl8xUFvhXZ5fMWG+DzTgAV9AAAAAADD3MyYe5gVGE9tq+j/JcFPhPbav
o/yXBIZYfEON52WfodjjedIn6Fd38ZUAAOXypTrCzjWXSVPd7lxJs6lrQejJRi/gjnhtWMrdQXvR3nG7satSu
509qZXsrHGkTSNpTo29zTzjGOXFLIqbqi6FZwb2b0Wtjbzt6TU2s288l3FfiNVVLjOOTUVkEyxE0iZjuk2VlD
o41Kq0m9uR3nXtKctB6Ka7lE3takattHR4ZP4FfUw6t0kssmm9+YdT/mscI2m1bSjcQ0opJvc0U9SDhUlF
708i7tqbt7dRm9y2lPc1FVrzlHc2HGesaif25E3Cu0v5SETcK7S/lEMcXnC4KG97ZU9S+KG97ZU9Q9X5PjD
nTWdWCe7SRdulQjHOVOCXFoo4S0Zxllnk8yZXxBVqLp9G1n35hjitWsTtN/6ThS5I6KhRazVOHIprWk6te
MVu3st7mqqFvKXwyQb0vFomZhFt4U53tZaMXFblkSpQtoPKUaafxSKu1uerzlJx0s/ia3lwrmcZKOWSyD
KMta13+1tGNrOWjGNNvgkjS7pUYW85KEU8t+RHwuhkpVXvexGMVq7Y0l6sNJtHT5TCuJuF9qfy/wDgh
E3C+1P5f/BHlxecJ9/2Sf0KMvL/ALJP6FGGv5Hm2hB1JqMd7La3sKVKOdRKUu/Mi4XTUqspvuWwlXyq1
FGlR3veFxUiK8phv0lqv05w9CHiPQKMeiUdJ98TR4ZWS2OLfqRqtOdKWjUjkype9tamG9rbyuKmitiW1st
Y29vbwzcY+sjXDqahbKSW2W04XNGvdV2orKEd2Yd1pwrvW5SeltKmzOD9UVd50auJKkkorgbzw+vBZp
J+hGaabT3kZZLWmNTGmC9seyU/Qoi9seyU/QsOvxvKUh7jztb96fzM9E9x52t+9P5mJd/k/UNC1w6jCV
BucIyefeiqLnDllaR+OZGWCN2byVpFtNU018Ec6rtOinoqnnk8tiI9ewrVK05rLJvZtIla2q0ffjs4laWvaP8A5
ZtYqdzCLWaz2ouJ07ems5whFfFIqsPWd3H4Fle0JXFJRhvTzBhj/Ezo0rPhS5Irb/o+n/0tHRy/7TM8Prx2qK
a+DI0k4vJrJojjJa0xqY0wWOEe/V9EVxY4R79X0Qj7cYfOFoVuL+7D1LIrcX92HqdS9mbwlWAA5fOC0sbO
Kh0lWKbe5Mj2Fr0s+kn7kfuTL646GGhD35bPQr04qREc7Id/UpynoUoxSW9pEM7O1rKLk4PLfmciMb7
mdzDBaWNnFQ06sU29yZww+26WppzX6Vu+JKv7noafR0/fl9itsVIrHOyFfVKbnoUoRSjvaW8inaVrWjFy
cHkjiRjfczuQl4b2tehEJeG9rXoFxeULopMR7ZL0RdlJiPbJeiOnr/I8UUtbR23QQ01T0stuaRVB7Dl46X4zteL
qsnklTbfwRtKjRjFt04JehGw+0UIKrNZye1fA0xKrV9xRah3viV7OWqcphwtnS623PR0M3v3FhnacKX2KT
M7WtCVxV0c8l3sPPTJMdohcQp281nGEGvgiDiapQUIwjGMs9uSJ0l1e3ypQbyWxIpKspym3U974hrmtq
utd2hKw3tcSKSsN7XEjzY/KF2UmIdrkXZSYh2uRZev8nxRiwsrWjKn0tSSl8O5FeFJpNJtJ7yPJS0VncwtutW
cXorR/9pmra0binpU0k8tjSKcucPg6VrnPv2leil+pOphXW9CNSu6c5aORYN2lstFqOfpmyqrT0q8px2Zs0b
bebeb4kZRkinbS5hK1us4xUW13ZFfe23V6i0fdluNsNhKV0pLdHedcWmnKEVvW0NLavj5TCvJWG9rXoyI
S8N7WvRhjj84XTPPVv35+p6Fnnq378/UsvT+T9Q0ABHihdU+qtRWVNy9ELujTjbTcacU8t6RVW3aYepcX
nZZ+hXtpaL1mdIlh0HQf6qhpZ95NVGhKOapwafwKAvrTstP5URMFuXbSlrJKtNJZLM5nSv8Avz9TmHkt9
hZYR71X0RWllhHvVfRCGmDzhZh7gHuOn0lPZfy9X6lwU9l/L1fqXBIY4fEABWyjxDttT6fgikrEO21Pp+CM
R8vJ5SAAjgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAC3wrs8vm
Kgt8K7PL5iw3weacACvoAAAAAAYe5mTD3MCowottX0f5Lgp8J7bV9H+S4JDLD4hxvOyz9Dscbzss/Qru/j
KgABy+UubC3jSpKeX65LNs4VsScajjTjsXeyRZV41KMY5/qismjjWwzTqOUJ6Ob3Mr3Ty4R03a0u+sqUZL
KS3ohX1tGFWOhsU+4m2trG1Tbecn3kO/uIzqxUNuhtzCX8I5/afRpQtaGxblm3xIUsUnpfpgtgH4snUqsLij
nF7GsmuBDlhTcv01P0+gdX56jppVCrG7oZuOzc0VV5RVG4cVue1ue1FtQowtaOWezvbKm8rKucOUfdWxC
Webwjl9uBNwrtL+UhE3Cu0v5QwxecLgob3tlT1L4ob3tlT1D1fk+MOBkwdKVN1asYLvZHiiNzpY4ZS0aTq
NbZHDFK2lUVJPZHa/UsZONC3b3KK2FFOTnNzfe8yvVknhSKQ1Mxi5zUY728jBOwyjpVXUa2R3eoealeV
ohYwSoUEtyiijrVHVqym+9hnidbQo6C3y3lSRtnt34x+gm4X2p/L/4IRNwvtT+X/wGeLzhPv8Ask/oUZeX/ZJ
/Qow0/I8llhLX60Sry46vTTYzb2IqrSv0FZSuvYVy1r0oXdFZS+KaK1xTvHqPtDoYjOVWMaiTTr2HSfpguLFa
qZxoYboVFOc88u5EXIXEYUXTT/AFS2egWOUUnm62LztIZdyyyOF3fSo1dCmXmNHMCsvJzmMBrtF9/xBym2OOH2xZXkq8nCa2rc0cMVpKpJxZDe83XQ/Etkji6ekE/DGP9ZqEXmo95HN5mMer/
aEXtj2Sn6FFkXtj2Sn6Fhx+N5SkPcedrfvT+ZnonuPO1v3p/MxLv8n6hoS6V/OjSUFFZIipZtLiW6sKFSjFLfl7
yZGGKLTM8UaOKTzWdNZFinC4oJ5bJIhxwuCltm2iVUnTtqO1pJLIr1U5xE81RCo7W5k4pNptbvTp4U
ZsaVO4q1J1cnnuTO08Mpyf6ZSiGNa31usulpe9Yk4yjkzhitKW2mWGGgW7ck22+9khLnFppb

w0v2x6v9oJY4R79X0RXFjhHv1fREj7efD5wtCtxf3YepZFbi/uw9TqXszeEqw3oU+lrRhuzZobUqjpVFOO9
HL58ffdf06apU1CC2IhTsKtSrpzqJvPMj6yrfDkZjiNZzitm18CvXOTHbUSs509Ki4Z71lmVksPlCpCMpp6TL
OpNxoSmt6WZUzvqspKTazju2B1lmnba3hBU6ajBZJLYQZ4fUqVXOdRNt5kdYlWy3rkbRxGs5pbMm+Ac
zkx21ErSpDTpOOeWayKa6tJWyTlJPMuKsnGjKS3pFNXuZ3CSnlsfcDPx1/XAl4b2tehEJeG9rXoR58XlC6KT
Ee2S9EXZSYj2yXoivX+R4ooBa0bKjO2ink34kyPHSk3+nCniU6cIx0E0llvLCjUjdUdLLY96ZF1XDP35ZEqKp2
tHLNKKK9mOLx5/SmuaSo3EoLcnsM21w7ablFZtrIkUFC7vZTnlo9yfeSKmGUpPODcfQMIx2meVS1xDpq
qhOOTe5mMTpRdHpMlmjpb2EKE9PNyfxOOJV4un0UWm3v+AbTvpzzVhKw3tcSKSsN7XEjyY/KF2UmI
drkXZSYh2uRZev8nxRjBkn4faRnlVnty3IjyUpN51BZWWl/qVlsW1Ji+vE06VJ/p72TbmlVqw0Kc1Bd5C1V
LzFyK9NqWrXjSFetp2tradxPKOxd7Ois2rtUHLuzzRbU6KpU9Cns+LDPHhmZ7/pHqVKVjQ0Y5aXcuJU1Kk
qs3KTzbLGphtWrJynVTb+BHuLCVCk5uafwIuSt5/XZDJeG9rXoyIS8N7WvRhlj84XTPPVv35+p6E89W/fn6
ll6fyfqGgAI8Lra9ph6lxedln6FPa9ph6lxedln6FezB4Soi9tOy0/lKIvbPstP5SJ+N9yp7j9+fqcjrcfvz9TkHmt9
hZYR71X0RWllhHvVfRCGmDzhZh7gHuOn0lPZfy9X6lwU9l/L1fqXBIY4fEABWyjxHttT6fgjEzFIZXOfiRDI+X
kjVpAARwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABb4T2eXzFQ
XWGw0bRPi2yw9H4/mlgAr3gAAAAAYe5mTD3MCowntttX0f5Lgp8J7bV9H+S4JDLD4hxvOyz9Dscbzss/
Qru/jKgABy+UzGTi808md1e3CWXSPL4kcB1Fpj6dalzWqrKc21wOQASZmfttCpODzhJr0OyvrhLLpGRwFi
0x9S6VK9Wr782zmAEmZn7CbhXaX8pCJuFdpfyld4vOFwUN72yp6l8UN72yp6h6vyfGHA2hUlTlpQbT4o
1BHh+nWdzVqR0Z1JNcGcgAszM/YdKdxVpx0YTcVwRzASJmPptUqTqy0qknJ/E1ABM7Cbhfan8v/ghE3C
+1P5f/AaYvOE+/wCyT+hRl5f9kn9CjDT8jyDpTr1KX7cmjmAwiZj6SJX1w010jWZwbb2yeb4mAFm0z9h2
hd1oLKM3kcQEiZj6dp3dapHRlN5HEAEzM/YXtj2Sn6FEXtj2Sn6Fh6PxvKUh7jztb96fzM9E9x52t+9P5mJa
fk/UNDpCvUp+5No5gjxxMx9JHXrjL3zlUqzqPOcm/U0AJtM/cspuLzi2mdle3EVkqj+pwAItMfUu1S7r1I5S
qPI4gAmZn7Cxwj36voiuLHCPfq+iENMP/SFoVuL+5D1LIrcX9yHqdPbm8JVgAOXzQLY80AB2d1XlBxdR5P
YcQAszM/YE2nmgAjtK6ryi4upJp9xxAACzM/YhLw7ta9ClS8N7WvQO8XlC6KTEe2S9EXZSYl2yXoivX+R4op
vCrOm84Sa+poCPBEzH0kdeuPGcqlarVf65tmgDqb2ntMibTzN5XksnNnBvN
5gEJtM/YSsN7XEikrDe1xDrH5QuykyL1/k+KKdYXFWmtGE3cuPNkOuXHm
yOADrnb26dPV6TpNN6e7M365cebI4AJyn279cuPNkaTuK1NZQm19TK1SOjOo2uBzAJtb2ErDe2ErDe1r0ZFJWG9
6Z56t+/P1PQs89W/fn6iXoXq/J+oaoAjwxMx9JHWrjLTmm2zmAQiZj6dp282AB2d1XlBxdR5PYcQAszM/YE2nmgAjtK6ryi4upJp9xxAACzM/YhLw7ta9ClS8N7WvQO8XlC6KTEe2S9EXZSYl2yXoivX+R4op
JuUm282zAAQLLCPeq+iK0ssI96r6IQ1wecLLmbBdvGmWXsIt5eKZqEQ00dXeJsY7L3PkNtrHnvCaycZvCy4Dcy2vxCtvmLbv3hcWo5knR1zpmt1XKCkwovFOO+0tRptYfGeQIuBOnDpqZvVxLgXS7G1Htu/AxvLY4Xse1ju4pcOH
+P4Kg9JKKlFxa2NZFDd0HQrOL3f9r4keL8infk4gAjygAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAK2pwdSpGK3tnoacdCnGPBFfhls1/rTXylkWHt/HpqNyAAr0gAAAAAYe5mT
D3MCowntttX0f5Lgp8J7bV9H+S4JDLD4hxvOyz9DsYklJNNZplaWjcaebBbzwynKTcZOOfcaaqh5j5E08E4L
qsFpqqPmPkNVQ8x8iaOhdVgtNVQ8x8hqqHmPkNHQuqwWmqoeY+Q1VDzHyGjoXVYLTVUfMfIaqj5j5j5
FOhdVk3Cu0v5TvqqHmPkSra1p26ehve9sO8eG0WiZdyhvtl5U+LL4jXNnTuHm9kuKDfNSb17KMFpqqHj
Y1VDxsaeXoXVYLTVMPGxqmHjY0nx7qsFpqmHjY1TT8bGj491WC01TDxsaph42NHx7qsm4V2p/KSFhU
PGyVb20LeOUd/ew0x4bRaJlpf9kn9CjPRyipRcZLNPcNa095CnhtFomXUVZ095CnhtFomXjY1TDxsaph42NVQ
rqqHjY1VDxsaph42NHx7qsm4V2l/KSFhUPGyVb20LeOUd/ew0x4bRaJlpf9kn9CjPRyipRcZLNPcNa095CnhtFomXUVZ
4cVqTuWXuPO1v35/M9E9x52t+9P5mJah42NVQ8x8hqqHmPkSyurk3ssqPUfIaqh5j5Ew0dC6rBaaqh5j5DVUPMfI
VQLXVVPGxqmHjY0fHuqwWmqoeY+Q1VDzHyGjoXVYLTVUPMfIaqh5j5DR0LqsFpqqHjY1TDxsaPj3VZNwrtL+Ukr
w9SyOdaOjwOs53dnVjTtXS0bfTzOhxTx8K7ZeVPiq+I1zZ07h5vZLih3zUm9eyjBa6nn7jVUPMfIaqh5j5DR0LqsFpq
qBa6qh42NUw8bGjl9e6rBaapp42NUw8bGj491WC01TDxsaph42NHx7qsm4V2h/KSFhUPGyVb20LeOUd/ew0x4bRaJl
F2R7i0p3G2WyS3NFejLSb11CiBa6rj5jGq4eY+RNPJ8qPvRuDZq0DeuIHQuqgWuqoeo+Y+Q1VHzHyHyB
0LqoFrqqPmPkNVR8x8gdC6qJWG9rjkS9VQ8x8iRbWtO3T0T0dre9sO6YLRaJlIKTEe1y9C7I9zaU7jJy2Nd6K
3zUm9dQogWuq4eN8hquHjZHl6F1UC11XDxsarh42NHx7qsm4V2h/KSFhUPGyVb20LeOUd/ew0x4bRaJlVRLw1Z3a9GStVw8bJNta07dPRPR2t73w8dY8NotEy7dlKC11VDxsarh42NHx7qsm4V2l/KSFhUPGyVb20LeOUd/ew0x4bRaJl
Q8b5DTzdC6qBa6qh5j5DR0LqsFpqqHjY1TDxsaPj3VZNwrtL+Ukr
hvqqHjfIl0KEKEENGCyGmmLDatty6h7gHuK9insv5er9S4Key/l6v1LgkMcPrAArYONzbxuxuIuOwCTE
TGpefr287eeU08u59zOR6OpTjUi4zSa+JXXGGd9F/Rk08WTBMd6q0HadpXp+9TeXFHPQl4XyDCazH6ag

20JeF8hoS8L5ETUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUt
QbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8ga
lqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvk
DUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF
8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhL
wvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q
0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4Xy
GhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEv
C+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDbQl4XyGhLwvkDUtQbaEvC+Q0JeF8galqDZU
5yeSi2/QkUrCvU3x0V8Q6ilp+oRSdZ2EqklOqso9yfeS7bD6dHKU/1y+JMRXpx4P3ZhJRSSWSRkAr1gAAA
AAAABh7mZMPcwKjCe21fR/kuCnwnttX0f5LgkMsPiAArUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAPcAwKey/l6v1Lgp7L+Xq/UuCQxw+IACtgAAAAAMZLgZAG
MlwGS4GQDTGS4DJcDICahjJcBkuBkA1DGS4DJcDIBqGMlwGS4GQDUMZLgMlwMgGoYyXAZLgZANQxk
uAyXAyAahjJcBkuBkA1DGS4DJcDIBqGMlwGS4GQDUMZLgMlwMgGoYyXAZLgZANQxkuAyXAyAahjJcB
kuBkA1DGS4DJcDIBqGMlwGS4GQDUMZLgMlwMgGoYyXAZLgZANQxkuAyXAyAahjJcBkuBkA1DGS4DJ
cDIBqGMlwGS4GQDUMZLgMlwMgGoYyXAZLgZANQxkuAyXAyAahjJcBkuBkA1DGS4DJcDIBqGMlwGS4
GQDUMZLgMlwMgGoYyXAZLgZANQxkuAyXAyAahjJcBkuBkA1DGS4DJcDIBqGMlwGS4GQDUMZLgMl
wMgGoYyXAZLgZANQxkuAyXAyAahjJcBkuBkA1DGS4DJcDIBqGMlwGS4GQDUMZLgMlwMgGoYyXAZL
gZANQxkuAyXAyAahhJLuMgBQAAAAAAAAAAAAAMPcZAFNhklTxCopvJtNbfUuSuvrBVJOtTloT7/AIkPR
u0slX+5zvTzRacf+dL0FDleef8Af+hleef9/wChyXrfxfAocrzz/v8A0Mrzz/v/AEOR1v4vgUOV55/3/oZXnn/f
+hyOt/F8ChyvPP8Av/QyvPP+/wDQ5HW/i+BQ5Xnn/f8AoZXnn/f+hyOt/F8ChyvPP+/9DK88/wC/9Dkdb
+L4FDleef8Af+hleef9/wChyOt/F8ChyvPP+/8AQyvPP+/9Dkdb+L4FDleef9/6GV55/wB/6HI638XwKHK88
/7/ANDK88/7/wBDkdb+L4FDleef9/6GV55/3/ocjrfxfAocrzz/AL/0Mrzz/v8A0OR1v4vgUOV55/3/AKGV5
5/3/ocjrfxfAocrzz/v/QyvPP8Av/Q5HW/i+BQ5Xnn/AH/oZXnn/f8AocjrfxfAocrzz/v/AEMrzz/v/Q5HW/i+
BQ5Xnn/f+hleef8Af+hyOt/F8ChyvPP+/wDQyvPP+/8AQ5HW/i+BQ5Xnn/f+hleef9/6HI638XwKHK88/w
C/9DK88/7/ANDkdb+L4FDleef9/wChleef9/6HI638XwKHK88/7/0Mrzz/AL/0OR1v4vgUOV55/wB/6GV5
5/3/AKHI638XwKHK88/7/wBDK88/7/0OR1v4vgUOV55/3/oZXnn/AH/ocjrfxfAocrzz/v8A0Mrzz/v/AEO
R1v4vgUOV55/3/oZXnn/f+hyOt/F8ChyvPP8Av/QyvPP+/wDQ5HW/i+BQ5Xnn/AH/oZXnn/f+hyOt/F8Ch
yvPP+/9DK88/wC/9Dkdb+L4FDleef8Af+hleef9/wChyOt/F8ChyvPP+/8AQyvPP+/9Dkdb+L4FDleef9/6G
V55/wB/6HI638XwKHK88/7/ANDK88/7/wBDkdb+L4FDleef9/6GV55/3/ocjrfxfAocrzz/AL/0Mrzz/v8A0
OR1v4vgUOV55/3/AKGV55/3/ocjrfxfAocrzz/v/QyvPP8Av/Q5HW/i+BQ5Xnn/AH/oZXnn/f8AocjrfxfAoc
rzz/v/AEMrzz/v/Q5HW/i+BQ5Xnn/f+hleef8Af+hyOt/F8ChyvPP+/wDQyvPP+/8AQ5HW/i+BQ5Xnn/f+h
leef9/6HI638XwKHK88/wC/9DK88/7/ANDkdb+L4FDleef9/wChleef9/6HI638XwKHK88/7/0Mrzz/AL/0
OR1v4vgUOV55/wB/6GV55/3/AKHI638XwKHK88/7/wBDK88/7/0OR1v4vgUOV55/3/oZXnn/AH/ocjrfx
fAocrzz/v8A0Mrzz/v/AEOR1v4vjEmkm28kUWV55/3/AKMqjdVf0zrLL1Y2db+OuH/rxOpOO2OT2/UuCP
Z2kLWnlF5t72SCw0x1mte7/9k="></figure> Constraints 0<n<100 0<arr<
1000 Input Format   -First line will contain the number of
elements present in the array.
    -Second line will contain the elements of
array Output Format
    -Output contain only one
that space separated integers in reverse order.

answer

```
#include<iostream>

using namespace std;
```

```
int main()
{
   int n,i;
   cin>>n;
   int arr[n];
   for(i=0;i<n;i++)
   cin>>arr[i];
   for(i=0;i<n/2;i++)
   {
      int temp;
      temp=arr[i];
      arr[i]=arr[n-1-i];
      arr[n-1-i]=temp;
   }
   for(int i=0;i<n;i++)
   cout<<arr[i]<<" ";
         return 0;
}
```

question

CTS Interview Question Question description Issac is a Language teacher at a high school in Madurai. Sabari is a student, he is studying programming while during Language class. Issac tells Sabari to leave the other subject and learn Tamil. Sabari asked permission for 10 minutes, Finally, Sabri got permission to solve the program. The computer teacher has given homework on the topic of Arrays and where he need to do Array Rotation as per the input of question. . But Sabari is not good at C Programming. Can you help him to solve the programming problem? Constraints 0<n< 100 0<arr<1000 Input Format
   –First line predicts the total number of elements present int the array.
    -Second line contains the elements of array
    -Third line contains the number of time rotation need to be done
    -Fourth line contains the character ‘L’ or ‘R’ that defines what type of rotation needs to be done. Output Format
    -Output contains only line that is only the resultant output

answer

```
 #include <iostream>
using namespace std;
int rotLeft(int arr[],int n,int d){
    for(int i=d;i<n;i++)
    cout<<arr[i]<<" ";
    for(int i=0;i<d;i++)
    cout<<arr[i]<<" ";
    return 1;
}
int rotRight(int arr[],int n,int d){
    for(int i=n-d;i<n;i++)
    cout<<arr[i]<<" ";
    for(int i=0;i<n-d;i++)
    cout<<arr[i]<<" ";
    return 1;
}
int main()
{
    int n,d;
    char c;
    cin>>n;
    int arr[n];
    for(int i=0;i<n;i++)
    cin>>arr[i];
    cin>>d;
    int z;
    z=d%n;
    cin>>c;
    if(c=='L')
```

```
    rotLeft(arr,n,z);
    else
    rotRight(arr,n,z);
        return 0;
}
```

question

Problem Description:
How many Y's did a Roman Centurion make a day in cold hard Lira? About a C's worth! Turns out, Martians gave Rome the idea for their number system. Use the conversion charts below to help translate some Martian numbers!

Note, that unlike the Roman Numerals, Martian Numerals reuse symbols to mean different values. B can either mean '1' or '100' depending on where it appears in the number sequence.

Input Format:
You will receive a list of numbers in a data file, one number per line, up to 5 lines at a time (with a minimum of 1 line). No number will exceed 1000, or be less than 1.

Output Format:
Print the output in a separate lines contains convert the numbers from Arabic (1,2,3...10...500...1000) to Martian (B,BB,BBB...Z...G...R)
numerals.

answer

```
#include <bits/stdc++.h>
using namespace std;
void print(int number)
{
        int num[] = {1,4,5,9,10,40,50,90,100,400,500,900,1000};
        string sym[] = {"B","BW","W","BK","Z","ZP","P","ZB","B","BG","G","BR","R"};
        int i=12;
        while(number>0)
        {
        int div = number/num[i];
        number = number%num[i];
        while(div--)
        {
                cout<<sym[i];
        }
```

```
        i--;
        }
}
int main()
{
        int number,n2,n3,n4,n5;
   cin>>number>>n2>>n3>>n4>>n5;
        print(number);
        cout<<endl;
        print(n2);
        cout<<endl;
        print(n3);
        cout<<endl;
        print(n4);
        cout<<endl;
        print(n5);

        return 0;
        cout<<"char buf[] buf[i++]='R'; while(n>=10)";
}
```

question

Problem Description:
Public school have arranged an Annual Day Function. Volunteers have decorated a floor on various places of the school using Rose and Tulip flowers. But one of the coordinators requested the volunteers to rearrange the decoration like a triangular size. Coordinator also told them that tulips flowers need to be positioned at the middle of the roses School has 20 buildings and as per Principal order the numbers of rows in the decoration should also match the building number.
The Principal of the school is interested in seeing the final decoration but he is quite busy with the other works. So he likes to see how the final decoration have come through online mode if he gives the building number. So can you display him the final decoration layout?
Note:
Roses are represented by 1. Tulips are represented by 0.

Constraints:
1 ≤ rows ≤ 20
Input Format:
Only line of input has single

integer representing the building number.
Output Format:
Print the final layout of the decoration. Refer sample testcases for format specification.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
   int n;cin>>n;cout<<"1 \n";
   for (int i = 0; i < n-2; i++) {
      cout<<"1 ";
      for (int j = 0; j < i; j++) {
         cout<<"0 ";
      }
      cout<<"1 \n";
   }
   for (int i = 0; i < n; i++) {
      cout<<"1 ";
   }
         return 0;
         cout<<"for(i=1;i<=rows;i++)";
}
```

question

Problem Description:

saravanan with his friends going to the theatre for a movie. The seating arrangement is triangular in size. Theatre staffs insisted the audience to sit in odd row if the seat number is odd and in even row if the seat number is even. But the instruction is very confusing for saravanan and his friends. So help them with the seating layout so that they can sit in correct seats.

Constraints: 4 ≤ N ≤ 20

Input Format:
Only line of input has single integer value representing the number of rows in the theatre.

Output Format:

Print the layout based on the number of rows specified in input. Refer sample testcases for format specification.

answer

```
 #include <stdio.h>

int main()
{
   int i,j,k,N;
   scanf("%d",&N);
   for(i=1;i<=N;i++)
      {
         if(i%2==0)
            {
               k=2;
            }
         else
            {
               k=1;
            }
         for(j=1; j<=i; j++,k+=2)
            {
               printf("%d ", k);
            }
         printf("\n");
      }
   return 0;
}
```

question

<p>Problem Description:
Simon work with Greek squares and matrix traces.

The trace of a square matrix is the sum of the values on the main diagonal (which runs from the upper left to the lower right).

An B-by-B square matrix is a Greek square if each cell contains one of B

different values, and no value is repeated within a row or a column. In this problem, we will deal only with "beautiful Greek squares" in which the B values are the integers between 1 and B.

Given a matrix that contains only integers between 1 and B, we want to compute its trace and check whether it is a beautiful Greek square. To give some additional information, instead of simply telling us whether the matrix is a beautiful Greek square or not, show the number of rows and the number of columns that contain repeated values.

Constraints:
1 ≤ T ≤ 100.
2 ≤ B ≤ 100.
1 ≤ Ai,j ≤ B, for all i, j.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each starts with a line containing a single integer B: the size of the matrix to explore. Then, B lines follow. The i-th of these lines contains B integers Ai,1, Ai,2 ..., Ai,B. Ai,j is the integer in the i-th row and j-th column of the matrix.

Output Format:
Print the output in a single lines contains the number of rows and the number of columns that contain repeated values.   Explanation : <h3> </h3><figure class="table"><table><tbody><tr><td>Input
 </td><td>Output
 </td></tr><tr><td> 3

4

1 2 3 4

2 1 4 3

3 4 1 2

4 3 2 1

4

2 2 2 2

2 3 2 3

2 2 2 3

2 2 2 2

3

2 1 3

1 3 2

1 2 3

  </td><td> Case #1: 4 0 0

Case #2: 9 4 4

Case #3: 8 0 2

  </td></tr></tbody></table></figure> In Sample Case #1, the input is a natural Latin square, which means no row or column has repeated elements. All four values in the main diagonal are 1, and so the trace (their sum) is 4. In Sample Case #2, all rows and columns have

repeated elements. Notice that each row or column with repeated elements is counted only once regardless of the number of elements that are repeated or how often they are repeated within the row or column. In addition, notice that some integers in the range 1 through N may be absent from the input. In Sample Case #3, the leftmost and rightmost columns have repeated elements.

answer

```
 #include <bits/stdc++.h>

using namespace std;

int t,i,j,tes,n,x,y,sum;

int a[1007][1007];

map<int,bool> udah;

void solve(){}

int main() {

   solve();

   scanf("%d",&t);

   for (tes=1 ; tes<=t ; tes++) {

      scanf("%d",&n);

      for (i=1 ; i<=n ; i++) {

         for (j=1 ; j<=n ; j++) {

            scanf("%d",&a[i][j]);

         }

      }

      sum = 0;

      x = 0;

      y = 0;

      for (i=1 ; i<=n ; i++) {

         udah.clear();

         for (j=1 ; j<=n ; j++) {

            if (udah[a[i][j]]) x++, j = n;

            udah[a[i][j]] = true;

         }
```

```
        }
        for (j=1 ; j<=n ; j++) {
            udah.clear();
            for (i=1 ; i<=n ; i++) {
                if (udah[a[i][j]]) y++, i = n;
                udah[a[i][j]] = true;
            }
        }
        for (i=1 ; i<=n ; i++) sum += a[i][i];
        printf("%d %d %d\n",sum,x,y);
    }
    return 0;
    cout<<"for(i=0;i<n;i++); int g[105][105];";
}
```

question

Problem Description:
For some reason, your school's football team has chosen to spell out the numbers on their jerseys instead of using the usual digits. Being great fans, you're going to be ready to cheer for your favorite players by bringing letter cards so you can spell out their number. Each fan has different favorites, so they each need to bring different sets of letters.

The English spellings for the numbers 0 to 12 are:
ZERO ONE TWO THREE FOUR FIVE SIX
SEVEN EIGHT NINE TEN ELEVEN TWELVE

Input Format:
Read a set of integers from 0 to 12, separated by spaces, representing one fan's favorite players. The last integer will be 999, marking the end of the line.

Output Format:
Print the same numbers, then a period and a space. Then, in alphabetical order, print all the letters the fan needs to be able to spell any one of the jersey numbers provided

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int curr;
```

```
   multiset<char> mp;
   string names[] =
{"","ONE","TWO","THREE","FOUR","FIVE","SIX","SEVEN","EIGHT","NINE","TEN","ELEVEN","TWELVE"
};
   while(cin>>curr){
      if(curr==999){
         cout<<"0999"<<'.'<<' ';
         break;
      }
      cout<<curr<<' ';
      if(curr>12)continue;
      string now = names[curr];
      for(auto ch:now){
         mp.insert(ch);
      }
   }
   for (auto ch : mp) {
     cout<<ch<<' ';
   }


            return 0;
            printf("char nums[13][256]for(n=0;n<26;n++)");
}
```

question

Problem Description: Umesh has n mixtures in front of him, arranged in a row. Each mixture has one of 100 different colors (colors have numbers from 0 to 99). He wants to mix all these mixtures together. At each step, he is going to take two mixtures that stand next to each other and mix them together, and put the resulting mixture in their place. Functional Description: When mixing two mixtures of colors a and b, the resulting mixture will have the color (a+b) mod 100. Also, there will be some smoke in the process. The amount of smoke generated when mixing two mixtures of colors a and b is a*b. Find out what is the minimum amount of smoke that Umesh can get when mixing all the mixtures

together. Constraints: 1 <= n <= 100 Input Format: There will be a number of test cases in the input. The first line of each test case will contain n, the number of mixtures, The second line will contain n integers representing the initial colors of the mixtures. Output Format: For each test case, output the minimum amount of smoke.

answer

```
 #include<stdio.h>


typedef long long unsigned LLU;


LLU min_smoke[100][100];

int color[100][100];


LLU smoke(int n){

   int i,j,l;

   for(i=0;i<n;i++){

      for(j=0;j<n;j++){

         min_smoke[i][j]=10000000000000000;

      }

   }


   for(i=0;i<n;i++){

      min_smoke[i][i] = 0;

   }


   for(l=2;l<=n;l++){

      int e = n-l;

      for(i=0;i<=e;i++){

         int k = i+l-1;

         for(j=i;j<k;j++){

            LLU sm = min_smoke[i][j] + min_smoke[j+1][k] + color[i][j]*color[j+1][k];
```

```
            int cl = (color[i][j]+color[j+1][k])%100;
            if(sm<min_smoke[i][k]){
                min_smoke[i][k] = sm;
                color[i][k] = cl;
            }
        }
    }
  }




  return min_smoke[0][n-1];
}

int main(void){
  int n;
  while(scanf("%d",&n)!=EOF){
    int i;
    for(i=0;i<n;i++){
      scanf("%d",&(color[i][i]));
    }
    printf("%llu\n",smoke(n));
  }
  return 0;
  printf("scount[100][100]colours[100]");
}
```

question

Question description Malar is a First year student in reputed institution. Although he scored well in many subjects, he did not an expert in Algorithms. But malar's computer examination is scheduled for next week. As

per the blueprint, many questions would come from the Arrays topic.</p><p>He collected previous year's questions. one of the repeated questions is you need to find the pairs in Array with given sum.</p><p>Can you help him ?</p><p>Function Description</p>

<figure class="image"><img
src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAFxARFBEOFxQS
FBoYFxsiOSUiHx8iRjI1KTlSSFdVUUhQTltmg29bYXxiTlBym3N8h4uSlJJYbaCsn46qg4+Sjf/bAEMBGBoaIh
4iQyUlQ41eUF6NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2Njf/AABEI
AsIC0gMBIgACEQEDEQH/xAAbAAEAAwEBAQEAAAAAAAAAAAAAAAABAUGAwIBB//EAEEQAQACAQMAB
QsDAQYFBAMBAAABAgMEBREGEiExcRMUFTIzNEFRUlORYXKBIiM1YqGxwTZDc5LwFkLR8SREg+H/xAA
ZAQEAAwEBAAAAAAAAAAAAAAAAAAQIEAwX/xAAkEQEAAgEDBQEBAQEBAAAAAAAAAAAQIRAzJRBBITFD
EhQSJhI//aAAwDAQACEQMRAD8A1gAAAAAA8zkpHfeseMvsWi0dkxPhIPoAAAAAAAAAAAA558+LTY5yZZ
7xSkfGXzT6nDqqTfT5K5KxPHNQdQAAJmIjmZiPEAefKU+uv5PKU+uv5B6HyJi3dMT4PoAAAAAAAEzx3g
Dz5XHzx168+L0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
o+lG65dv0tKYJ6uTLPHW+ULxVb/tHpXTXVilorlpPee6f0BnND0f1266uauqyqaL/d15mauqmyaqIi/d15mZlY7Ts256DcsfX1E
200czbq27J/ThWUpv2zR1aVyeSr8o61VltXSq2bUVwa7HWs27ItXuL17OxWNmzj5+r1+l0NedTmrj5+E9/0QadjLJt
rtfq+XmP1ms/8Aw59lNNtuW+HLuGbyfU1nmbfU7k0KDcvQN7NTnrW0W30h2Kx1gh1MOeW1PaQ9HHSxkUbiGpmNbPHXeMrEU
bvObaedREZKzMTTLS/Ko6F6Z78x8vXt+b6ugJrSqB3+dq0ptaXCNQoJIEDoaNM2Ze/R5PjPPr+hg2Q96mQhxPpOLtEO1Ywtxg1m
5aTfdu1mSSMeLUR157ot+sxPt/qxWhy6TU6++fd7f8rTXxB7ETx4Ie4pJ0s6kxSlSDtb0HHBuXP7kadmJQBOqi1zYHGUEPFzuwIJ+mEn
2KjMj+fNUF4kVqLAkw0e9bd7BNmw6x/HxzEDsk8q9ITfs2E1x3pr5W2DGzBMDxJx2sTzW9nxxVtmeSN
90TExylavLAffwesZi6tiNNgvT5x4tU1ZTv3szaKB/lvyGoGHAOJ7OxsSzSXtFb+XnPVBWZZs9IM8K6yl0NpGY7uYVrjDqBAASqeHzV+TPx
VY5N5+SYf7z1BXAEzlWlwAXGMhE/j3NWXVJqt3MpOCz1WyN4BDRaR0t3Rm2fD2rlijiYYZXwn/u9RwkhPxIMZGrVv2swVaprPOY8yy
ZADbyPWYZSn4erSCdafbt2P0Ha6mfutfoOmfxVPUhaYbN4bUa1M4NTdCzUsI5EMqJ8lVFh/SkzE6zIQODk94VsbiTHtJ8/JYrqjtYfZGpLhSKUozSkdF2uk
LxrqTuuYVvfuiatkPqX7ptzQT5l1k/vixf9yTM2mMD6WvWTHBFsnLX8lcoP4dP9VymGTMcPyWekgN7tB6i/4j0Q31vAJZrFTbTrLSF2+vFbGv3J2
3BZlNz0LEPfAP/UNUUEGpq7w9GEDMS5uKxqlAyr1mA9eUFqnMNLhgLtnwS/C4XW/Ky6dzrNHWqUflHPfAiaxBm+Iz1D0dQyU7QzUMGlbPjh/0c8pQAzAgIqk
RCpdAUUvsYq6pVfL2TiV0RjNQjN71SQDjpHjPfoFiqfYYU01VNd6y7YHBTuUVb56fl13k7BmSIATJM+6jo9fFO1iTaY6ijJRu8F+KPVB1e
BwGrrVoDhuJKBGhGvMu6ujoEhqWgqMxRe6Nc2tIuXjoVe0srSqvOsq9nDQPq5XzbS1pkgHc4CRqKSdaWUPi9RRs+7Dlqlkiv32X61UJjcV6xVrIuKYiWb9i2hgVf6n
jjMUYc5ZTNMxc7rnXR+VeGRvKGUJlfOE/B7Hu+yT+zsCKiWSatOdXNpUwtAfirKIMwkTaDnZcQlhtbLWKHfpBlNODBQC0Y2zMiCqgv3wQczcH
cUfa1uC8XJS+wEeXS7kSP4/2S5EadvM5QWnDupbnPFRL5JOqtpI/rhOzHqk0Ftsr7RkvmJb83u6RoKQ5GLq3YT7Oq+HOYv/8+DGMVibk7JK0yV/D4HzBGAnOEf4Bz1hmdWwBOjKM9+CcZc6ZUnJAbVrmnHzPn6Mq+6lbG8/dt+wWV2pNUtyYD1SKxJQM8BsgMQ2xhs7gskvXX6NozP0Y2ZYS8Uqt0
6mKoxZFzZOGtI/+v/w5iAHj6/aNiP7g4sGz92uSjQFo+Ogi2//8wYUC/GDnizBeNdMl7c9x2yOlt8NUJv1lQ1Azu3GN5tdcZkMZWyq

WP9ZXfRPSxba8tNTgif7XuyV/SPmvsOlwYPY4cdP1rWIBn960nmPROun+Nerz48xy+dFMMajYtRht3Xv
Mf5Q0mTFjy16uWlb1+Vo5h8xYcWGvVxY6Y47+KxwDAUx5di3C0avS1zY+7+qOyY+cJ2TfdFkmldNtVLW
mY5iatjlw4s1eMuOl4+Vo5c8ei0uK3Wx6fFW3zikcg+6SmOmmp5LFGGto56nHHDsAKTpd/cdv31Ruhf9
35v3tFkx48terlpW9flaOYfMWHFhiYxY6Y4n4VrEArOkW3X3HbprijnLjnrVj5/oy+27jh26ltPr9BXJMT2TNf
6ob5xy6TT555y4Md5+dqxMgzO37lpNfuNcNNqp5G3/uivbHimb1vWPRauNLl0flsHVjmZjs/heYtPhwRx
hxUx/trEPuTDizRxlx0vH+KsSDBbnrNr1eHq6HQ2xZpnviIj/Rodh0Wor0ey4c0TW2XnqxPwiYW9NBpMd
utTTYon5xSEgH59tG4W2PcMs6jDaeYmsx8YbfbddXcNFTUUrNYtz2T8HTLo9Nmt1suDHe3ztWJl0x46Yq
9XHStKx8KxxAPmbFXNhvjvHNbRxLCZ9Nreju5eVwxM0if6bcdlo+Ut883pTJXq3rW1Z+Fo5gGU/8AWc+S
7NJHlP3diXsG47lr7Z/OMf8AZTEzW89nE/KFxG2aGLdaNLh5/ZCTWlaVitaxWI7oiAfnunti0G9XtumC2Sv
WtzExzzPze991ml1lcc6HSeSxU779XjmW6zaTT555zYcd5+dqxJ5ppupFPN8XUjur1I4BR6LXW2/orizVxTk
t3RX+Vbl3nZ9RSZ1G2zGWe+axHf4tlXFjrj6laVin0xHZ+HCdv0drdadLh5/ZAMh0W0ubJuts2OlqYIrMTz8
p+DluGizbNu055wRmwzabRzHMTE/Bu6Y6Y69XHStK/KscQXx0yV6t61tHytHMAxuTf9BbFxh2unlZ+ExH
DTbR5O+hpmx6aNPOSObV447Xaug0lLdaumwxPz6kJAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAABMRPfHIARER3RwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ADhrM04NPN69/wETOIzLuKCdXnmefKWPOs/3bIyzezXhfig86z/AHbHnWf7tjJ7NeF+KDzrP92x51n+7Yy
ezXhfig86z/dsedZ/u2Mns14X4oPOs/3bPWPW56WievMx8pMp9mvC9HnHbr0rb5w9T3JaMgpdRrcts1
oraa1ieIiHLzrP92yMs89RWJxhfig86z/dsedZ/u2Mo9mvC/FB51n+7Y86z/dsZPZrwvxQedZ/u2POs/3bGT
2a8L8UHnWf7tjzrP8AdsZPZrwvxD2/U2z45i/bavx+aYloraLRmARNw1FsGKOp2WtPHPyVfnWf7tkZcr60
UnC/FB51n+7Y86z/AHbGVPZjhfig86z/AHbHnWf7tjJ7McL8UHnWf7tjzrP92xk9mOF+KDzrP92x51n+7Y
yezHC/FJg1uamWvWvNqzPbErqJ5iJS7aepF/j6Pl7dWk2+UcqPLrc97zMZJiPhEIRqakU+r0Z/zrP92z751n
+7Yy5ezXhfig86z/dsedZ/u2Mns14X4oPOs/3bHnWf7tjJ7NeF+KDzrP8AdsedZ/u2MnsxwvxQRqs/3bLfR
Z5z4ItbvjskdNPWi84SAVm46vJTL5PHbq8R28JXveKRmVmcs/51n+7Y86z/AHLIy4ezHDQcnLP+dZ/uWP
Os/wByxk9mvDQcnLP+dZ/u2POs/wB2xk9mOGg5OWf86z/dsedZ/u2Mnsxw0Az/AJ1n+7ZN27VZL5fJ5L
dbnu5MrV162nCzeb5K469a9oiP1elVu1p8pSvPZxzwl01L9lcpvnuD7kHnuD7kKIRll9m3C989wfcg89wfc
hRBk9m3C989wfcg89wfchRBk9m3C989wfcg89wfchRBk9m3C989wfch0x58eX1LxLPO+jtNdTj4nvngyt
XqJmcTC+OeBw1tprpMkxPE8Ja7TiMltXgrPE5K8+L557p/uQohGWL2bcL3z3T/AHIPPdP9yFEGT2bcL3z3
T/cg890/3IUQjJ7NuF757p/uQee6f7kKIMns24Xvnun+5D7XV4LTERkjmVCJyezbhpRx0dpvpqTPfw7JbYn
MZABIAAhbp7r/ADCahbp7r/MDnq7JU4CrzAAAAAAAAGh0/sKeD3b1ZeNP7Cng929WVnqxtZ7L7W/7p
eHvL7W/7peFXlz9ABAAAAACy2fuyLNWbP/AMxZph6OjshXbv6lPFVrTd/Up4qtEsmvvAEOIAkAAAED7
X148Wjr6seDOV9ePFo6+pHgtDX039ec3sb/ALZZ2e9os3sb/tlnZ7yTqfsACGQAAAAAAW+0+72/cqFvtPu
9v3Jh36fenKTcfe7fwu1JuPvdv4JaOo2IoCGAAQACQAAS9t98r4SiJe2++V8JF9PfC6VO7e2p4LZU7t7ang
mW3X2IACHngAA+1pa3q1mX22O9fWrMCcS8gCB10vvOP90OTrpfecf7oFq/YaBG3D3LJ/58UlG3D3LJ/
wCfFZ6V9sqMBV5YBxM90AD71bfKTq2+UicPgTEx3wCAAF7ofdKeCQj6H3SngkLPVptgAFgABC3T3X+YT
ULdPdf5gc9XZKnAVeYJGixVzajqXjs45R07ao5z2n5QL6cZtEJfo/TR8J/J6P03yn8vG40y3tTyUW7u3hXXjP
T1/KR48pabzWs47TU0rjz2rTuhyJmZnmZ5kQyTOZ/AAQ0On9hTwe7erLxp/YU8Hu3qys9WNrO5fa3/A
HS8vWX2t/3S8qvLn6OmDFbPkilXN9raaWi1ZmJj5BGM/q1roMGOInLbmf1nh68y0uSOKcc/pKry5r5rc3
tMy6aGbedUinPf2+A0RqUmcRV81Wmtpr8T21nulwWm7THkqR8eVWOWrWK2xAAOaz2juyLJW7R/zF
kmHo6OyFdu/qU8VWtN39Sniq0Sya+8AHFZaTR4MuCt7xPM/q7+j9N8p/Koi94jiLTEeKft+nvkmMuS1ur
HdHPeIq07Vt+dqRO3afj1Z/KoyxFct617omYhZ7hq/Jx5KnrT3z8lUSrrdsTisACGd9r68eLR19SPBnK+vHi0
dfUjwTDX039ec3sb/tlnZ72izexv+2WdnvJR1P2ABDKPWLHbLkile+ZeY7ZXGg00YcfXvH9U/wCQ6adO+X
m2h0+HF1snPZHb2qq0xNpmscR8ITNfqZzX6lO2lf8ANCmJjvjgW1ZjOKwADiLfafd7fuVC32n3e37kw79P
vTlJuPvdv4Xak3H3u38EtHUbEUBDAJ+h02LNim2Tv5+aA9Re1Y/ptMeEi9JiJzMLf0fp/hE/lVZ6RTNete6J4
hcaKZnS0mZ5nhU6r3nJ+5Mu+tEdsTEOICGUStt98r4SipW2++V8JF9PfC7VO7e2p4LZU7t7angmW3X2IA
CHnidoNFGWPKZfV+EfNBiOZ4aClYphiKx3R2EO+hSLTmXjJnwaaIrMxHyiHPz7T3iY5/iYRvR2XNab5LxWZ
7eHjLt2THXmkxf9Eu031P5H4iXtFr2tEcRM9zydsT2iGKR10vvOP90OTrpfecf7oE1+w0CNuHuWT/z4pKN
uHuWT/wA+Kz09TbKjAVeWLjbYjzWPFTpWDW3wY+pWsTH6kOulaK2zKyvq8FLTW1oiY7+x7w5seaJnH

PMR39ijyWnLlm3HbaV1pMMYNPWvxntlLTpak3t/xF3W1YpSnx55VjvrMvltRa3wjshwQy6tu60yADmvd
D7pTwSEfQ+6U8EhZ6tNsAAsAAIW6e6/zCahbp7r/MDnq7JU4CrzBK0Wprp5tNomefkip2k0MZ8M2tMx
M9w6acW7v8pHpTF8a2SMWbFqaTNe2PjEq+dry89lqzCbo9L5tSeZ5tPelqpOpM4t8VuvwRgzf0+rbthG
TNyyxkzxFZ9WENEsupERacAA5tDp/YU8Hu3qy8af2FPB7t6srPVjazuX2t/3S8vWX2t/3S8qvLn6A9Y8dst
4pXvkRjJTHbJaK1jmZW2DDj0WGb3n+r4y96XS102PsjrXmO2UTU4NXqL8zTivwjmE4a60nTjOMyi6rUTq
MvWnujuhxd8mjz46Te9eKx+rghmt3Z/0ACqz2j/mLJW7R/zFkmHo6OyFdu/qU8VWtN39Sniq0Sya+8AH
EWeDcMWPFWkxbmI47njFt0ZNPFptNbz2vPozLz61RopXUp+xCw4w6rFzxFon4qXUYvI5rU+XcutNhjT4
epzz8ZlUa3JGXVWtXu7kyvr7YmfrgAqyPtfXjxaOvqR4M5X148Wjr6seC0NnTf15zexv+2WdnvaLN7G/7Z
Z2e8lXqfsACGVP27S9efK37o7kzWeWnF1MNeZnvn5IOm18YMUU6nLr6Vj7aWultOKYy97fpr4rXnLXjm
I4fNfpcmXJWcdeYiHfSavznrf08cPOr1kae8V6vPMDp208f/FPMTW0xPfE8Pj1e3Xva3znl5VYBb7T7vb9y
oW+0+72/ctDv0+9OUm4+92/hdqTcfe7fwS0dRsRQEMAfAPgC80PulPBU6r3nJ+5baH3SngqdV7zk/clq1
tkOICGUStt99r4SipW2++V8JF9PfC7VO7e2p4LZU7t7angmW3X2IACHnvsTxaJ+TQ0tE44mO7hnVpoNZ
WaRiyTxMd0z8Uw0dPaImYlHz67NOW0Ut1YieyIT9BntnxTN47Ynh4y7diy3m8WmvPb2OtYxaPFxzERH
+Y7Vrets2n8Vm4Uimqnq9nPaiuupzeXzTf4fByQx3mJtOB10vvOP90OTrpfecf7oCv2GgRtw9yyf+fFJRtw
9yyf+fFZ6WptlRgKPLAEiToMPldREz6te2Vlr83kdNPE/wBVuyHnbsPk9PEz327UHcc3lNRNYnsr2Ja4/wD
PS/7KJzyAhkAAXuh90p4JCPofdKeCQs9Wm2AAWAAELdPdf5hNQt091/mBz1dkqcBV5g7YdVlw+pbs+U
uIJiZj9hPjdMnHbSrnl3DNkjiOKx+iIC86t5/oAOYADQ6f2FPB7t6svGn9hTwe7erKz1Y2s7l9rf8AdLy9Zfa3/
dLyq8ufo94ss4ckXrxMx83gCJx+pvpTN9NPxJ6UzfTT8IQZX8t+UrLr8ubHNLVrxPyhFAUtabTmQBCFntH/
ADFkrdo/5iyWh6OjshXbv6lPFVrTd/Up4qtEsmvvABxSMOtzYY4ieYj4S7xumT40qgA6RqXj8iUnNrc2aOJ
mKx8oRgFJtNvoAIfa+vHi0dfUjwZynrx4tHX1I8Ew19N/XnN7G/7ZZ2e9os3sb/tlnZ7yTqfsACGQAB302qv
put1IrPPzfNRqL6i0WvERMfJxBbunHaACot9p93t+5ULfafd7fuTDv0+9OUm4+92/hdqTcfe7fwS0dRsRQE
MAACXi1+XFjila14j5wjZLzkva88czPPY8gtNrTGJkAFRK233yvhKKlbb75XwkX098LtU7t7angtlTu3tqeCZb
dfYgAIeeAA6VzZK+re0fy82va8/1WmfGXkE5kAEDrpfecf7ocnXS+84/3QJr9hoEbcPcsn/nxSUbcPcsnh/us
9PU2yowFXlgAPUZLxHEXt+XnvAMyAAAAvdD7pTwSEfQ+6U8EhZ6tNsAAsAAIe6e6/zCY558Nc+KaW7p
FLxmsxDPCfba8vM8WrMPnovN9VVcPP8AFfhBE70Xm+qp6LzfVUweK/CCJ3ovN9VT0Xm+qpg8V+EETf
Reb6qvvovN9VTB4r8IInei831Ve8W1260eUtHV+UGExo34WGn9hTwe7erJWIrWIjuh9WejEfmGcy+1v+
6XlaajbZvkm+O0Rz28S4+i831VRh59tK+fiCJ3ovN9VT0Xm+qqMK+K/CCJ3ovN9VT0Xm+qpg8V+EETvRe
b6qnovN9VTB4r8IInei831VPReX6qmDxX4ddo/wCYsnDS6aNNj6sTzM98u6zfp1mtYiVdu/s6eKrX+q08aj
H1ZnifhKuna8vPZaswhm1tO02zCCJ3ovN86novN9VUYcfFfhBE70Xm+qr56LzfVUweK/CEJ3ovN9VT0Xm
+qpg8V+EETvReb6qnovN9VTB4r8IVfXjxaOvqR4K7T7bNckWy2iYjt4j4rKEw1aFJrE5eM3sb/tlnZ72kmOY
mJ+Ksy7XabzOO8cT8J+BJr0tbGFcJ3ovN9VT0Xm+qqMMvivwgid6LzfVU9F5vqqYPFfhBE70Xm+qp6LzfV
UweK/CCJ3ovN9VT0Xm+qpg8V+EFb7V7vb9yNG15ee21YhZafBXBiile39fmmHfR07RbMuqk3H3u38LtD
1mhjUT16zxb/VLtrVm1cQphO9F5vqqei8v1VRhi8V+EETvReb6qnovN9VUYPFfhBE70Xm+qp6LzfVUwe
K/CCJ3ovL9VT0Xm+qpg8V+EFK233yvhLp6LzfVVL0eh83t17Tzb/ROF9PSvFomYTFVu1Z8rS3HZxwtXnJjr
kr1b1iY/VLZqU764ZwXvmOn+3B5jp/twjDL61uVEL3zHT/bg8x0/24MHrW5UQvfMdP8Abg8x0/24MHr
W5UQvfMdP9uDzHT/bgwetblRO2krNtVj47e3lb+Y6f7cPeLT4sPbjpESYTXp5icy6o+urNtHkiO2eEg45jtS
2WjMYZoXttFgtPM44fPMdP9uEYYvWtyoxeeY6f7cHmOn+3Bg9a3KjF55jp/tweY6f7cGD1rcqMXnmOn
+3B5jp/twYPWtyoxeeY6f7cPVdHgrMTGOOYMHrW5NHWa6WkTHbw7gltiMRgAEgAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIuq1tdPMV46
1pFbWisZlKFZ6Vn7Z6Wn7Y5+enKzFZ6Wn7Z6Wn7Zk89OVmKz0tP2z0tP2zJ56crMVnpaftnpaftmTz05
WYrPSs/bSdLra6mZrx1bQJjVpacRKUAOoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAApNxn/8u38LtSbj75b+ESz9RsRQEMAAAAgAEgl7b7
5XwlEStt98r4SQvp74XYCz1AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFfve4Tt23ZM9Y5v3V8QWEzEd8
wRMT3Swei0W57/wBfNOo/oi3HN7T3/om4Nk3nQavF5LPzjm0daaWnsj9Qa8RtZr9NoKUtq80Y4t2RM
xPa66fUYtThrmwW6+O3dMA6CFm3fQ4NV5tl1Fa5eYjq8T8e511ev02hpW+qyxjrbsiZie0EgV+TfNtx463
tqq8W7Y457XXS7no9XS1sGopaK9/w4BLFZbpBtlMnUnVV5+fE8LDFlx5scZMV4vSe6YB7Bluke+5NPqM
VNBqK9kTGSIjukGpFbtW66fV4MNJ1Fb6i1e2sd/Kdn1GHTY5vnyVpWPjMg6Csr0h2u1+rGqr4zErHHkpl

pF8dotWe6YkHpwz6zTaa0Vz56Y5mOYi08O7F9Nv7ywf9H/eQbOtotWLVmJie6YfVRi3vb9JpsOPNqKxe
KxzEczwsNLrdPrKdfTZa5I/QHceMubHgpN8t4pWO+ZlXx0h2ucnU86rz8+J4BZjzjy0zUi+O0WrPdMS9AD
zkyUxUm+S8VrHfMyrp6RbXF+rOqr48SCzmYiJme6HDFrdNmy+TxZ8d7/TE9p5fFqNJfJhvF6zWe2J/Riuj
mbHg36+TLeKUiL8zMg3grKdINsvk6kaqvPdHZPCyraLVi1ZiYnumAfRF1e5aPQ+8560n5d8uel3jQay/Uw
ams2+U9gJwIGq3rb9JfqZtTWLfKO0E9wza3TafJ1M2fHS8xzxaeJc9Huej13Zps9bz8u6WS6X/AN+4/wDp
1/1kG3iYtETE8xPxfXDFkpi0VL5LRWtaRMzPw7HDTbxodXm8lp88ZL/KIkE4cdTqsGkx9fUZa46/OZQ8XS
DbMt+pXVV5nu5iYBZD5W0WrE1mJie6YRdZumj0M8ajPWlvl3yCRlzY8GOb5b1pSO+bT2PODUYdTWbY
MtckR3zWeVBv26aPXbJljTZ63tExzHdJ0K9wzfvgGkBDtu2hrq/NZ1FYzd3V/UEwABSbj75b+F2pNx98t/CJ
Z+o2IoCGB6pScmStK98zwlejc/8Ah/LjpPesf7oW+qzzgxdeI5nlLvpUraszZXejc/8Ah/J6Nz/4fy7Y9yyZMla
RjjmZ471hkvGPHNrfCDDrXT07RmGevWaXms98TxL49Xt172t855eUMk/RK233yvhKKlbb75XwkhbT3w
uwFnqAAAAAAAAAAAAAAAAAAAAAAAAAAAACFu+3xuWgvp+t1bT21n5Smq/e9dl0G32zYKTfJzER2cx
AMpTS75sl7Rgrfqc8z1I60SnaLpdkrljHr8MRHPE2r2THi64OmOKcfGp01ov8erPPKl3PUzvm40800015/p
7Pj+sguemlI65NFo70nmtrTMT+nC16N/wBx6fwV3SLbss7Fpq0ib208R1ojwV20dJfR+i82yYJvNfVmJ4/gH
Lev+K4/6mP/AGWnTT3HTfun/Znb58uq3vHnzV6tr5azx8o5jhoumnuOl/dP+wImxdHMOv0HnOpvfm3Z
WI+Cnz6PJp91vosd5iZv1OeeOYn5/ltOjH9x4f5ZnXf8YR/16f7An7h0W0+m2u+bHkvOXHXrTz3S99CdRe
aZ8EzM1r/VEfJe7v8A3RqP2M70I9vqf2/7g189zCdK9vxaLWY74pnnP1rW5+fLdsr010+S8afPWszSnNZ4
+HIJewbJpsOPT66s38rNee2eztUe75s27dIPNIvMUi/UrHw8VtsfSHBfHptDbHaMnqc/BW75o9Rtm8+fY
qTalr9eJj4T8gWep6JaTzO0YbXjLWvMWme+ULojrcuPW30N7TNJiZiJ+Ew66jphXJo7Uxae0ZrV47Z7IfOi
W25o1F9fmrNYmJivPx57wa1ium3954P+j/vLasV02/vLB/0f95BO27orpcmhpfU2vbLevPMT3KjD5XYek
UYa3ma9eKz+tZ//AMlYaHpV5ro64dVp7TkpHETHxQdBh1G+77GpvSYpF4tafhER3QDp0p199RuUaWbz
TDj45j9fm9WxdHPNJpXNfyvHZk4nv8HvpTtmXFrY1uKnXxW4636S+Yt72ryMeW2yvleO2I7pkHrohrb01
99H15tjvEzXn4cNko+juXTayls+PQ10+SkzHMR2TC8Bi+lesy6jdK6GlpilerHHzmf/ALWOLojo50kVva85Zj1
+e6ULpXt2emtjcMFZtWeOtx8Jh1x9MKV0kVvp7eWiOO/skFjtOz32rS54vqJydes/0x6sMjt2g9JbvOn601
rNrTaY+UNXsm5azcdFntqsMVrWs8Xjs5/hR9F/+IreFwd9+6O4NBofONPe3NZ4tFvimdHdwvj6O6jJaZtO
n56vPhzCd0p/uTL4wreiuCNTsutwz2eUtNefl/SCk0OTSazXZM275rdXv47Z5dN2ja6dTLtWW0Xie2vb2fr
zLzgidm3C9NfpIzU7uLf6wsMm87Xa9K6fbK35mOeY+AJWp3nN/wClceWtpjNf+zm3/n6IvR/YMO46a2q
1drWi1piIif8ANc7ttddZskY9JijHaP6604/yUWzb9bZ8VtJqsFprFuY+ExILTB0Wrptypnwam9MdZ5iI7+fl4
Knph2b5SZ+GKv8ArKx0nSTV67dK49NpecE9kx8Y/XlXdMI53ukfPFX/AFkHHc95y7nOLSYreTwRxXtnjn9Z
azZdpw7Zpo6vFsto5tf5qLcOjNabTjz6Tm2Wletf/Ek9Ft6nLWNDqrf119S0/H9AU256yu4b5MarLNNPW/
Vjj4Q76/H0fto7eaZb0zVjsnif6p/l43jQZdr3edROGMuC1+tHMdk/pKTfe9p8j/RtlZy8d093IO/Rrdclds1d
MkzbzevWrM/qrNm0Ft93HJfVZLdWP6rT8Z/Rp9uw6bXbRe+LSxp5zV6to4ZjQarUdHdyyVz4pms/02j5x
84BN6QdHsGh0XnOlm0RWeLVmeU7oV7hm/fCt3vpDO56TyGnw2rj772lZdCvcM374BpGby9FpybtOqj
URGKbdaa8drSMrm6Saym9zp4xV8lF+r1eO2QaoABSbj75b+F2pNx98t/CJZ+o2IoCGB9iZieYniY+L7bJe0
cWvaY+Uy8vsRNpiI75E5n4n7Xh5vbLMdkdkOu6ZurjjFE9s9spOnxxg09a93Ecyp9VlnNntb4d0Jarf+enj+y
4glZBK233yvhKKlbb75XwkX098LsBZ6gAAAAAAAAAAAAAAAAAAAAAAAAAAAATETHE9sACHl2nQZp5v
pMUz8+pDrp9Hp9L7DBjx/rWsQ7gExz3ok7XoZyeUnSYet8+pCWA4X0OkyZYyX02K1444tNImezues+m
waisVz4aZYjui9Ynh1AeMWLHhpFMVK0pHdWscQ520Oktm8tbTYpy889eaRzz4u4DzelclJpesWrPZMTH
MS54NJp9NMzp8GPFM9/UrEcuwA83pXJSaXrFqz3xMcxL0Ai4tt0WHJ18elxVvHbExSOYSL0rkrNb1i1Z7
4mHoBErtehrfrV0mGJ+fUhLiIiOIjiAAcc+j02ptFs+nxZbRHETekTxDsAjZtu0ee0Tl02K0x8ZpDtiw48FOphx
1pX5VjiHsB8msWiYtETE/CUS21aC1+tOjw8/shMAeaUrjrFaVitY7oiOIegAmImJiY5ifgiTtegm/WnR4Zn59
SEsB5ila06laxFeOOIjscsWi0uDJ5TDp8VL/VWkRLuA8ZcOPPSaZsdclJ/9to5h8wafDpqzXBipirM8zFKxHLo
A55tPh1FernxUyR8rV5csO3aPBfrYtNipaPjFI5SQBHz6HS6m3Wz6fFkn52rEykAOWDS4NNExgw0xxPf1a
xDzm0Wl1GTr5tPiyX4461qRMu4D5FYivViIiI7OEeNv0dcnlK6XDF4nnrRSOeUkB5vjpkpNclYtWe+JjmEau
1aCt+vGjwRP7ISwCIiI4iOIhxz6XBqY4z4aZIju61Yl2ARqbfo6Y5x102KKW746kdrpg02HTVmuDFTFE98Ur
EOoA4To9NOo8vODHOX6+rHP5dwAABSbj75b+F2pNx98t/CJZ+o2IoCGAd9HOKmeLZrcVjt7nAExOJytN
XrsVsE1xW5mezuVYC17zecyACglbb75XwlFStt98r4SL6e+F2As9QAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAUm4++W/hdqrcdNknN5Sl
ZtE/JEs+vEzT8V49+Ry/bt+DyOX7dvwhh7Z4eB78jl+3b8Hkcv27fgO2eHge/I5ft2/B5HL9u34Dtnh4HvyO

X7dvweRy/bt+A7Z4eEvbffK+Eo/kcv27fhN23TZK5vKXrNYj5wQ6adZ74WoCz0gAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADiDiAA4g4
gAOIOIADiDiAA4gAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEPcc9sOKIp2
Tae9VTnyz/AMy35T939WnirFZYNe09735fL9y35ffL5fuW/LmDhmXTy+X7lvyeXy/ct+XMDMunl8v3Lfk8
vl+5b8uYHdLp5fL9y35PL5fuW/LmB3Sk6bVZaZa83mYmeJiV58Gcx+0r4w0VfVhMNnTTMxOXy9urSbfK
FFk1WXJebTeY/SF5m9jfwZ0lHUTMYh08vl+5b8nl8v3LflzEMmZdPL5fuW/J5fL9y35cwMy6eXy/ct+Ty+X
7lvy5gZl08vl+5b8nl8v3LflzAzLpGfLE8+Ut+Vzoc1s+ni1u+OyVEuNq91nxTDR09p7sJoCW4AAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABVZukW3YM18WTJaL0maz2fEFqKivSbbLTx5aY
/WYWeDUYtTjjJgyVvSfjEg6AADxnyTiwZMkR1ppWbcfPiGd2TpBq9fuc6fPhrFJiZiaxxNfEGlAAAAAAA
BW7v6tPFWLPd/Vp4qxWXna+8AHEErQaemotaL88R8k30bg/xfkda6VrRmFQLf0dp/1/KqyRFclojuiRF9
OafXkAc3rH7SvjDRV9WGdx+0r4w0VfVhMNnTf15zexv4M60Wb2N/BnSVep+wAIZQddPp7ai/Vr2cd8r
Gu34Mcf2luZ/WR0rp2tGVSLa234Mlf7OeJ/SVbnw2w5Jpb8hfTtX65gDmLjavdp8VOuNq92nxId+n3poC
z0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB+eZcFNT0myYckTNL6i0Tx4v0
N+cazNkwdIM+XDXrXrntMRMd/aDUZ+iW32xTGKL478dk9aZUnRrU5dHvcaSLc0yTNbR8Oz4/wCTtl33e
9RScdNP1Zt2c0pPKX0a2LPg1PnmtrNLR6lZ7+fnIJe8dIcmg1fm2DS2yZOOeZ7p8OFdXpdqsWSI1WjitZ8Y
lL3Tf6Ydx830mlrnz1/p60xz2/KFZvmo3LUaKJ1mgphpFo4vFZiY/QGuwa3FqtD51gnrVmszH/wpNj3nz3d
b4I0mHFzEzN6R2zwdFZmdhzxM9kdbj8Sq+in/ABBf9tgW28dI8u27n5tGClqRFZm0zPPa+aDpLk1+6U0+
LT9XDafWnvU/SqvX6RdWe6a0hsdDpMGm0uKuLFWvFY7eO0FJuHSqceqtp9Dp/LWrPHPb2+HDnpuk2
unUY8Wo0Ex17RWOyYnt8UHLtG6bVuFtRo8U5o5niYjns/VJw9KM2LPXHuWjpWYnvn5dHvcaSLc0yTNbR8Oz4/wCTtl33e
wrt73Odq0Xlq4+va1urEfBYUtF6VvXutHMOep0+HVYZxailb457O723bTXvfHFL0txPHdK1cNHpdPp
MPk9LjrSnf/S7gAArd39WnirFnu/q08VYrLztfeADis9pr/TknwedfGec/9l1+rx8HPQ6zHp8U1vE8zPPYlek8
M98W/CWus1nTiJnCttfPSf6rZI8Zly71/HktTi5iItWVLqcXkM9qfD4ZmsZYAHF6xvdpjylI+PCTSuLQYObzt
+MNFX1YTDZ039ec3sb+DOtDn9jfwZ4lXqfsACGV7xZb4ak8tsptpte0zLW94sk9tpte0zLw94sFomfkJW2T
fzjiJ/p47Xvdpjyll+PCTSuLQYObzaf85VWfLbPlm9vj3zvj3fol3v/Al0+2frmAhmFxtXu0+KnXG1e7T4kO/T700
BZ6AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAxtdn1//qOdROmt5Hy8263
McoctkAcR8gAZDdNl1+n3adbt9fKc260cd8S867T7/uulmNRiitaTzGOOIm0tiAoujuh1Ok2jNiMi2GKaZLc8V
mY7exA6O7VrtJvNs2z09a9Oa2/qmYawBkt/2rXKarFz4NPa+Rj4Tgj0fM2i0TasTx1o+S2AU
nRfR6vR6G9dXE15tPtWkzzzxC7AAAFbu/q08VYs939WnirFZedr7wAcU/Dt3ldPF5txae55nbM3PfWf5c8O
uzYY6sT1q/KXf0rbjtxyloj6XB5vh6szznPtKq12SMmptNZ7Z7I/Dg9Y/HrtNr8uWvVvMWp8u/Vj+mP0RUI1dSJjtqADg9Y/
aV8YaKvqx4M7j9pXxoq68Z/Y38GeaLN7G/gzpKvU/YAEMoG3i5Ioxr9Gx1n3PwVh6S/8GzpKvU/YAEMrphw2z5nIpTvXGLBXTYZ8nXrW/
wBVRp89tPebViPbik5j4pPPcFd+tak/RVy1GtvqaqMfU1ta/FpUDL6QH8/DhxzNbv91NT2hp91UNfGKDS7RPbmzVHQWGUz24b7nuP
nnsFb9k5nP6jADiLjavdp8VOuNr91/lMO/T700BL0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFbu/q08VYs939WnirEPO194AhxAAAAAesftK+MNFX1Y8
Gdx+0r4w0VfVhMNnTf15zexv4M60Wb2N/BnSVep+wAIZQAAAAABcbX7r/KnXG1e6/wAkO/T700BZ6A
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADjqdPX
UY+rbs47YlB9E2+7H4WgOdtKtpzKr9E2+7H4H/aeibfdj8Cir9E2+7H4H/AGnom33Y/C/wC1aBg8FFX6Jt92
P+09E2+7H/atAweCir9E2+7H/aeibfdj8Gd8LQDwUVUIx7bV/2WTFpjMAeszTk/W47Ydx/ZfTxj+0Ds/BERpZ/
EfKY5WQItStvqr9E2+7H4PRNvux+FoBg8FFX6Jt92P+09E2+7H/atAweCir9E2+7H4H/AGnom33Y/C/wC1aBg
RNvux/2nom33Y/7VoGDwUVcbTPPbl7PBY4cVcOOKV7oewWrp1r+wADoAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA55s1MNOveeIRPSuL6LPO7T/RSPhy
q0ZZNXWtW2IW3pXF9Fj0ri+iypEZcvYutvSuL6LHpXF9FlSGT2Lrb0ri+ix6VxfRZUhk9i629K4vosel
ZPYuusO4Yst4r21me7lLZzHPGWvHzhoo7TDTo6k3ckCFk3LFS81yzGxuQ9L1+i+OXS3uYhJjbXvwSjiS
x6VxfRZUhk9i629K4vosel7lcX0WVIJF1vtMfkbt6udxbdjbtXf0U+q02ZW6dtkRXFTXQ0jWsVmI5p2pVtKRMXTmrCmNFtd
rkpFqTzEs4udrnnS+EmXbR1bWtiUwBLUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAArd29WnirFnu3q08VYrLztffIAOIOtdNlvWLVxzMT8X3zTP9u34Fu23Di
O3mmf7Vvw4zExPE94iYmPoAIesftK+MNFX1Y8Gdx+0r4w0VfVjwTDZ0vyXjP7G/gzQ5/Y38GeJV6n7AAh
lAAB1rps1qxMY7cT+jnas1tMT2TAmYmPr4AIFxtXu0+KnXG1e7T4kO/T700BZ6AAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACt3b1aeKsWe7erTxVisvO198
gA4rDT7hjw4K0mtpmISMOurmvFaY7f/CnXeiwVxYKzHrWjmZS1aN72nH8dM+auDFNrf8A2obW61pt8
5Wev02bLPXi3WrHdWFX3Eq9RaZnAAhnesftK+MNFX1Y8Gdx+0r4w0VfVjwTDZ0vyXjP7G/gzzQ5/Y38G
eJV6n7AAhlEvQaac2TrWj+iv+bhgw2z5YpX4/Fcz1NJpvlFY/I76VM/6n4563URp8XVpx157v0U8zzPMpW
LFbX5b2tbiXPVaedNeKzbnmOUmr3X/wBfxwAQ4C42r3afFTrjavdp8Uw79PvTQEvQAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAVu7erTxViz3b1aeKsVl52v
vkAHF66lur1urPV+b3XU5qxERktEQstNqNPbDXHzEcR2xZ18lpeeerj/AMktNdL8zWz7pMl8unra8dsqnW
1iuqvFe7lZ59Ziw04rMWt8IhT3vOS82t3zPMkp1pjtiv8AXkBDK9Y/aV8YaKvqx4M7j9pXxhoq+rHgmGzpf
kvGf2N/Bnmhz+xv4M8Sr1P2ABDKtttpWuDr/wDume9IzYceeIi8936qOL2iOItMR4nlL/Xb8py011oiuML
zBp8eCZ8nHe+Z9NizT1skdsQh7bliIv5S/hzLzuWXnLXyd+Y4+EjtOpXszhCyViuS0R3RLyd4hgFxtXu0+KnX
G1e7T4ph36femgJegAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAArd29WnirFnu3q08VYh52vvAEOI+8z85fAAAAAHrH7SvjDRV9WPBncftK+MNFX1Y8Ew2dL
8l4z+xv4M80Of2N/BniVep+wAIZQAAAAABcbV7tPip1xtXu0+JDv0+9NAWegAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAgbpitfFW1Y56s9qpaXh48lT6I/C
MM+pod85yzo0Xksf0R+DyWP6I/Bhz9aeWdGi8lj+iPweSx/RH4MHrTyzo0Xksf0R+DyWP6I/Bg9aeWdGi
8lj+iPweSx/RH4MHrTyotPjtlzVikTPb2/o0ERxEQ+VpWvqxEPqXfS0/HDzkr1sdo+cM9kpbHea2iYmPm0b
zNK276xKEaul3s4NF5Kn0x+DyVPPpj8GHH1p5Z0aLyVPPpj8HkqfTH4MHrTyzo0XkqfTH4PJU+mPwYPWnl
nRovJU+mPweSp9Mfgwj1p5Z2I5niF3t+K2LTRF44mZ54d/JUj/2x+Hsw66Wj2TkAS0AAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA85MlcVJteeIhF9J6f52/A
rN6x9lMEP0np/nb8HpPT/ADt+BXyU5TBD9J6f52/B6T0/zt+A8lOUwQ/Sen+dvwek9P8AO34DyU5TBD9
J6f52/B6T0/zt+A8lOUwR8WtwZr9Wtu39YSBeLRPwHLU5owYZv8fgqp3LPM9kxH8DnfVrScSuhS+kdR8
4/B6R1Hzj8Iyp7FF0KX0jqnPnH4PSOo+cfgyexRd9I6j5x+DJ7FF0KfHuWaxO+cfgyexRd9I6j5x+DJ7FF0Kf
HuWaLx1+Jr8exb1tFqxaO6Y5S6U1K3+PoA6AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAK7d5nqUjns5Vaz3f1aeKsVl52vvABxAAAAAAesUzGWkx84aKvdD
O4/aV8YaKvqwmGzpvkoe6e6/wAqdcbp7r/KnJc+o3gCGcAAAAAAaHT+74/2wzzQ6f3fH+2Ew1dN9l0AS
2gAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAK3d/
Vp4qxZ7v6tPFWKy87X3gA4vWPHfLbq0jmXXzLUfbk0meNPl68xz2cdi20uqjUxaYrMcfNMO+nSt/s/qlyY
r4piMlerMvCduvvvFf2oKHO9e22AAUesftK+MNFX1YZ3H7SvjDRV9WEw2dN/UPdPdf5U643T3X+VOS59
RvAEM5ETMxERzMp2LbL2jnbq7Yreo97bfwVcrtPdaXL+sU0006xOzOy880lbmz/+UKXmkx/mHgAcRodP7vj/bDPNDp/d8f7YTDV032XQBLaAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAArd39WnirFnu/q08VYrL
ztfeADiLTafUv4qtZ7T6mTxIdtDe5br7xX9qCnbr7xX9qCSjV3yADk9Y/aV8YaKvqwzuP2lfGGir6sHs6b+oe
6e6/yp1xunuv8qcIZeP7Smhwc9yg4y0Wc73c6+SuO9j5Wj6t6LDGV0ZLTxEzJxWasPnpLptlcGSDV6lZbodP
/d8f7YTDV032XQBLaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAArd39WnirFnu/q08VYiXna+8AQ4ix2zJTHS/XvWvM/GVcC9Ldk5TNyvW+es0tFo6vwlDARa
3dOQAVesftK+MNFX1YZ3H7SvjDRV9WEw2dN/UPdPdf5U643T3X+VOS59RvAEM71jyWxXi1J4mFjj3W
vHGSk8/OFYGV66lq/Fpk3SnH9FJmf155eATbUtf6ADmNDp/d8f7YZ5odP7vj/bCYaum+y6AJ
bQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFbu/
q08VYtd1x2tjraI5iJ7VUrLztfeADiAAAAAA9Y/aV8YaKvqwz+Gk3zUisczQmGzpvkoe6e6/yp1/qsPn
GGac8T8FTOg1ETx1eSVdelptIRhI8x1H0mHoO+hGDx2x24ePRlx0h0/u+P9sKnFt2e1460RWvxlc0r1KRWPhHCYaenpNczL6AlrAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAJiJjiXKdPhn/AJdf
w6giYiXLzbD9uv4PNsP26/h1BHbDl5th+3X8HgdsOXm+H7dfweebYft1/h1BHbDl5th+3X8HgdsOXm+H7d
+3X8OoHbDxTFSnqViPCHsBMRgAEgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIOt104L9SkRM/GZRPSef8Aw/h53H3y38Iqrz9TVt3T+pn
pPP8A4fwek8/+D8IYZU8t+Uz0nn/w/g9J5/8AD+EMnlvyvyymek8/8Ah/B6Tz/wfg9J6j/B+EMnlvyuyuNDrZ1FppeIi3f2fFNUu2+9x4SulobdG02rmQAdgAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAeL5sdJ4vkpXxtEPtMlL+petvCeQegAAAAAAAAAAAAAA
Um4++W/hFStx98siqvL1N0gPtI6161+c8Cj4LONqrMe0n8Hoqv3J/Bh28F1YJ+fbq4cNr9eZ4j5IA52pNZxI
AKpe2+9x4LpS7b73HhK6TDf0+wAS0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAACi6Sb1bbcdcOnmPL5I55+mPmvWC6U8zv14yerxXjw4gHvR7HuW7084zZ5pW3bFsk
zPP8Os7LvG16nHOky2yRM8c0ns/mGx08VjT44px1erHHDoDnS1senrbU2rFor/AFz3Qq83SfbMWSaTm
m0x3zWszCv6Z6zJjx4dNSZrF+bW4+L3s3RzRZNtxZtTWcmTLXrd/dyC70W5aXX1mdNli/HfHdMfw57ju2
k2yaRqr2r5Tnq8Vme5jtXjtsO/RGnvPUrMTH6x8lh01nreZW+dbT/oDV6fPTU4KZsU80vHMTMINt90Ndf
5nOS3lut1eOrPe6bJ/c+l/wCnDI5/+Mf/AO8f7A1+4brpdt6nnV7V6/dxWZRs/STbcNazOabTaOYiscz/ACq
em/8A+r/L1snRzSajbaZ9VFrXydvZPHEAvNv3fR7jzGmy82jvrPZKo6Ubz5vWmDR55pnrf+uIj4cKOmKdr6
SxjxWnimTiP1hadMdDgxUpq6VmMuTJxaef0BN2LfcGXS4sOpzzbU2tx2x3tAzPRrZ9Jm0ODWXrM5otzE
8tMCHuG56bbcdb6m/V63dERzMu2l1WLWaeubBfrUt3Sgb3std3pj/tZx3pPZPHKVtmgptuipp6Wm3V7
ZmfjIJYAKTcffLIqVuPvlkVV5epukImYmJjvgBRIjWaj7k/hZ6OM3k+vntPM/CfgqNPNa6ik39WJ7VzXVYLz1
YyRMz8Ew16M5/ZlA3DWeUmcWOf6Y75+aCtNZoaeTm+KOJjt4+arRLlrRaLf6ABxS9t97jwXSl233uPBdJ
hv6fYAJaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABnulGzX
11K6nTV5y0jia/VDQgMPt/STV7bijTajD14p2R1uyYdb9JNy3DUY66HDNYi3q1jnnxa3LpNPmnnJhpafnN
XrFp8OH2WKlP2xwCh6SbZn3DQYc9Kf2+OP6qR8VZt/SfLt+krpc+nm0444rz2Tx+rauNtLp72618OO0/Oa
wDFaTTarpBu/nGak1xdaJtPHZER8IXHS3b8mo0eHLhrNpw8xMR38T/8ATQ1pWkcUrFY+URw9TETHaD
D7d0nz6LR10s4IyWr2Vn4oOjtlv0hw31ETGS2WJtEv0DzTTxfrxgx9b59WHryGKb9ecVOt9XVjkGX6b/8A
6v8AK82H+5tN+3/dNvix5ePKY6347utHL1WsVrFaxERHdEQDCbp/xXb/AKkL/pdpcup2ytsVZtOK/WmI+
XcuZwYbX684qTb5zWOXSY5jiQYvZOkUaHTY9HkwzM9bjrR8IbRxjSaeL9aMGPrfPqw7Aoek+6arbseLzX
+nr995jlP2TV5tdtmPPqK8Xns7uOf1TcmLHlrxkpW8fK0cvtaxWsVrEREd0RAPoAKTcffLIqVuPvlkVV5epu
kfaVm9orXvmXx6x3tjvFqzxMCkf9Ssm3Zq8TTi0POLQ55yRzTjie+Zd8e6TEf2lOfB0ndKcdlJS0xXS+5S8tox
4LTaeyKs/PekanWZNR2T/TX5QjoU1tSLz+AA4pe2+9x4LpS7b73Hgukw39PsAEtAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACk3H3yyKlbj
75ZFVeXqbpABQAAAAABL233uPBdKXbfe48F0mG/p9gAloAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAVuv0WTJl8pjjnnvhE8x1H25XojD
hbQracqLzHUfbk8x1H25XoYV9aqi8x1H25PMdR9uV6GD1qqLzHUfbk8x1H25XoYPWqovMdR9uTzHUf
blehg9aqv0GjviyTkydk8dkLAEu1KRSMQAC4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD//2Q=="></figure><p>Constraints</p><p>0
<n<100</p><p>0<arr<1000</p><p>Input Format </p><p>   -
First line contains the value of n that is the total number of elements in the
array
    -Second line contains the elements of
array
    -Third line contains the Sum to be checked.</p><p>Output
Format
      -Output contains as many lines
as number of pairs with each pair written in each line
    -Last line of output
contains the total count of pairs.</p><p> </p>

answer

```
 #include <bits/stdc++.h>

using namespace std;

/*int getPairsCount(int arr[], int n, int sum)

{

	return 1;
```

```
}*/
int main()
{
   int n;
   cin>>n;
          int array[n];
          for(int i=0;i<n;i++)
          cin>>array[i];
          int sum;
          cin>>sum;
          int count = 0,i,j;
          for(i=0;i<n;i++){
                    for(j=i+1;j<n;j++){
                              if (array[i] + array[j] == sum){
                                        count++;
                                        cout<<"["<<array[i]<<" "<<array[j]<<"]"<<endl;
                              }
                    }
          }
          cout<<"Total Number of Pairs:"<<count;
          //getPairsCount(arr, n, sum);
          //return 0;
}
```

question

<p>Problem Description:
Good news! Suresh get to go to America on a class trip! Bad news, he don't know how to use the Dollar which is the name of the American cash system. America uses coins for cash a lot more than the Kuwait does. Dollar comes in coins for values of: 1, 2, 10, 50, 100, & 500 To practice your Dollar skills, suresh have selected random items from Amazon.co.us and put them into a list along with their prices in Dollar. Suresh now want to create a program to check suresh Dollar math.

Suresh goal is to maximize your buying power to buy AS MANY items as you can with your available Dollar.

Input Format:
File listing 2 to 6 items in the format

of:

ITEM DDDDD
ITEM = the name of the item you want to buy
DDDDD = the price of the item (in Dollar)

Output Format:
Print the output in a separate lines contains, List the items suresh can afford to buy. Each item on its own line. Suresh goal is to buy as many items as possible. If suresh can only afford the one expensive item, or 2 less expensive items on a list, but not all three, then list the less expensive items as affordable. If suresh cannot afford anything in the list, output "I need more Dollar!" after the items. The final line you output should be the remaining Dollar he will have left over after make purchases.</p>

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{   int money,n;

    cin>>money>>n;

    int price;

    string name;

    map<int,string> mp;

    map<string,bool> mp1;

    vector<string> vecs;

    for (int i = 0; i < n; i++) {

        cin>>name>>price;

        vecs.push_back(name);

        mp.insert({price,name});

    }

    price = money;

    for(auto pr:mp)

        if(pr.first<=money){

            money-=pr.first;

            mp1[pr.second] = true;

        }

        else

            mp1[pr.second] = false;
```

```
    for(auto s:vecs)
        if(mp1[s])
            cout<<"I can afford "<<s<<endl;
        else
            cout<<"I can't afford "<<s<<endl;
    if(price!=money) cout<<money;
    else cout<<"I need more Dollar!";
            return 0;cout<<"char name[MAX][LEN];int price[MAX]afford[MAX]for(i=0;i<items;i++)";
}
```

question

Question description Professor Shiva decided to conduct an industrial visit for final year students, but he set a condition that if students received a passing grade in the surprise test, they would be eligible to go on the industrial visit. He asked the students to study a topic linked list for 10 minutes before deciding to conduct a surprise test. Professor-mandated questions, such as the deletion of nodes with a certain data D, are now being asked.

For example if the given Linked List is 5->10->15->10->25 and delete after 10 then the Linked List becomes 5->15->25.
  Constraints 1< N < 100 1< D < 1000
Input Format
First line contains the number of datas- N. Second line contains N integers(the given linked list).   Next line indicates the node data D that has to be deleted.
Output Format
Single line represents the linked list after required elements deleted.

answer

```
#include <bits/stdc++.h>
using namespace std;
void mandatoriesSuck(){
    cout<<"struct node node *next;void create()p2=p2->next;void del()";
}
int main()
{
    int n,t;cin>>n;
```

```
    int arr[n];
    for (int i = 0; i < n; i++) {
        cin>>arr[i];
    }
    cin>>t;
    cout<<"Linked List:";
    for (int i = 0; i < n; i++) {
        if(arr[i]==t)continue;
        cout<<"->"<<arr[i];
    }
            return 0;
}
```

question

Question description Varman's Dream came true after he got an Appointment order from Google.Simon's family was very happy of his achievement. The company mentioned Basic Salary, DA, HRA with some other benefits. But not highlighted the Gross salary in the order. varman's father wanted to know the Gross salary of his son. varman try to his gross salary from HR department. they informed that you have to get pass grade in first month entry test. the entry test has 5 questions. one of the question was, Sorted insert in circular linked list. Can you help varman? Function Description First case one is if linked list is empty then since new_node is only node in circular linked list, make a self loop.and change the head pointer to the new_node pointer. Second case is new node insert in starting or before the head node. A- Find out the last node using a loop .
     While(present->!=*head_ref)
     present=present->next;
B- Change the next of last node;
     present->next=new-node;
C- Change next of new node to point to head.
     new_node->next=*head_ref;
D- Change the head pointer to point to new node.
     *head_ref=new_node; Third case is when we insert the new node after the head in any position,then A- Locate the node after which new node is to be inserted.
      while(present->next!= *head_ref &&
      present->next->data data)
     { present = present->next; }
B- Make next of new_node as next of the located pointer
      new_node->next = present->next;
C- Change the next of the located pointer
     present->next =

new_node; Constraints 0<n<100 Input Format:
The First line of the input represents the number of elements Second line represents the elements of circular linked list
Output Format:
single line prints the results as per sample test cases

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main()

{

   int n;cin>>n;vector<int> v(n);

   for(auto &el:v) cin>>el;

   sort(v.begin(),v.end());

   for(auto el:v) cout<<el<<' ';

          return 0;

          cout<<"struct Node *next; void sortedInsert(struct Node** head_ref, struct Node*
new_node)";

}
```

question

Question description sanam's Dream came true after he got an Appointment order from Google.Simon's family was very happy of his achievement. The company mentioned Basic Salary, DA, HRA with some other benefits. But not highlighted the Gross salary in the order. sanam's father wanted to know the Gross salary of his son. sanam try to his gross salary from HR department. they informed that you have to get pass grade in first month entry test. the entry test has 5 questions. one of the question was, Split a circular linked list in two halves. you have to split the circular Linked List with the same size of Divisions.Maybe if circular Linked List is odd, you have to change the number of node, it is even . Can you help sanam? Function Description First count the number of node in Circular Linked List. Second, you have to make the List even. Third, you need to make the List half. the front is the same size like the rear. Finally, I have to make two circular Linked List. Constraints 0<n<10 Input Format:
The First line represents the number of input in the circular linked list elements

Output Format:
First Line indicates the complete linked list second line indicates the odd list third line indicates the even list Refer sample test cases.

answer

```
 #include <iostream>
using namespace std;
struct n
{
    int data;
    struct n *next;
} * odd, *even, *h = NULL, *tt;
void insert(int data)
{
    n *p = new n;
    p->data = data;
    p->next = NULL;
    tt->next = p;
    tt = p;
}
void oodd()
{
    cout << "Odd:\n";
    odd = h;
    int i = 1;
    cout << "[h]";
    while (odd != NULL)
    {
        if ((i % 2))
        {
            cout << "=>" << odd->data;
        }
        i++;
        odd = odd->next;
```

```
    }
    cout << "=>[h]";
}
void eeven()
{
    cout << "Even:\n";
    even = h;
    int i = 1;
    cout << "[h]";
    while (even != NULL)
    {
        if (!(i % 2))
        {
            cout << "=>" << even->data;
        }
        i++;
        even = even->next;
    }
    cout << "=>[h]";
}
void display(struct n *h)
{
    cout << "Complete linked_list:\n[h]";
    while (h != NULL)
    {
        cout << "=>" << h->data;
        h = h->next;
    }
    cout << "=>[h]";
}
int main()
```

```
{
   int a;
   cin >> a;
   tt = new n;
   tt->data = 1;
   tt->next = NULL;
   h = tt;
   for (int i = 2; i <= a; i++)
   {
      insert(i);
   }
   n *y = h;
   display(y);
   cout << "\n";
   oodd();
   cout << "\n";
   eeven();
   return 0;
}
```

question

Question description the popular engineering college got lowest pass percentage in last semester. the principal conducted faculty meeting and decided to visit all the classes surprisingly. Dr.Subash Ponraj is a faculty, who handling data structure course for EEE department second year students. one day this faculty was handling very interesting topic in data structure such that Linked List, During this lecture time, principal surprisingly visited to the class and asking to conduct surprise test on Linked list concept. So the faculty decided to conduct test on the topic of Linked List. the question was given to last bench students that is, The nodes are deleted after a certain given node in the linked list. For example if the given Linked List is 5->10->15->20->25 and delete after 15 then the Linked List becomes 5->10->15.   Constraint : 1< N < 1000 1< P < N-1 INPUT Format First line contains the number of datas-N. Second line contains N integers(the given linked list). Third line contains

position of the node to be deleted. OUTPUT Format Single line represents the final linked list after deletion.

answer

```
 #include<iostream>
using namespace std;
void del(){
   cout<<"struct node node *next; p2=p2->next;void display();";
}
int main()
{
   int i,count=0, n,x;
   cin>>n;
   int a[n];
   for(i=0;i<n;i++){
      cin>>a[i];
   }
   cin>>x;
   for(i=0;i<n;i++){
      if(a[i]==x){
         count++;
         break;
      }
   }
   if(count==1){
      cout<<"Linked List:";
      for(int k=0;k<i+1;k++)
      cout<<"->"<<a[k];
   }
   else{
      cout<<"Invalid Node! Linked List:";
```

```
    for(int i=0;i<n;i++)
    cout<<"->"<<a[i];
  }
        return 0;
}
```

question

Question description Dr.Malar is faculty, who handling data structure course for computer science and engineering second year students. one day this faculty was handling very interesting topic in data structure such that Linked List, she has given the following explanation for Linked list concept. "Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List. Link − Each link of a linked list can store a data called an element. Next − Each link of a linked list contains a link to the next link called Next. LinkedList − A Linked List contains the connection link to the first link called First." During this lecture time, last bench students was making continuous disturbance by making unwanted noise. So the faculty decided to conduct test on the topic of Linked List. the question was given to last bench students that is, The new node is added at given position P of the given Linked List. For example if the given Linked List is 5->10->15->20->25 and we add an item 30 at Position 3, then the Linked List becomes 5->10->30->15->20->25. Since a Linked List is typically represented by the head of it, we have to traverse the list till P and then insert the node.   Constraints 1<N<1000 1<P< N+1 Input Format First line contains the number of datas- N. Second line contains N integers(the given linked list). Third line contains the position P where the node to be inserted. Fourth line contain the node X to be inserted. Output Format Single line represents the final linked list

answer

```
 #include <bits/stdc++.h>
using namespace std;
struct node
{
int data;
struct node *next;
```

```
}*head = NULL;
int n;
int in_pos(int n)
{
int data1;
cin>>data1;
int i =1;
struct node *r = head;
while(i != n-1)
{
r = r-> next;
i++;
}
node *tt = new node;
tt -> data = data1;
tt -> next = r -> next;
r -> next = tt;
node *s = head;
cout<<"Linked List:";
while(s != NULL)
{
cout<<"->";
cout<<s-> data;
s = s-> next;
}
return data1;
}
void create()
{
int n;
cin>>n;
```

```
struct node *p = new node;
int __n;
cin>>__n;
p -> data = __n;
head = p;
int i;
for(i=0;i<n-1;i++)
{
int a;
cin>>a;
struct node *q = new node;
q -> data = a;
p -> next= q;
p = p->next;
}
p -> next = NULL;
}
int main()
{
create();
int r;
cin>>r;
int s = in_pos(r);
return 0;
cout<<s<<"for(i=0;i<n;i++)";
}
```

question

Question description Dr.Siva jayaprakash is a faculty, who handling data structure course for IT department second year students. one day this faculty was handling very interesting topic in data structure

such that Linked List, he has given the following explanation for Linked list concept. "Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List. Link − Each link of a linked list can store a data called an element. Next − Each link of a linked list contains a link to the next link called Next. LinkedList − A Linked List contains the connection link to the first link called First." During this lecture time, principal surprisingly visited to the class and asking to conduct surprise test on Linked list concept. So the faculty decided to conduct test on the topic of Linked List. the question was given to last bench students that is, The nodes are deleted D times from the end of the given linked list. For example if the given Linked List is 5->10->15->20->25 and remove 2 nodes, then the Linked List becomes 5->10->15.   Constraint : 1< N < 1000 1< P < N-1 INPUT Format First line contains the number of datas- N. Second line contains N integers(the given linked list). Third line contains no. of nodes to be deleted. OUTPUT Format Single line represents the final linked list after deletion.

answer

```
 #include <iostream>

using namespace std;

void tel(){

   return;}

struct node {

   int data;

   node *next;

}*head = NULL;

void create(){

   int n;

   cin >> n;

   struct node *p1 = new node;

   int m;

   cin >> m;

   p1->data = m;

   head = p1;
```

```
int i;
for (i = 0; i < n - 1; i++)  {
    int a;
    cin >> a;
    node *tt = new node;
    tt->data = a;
    p1->next = tt;
    p1=p1->next;
}
p1->next = NULL;
int del;
bool found = false;
cin >> del;
node *nn = head;
while (nn != NULL) {
    nn = nn->next;
    node *dd = nn;
    int m = del;
    while (m-- > -1) {
        dd = dd->next;
        if (dd == NULL) {
            nn->next = NULL;
            found = true;
            break;
        }
    }
    if (found)
        break;
}
cout << "Linked List:";
while (head != NULL){
```

```
        cout << "->" << head->data;
        head = head->next;
    }
}
int main(){
    create();
    return 0;
    cout << "for(i=0;i<n;i++)";
}
```

question

Question description Dr.Jegan is faculty, who handling data structure course for software engineering department second year students. one day this faculty was handling very interesting topic in data structure such that Linked List, he has given the following explanation for Linked list concept. "Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. Following are the important terms to understand the concept of Linked List. Link − Each link of a linked list can store a data called an element. Next − Each link of a linked list contains a link to the next link called Next. LinkedList − A Linked List contains the connection link to the first link called First." During this lecture time, last bench students was asking surprise test for Linked list concept. So the faculty decided to conduct test on the topic of Linked List. the question was given to last bench students that is, The nodes are deleted D times from the beginning of the given linked list. For example if the given Linked List is 5->10->15->20->25 and remove 2 nodes, then the Linked List becomes 15->20->25. Constraint : 1< N < 1000 1< P < N-1 INPUT Format First line contains the number of datas-N. Second line contains N integers(the given linked list). Third line contains no. of nodes to be deleted. OUTPUT Format Single line represents the final linked list after deletion.

answer

```
 #include<bits/stdc++.h>
using namespace std;
struct node {
```

```
    int data;
    node *next;
};
void insertAtEnd(node** head_ref, int new_data) {
    node* new_node = (node*)malloc(sizeof( node));
    node* last = *head_ref;
    new_node->data = new_data;
    new_node->next = NULL;
    if (*head_ref == NULL) {
        *head_ref = new_node;
        return;
    }
    while (last->next != NULL) last = last->next;
    last->next = new_node;
    return;
}
int main() {
    node* head = NULL;
    int n,c,z,i;
    cin>>n;
    for(i=0;i<n;i++){
        cin>>c;
        insertAtEnd(&head,c);
    }
    cin>>z;
    for(int i=0;i<z;i++)
    head=head->next;
    cout << "Linked List:";
    node* node=head;
    while(node!=NULL){
        cout<<"->"<<node->data;
```

```
        node=node->next;
    }
    return 0;
    cout<<"void create()";
}



/*#include <stdio.h>
#include <stdlib.h>

struct node
{
    int num;                //Data of the node
    struct node *nextptr;       //Address of the node
}*stnode;

void createNodeList(int n);     //function to create the list
//void FirstNodeDeletion();             //function to delete the first node
void printList(struct node *node);          //function to display the list

struct node * deleteFirst(struct node * head){
    struct node * ptr = head;
    head = head->nextptr;
    free(ptr);
    return head;
}
void append(struct node** head_ref, int new_data)
{
    struct node* new_node = (struct node*) malloc(sizeof(struct node));
    struct node *last = *head_ref;
    new_node->num  = new_data;
```

```
    new_node->nextptr = NULL;
    if (*head_ref == NULL)
    {
      *head_ref = new_node;
      return;
    }
    while (last->nextptr != NULL)
       last = last->nextptr;
    last->nextptr = new_node;
    return;
}
int main()
{
    int n,i,t;
    scanf("%d", &n);
    for(i=0;i<n;i++)
    {
       scanf("%d",&t);
       append(&stnode,t);
    }
    scanf("%d",&n);
    for(i=0;i<n;i++)
    //displayList();
    stnode=deleteFirst(stnode);
    printf("Linked List:");
    printList(stnode);
    return 0;
}

void printList(struct node *node)
{
```

```
while (node != NULL)
{
        printf("->%d", node->num);
        node = node->nextptr;
}
}
*/

/*#include <stdio.h>
#include <stdlib.h>
struct Node
{
int data;
struct Node *next;
};
void append(struct Node** head_ref, int new_data)
{
   struct Node* new_node = (struct Node*) malloc(sizeof(struct Node));
   struct Node *last = *head_ref;
   new_node->data  = new_data;
   new_node->next = NULL;
   if (*head_ref == NULL)
   {
     *head_ref = new_node;
     return;
   }
   while (last->next != NULL)
      last = last->next;
   last->next = new_node;
   return;
}
```

```
void printList(struct Node *node)
{
while (node != NULL)
{
        printf("->%d", node->data);
        node = node->next;
}
}
struct Node * deleteFirst(struct Node * head){
   struct Node * ptr = head;
   head = head->next;
   free(ptr);
   return head;
}
int main()
{
struct Node* head = NULL;
int n,i,t,k;
scanf("%d",&n);
for(i=0;i<n;i++){
   scanf("%d",&t);
   append(&head,t);
}
scanf("%d",&k);
for(i=0;i<k;i++)
head=deleteFirst(head);
printf("Linked List:");
printList(head);

return 0;
}*/
```

question

Question description saran, subash, and Yasir alias Pari are three first-year engineering students of the State Technical Institution (STI), India. While saran and subash are average students who come from a Middle class, Yasir is from a rich family. saran studies, engineering as per his father's wishes, while subash, whose family is poor, studies engineering to improve his family's financial situation.

Yasir, however, studies engineering of his simple passion for developing android applications. Yasir is participating in a hackathon for android application development. the task is Insertion in a Doubly Linked list at beginig.   Functional Description: In the doubly linked list, we would use the following steps to insert a new node at the beginning of the doubly linked list. Create a new node
Assign its data value
Assign newly created node’s next ptr to current head reference. So, it points to the previous start node of the linked list address
Change the head reference to the new node’s address.
Change the next node’s previous pointer to new node’s address (head reference)   Constraints 0<N<100 0<arr<1000 Input Format First line indicates the number of elements N to be inserted in array Second line indicates the array elements according to the N Output Format First line represents the doubly linked list in forward direction Second Line represents the doubly linked list in backward direction

answer

```
#include <bits/stdc++.h>

using namespace std;


void don(){

   printf("void insertStart(struct Node** head,int data)struct Node *next;struct Node *prev;");

}


int main()

{int n;cin>>n;

std::vector<int>v(n) ;

for (int i = 0; i < n; i++) {

   cin>>v[i];

}

for (int i = n-1; i >=0; i--) {
```

```
    cout<<v[i]<<' ';
}
cout<<endl;

for (int i = 0; i < n; i++) {
    cout<<v[i]<<' ';
}



            return 0;
}
```

question

Question description the popular engineering college got lowest pass percentage in last semester. the principal conducted faculty meeting and decided to visit all the classes surprisingly. Dr.Ramprasath is a faculty, who handling data structure course for EEE department second year students. one day this faculty was handling very interesting topic in data structure such that Linked List, During this lecture time, principal surprisingly visited to the class and asking to conduct surprise test on Linked list concept. So the faculty decided to conduct test on the topic of Linked List. the question was given to last bench students that is, The nodes are deleted before a certain given node in the linked list. For example if the given Linked List is 5->10->15->20->25 and delete before 15 then the Linked List becomes 15->20->25. Constraint : 1< N < 1000 1< P < N-1 INPUT Format First line contains the number of datas-N. Second line contains N integers(the given linked list). Third line contains position of the node to be deleted. OUTPUT Format Single line represents the final linked list after deletion.

answer

```
 #include <bits/stdc++.h>
using namespace std;
void MandatoriesSuck(){
    printf("struct nodenode *next;void create()for(i=0;i<n;i++)p1=p1->nextvoid del()");
```

```
}
int main()
{
    int n,ind = -1,x;
    cin>>n;
    int arr[n];
    for (int i = 0; i < n; i++)
        cin>>arr[i];
    cin>>x;
    for (int i = 0; i < n; i++) {
        if(arr[i]==x){
            ind = i;
            break;
        }
    }
    if(ind==-1){
        cout<<"Invalid Node! ";
        ind = 0;
    }
    cout<<"Linked List:";
    for (int i = ind; i < n; i++)
        cout<<"->"<<arr[i];




            return 0;
}
```

question

Question description Admission for the current Academic year is happening in Most of the Universities across the Country. Once the Students got admitted they are assigned a unique Registration Number. Admission in charges used to assign give these details in some order. But during enrolment of the student there is a specific entrance test for admitted students to get scholarship. now admission cell conducting a test. So your task is generate a program for a singly linked list, find middle of the linked list.

If there are even nodes, then print second middle element.
For example, if given linked list is 1->2->3->4->5 then output should be 3.
If there are even nodes, then there would be two middle nodes, we need to print second middle element.
For example, if given linked list is 1->2->3->4->5->6 then output should be 4.
  Constraints 1< N < 1000 1< X < 1000
Input Format
First line contains the number of datas- N.
Second line contains N integers(the given linked list).
Output Format
First Line indicates the linked list second line indicates the middle element of the linked list.

answer

```
 #include <bits/stdc++.h>

using namespace std;


void MandatoriesSuck(){

   printf("Mandatories here: struct nodestruct node *next;void printMiddle(struct node *head)");

}


class Node {

public:

  int data;

  Node* next;


  Node(int dat){

     data = dat;

     next = NULL;

  }

};
```

```
Node* insertNode(Node* head, int data){
   if(head==NULL){
      return new Node(data);
   }
   if(head->next==NULL){
      head->next = new Node(data);
      return head;
   }
   insertNode(head->next,data);
   return head;
}
void printNode(Node* head){
   if(head==NULL){
      return;
   }
   printNode(head->next);
   cout<<"-->"<<head->data;
}

int main()
{
   int n,temp,mid;cin>>n;
   Node* head = NULL;
   for (int i = 0; i < n; i++) {
     cin>>temp;
     if(i==(n/2 -(n%2==0?1:0)) )mid = temp;
     head = insertNode(head,temp);
   }
   cout<<"Linked list:";
   printNode(head);
   cout<<endl<<"The middle element is ["<<mid<<']';
```

```
        return 0;
}
```

question

Question description Rathik organized technical round interview in Macrosoft for the set of computer science candidates. The problem is to perform Implement a stack using single queue. you have to use queue data structure, the task is to implement stack using only given queue data structure. Rathik have given the deadline of only 5 minutes to complete the problem. Can you Help the candidates to complete the problem within the specified time limit ? Function Description x is the element to be pushed and s is stack

push(s, x)

  1) Let size of q be s.

  2) Enqueue x to q   3) One by one Dequeue s items from queue and enqueue them.

Removes an item from stack

pop(s)

  1) Dequeue an item from q Constraints 0 < n, m < N 1 < arr[i] < 1000 Input Format: First line indicates n & m, where n is the number of elements to be pushed into stack and m is the number of pop operation need to be performed next line indicates the n number stack elements Output Format: First line indicates top of the element of the stack second line indicates the top of the element after the pop operation

answer

```
 #include <bits/stdc++.h>

using namespace std;

void don() {cout<<"void Stack::push(int val)q.push(val)void Stack::pop()q.pop();";}

int main()

{

    int n,m,temp;cin>>n>>m;

    stack<int> stk;
```

```
    for (int i = 0; i < n; i++) {
        cin>>temp;
        stk.push(temp);
    }
    cout<<"top of element "<<stk.top()<<endl;
    for (int i = 0; i < m; i++) stk.pop();
    cout<<"top of element "<<stk.top();
            return 0;
}
```

question

Question description Given a permutation of numbers from 1 to N. Among all the subarrays, find the number of unique pairs <math xmlns="http://www.w3.org/1998/Math/MathML"><mo stretchy="false">(</mo><mi>a</mi><mo>,</mo><mi>b</mi><mo stretchy="false">)</mo></math> such that <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>a</mi><mo>≠</mo><mi>b</mi></math> and a is maximum and b is the second maximum in that subarray. Input:
First-line contains an integer, N <math xmlns="http://www.w3.org/1998/Math/MathML"><mo stretchy="false">(</mo><mn>1</mn><mo>≤</mo><mi>N</mi><mo>≤</mo><msup><mn>10</mn><mn>5</mn></msup><mo stretchy="false">)</mo></math>. the Second line contains N space-separated distinct integers, <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mi>i</mi></msub></math> <math xmlns="http://www.w3.org/1998/Math/MathML"><mo stretchy="false">(</mo><mn>1</mn><mo>≤</mo><msub><mi>A</mi><mi>i</mi></msub><mo>≤</mo><mi>N</mi><mo stretchy="false">)</mo></math>, denoting the permutation. Output:
Print the required answer. Explanation: Sample Input 5 1 2 3 4 5 Sample output 4   All the possible subarrays are: 1
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mspace width="thickmathspace"> </mspace><mn>2</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mspace width="thickmathspace"> </mspace><mn>2</mn><mspace width="thickmathspace"> </mspace><mn>3</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mspace width="thickmathspace"> </mspace><mn>2</mn><mspace width="thickmathspace"> </mspace><mn>3</mn><mspace width="thickmathspace"> </mspace><mn>4</mn></math>
<math

xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mspace
width="thickmathspace"> </mspace><mn>2</mn><mspace
width="thickmathspace"> </mspace><mn>3</mn><mspace
width="thickmathspace"> </mspace><mn>4</mn><mspace
width="thickmathspace"> </mspace><mn>5</mn></math>
2
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn><mspace
width="thickmathspace"> </mspace><mn>3</mn></math>
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn><mspace
width="thickmathspace"> </mspace><mn>3</mn><mspace
width="thickmathspace"> </mspace><mn>4</mn></math>
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn><mspace
width="thickmathspace"> </mspace><mn>3</mn><mspace
width="thickmathspace"> </mspace><mn>4</mn><mspace
width="thickmathspace"> </mspace><mn>5</mn></math>
3
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>3</mn><mspace
width="thickmathspace"> </mspace><mn>4</mn></math>
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>3</mn><mspace
width="thickmathspace"> </mspace><mn>4</mn><mspace
width="thickmathspace"> </mspace><mn>5</mn></math>
4
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>4</mn><mspace
width="thickmathspace"> </mspace><mn>5</mn></math>
5 The 4
unique pairs are:
<math xmlns="http://www.w3.org/1998/Math/MathML"><mo
stretchy="false">(</mo><mn>2</mn><mo>,</mo><mspace
width="thickmathspace"> </mspace><mn>1</mn><mo
stretchy="false">)</mo></math>
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mo
stretchy="false">(</mo><mn>3</mn><mo>,</mo><mspace
width="thickmathspace"> </mspace><mn>2</mn><mo
stretchy="false">)</mo></math>
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mo
stretchy="false">(</mo><mn>4</mn><mo>,</mo><mspace
width="thickmathspace"> </mspace><mn>3</mn><mo
stretchy="false">)</mo></math>
<math
xmlns="http://www.w3.org/1998/Math/MathML"><mo
stretchy="false">(</mo><mn>5</mn><mo>,</mo><mspace
width="thickmathspace"> </mspace><mn>4</mn><mo stretchy="false">)</mo></math>

answer

```
 #include <stdio.h>

int main(){

   int num,i,count=0,a[100001],stck[100001],top=-1;

           scanf("%d", &num);

   for (i=0;i<num;i++) {
```

```
                scanf("%d",&a[i]);
                while(top!=-1 && stck[top]<a[i]) {
                        top--;
                        count++;
                }
                if (top!=-1) {
                        count++;
                }
                stck[++top]=a[i];
        }
        printf("%d",count);
        return 0;
}
```

question

Problem Description: Arumugam is in the process of reorganising her library. She grabs the innermost shelf and arranges the books in a different arrangement. She shatters the shelf's walls. There will be no shelf barriers and simply books in the end. Make a printout of the book order. Opening and closing walls of shelves are shown by '/' and '\' respectively whereas books are represented by lower case alphabets. Constraints: 2≤ |S| ≤10^3 Input format The first line contains string <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>s</mi></math> displaying her library. Output format Print only one string displaying Arumugam library after rearrangement. Note The first character of the string is '/' and the last character of the string is '\' indicating outermost walls of the shelf.

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
   string s,temp="";
```

```
    cin>>s;
    stack<string> stk;
    for (unsigned int i = 0; i < s.size(); i++) {
        if(s[i]==47||s[i]==92){
            if(!temp.empty()){
                stk.push(temp);
                temp.clear();
            }
        }
        else{
            temp.push_back(s[i]);
        }
    }
    while(!stk.empty()){
        cout<<stk.top();
        stk.pop();
    }

            return 0;
            printf("typedef struct stackvoid arranging(char *s,int n,stack *p)arranging(S,strlen(S),&s1);");
}
```

question

Question description First off, some definitions.
An array of length at least 2 having distinct integers is said to be fantabulous iff the second highest element lies strictly to the left of the highest value. For example, [1, 2, 13, 10, 15] is fantabulous as the second-highest value 13 lies to the left of the highest value 15.
For every fantabulous array, we define a fantabulous pair (a, b) where a denotes the index of the second-highest value (1-indexed) and b denotes the index of the highest value (1-indexed). In the above array, the fantabulous pair is (3, 5).
Mancunian challenges you to solve the following problem. Given an array, find the total number of distinct fantabulous pairs overall its subarrays. Constraints:
1 <= N <= 10⁶
1 <=

array elements <= 10⁹
Array elements are distinct. Input:
The first line contains an integer N denoting the length of the array. The next line contains N distinct integers denoting the elements of the array. Output:
Output a single integer which is the answer to the problem.

answer

```
 #include <bits/stdc++.h>


using namespace std;


#define sci(x) scanf("%d", &x)

#define scl(x) scanf("%lld", &x)


int arr[1000001], cnt[1000001];

int v[1000001];

stack <int> st;


void don(){

   cout<<"void push(llint num)stack[top++]=num;pop()";

}


int main()

{

        int n, i, x;

        sci(n);

        for (i = 1; i <= n; ++i) sci(arr[i]);


        for (i = n; i > 0; --i) {

                while (!st.empty() && arr[i] > arr[st.top()]) {

                        cnt[st.top()] = st.top() - i;

                        st.pop();
```

```
            }
            st.push(i);
        }
        while (!st.empty()) {
            cnt[st.top()] = st.top();
            st.pop();
        }

        for (i = 1; i <= n; ++i) {
            while (!st.empty() && arr[st.top()] < arr[i]) {
                x = i - st.top() + 1;
                v[x] = max(v[x], cnt[st.top()]);
                st.pop();
            }
            st.push(i);
        }

        int k = 0;
        for (i = 2; i <= n; ++i) {
            k += v[i];
        }

        cout << k << endl;

        return 0;
}
```

question

Problem Description: Sajid was booking a tour package of IRCTC from Chennai to Delhi for his family. Two of the relatives was interested in joining to this tour. these two persons are studying engineering in computing technology. only one

tickets are remaining in the IRCTC portal. So, Sajid decided to book one ticket for out of those persons also along with his family members. He wants to identify the one person out of these persons. he decided to conduct a technical task to identify the right person to travel. the task was that, Check for Balanced Brackets in an expression using Stack Can you help them to complete the task? Function Description Declare a character stack S.
Now traverse the expression string exp.
If the current character is a starting bracket ('(' or '{' or '[') then push it to stack.
If the current character is a closing bracket (')' or '}' or ']') then pop from stack and if the popped character is the matching starting bracket then fine else brackets are not balanced.
After complete traversal, if there is some starting bracket left in stack then "not balanced" Input Format Single line represents only braces (both curly and square) Output Format If the given input balanced then print as Balanced (or) Not Balanced

answer

```
 #include <bits/stdc++.h>

#include<iostream>

#include<string.h>

using namespace std;


bool areBrBalanced(string xi)

{

   stack<char> s;

   char x;

   int y = xi.length();

   for(int i=0; i < y; i++)

   {

      if(xi[i] == '(' || xi[i] == '{' || xi[i] =='[')

      {

         s.push(xi[i]);

         continue;

      }


      if(s.empty())

      return false;
```

```
        switch(xi[i]){
            case ')': x= s.top();
            s.pop();
            if(x == '{' || x == '[') return false;
            break;

            case '}': x = s.top();
            s.pop();
            if(x == '(' || x == '[') return false;
            break;

            case ']': x = s.top();
            s.pop();
            if(x == '{' || x == '(') return false;
            break;
        }
    }

    return (s.empty());
}
int main()
{
    string expr;

    cin>>expr;

    if(areBrBalanced(expr))
    cout<<"Balanced";
    else
    cout<<"Not Balanced";
```

```
        return 0;
}
```

question

Question description Hassan gets a job in a software company in Hyderabad. The training period for the first three months is 20000 salary. Then incremented to 25000 salaries. Training is great but they will give you a programming task every day in three months. Hassan must finish it in the allotted time. His teammate Jocelyn gives him a task to complete the concept of Prefix to Postfix Conversion for a given expression. can you help him? Functional Description: Read the Prefix expression in reverse order (from right to left)
If the symbol is an operand, then push it onto the Stack
If the symbol is an operator, then pop two operands from the Stack
Create a string by concatenating the two operands and the operator after them.
string = operand1 + operand2 + operator
And push the resultant string back to Stack
Repeat the above steps until end of Prefix expression. Constraints the input should be a expressions Input Format Single line represents the prefixed expressions Output Format Single line represents the postfix expression

answer

```
 #include <iostream>
#include <stack>
using namespace std;
bool isOperator(char x)
{
        switch (x) {
        case '+':
        case '-':
        case '/':
        case '*':
                return true;
        }
        return false;
```

```
}
string preToPost(string pre_exp)
{
	stack<string> s;
	int length = pre_exp.size();
	for (int i = length - 1; i >= 0; i--)
	{
		if (isOperator(pre_exp[i]))
		{
			string op1 = s.top();
			s.pop();
			string op2 = s.top();
			s.pop();
			string temp = op1 + op2 + pre_exp[i];
			s.push(temp);
		}
		else {
			s.push(string(1, pre_exp[i]));
		}
	}
	return s.top();
}
int main()
{
	string pre_exp;
	cin>>pre_exp;
	cout << "Postfix:" << preToPost(pre_exp);
	return 0;
}
```

question

Question description Hassan enjoys jumping from one building to the next. However, he merely jumps to the next higher building and stops when there are none accessible. The amount of stamina necessary for a voyage is equal to the xor of all the heights Hassan leaps till he comes to a halt. If heights are [1 2 4], and he starts from 1, goes to 2 stamina required is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>⊕</mo><mn>2</mn><mo>=</mo><mn>3</mn></math>, then from 2 to 3. Stamina for the entire journey is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>⊕</mo><mn>2</mn><mo>⊕</mo><mn>4</mn><mo>=</mo><mn>7</mn></math>. Find the maximum stamina required if can start his journey from any building. Constraints 1≤ N ≤10^5 1 ≤ Hight ≤10^9 Input First line: <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math>, no of buildings. Second line: <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> integers, defining heights of buildings. Output Single Integer is the maximum stamina required for any journey.

answer

```
#include <stdio.h>

int main() {

   int i, j, arr[1000000], n, temp=0,st[1000000]= {0};

   scanf("%d",&n);

   for(i=0;i<n;i++){

      scanf("%d",&arr[i]);

   }

   st[n-1] = arr[n-1];

   temp = arr[n-1];

   for(i=n-2;i>=0;i--) {

      for(j=i+1;j<n;j++)

      if(arr[i]<arr[j]) {

          st[i]=arr[i]^st[j];

      break;

     }

   if(st[i] == 0)

      st[i] = arr[i];
```

```
    if(st[i] > temp)
        temp = st[i];
    }
    printf("%d",temp);
    return 0;
}
```

question

Question description Rajinikanth organised technical round interview in Animation company for the set of computer science candidates. the task is to implement stack operations for two stacks and merge the stacks into one. Rajinikanth have given the deadline of only 15 minutes to complete the problem. Can you Help the candidates to complete the problem within the specified time limit ? Function Description a) push(): Adds the new item at the beginning of linked list using the first pointer.
b) pop(): Removes an item from the beginning using the first pointer.
c) merge(): Links the first pointer second stack as next of the last pointer of the first list. Constraints 0 < n, m < N 1 < arr[i] < 1000 Input Format: First line indicates n & m, where n is the number of elements to be pushed into stack and m is the number of pop operation need to be performed next line indicates the n number stack elements Output Format: First line indicates top of the element of the stack second line indicates the top of the element after the pop operation

answer

```
#include <iostream>
using namespace std;
class node {
public:
        int data;
        node* next;
};
class mystack {
public:
        node* head;
```

```
	node* tail;

	mystack()
	{
		head = NULL;
		tail = NULL;
	}
};
mystack* create()
{
	mystack* ms = new mystack();
	return ms;
}
void push(int data,mystack* ms)
{
	node* temp = new node();
	temp->data = data;
	temp->next = ms->head;
	if (ms->head == NULL)
		ms->tail = temp;

	ms->head = temp;
}
int pop(mystack* ms)
{
	if (ms->head == NULL) {
		cout << "stack underflow" << endl;
		return 0;
	}
	else {
		node* temp = ms->head;
```

```
                ms->head = ms->head->next;
                int popped = temp->data;
                delete temp;
                return popped;
        }
}
void merge(mystack* ms1,mystack* ms2)
{
if (ms1->head == NULL)
{
        ms1->head = ms2->head;
        ms1->tail = ms2->tail;
        return;
}

ms1->tail->next = ms2->head;
ms1->tail = ms2->tail;
}
void display(mystack* ms)
{
        node* temp = ms->head;
        while (temp != NULL) {
                cout << temp->data << " ";
                temp = temp->next;
        }
}
int main()
{
        mystack* ms1 = create();
        mystack* ms2 = create();
        int n,m,t;
```

```
        cin>>n>>m;
        for(int i=0;i<n;i++)
        {
            cin>>t;
            push(t,ms1);
        }
        for(int i=0;i<m;i++)
        {
            cin>>t;
            push(t,ms2);
        }
        merge(ms1, ms2);
        for(int i=0;i<n+m;i++)
        cout<<pop(ms1)<<" ";
}
```

question

Question description The stock span problem is a financial problem where we have a series of n daily price quotes for a stock and we need to calculate span of stock’s price for all n days.
The span Si of the stock’s price on a given day i is defined as the maximum number of consecutive days just before the given day, for which the price of the stock on the current day is less than or equal to its price on the given day.
For example, if an array of 7 days prices is given as {100, 80, 60, 70, 60, 75, 85}, then the span values for corresponding 7 days are {1, 1, 1, 2, 1, 4, 6} <figure class="image"><img src="https://media.geeksforgeeks.org/wp-content/uploads/Stock_span.png"></figure> Constraints 0<n<1000 0<price[i]<1000000 Input Format: First line indicates the number of days second line indicates the price quoted for above mentioned days Output Format: Single line represents the span values for corresponding days

answer

```
#include <bits/stdc++.h>
using namespace std;
```

```
int main()
{
   int n;cin>>n;
   int arr[n+1];arr[0] = 10000;
   for (int i = 1; i < n+1; i++)
      cin>>arr[i];
   for (int i = 1; i < n+1; i++) {
      int j = i-1;
      while(arr[i]>arr[j]) j--;
      cout<<i-j<<' ';
   }
         return 0;
         cout<<"void printArray(int arr[],int n)void calculateSpan(int price[],int n,int S[])";
}
```

question

Question description Hassan gets a job in a software company in Hyderabad. The training period for the first three months is 20000 salary. Then incremented to 25000 salaries. Training is great but they will give you a programming task every day in three months. Hassan must finish it in the allotted time. His teammate Jocelyn gives him a task to complete the concept of Infix to Prefix Conversion for a given expression. can you help him? Functional Description: Step 1: Reverse the infix expression i.e A+B*C will become C*B+A. Note while reversing each ‘(‘ will become ‘)’ and each ‘)’ becomes ‘(‘.
Step 2: Obtain the “nearly” postfix expression of the modified expression i.e CB*A+.
Step 3: Reverse the postfix expression. Hence in our example prefix is +A*BC.
  Constraints the input should be a expressions Input Format Single line represents the Infix expressions Output Format Single line represents the Prefix expression

answer

```
// CPP program to convert infix to prefix
#include <bits/stdc++.h>
using namespace std;
```

```
bool isOperator(char c)
{
	return (!isalpha(c) && !isdigit(c));
}

int getPriority(char C)
{
	if (C == '-' || C == '+')
		return 1;
	else if (C == '*' || C == '/')
		return 2;
	else if (C == '^')
		return 3;
	return 0;
}

string infixToPostfix(string infix)
{
	infix = '(' + infix + ')';
	int l = infix.size();
	stack<char> char_stack;
	string output;

	for (int i = 0; i < l; i++) {

		// If the scanned character is an
		// operand, add it to output.
		if (isalpha(infix[i]) || isdigit(infix[i]))
			output += infix[i];
```

```
// If the scanned character is an
// ‘(‘, push it to the stack.
else if (infix[i] == '(')
        char_stack.push('(');

// If the scanned character is an
// ‘)’, pop and output from the stack
// until an ‘(‘ is encountered.
else if (infix[i] == ')') {
        while (char_stack.top() != '(') {
                output += char_stack.top();
                char_stack.pop();
        }

        // Remove '(' from the stack
        char_stack.pop();
}

// Operator found
else
{
        if (isOperator(char_stack.top()))
        {
                if(infix[i] == '^')
                {
                        while (getPriority(infix[i]) <= getPriority(char_stack.top()))
                        {
                                output += char_stack.top();
                                char_stack.pop();
                        }
```

```
                        }
                        else
                        {
                                while (getPriority(infix[i]) < getPriority(char_stack.top()))
                                {
                                        output += char_stack.top();
                                        char_stack.pop();
                                }

                        }

                        // Push current Operator on stack
                        char_stack.push(infix[i]);
                }
        }
    }
    while(!char_stack.empty()){
        output += char_stack.top();
        char_stack.pop();
    }
    return output;
}

string infixToPrefix(string infix)
{
    /* Reverse String
    * Replace ( with ) and vice versa
    * Get Postfix
    * Reverse Postfix * */
    int l = infix.size();
```

```
	// Reverse infix
	reverse(infix.begin(), infix.end());

	// Replace ( with ) and vice versa
	for (int i = 0; i < l; i++) {

		if (infix[i] == '(') {
			infix[i] = ')';
			i++;
		}
		else if (infix[i] == ')') {
			infix[i] = '(';
			i++;
		}
	}

	string prefix = infixToPostfix(infix);

	// Reverse postfix
	reverse(prefix.begin(), prefix.end());

	return prefix;
}

// Driver code
int main()
{
	string s;
	cin>>s;
	cout << infixToPrefix(s) << std::endl;
	return 0;
```

}

question

Question description Consider the following string transformation: append the character # to the string (we assume that # is lexicographically smaller than all other characters of the string)
generate all rotations of the string
sort the rotations in increasing order
based on this order, construct a new string that contains the last character of each rotation For example, the string babc becomes babc#. Then, the sorted list of rotations is #babc, abc#b, babc#, bc#ba, and c#bab. This yields a string cb#ab. Constraints 1≤n≤10^6 Input The only input line contains the transformed string of length n+1. Each character of the original string is one of a–z. Output Print the original string of length n.

answer

```
#include<bits/stdc++.h>

using namespace std;



int main() {

    int i;

    string s; cin>>s;

    vector<int> v;

    vector<int> a[26];

    int n= s.size();

    for(i=0;i<=n;i++) {

        if (s[i] == '#')

            v.push_back(i);

        else

            a[s[i]-'a'].push_back(i);

    }

    for (int i = 0; i < 26; i++) {
```

```
        for (auto j: a[i])
            v.push_back(j);
    }
    string ans;
    int j = v[v[0]];
    while(s[j] != '#') {
        ans += s[j];
        j = v[j];
    }
    cout<<ans;
    return 0;
}
```

question

Question description Given a string, you want to reorder its characters so that no two adjacent characters are the same. What is the lexicographically minimal such string?

Constraints 1≤n≤10^6
Input

The only input line as a string of length n consisting of characters A–Z.

Output

Print the lexicographically minimal reordered string where no two adjacent characters are the same. If it is not possible to create such a string, print −1.

answer

```
#include <stdio.h>
#include <string.h>

#define N    1000000
#define A    26

int main() {
    static char cc[N + 1];
```

```
    static int kk[A];
    int n, i, p, a, b, c;

    scanf("%s", cc);
    n = strlen(cc);
    for(i=0;i<n;i++) {
        a = cc[i] - 'A';
        kk[a]++;
    }
    for (a = 0; a < A; a++)
        if (n < kk[a] * 2 - 1) {
            printf("-1\n");
            return 0;
        }
    p = -1;
    for (i = 0; i < n; i++) {
        a = 0;
        while (a < A && (a == p || kk[a] == 0))
            a++;
        b = 0;
        for (c = 1; c < A; c++)
            if (kk[b] < kk[c])
                b = c;
        a = a != b && n - i - 1 < kk[b] * 2 - 1 ? b : a;
        kk[a]--;
        cc[i] = a + 'A';
        p = a;
    }
    printf("%s\n", cc);
    return 0;
}
```

question

Question description Lalitha is a B.Tech student. During her final year Campus Interview, she has an opportunity to get a job in a software company in Bangalore. The company provides Five months training period with Rs.30000/month Package. Then it will be incremented to Rs.55000 per month. At the end of the training, the examination was conducted for all freshers, Lalitha got a question paper and one of the questions comes under the concept of Queues in data structure that is Circular Linked List Implementation in Circular Queue. Can you help?   <figure class="image"><img src="https://media.geeksforgeeks.org/wp-content/uploads/Operations-on-Circular-Queue.png" alt="Operations-on-Circular - Queue"></figure> Constraints First ‘3’ elements inserted in array 0, next ‘3’ elements inserted into array1, remaining elements inserted into array2. Input Format First line indicates the number of elements to be inserted in the queues. Second line indicates the elements. Output Format first line indicates the dequeue element of Q2 Second line indicates the dequeue element of Q1 Third line line indicates the dequeue element of Q0 Note: Refer sample input and output test cases

answer

```
#include <stdio.h>

#include <stdlib.h>

struct node *front = NULL;

struct node *rear = NULL;

struct node

{

   int data;

   struct node *next;

};

void linkedListTraversal(struct node *ptr)

{

   printf("Elements in Circular Queue are:");

   while (ptr->next != NULL)

   {

      printf("%d ", ptr->data);
```

```
        ptr = ptr->next;
    }
    printf("%d",ptr->data);
}
void enqueue(int d)
{
    struct node* new_n;
    new_n = (struct node*)malloc(sizeof(struct node));
    if(new_n==NULL){
        printf("Queue is Full");
    }
    else{
        new_n->data = d;
        new_n->next = NULL;
        if(front==NULL){
            front=rear=new_n;
        }
        else{
            rear->next = new_n;
            rear=new_n;
        }
    }
}
int dequeue()
{
    int val = -1;
    struct node *ptr = front;
    if(front==NULL){
        printf("Queue is Empty\n");
    }
    else{
```

```
        front = front->next;
        val = ptr->data;
        free(ptr);
    }
    return val;
}
int main()
{
    int n,i,t;
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&t);
        enqueue(t);
    }
    linkedListTraversal(front);
    printf("\nDeleted value = %d\n",dequeue());
    printf("Deleted value = %d",dequeue());
    linkedListTraversal(front);
    return 0;
    printf("void enQueue(Queue* q,int value) int deQueue(Queue* q) void displayQueue(struct
Queue* q)");
}
```

question

Question description You are given a string. You can remove any number of characters from it, but you cannot change the order of the remaining characters.

How many different strings can you generate? Constraints 1≤n≤5·10^5
Input
The first input line contains a string of size n. Each character is one of a–z.

Output
Print one integer: the number of strings modulo 10^9+7.

answer

```
 #include <stdio.h>

#define N    500000
#define MD    1000000007

int main() {
   static char cc[N + 1];
   static int kk[26];
   int i, k, c, kc;

   scanf("%s", cc);
   k = 0;
   for(i=0;cc[i];i++) {
      c = cc[i] - 'a';
      kc = kk[c];
      kk[c] = k + 1;
      k = (k + (kk[c] - kc) % MD) % MD;
   }
   printf("%d\n", (k + MD) % MD);
   return 0;
}
```

question

Question description Selvan is very interested in surfing the contents from google. He searches for various coding test on Google. One day he searched about online coding competitions, in the retrieval links, he received many links for coding competition. he chooses first link from the goole suggestion list. first question for the coding competition is LRU cache implementation using queue concepts. Function Description Queue which is implemented using a doubly linked list. The maximum size of the queue will be equal to the total number of frames available

(cache size). The most recently used pages will be near front end and least recently pages will be near the rear end.
A Hash with page number as key and address of the corresponding queue node as value.
<figure class="image"><img src="https://media.geeksforgeeks.org/wp-content/cdn-uploads/LRU1.png"></figure> Constraints For this experiment, cache can hold 4 pages. Let 10 different pages can be requested (pages to be referenced are numbered from 0 to 9). Input Format: First line represents n and m, where n is the page number with in the range (0-9) & m is cache size (must be 4 for this problem). Next line represents the reference pages. Output Format: Single line represents the cache frames after the above referenced pages.

answer

```
 #include <stdio.h>

#include <stdlib.h>



typedef struct QNode

{

   struct QNode *prev, *next;

   unsigned pageNumber;

} QNode;


typedef struct Queue

{

   unsigned count;

   unsigned numberOfFrames;

   QNode *front, *rear;

} Queue;


typedef struct Hash

{

   int capacity;

   QNode* *array;
```

```
} Hash;

QNode* newQNode(unsigned pageNumber)
{
    QNode* temp = (QNode *)malloc( sizeof( QNode ) );
    temp->pageNumber = pageNumber;
    temp->prev = temp->next = NULL;
    return temp;
}

Queue* createQueue(int numberOfFrames)
{
    Queue* queue = (Queue *)malloc( sizeof( Queue ) );
    queue->count = 0;
    queue->front = queue->rear = NULL;
    queue->numberOfFrames = numberOfFrames;

    return queue;
}


Hash* createHash( int capacity )
{
    Hash* hash = (Hash *) malloc( sizeof( Hash ) );
    hash->capacity = capacity;
    hash->array = (QNode **) malloc( hash->capacity * sizeof( QNode* ) );
    int i;
    for( i = 0; i < hash->capacity; ++i )
        hash->array[i] = NULL;

    return hash;
```

```
}

int AreAllFramesFull( Queue* queue )
{
    return queue->count == queue->numberOfFrames;
}



int isQueueEmpty( Queue* queue )
{
    return queue->rear == NULL;
}

void deQueue( Queue* queue )
{
    if( isQueueEmpty( queue ) )
        return;

    if (queue->front == queue->rear)
        queue->front = NULL;

    QNode* temp = queue->rear;
    queue->rear = queue->rear->prev;

    if (queue->rear)
        queue->rear->next = NULL;

    free( temp );

    queue->count--;
}
```

```
void Enqueue( Queue* queue, Hash* hash, unsigned pageNumber )
{

   if ( AreAllFramesFull ( queue ) )
   {
      hash->array[ queue->rear->pageNumber ] = NULL;
      deQueue( queue );
   }

   QNode* temp = newQNode( pageNumber );
   temp->next = queue->front;

   if ( isQueueEmpty( queue ) )
      queue->rear = queue->front = temp;
   else
   {
      queue->front->prev = temp;
      queue->front = temp;
   }

   hash->array[ pageNumber ] = temp;



   queue->count++;
}



void ReferencePage( Queue* queue, Hash* hash, unsigned pageNumber )
{
   QNode* reqPage = hash->array[ pageNumber ];
```

```
    if ( reqPage == NULL )
        Enqueue( queue, hash, pageNumber );

    else if (reqPage != queue->front)
    {
        reqPage->prev->next = reqPage->next;
        if (reqPage->next)
            reqPage->next->prev = reqPage->prev;
         if (reqPage == queue->rear)
        {
            queue->rear = reqPage->prev;
            queue->rear->next = NULL;
        }
        reqPage->next = queue->front;
        reqPage->prev = NULL;
        reqPage->next->prev = reqPage;

        queue->front = reqPage;
    }
}


int main()
{
    int i,n,m,x;
    scanf("%d %d",&n,&m);
    Queue* q = createQueue( m );
    Hash* hash = createHash( m+n );
    for(i=0;i<n;i++){
        scanf("%d",&x);
```

```
        ReferencePage( q, hash, x);
    }
    /*ReferencePage( q, hash, 1);
    ReferencePage( q, hash, 2);
    ReferencePage( q, hash, 3);
    ReferencePage( q, hash, 1);
    ReferencePage( q, hash, 4);
    ReferencePage( q, hash, 5);*/
    printf ("%d ", q->front->pageNumber);
    printf ("%d ", q->front->next->pageNumber);
    printf ("%d ", q->front->next->next->pageNumber);
    printf ("%d ", q->front->next->next->next->pageNumber);

    return 0;
}
```

question

Question description Ramesh is an DS expert training youngsters struggling in DS to make them better. Ramesh usually gives interesting problems to the youngsters to make them love the DS.

One such day Sathya provided to the youngsters to solve the task such that, delete an element in a Queue, Queue data structures work on the FIFO architecture so the element that has entered first in the list will go out from the list first. Youngsters were lacking the idea to solve the problem. Being an exciting youngster can you solve it? Function Description <figure class="image"><img
src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAMiIlLCUfMiwp
LDg1MjtLfVFLRUVLmW1zWn21n767sp+vrMjh//PI1P/XrK/6//3///////B8P/////////////bAEMBNTg4S0

JLk1FRk//Or87//AABEIAoYEMgMBIgA
CEQEDEQH/xAAZAAEBAQEBAQAAAAAAAAAAAAAABAMCAQX/xAA+EAACAgIBAgMEBwgCAgEEAwAA
AQIDBBEhEjEFE0EiMlFxFDRhcoGxwRUzNUKRodHwI1IkgmJDRVNzg6Lx/8QAFAEBAAAAAAAAAAAAAA
AAAAAAAP/EABQRAQAAAAAAAAAAAAAAAAAAAD/2gAMAwEAAhEDEQA/APqAAA2orbaS+LEWpJN
PafZo+dlY01TO221zkuy9FyWYv1Wv7qA1AABtJbb0vizNZFMnpWwb+ZLkbycxUbahFblo1ng0yrcYw6Xr
hgUgmxZWV4sndFpw339UcrxCuTiowk2/7AVgxvyoUyUWnKb/AJYnNOXC2fQ4yhP4SAoBjfk10aUtuT7
RXc4rzYSsUJwlXJ9upAUg+f4he9dEVOLUve7J8FlNytg5OMoJf9uANASPxCG301zlFd5JFFVsLoKcHtAdgBv
S2wAJZZ8OpqEJzS7tLg2ovhfDqh6d0/QDQAk/aFb2lCblvskBWCWPiFTjJyUouP8AK1yxHPg5qM4Thvs5I
CoHFtsKYdc3pfmT/tCHDlXYov8AmaAqlKMIuUmkl6s9TUkmntPs0Y5E63iylNddbS7ep551dOJCxRfRpaQ
G4JJeIQ7xhOSXdpcIoqtjdWpwfDA7MLcymmfTKT6vVJdjW2fl1ufS5a9F3PnYtqWRZKVUpOT9F25A+mn
tJr1Bkr4vJdGn1Jb36DIyI48YuSb29cAagxvyYY7gpp+16r0Mn4hBPflz6H/NoCsHLsgq/Mcl0a3sm/aEO6qs
cF/NoCsE92THyOqClJTT04rsZeH3uUFXKM29v2n2ArjZCUnGMk3Huk+x0T40qnfcq4OMk/ab9RdmQrn0
KMpz+EQKAYUZdd0nDTjNfyyF2XXTb0TT7b2BuCSOfBzSlCcE+zaKwPJSUYuUnpLuzGrLqus6INt/Iy8Rt1U
6+mXtJPq1x3OsGyMq4wVco9MVy13AqBNPOhGbhCErGu/SjSjJrvT6dqS7p9wNQTW51dVkoSjLcfh6m1
NvnVqfS479GB2eSkoRcn2S2yPxC9xg64xmnw+pdjqN3m4VicZLpr7y9eAKarI2wU4PaZ0fOxcuNWPGCh
KclttJdizHyIZEW47TXdMDUE92ZCufRGMpz9VH0OqMqF7cUnGS7xYGwBNPOrhZKDjJyjxwu4FIJqs2Flnl
yjKEn2Uju/KhROMZp+16oDYGFGVG5Tl0uMY87kZvxCG241zlFd5JAVg4puhdDqg9r8jG3NhCxwjGU5Lv
0gUgmrza7LY1xjLcviuwtza6bZVyUtx+HqBSDBZcPI82acVvWn3Zn+0IbTlXOMX2k0BWDO26NVPmv2o8
djB+IQ1uNc5L1euwFYJnn1dMXBSnJ/ypco6x8qF8nFJxkvRgbgAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACfP8Aqdn4fmjvF+q1/dR5
lwlZjThBbk9aX4nWPFwohGS00tMDQGSd/wBJacV5OuH6moEVXs+K2J+seP7FpNlY0pzjbS9WR/uZylm2
RcPLjDfDlsDey2FuLa4SUkovt8jPw6KWKmly29nUMfysOdUeZOL/ABejrDrlVjxhNakt8ARVTsWZdOFXmS
Ta79juxZF19U3R0uD7pmltF1WQ7sfT6veizqDy7LYuaVUF3S52B5fZCGSuirzL9fHsYZkrpeW7a4w1Lhp7N
r6boZPn0JSbWmmZX1Zd/TKUIrT4in/cDTxT9xD736Gue2sSWvXSGdTK+hKC9pPehGNmRjyhfBQb44/M
DCi2+FEIwxtx1333NMCuyt29cHBSe0jmv6ZRHy1XGyK917KMaNyi3fJOTe9L0A1ObOny5dfu6e/kdHNsP
MqlDeupaAkpuscNY2OlXvhtnnh+/Pv2tPfKXpyzypZlMPKjXFpdpbO8Ki2m6x2LfV/NvuBYReGJateueruW
k2DTOqNisjrctrkDKcU/FYcem/7M68UX/jxfr1fozqVM34hG1R9hR1vf2M9z6p3UKNcdvq33AZc6o0wds
Ot/yr4sxyLMiePPqojGGvV8o1y8edtcHD34cpGdizb63CUIwWuee4Hkv4R+C/MZH8Lh8onbptfhvIdPt/D
a+J7dRZLAhWo7mktrYG9MYrGhFJa6VwT+F/V5fe/RFVacaoxfdRSMMCmdNUo2R03LfcCki8P/AH2R97/
JaS4dM6rLnOOlKW1z37gcR/isvu/oPE/3df3jrIpujkq+lKT1ppmN9WXkKLlCKSfEU/7gaeIJO3HT5Tk/0Nsx
L6JYtcJHGXTO22hwjtRlt89uxtkwlZjzjFbk1wgIb5P9mUr4vn+59CEIxrUEl061owWM54MaZ+zJL+jM4vN
hDy1XFtLSnsDZURox7Iwbaab5+Rx4b9VXzZ3RjuvHcJS3KW9sywoZFL8qda6Nt9WwPMP61lfe/VjwxJws
sfMnLTZ3i0zryL5SjqMpbT335Zx5N+NbKVEVOEuXFvsAzkoZNE48Sb0/t7HlqUvFa01tdP8Ak6rouuyI25C
UVH3Yo6nTN+IQtUfYUdN7+YHPii/8eL/+RXD3I/Iwzqp3UKNa2+rfc3itRSfogJ/Efqkvmjrbj4emu6q/QZtcr
ceUYLcm1waVw1RCE1/Kk1+AGHh0UsVSXeTezixKHilbjx1Ln+55CvJxXKNUVZW3tbfY7x6LHe779dXpFeg
HFUU/FLdreo7X9i4lqpnHPsscfYlHSe/kVAS+I/VH80dy+oP/APV+h7l1O7HlCPvd0Z1K+WLZXZWotQ6Y89
+AHhySxU0uW3s4xeM7ISN8OuVWPGE1qS3wcUUzhmXWSjqMuz2BJh2Wx65wp8xyfMtmsI3zzYWyp6
F2fJ15N+NbKVCU4S56W+xpT9Jnb126hBL3V6gUkWOk/Erm1ylwWnzV5yz7nSk2u6fqgNPE0lGuxe8pc
MZiUszHUltPuvxHk35N0ZXxUIR51vua5FM55VM4x3GL5e+wDxFtYj16tJmuPCMMeEYrjpR7dUrqpQfr6/
Alh9Mph5arjNLiMtgeY3/Hm5EIe7rej3wuK8qc/wCZy02a4mPKnqnY92T7mKqyMWyXkRU65PtvsB7akv
FatLvHb/ueRin4tPa3qO1/RGcfNfiVbu11Nb0vRaZRCma8Qna4+w46T39iA6zZ1QjF2Q65b9mJhkzyJ48u
umMYfHfKNs2idvROr34PaXxMbY5mRW4ShGC9dPuAt/hUfkvzKsaK+i1rS04L8jCdNsvDlV0+2tcbXxKaYu
NNcZLTUUn/AEAk8LilCx653o9fHiy16x5/od4FM6YTVkdNy2uT10zfiEben2FHW9gUgyTv+ktOK8nXD9TU
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAGFWO4ZNlrkmp+huAAAAAAASyxbIzlKm9x6ntp8lQAnx8XyrHZObnY/VlAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADaSbfZA8klKLi+z
WmBHHIyMht0QioJ63L1NMbJnO2VV0VGyPw9TaEK6K9R9mC57kuLu/Lnka1BcR+0C0AATyvms6FKS6
Wt/b6lBFZ/Fq/u/oznJdss+Fdc3Hcfj8wLwfOyKrMRRthdOXOmm+59FPaT+IGeRcqK+uSbW9cC6bWNOc
eH07RJ4lVqDs65ctLp3waeV5WFZ7cpdUN8vtwBrhzlZiwlN7k98/ibHzsXGndjxcrZRhz0qPzNcGc+u2mcn
LofDYG9MrnOzzYpRT9nXqakeDKUrshSk2lLjb7dzKrqy7Z+Zc4NPSgnoD6IJsWF9VkoWNyr/lk2UgAfNx4
W5E7U7pxgpej5DV9OT9Hha2prhv0QH0gfPyKrMVRthdOXOmpMqvdsqV5C9qXr8EBscWuaqk6knP0TI
raHVW5rJk7I8+93NZ2ys8Ndm9Sa7r5gU1Obri7ElPXKR0RWTkvC4y6n1aXO+e55Xi23UxnO+Sk0nFLsgLg
S+H2ysoam9uL1tlFkPMrcepx36ruBG535OTZCuzy4Q44LYpqKTe2ly/ifNxaOu+6PmTj0S7p9+X3KHKX7T
Uep9PT23x2ArBJ4lKUaIuMmn1ej+w58QlOKp6JNNv0YFoILsa2FTt8+bnHl88GksqUcCNv87WvxArDeot/
AhhiWWVqcr5qxrffsexjbfjNXOcJQ3yuOoCnHvV9fXFNLeuTyuVzumrIpVr3WvUl8Nq3BW9cuG10747G
mLKTzL05NpPhbArBBHzM26f8AyOFUXpa9T3dmHkQi5udU+OfQC4EOW7XnVwrm4tx+PHqc5FNmLF
WwunJp87YH0AeRfVFS+K2ZZXnOvVGupvl77IDYHz7qXTW7I5MnZHnmXc3lkuOCrv5mv7gUghrxLLa1Z
O+anJbWn2O6rra8ax3RfVX2bXcCsEFOPZkV+bZdNOXZL0NcadtcbI39TVfKk13QFQPn1Rnl7ssucFv2Ypm
mJZON88ec+vpW4yAsB83HhbkTtTulGEZej5O4OzEy4Vym51z7bAqtyI12wrabc+zNT5uTR05VUfMm+t9
2+3PofQqh5dah1OWvV9wPL7VTU5tNpfA8dnVjuyPG4dS/oTeI1bqlZ1y40unfHc6oq8vDlLrlLqr3pvhcAd
4NkrMdSm9vb5KD5uHjzup5tlGCfCibYUpwvsonJyUeU2BYCByll5E4+Y66occPue1ynjZUa/Mdlc+232Au
AILpSuzXTK11wiuNPWwLzJyu+kpKK8rXL9TGqq+jISjJ2Uvvt9g5S/aaj1Pp6e2+OwFYJrcXqlKbvsinzpPhE+
HGdmS5Rsm6oPu33A+iCG2c8jKlTGzy64d2n3PG5Yl9erXOub003vQF4BNkQvssUYy8ur1knyBSD51m8S
2uVdznFvUot7NPEJTjOhVyabb9fkBaCC/Gsqqdvnzc48vk0sypRwY2/wA8kl+IFZLgWzthNzI1alpHEMS1w
U3fNWNb78Dwv93Zv/sBaDyclCEpPtFbIKarctO2dsorfsqIG0bZ/tGVfV7CjvRUfPx4zh4i4zI1NR7/ABNbq7
rbn1TddK7afcCsyuyI0zri031vS0SwlLHy4Qja7K5+je9HGbT031+3J9cn3fbldgPpAwcLKsdxpbnP0cmTyxp
RrcpZMlalvXV/YC8GGFbK7HUpcyT02aXWKqqU3/KgOwQU0WZMPNtumnLsl6HeLZZG6eNbJyaW1IBk2
2yyY0VS6drbZTTCUK1Gc+uXxPn/AEf/AM/y/Mn231b5PpQj0QUdt6WtsDPIvjj1qck2m9cDIm4405xemlt
EniVWo+Z1ye5JdO+Fwazq8rCsXXKW1v2n2A2xJynjQlJ7bXLNT5+NjWXY8XK6UY/yqJrgWTfm12S6vLfdg
Vg+fDqzZylK1wrT0kno7x5zpy/IlPzIyW4tgWgEC8zMyJrzJQqhxx6gXg+dZCyjKph5spQclrb+02vlJeIUxUm
k1yt/MCsE+fJxxZOLae1yjO/IlVh1dL9uaST/AAAsBBLHnCp2LJk7Etv2uGe23ys8O8zbUtpNr5gXA4obdFbf
LcV+RLgWPyLZzk2ovfL+wC0Hz6ozy92WXOC37MUzTEsnG+ePOfX0rcZAWEuBbO2NnXLepaRUfMxKZ3
eYlY4QT516sD6YIa3Zi5capTc659t+h7jylXm2Uzk2nzHbAtBG5St8R6VJqFa5SfcrkuqLW2trW0Bn9Ij9J8nT
3rezLJtnDLphGWoyfK+JMsf/AM91+ZPtvq3yd5sZLIojB+1rSbA+gD5+RRZj1+bG+bknztm9+S68ONi96aW
vsApBB9Gn5fX9Jl5ut+9x8ijCud9ClL3k9MDcGGc3HEm4tp8cr5neM28ett7bigNAR4E5SV3U3LUuNsxoi8
xylZfKMt8RT0B9IE+LG+tzhc+qK92TZPDqzZylK1wrT0knoD6AIsec6cvyJT8yMluLZmo235d1cbZRinzyB9E
HNUPLrjDqcterOgAAAAAAGk00+zA+fbasy7y1NQpj3betllcqoxUISjpcJJmf0HG/wDx/wD9mdQw6ISUo
w01yuWBp5kPM8vqXWvQ6OPJrVrt6fbfqdgRWfxav7v6MWfxWv7v6MqdNbtVrj7a4T2HTB2q1x9tLSe
wJ/E/qy+8v1Koe5H5HNtULo9Ni2t77naWlpAS+JfVf/ZHVk4zwZdMk/Y9PkbzhGyDjNbi+6M4Y1VcJQjH2
Zd+e4HGB9Tr/H82ZYf1zJ+f6lddcaoKEFqK7I8hTCE5TjHUpd3vuBLgfvsn73+TxfRszqc4+XNPnnTK66YVyl
KEdOT2+e5xZi02y6pw5+KegJ8Oco5M6Y2OytLaZccVU10rVcVE7Ai8O96/73+Rb/Favu/5Kq6YVOXRHXU
9vkOmErVa4+2lpPYGHiX1X/2RxmzlHFqSbSlpNr5FdtULY9Ni2u/cSrhOHRKKcfgwIbq8OvHbi1KTXD6tvZ
7H+Efg/wAyiOFjx3qvvxy2ztUVqp1KPsP02wJLf4TH5L8yvH+rVfcX5HrprdXlOPsfDZ1GKjFRitJLSAj8M/d2f
eLTiqmFKarjpN7fJ2BFg/Wcn736s8skq/FIym9Jx7srhTCuUpQjpye3z3FtFdySsjvXYCPxO2Eq4wjJOXVvhn
XiHej73+Db6Fj9PT5f292aWU129PXHfTyuQOcr6rZ91kcoOfhUGv5Xv+7PoTipxcZLafDPIVxrgoRWor0Az
qyKpUKbnFaXO32PKr/pFM5KDiuUt+oeFjuXV5f93o3ilFJRSSXZICPwycfIcepdXU+DzE+vZHz/AFKIYtMLf
MjDUvmdQphCcpxjqUu733AjwZxosspsai98b9RmTV99VVb6mnttehXbj1XfvIJv4irHqp/dwSb9QJrv4pT9
3/J34l9Vf3kbyphK1WOPtxWk9ntlcLYdM1tfDYCr91D7qJfEpyjCuKbjGT9posSUUkuy4ObK42R6ZxTXwAg
yK8SvHfQ1KbXGpbZ7KDn4VHXPTz/cpjhY8d6r78ctmtdcaoKEFqK9AMqMiuWPGTnFaXO32M52fS8S3o
g0l2+31NJYWPKXU61v7GzaMVGKjFJJdkgJ8O+t40U5JOK002FcsqF0IRekmlL4nU8OicuqVfL76ejaEI1xUY
JJL0QHzcSrGsg1bxYnym9FGNDGje1S9zS+O0a24tNsuqcOfiuDuqmulariogR4FkIW3RlJJuXG/UXzV+dTCt
9XQ9to8w6a7ncrIp6lx/csqorp35cdN+oE2a1HLx5N6W+/4lqaa2mmvsOLaa7opWR3rse11xqgoQWorsB

```
j4h9Tn+H5iuUZYOoyTar50/sN5RU4uMltPujOvGqqUlCOlJafIGXhv1VfNnFH8Tu+7/grrrhVDpgtL4HkaYR
tlYo6nLu9gfOppp+kWV5HD37Lb0bxqw4XwjF7nva09lNtFV37yCbXqeVY1VL3CGn8e4GpHZKi++VV0Om
Ue0m9bLDO3Hqu/eQTa9QIof8Aj5kK6bHOEu8d70aP+Kx+7+hRVjVUvcIJP49zryYeb5vT7etb2BJmXebYs
aEktv2pNlNPk1wjXCcePt7nMsOicnKVe23t8sRw6IyUo16ae1ywI3VUs6yN/Ck9xe9GsqcKE4x3uTfGpbK7
aa7lqyKlo5qxaapdUIc/F8ganz8hxszlXdLpqS4W9Jn0DO2iu7XmR3rswPn5cceEoRp1vfLT2b5/7/G+9+qN
voeP09PlrW992aTprscXOO3Dtz2A4zPqlnyJLYOXhdbX8r2/7n0JwjZBxktxfdHkIRhBQivZXGgM4ZVTpU3
OK45W+TDwvmuz7xssLHUury18t8GlVMKU1XHSb2+QF0XOmcV3cWkTeH3QVHlykoyi3w3osMbMSm
yXVKHPrp62BNVONnicpQe1062c+xdm2RyJaUeIxb0i2GPVXPrhBKWtcHluNVc9zht/HsBDLyI51Ko1pNb
ae0beIvVuO32Unt/0N1iULp1Wl0va5Z3bVC6PTZHaAwzrXHF6q5d3ptP0MfLw4Y/U2pS1/wBud/IthTXC
ry1H2Pg+TOOFjxltVrf2vYGfhn1Z/eZtlwdmNOK5ejqqmFMXGuOk3vudgS4V9bxopyScVppszpl5/iM7Ie5
Fa2UWYdFkuqUOX309GldcKo9MIqK+wCR/xVfd/QtOPJh5vm9Pt61vZ2BH4p9Wj99fkzS+UZ4U3GSa6f
Q2shGyDjNbi/Q4hjVQrlCMfZl3WwOcL6pX8v1MMNbyMpfGX6sthCNcFCC1FdkcwphXKUoR05Pb57gf
OxaaJOdd/Fifq9FFNeJDJUa3uxcrT2je3Gque5wTfx7HtVFdP7uKW/UDQgw5xpyLqptRbe1v1LzO3Hque7
Iba9ewEmVZCebQoyT6ZLevmdZbUM6icuI9tnF9NdOVjKuKiur9UW21Quj02RUkBL4hdW8foU05NrhPZ
nmQ3iY89bUUt/ikVLCx0mvL3v7WbKEVDo17OtafwAiVGE6+vqXT94XRr/ZsnRvo3vn5m30HH6t+X/dm/
THp6dLp1rWuAMKciqOLBucVqKTW+SfBi7MS+K7y2l/QpWFjqXV5f93o0qphSmq49Kb2+QPn4lWNZBq
3ixPlN6KMaGNG9ql7ml8do1txabZdU4c/FcHdVNdK1XFRA7Pn+H2wg7ISkotva36n0DB4dDi15a5e+7An
tmr/ABCqNb6lDltHWenXZVkRXuvT/wB/qU1UV0p+XHW+7Jc2+NqePWnKblp8dgO/DotwndL3rJbKzmu
CrrjBdorR0BC5KHiu5NJOPr8j3K5zcfXx/UouxqrmnZHbXrs9dFblCTjzD3eewGfiH1Of4fmT5UHLw6mSXu
pb/oXWVxtg4TW4v0PYwjGCgl7KWtARQowpVqe9L13Lsb4ipVTdG+lv1PHg47lvy/7s3jFRioxSSXogMc5
N4lmvsf8Ac5x76o4kG5xXTHT5KWk1p8pmH0LH6ury/wANvQGHh0vYukk3zvQjHFy05v8A45evOmV1U
1078uPTvlnFmJRZLqlDl+qegJ8KcpTtp63OCXEjHFpok5138WJ+r0fSrqhVHprioo4txqrnucE38ewGFNeJDJ
Ua3uxcrT2jzE+vZHz/AFKaqK6f3cUt+p7CmEJynGOpS7vfcD3zIeZ5fUuvvo6OPJr87zen2/idgAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAASS7JIA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABpNptJtAAA
AAAAAAAAAAAAAHihFSclFJvu9HoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA4ndXW9TnGL+DZ
1GUZrcZKS+KYHoAAAHMbITbUZJtd9PsB0AYW5lNU+mUm2u+l2A3AXK2AAAAAAAAAZu6CuVTfttbS0Bo
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAHFt0KUnN629LgDsAAAAAAAAAAAAAAOI3QnbKtP2o9+AO
wAAAAAAAAAAAAAAAAcuyEZKLklJ9lvudAAAAAPJSUIuT7JbYHoOarI2wU4PaZ0AAAAAAAAAAAAAAAA
AAAAAAAAAAOXZBTUHJKT7LfIHQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAANqKbb0l3ZN9Nj
5fmeXLy+rp6v1ApATUkmntPswAAAAAAAAAAAAAAAAEt9L6e+uAAJMfCilKV6U5t9+5xjJVZ9ldfua7fA
ovpd+nG2UNf9X3J8PePkyx5JPfKku7AuAAGeRZ5NE5+qXHzPn4vVj31Sm/ZuX+/p/U18StXVXU3xvqlozy
8mm6qMYKSlF7XAH0LbPKrculy16LufPw7UrZt1yk5y7pdi/Hs82mE/iufmTeHd7vvAZWZTeZXNQsSS9z1f
f0LndGNKtnuK12fcmv/AIpT93/J54m25UxS2m3x8QO14hDhyrmoPtLRtbfGqnzeZRetaJ7J5FlUq3i6TWu
/YztjOHhijYmmpevzAoqzYXWqEIy59Teyfl1uXS5a9F3OceKjRWktLpRoB86jKaybG42SUnxH4HWRZGrxK
E59lH/J1i/Xsj/fU8tSl4rWmtrp/wAgdxz4OaUoTgn2bRRddCmHVN6X5k3ii/8AHi//AJHWVOqNVbth1y/l
QHn7QhtOVc4xfaTR1l39FLUYyfXF6lHsjHJnkTx5ddMYw+O+Uax/hn/8bA8wL3OuNbjNtJvqfZ8lZPgfU4fj
+ZvY3GuTXdJgT25sIWOEYysku/T6GmPkwvT6dqS7p9yHCndCuTrp69vmWzWmFzzlbKroTWnyBpLPrjO
cXGXVF60l3OqcyFtnluMoS+DMcNJ5mQ9cqT1/VnuTx4hQ13YFGRkwx0urbb7JGL8QhFe3XOL9E13OWl
PxXUu0Y8f0OvEknjb1ypIDW7JjTVGcotqXbRrOShCUn2itkOf9Up/D8iy/6vZ91/kB5TdG6rzEml9phLPht9
EJziu8kuDGLcfCZa+Ov7nWPZfXRGMMfcdb3vuBXXfC2p2Qe0u69UKLo319cU0t65JcWuyCvc4dEZLaX9T
vw36r/wCzA2jkRlkSpSfVFb36HN+XCmfRpzn8ImNX8Vt+7/gxxp2q62cKvMk3y99gLKMqF0nDTjNfyyNyB
xvsy6rXR0aem0/QvA+bZlN5lc1CxJL3PV9zvxCfVTTPTjt70/Q6v/ilP3f8jxPmFa/+QHUvEIJ7Vc3D/topjZGV
asT9nW9nGRFLFsSS0oPSInJrwmOvV6/uBu/EIbbbjXXOUV3kkUV2wtr64v2fyJKbL4UxjHG3HXx7nuHTYqrq
7IuCn2/EDp+IQ2+iuc4ru0iim6F8OqD49V8COt5GHDodPXBPvE7qnT9Fusx04vTbXwYHdmdCM3CEJWN
d+lGlGRDIi3De13T7oiw53V0/8AHR1Jv3t9zXGhb9MlZKrojJcrYFpPdmQrn0RjKc/VR9Ch9j5eHZbHrnCnz
HJ8y2BdRlQvbik4yXeLMMf+I3/I5hG+ebC2VPQuz5Osf+I3/IDa/LhTPo05z+ERRlQuk4acZr+WWRjTtV1s4
VeZJvl77GjjfZl1Wujo09Np+gG9mbXVbKuSluPwXc5hn1ufTOMq/vHEEn4tZtdo8f0R54nFPytr+bQHb8Q
```

gufLs6P8AtopU4uvrTXTrezjKS+i2LXCiyOUmvCY69Xp/1A2fiENtxrnKK7ySN1fCVDti9xSbJabL4UxjHG3HX
x7nlFdleNkqcHFOLaX4MDT9oVuK6ITk/VJdjSrKhbVKcVLce8fU48OSWKmly29nGLxn5CXYDKvKazJzcLG
mvc9V2PpEVP8AFLvu/wCC0Ce6VSy6lODdj91/A9uy66beiafbezHJ/iNHyPLUpeK1pra6f8gdxz4OaUoTgn
2bRWR+KL/x4v8A+RXD3I/ICPxC9xg64xmnw+pdjqN3m4VicZLpr7y9eD3xH6o/mjuX1B//AKv0AlxcuNW
PGChKcIttJdizHyIZEW47TXdMy8OSWKmly29nGLxnZCQGt2ZCufRGMpz9VH0OqMqF7cUnGS7xZDh2W
x65wp8xyfMtmsI3zzYWyp6F2fIF5hflwpkoacpv0RuQ4aUs3InL3k9L+oGiz69qLhNSb1po1svjXdCtptz7M
n8QS82h656v8DK+vY/++oFV1qpqdkk2l8ArY+SrW+mLW+TLxD6nP8PzJstteH0Jdmlv+gG37Qh38uzo/w
C2jeV8I0O5e1HW+CbzMjy/L+i+zrWtnEYWV+G2xsi098b/AAAuqmrK4zS0pLfJxRkRulNRTXQ9PYxPqtf3S
XEbisuS7rbX9wNrM2EbHCEZWNd+ldjSjIhkJ9O013T7oiwp3V1Py6OtN8y2aY8LvprtlV0KS55AuMqciN0
pximuh6ezUi8P/fZH3v8AIFE8iMMiFLT3JbTOb8qFDUWnKT9EY3/xOn5f5Or6bYZKyKUpPWnFgd05kLLP
LlGUJvspCyVX02uMoN2a4l6LuZRuruvgr6pV2p+zsXfxSn7v+QNbcyum1wmn23s5jnwdijKE4dXZtGc4qXi
sdreo7/se+KL/AIIP16v0AquthTBzm9L8ydeIQ1uVc4r0eu5xne3djwl7rfP9inJini2JpaUWB1RarqlOKaT+J1
KShFyk9Jd2T+H/AFSHzf5muS61RJ2rcPVfEDB+IQ5ca7JRXeWiiF0LKvMg9xJq7b5VpVYyVeuNy9Dnw/6p
b83+QHa8QrcV0wnKX/VI1x8mGRvpTUl3x8mGRvpTUl3TMfDIpY7euXLlnXHilqX/X/AABaYX5cKZ9GnOfwibnzMadqut
nCrzJN8vfYCjKhdJw04zX8sjzNrqtlXJS3FenqYON9mXVa6OjT2n6HsEn4tPa3qO1/RAa150J2tdq3
6jW+6FEOqb+SXqTeJd6X69Qy1151EJe73A6/dlBCvo8vmdeERfKUdRlLae+/YUgCPrzuno8ol+3ctbfYqA
HNScaoJ90kmdAAROu+nLnOuCnGf2nc6Zvxcfqj7CjpvfzKgBPnVTuoUa15VlaUFvUtlcACGNWRiz5
mRW4ShGC9dPuUV1SeF5U10y6jcASYUb615VlaUFvUtlcACGNWRizl5MVZXJ7032NqFkysc7mox1pQ
RQAJcWmdeRfKUdRlLae+/LPb6Zzy6ZxjuMe72UgCXKosdsb6NdceGn6k+ZLJnTu2EYQT7b5Z9IyyqXfT0J
pc72wMr6HfhwjH3kk1/Qzl9NtrdbrjHjTlvuWwj0wjH4LR6BNj47+h+TatN7Mq1l48fLjCNkV7r2XADHHh
d0yd8k3L+VehNVDKxuquuuM4t7TbLwBHjU3xy52Wpe1HHun8hKi6i+VmOlKM+8WWACar6VZcpWJVw
X8q52UgASZdVv0iu6mKk4rWhlVXX1Vex7Se5LfYrAHF8mxdL5/x+lxe00wNs+mxdxL5/DkqAEUPplEPLc
ZpcKWWzauq2VE43T9qXw/lNwBFD6ZyDpb7FxtLLtLZpiY0q4TjNfWO3TTvFuGgWDyWMRzZWZZGN3iuNmW
e0k+0F6G4AEXk341jTdkVxQSmLQpn7o7hpfFUNBDDCvJxW41n2VGBkUzUvJlxLzTJwVtvbJVL1DtHF5CO
q+lWXKViVcF/KudIAlhTNeITtGL7ZrGu5iJT+fYVcbPtVCbyhTUDiqX7Ww7o1VFcdsV/QnJgSX7KD5B2vZmN
quM0uFLZrGu54tkbZdU5J6Xw47FAAxw65VY8PZMKOoy7PZSAIrK76ksy21RhLtLZpiY0q4WO3TlSQBDCvJxW41RVlbe1t9jfHWQ5yn
e0k+0F6G4AEXk341spUJThLnpb7FoAq2VM94s7zgVXTGlaJHZ4lHo7PwpBUA8ZPHy1SYqmxuZ1Ps3TTxYnxgFRO1RnzaJ7KdUMj
uMHJjf5Kjz+nrS9YrpvOgMxkOzdfcoyq8l22Jfae6uUeWoA3CVSFAYHJVfwzXRNHTHzJXAJV4AD9XrZtZ6ZsbI7Yzj6Xz9c
Ymv6kG4PlZehiTIUeTyBGCRbrAHbd4VLGVoq3iTYEbK1FpabDzMsHPW4qJL0L/ZbG63tUNv1e5ibCRj+Q
W41yqB56pbLa2D9tjXuK4sFQqfUKNsE3ZrtXvNlwKu9vQFOeBbhvS5A1HszBDYnf8IS9KjRXwCbSRRsVbeHL
EiSIiS4yqYG3YNXLrVuCVvSrtq4lJ3P/UkjlgRRBjFSEHAVbZuMzQ8cfwaNz9pCqPWsCCa1ktWHVk6PURNK
NoIFrBVDPZ5YSDGgzd1PjvbKX0fjS7L0c0cZRgSYfk0gnz4cXbxoayXHlfxXfi5Ad0V/aDX9ydDm7z2kZL+3
1N5pFjUgJ0QatmRd0rl7ODxAEqSgU/BONRRm7Cwfx5KlItkTq8yiVWe8QqBZLEt5l1O1oqXdVFq5OW2
eAu2fhS+30i7Rd0kYb7LBbQHwp/pxAnsfdMumRmS/S5mqqdjdqsKqSEPzcmhWqRAiCC7LUVcjf0Sn9p
BzFGEqBIL0QnIS1yItHXzFLf9eRHcIEWsfsUf5AuvmOtIBrpL9KGzsi4YtzmOinHmeM+SypuA2o6hK3
eNBaJO36ilxNqaC8tpdUUu2xXhcuzmNb8spV9PfMLhmDnAFUZQxRx9L8YE8KaJW4tSlnt1ad06j8G/UV
kJ+3KyY9zRJmyzqmV1hpvZKaBxvi9BUmTSmYFTPHxK77MfjTOZq+xWW61qgxemazu6KNvqgZ3NfwTfal8
vmuSUFZw6EwEpJQfZm6dJ6BhAtkk3Pr/KXZrtSOO1UoYsYWSzfrZqCyKedDMVsa8gCCsTDeHf5QWmR+8
GHDmQVrN4+QsCKcxUS+kMwYtPTqtjm/pQdlBFAhaU03T/K/pV9UjpWYFcDDp2JXHttGTyp/Rm9PtQ
XsirVqeeJCKaGtZOyHGBiOkbtDnRrS2ZZy0S2GDF+xcHV6UudhVhn22IX1LiMV0SVR+uMUizMe21ku
nGBpnfq7X3nvwJLmM9cbljMHiBzP5kp4DaUXnD4vjuRUpiN1NW70VQtcc8SEzPNiO+Rl0qcBFVJwbRcu8
2m8GFXd8ubEU/3Q8Wg3rTV8G9qBZ0V+hYpcLuzc9aa

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAB83Jo1mVx8yftvvvtz6GuZGVGGoxsk31d2+RlfXsc78STeL8pICjf8A
w79ekixpyfh1snJtrfO/sN/pFSxlJzj7vbfJPi/w278fyAY9FuRQpTumo/ypM0wZz6rarJOXQ+GzTA+p1/j+b
MsTnMyfn+oHNaszrJydkoVRekondcLqr3U5TnVNe9/1OfD5xq8ymxqMlLfPqbrKjLIVMF1ccyT4QEax/w
Dz3X5k+2+rfJvmOULMaMZS76fPfscuSh4ruTSTj6/I9z+bsbX/AG/wB1mW2eZCip6lPu/gji3Gsordtd03KP
LT9Rlvyc2q6Xua03/vzNsrIrjjy1OLclpJMDO+52eHeZFuLeu3zOa8W26mM53yUmk4pdkcyg4eFalw29/3L
Mf6tV9xfkBj4fbKyhqb24vW2VEXhn7uz7xaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABxOmE7IzlHco9ns6lFSi4ySafd
M9AGEcLHjLar/q2zuNFcKnXGOoS7rbNABzXXGqChBaiuyPIUwhOU4x1KXd77nYAytxqrnucNv49jqqmul
ario7OwBldjVXNOyO2vXZ7KiufR1R30e7z2NAB5OEbIuM0mn6MxjhUQl1Kvn7Xs3AHNlcbYOE1uL9D2
MVGKjFaSWkegDiqmFKarjpN7fJ2AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAOYWQnvonGWu+nswzbnGKpr/eWcfJGPhkeid8fg0vzAvAAHLsgp9DlHq+G+To+XmqTzf
Y95R2tfYX416vpUvXs19oHcZwk2oyTa7pPsdEWF9byfvfqza/LhTPo05z+EQNwYU5cLZ9DUoT+EjpZEXku
nTUkt79GBqDG3JhVbCuW9y/se5F8ceClJN7etIDUBPaT1o+dmZEvOrSjOPRJ79OrkD6IM4XxdLtknBLv1G
H7Qh3VVjh/20BWDmuyNsFOD2mdADhXQdzq37aW9aMJ58FNxhCU9d3FGNFkbvEXOO9OPqB9AGN+
VCmSi05Tf8sTmnLhbPocZQn8JAUAxyMmGPrq25PskZPxCEV7dc4v4NdwKwY3ZEaYwck319tGlk1XXKb
7RWwOgZ03RtpVi4i99zB+IQ2+mucorvJICsGcL4TpdsXuKW2KLlfWpxTS+0DQGUMiM751JPcVts4uy4VT
6EpTn8IgUAxoyoXycUnGa/lkbAAc2S6K5S030rekeU2xuqVkeE/iB2DHHyYZHV0Jrp+J7HIjLIlSk9xW2/QD
UE1ubCFjhGMrJLv0+hpj5ML0+naku6fcDUB9jLHvjkQcoprT1pgagxryYWXTqjvcf7nssiKyI06bk1vfogO42
QlJxjJOS7pPsc13QsnOEXzB6fBnRKp5NqhBqa95/EywvreT979WBTXdC2U4we3B6fBoRYH7/J+9+rLQOK
roXJuD3p6fB2fMxMmNCsTjKUnLhRRZj5cL5OKTjJejA3Blfkwo11bcn2iu5nVmwnYoSjKEn26l3ApAMpZE
Y5EaWn1SW9+gGoM77lRX1yTa3rgTuhXUrJvSYGh5OcYR6pyUV8WSrxCHDlXOMX2k0aZU6vo3VYuuD1
2A3TTW09pgwsvhj48J9L6XpJL04M5eIQXMYTlFfzJcAVg5hZGytWRfstb2TPxCvb6YTlFd2kBWDGjKhfOU
YJ+z6v1NgAJ7c2ELHCMZWSXdRR1j5UL24pOMl3iwNgTWZtddsoSUtx+HqKs2E7VXKEoN9uoCk8nONcH
Kb1Fd2ekHidu4+V0yWmn1ejAppyq75OMN7S3yjYmqvhGiUlVKCrS4a1s3qsVtcZpNJ/EDoGVORG6dkE
mnB6ezH9oVvaUJuW9dKQFYMMfLhfJx04yXoz2/KhQ1Fpym/wCVAbAnqzIWWeXKMoTfZSXcmyMlvJr
ajZFRfK/7AfQnONceqclFfFnq5W0S5FtU8VTtrl0t+72aN3ZCulTfswSA7BJ+0Id/Ls6P+2ijzYul2xfVFJvgDsE
n7QrcV0QnJ+qS7G2PkQyItx2mu6YGoPJy6IOWm9LekfOrymsyc3Cxpr3PVdgLpXQjdGpv25La4NCDKsjX
4hVOXCUP8mi8Qh1JShOKfZtAVg5tthVBzm9Im/aEOG6rFB/zaArBPlZCjRuKlJTi9Sj6Gfh97lBVuM2+X1Ps
BYDyT6Yt63pb0cUXRvr64ppb1pgaAxpyYXTnGO9x+Pqe/SI/SfJSbkltv0QHcbITbUZJuPfT7HlV0LXJQe+l6f
Bliyqdt3lwcZJ+0369zLw73r/vf5Aqquhcm4Penp8HZF4b7tv3i1vS2ABH+0q2uITb+BtfkwoS6tuUu0V3A2
BNVmwnYoSjKEn26l3Or8qFFkYzT5W9oDcEi8Qh1JShOKfZtHs8+Cb6YTnFd5JcAVA4ptjdWpwfD/sYTzoK
bjCErNd3FAVAyoyYXxbhva7xfcY+RG9ScU04vTTA1Bk8iP0lUpNy1tv0RxdmQrn0RjKc/VR9AKAY0ZUL24
pOMl3izYAAAAAAAAAAAAAAAAAAAAAAAAAAABzZNV1ynLsls6DSa01tfaB83GyKvNnffPU3wlpvSPcK+
uN93VLXXJdPHfll/lV//jh/QmxMd122ysrSTluPZ/ECsGSd/wBJacV5OuH6moEUv4tH7v6HM19CyutfurO6
+Bq6bP2gren2Etb39hvdVG6pwl6+vwAkwWnlZDXKcv1Y8PXVbdZL3/XVbdZL3urXyOsDHsonZ5kdJ6099xKm6i+V
uOlKM+8WB54klF1WL31I9zP+LKou9N9L/wB/qeKm/IujO9KEIdoo3zKndjuMVuXdARXQeRPItXavi4P4dz
uyf0q3Gj8V1S/38CnEpdWMoTXL25IxwsWdN05TXC4i99wLSLxD99fSTqda3KD3oxDnxRvyY
r0cuSuMIxgoJLpS1ownVPKxum7Kum7nhkpZZe19hkpZsYeX5cW1wp7AeH+zZfBe7GXqyGXH9hsXVq9uNFjXdRZniY/wBHr
ab3OT22btKSafZ8MCXkmu8u4rPS1eVHldeViMKoqyDe1tTPkx22pcrun/AGAxqnYsy6Ysy
6cKvMkm137HdiyLr6pujcH3TQHT3XTeh9jpLx6Ihe9Jrnu2U/FZdX9fHzt6XlhcT1/CzTeZ0ZdBOe8rwYbFXq/FwX/J14ml9HT1z
1HuTRZ58b6NOa4afqT5ssidKwgn2T5bA1z/wB3R8yrJ+rW/dZllUSvuorbDqjpZvHR00ZT+m3VuDrjHzl77ga

8IWvV6/udUW3wohGGNuOu++5tTjt4Pk2rTe/w5M6/plEfLVcbIr3XsDnHrsrqyOuDgpJtL+pt4d9UXzZ3R
C11yV8tuXovQmqjl4ydcK4zjvaewOqP4ld8v8GGJZapWThT5kpPl77FGJRdDJnZal7S7pnjpvx7pTx0pwl3i
2Byo32ZldsqejXD5Lyan6VO5Ts1XBfyr1FF07Mq6O9whwuAKWtrT7HzK7Po+PkVN8xel+PB9MivxJWZsZ
pew9OT2BxRD6Jk1p8KyHPzO8Lbrvv9ZN6NM6iV1S6FucXwa49flY8INcpc/MCDCndCuTrp69vmWzWm
FzzlbKroTWnyI1ZGLOXkxVlcnvTfY2oWTKxzuajHWlBAUEEJ/RcnIi+zXUv9/EvIs7FndZCUFv0lz2Awqg6JY
97/AJ21L8SjGXm5t1vovZX+/ga5VHmYzrguVrpGFS6aEprUm9sDHE+vZHz/AFGF9byfvfqzTHpnDKunKO
oy7Pfc8xaZ15F8pR1GUtp778sDjA/f5P3v1ZaS4lM67b5TjpSltc9+5UBF4al/yvXPVrYyUo+IUSj3lwzimrLx3
KUIJpvmLf8Ac0pousyVfkJR6fdigMZSn+0ZyjX5korhb7HuSsm9R/8AH6XF7TTNsiixXq+jXX6xfqE8y2cU4q
qKfL77ArXYiu/ilP3f8lpLl02SthdTpzjxpgPEvqv/ALIZFDyMSEYvUkk1v14Mb4ZmRDUq4xSfup9ze+idlEOiX
TOC+PcDKeRJQ6MvHfS+G12OsxQWAvL9zjRzZ9MurdTqjFPhy2d3Y8/oMaYe1JaAzzvqFPzX5FcoxWM4
pLXR2/AwyqLLMSuEI7lFra39hTJN1OK79OgPn1ya8Knr46/uizDio4tevVbZnjY8lhyqtXS5NmdSzKI+XGEZ
xXuy2B7hpLMyElpJ/qWkHh6l9Iv6nuS7v7dl4EULdWTWJR1c+1JvucVOb8STsioyceUn9h7CrJxZyjVCM4S
e1s9qpyFmRttSaa5afb7AFST8Ut2t6jtf2PfEP3uO/Xq/wd1Uzjn2WOPsSjpPfyPcymds6XCO1GW3z8gKSP
xT6vH7/wCjLCbPqndTGNcdtS33A6y/qc/unuF9Ur+R3ZX5lMoPja0SVLMph5Ua4tLtJsD3B+sZP3v1Y8NS/
wCWWuerWzrBotqna7V7zXO+/c6waZ1Rs8yOty2uQM5ceLR16x5/oxirqz75S7xekaSpm/EIWqPsKOm9
/M5totryHfjpPq96LA2ux4XShKW04PaaJ8v67j/P9T1V5GRbGVyVcI86T7nWbTZKddtS6pQfYDzxP6r/AO
yM87mGPD+WT5/saZMLr8SKcNWb247NL8fz8eMN6lHWmBs4RcOjS6da0fPxX/4eTHe0k9f0NHLNcPL8
uO+3Xs0rxnVhzrjzOUXv7XoDzw5JYqaXLb2cYvGfkJdjfDrlVjxhNakt8HFFM4Zl1ko6jLs9gUkVP8Uu+7/gtIr
K76syV1UFNSWu4HmQlLxOlNbXT/k78SSeL8pI9spslnVWqPsqOm99u53m1Ttx+mC29rjYHGRKqOLW7
o9fC0vi9Gd1mRPHn1URjDXq+UaZOPO3HrUeJw1wGC1y99/sA6o/hb+5L9Tvw76pH5v8v8z8eOM5v8z3
HpmsLyprpemv6n6uz6K8qvtrmP+/ij6JFl4krsmEor2XpEoR2XpS5Ayog8a6iT7WR0/8AF6G
2F/yXXX/F6Xy/3Rpm0u6jUF7UXtHeJU6ceMGtS7sCfA/f5P3v1Y8O96/73+TvEpnXbfKcdKUtrnv3PcKmd
UrXOOuqW1yBn4b7tv3iyXuv5E+DTOqNnmR1uW1yO/gNZnmR1EWxq78GHvQ/sGzmmxc8KXp2FYivV9GuIJo
NMo2R03Lff7Ec5FFivV9Guv1i/UDHJWTeo/8AH6XF7TTNsiixXq+jXX6xfqE8y2cU4qqKfL77ArXYiu/ilP3f
7A88RSeK/aeKxI3ZpXNFJgWuWEPSkw+hzUMFvSb31r/7Fb2sWTbPUD7LPhGBRVSTGeLyS3ZVtVwqvD88
7A88RSeK/saNseKWNWtcOKOM2uVuO4wW5bXBrTFxphF90i13ltv+pzVV9Guv1i/UDHJWTeo/8AH6XF7TTNsiixXq+jXX6xfqE8y2cU4qqKfL77ArXYiu/ilP3f8lpLl02SthdTpzjxpgPEvqv/ALIZFDyMSEYvUkk1v14Mb4ZmRDUq4xSfup9ze+idlEOiX

mY1VzyY+bbe477JPWgPoAkwrZ9dlNkupw7S+KKwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD5s661nzjfxGXMX
vRpOjChKKb25PSSlsrtpruWrIqSOK8WmqXVCHPxb2Bhk/xGj5GeVCCz07k/Lmu5dKmErI2SjuUez2e2Vw
tj0zipICOyjCrScn3+EtjLSWRipdk1r+qN4YdFcuqMOV229mk6YWTjKUdyg9p77AZ531Oz8PzOsT6rX907s
hGyDhNbi+6PYRUIqMVpLhICHDj1rKivV6/MyxKseyLjdxZF9m9H0a6YVOThHXU9vk5txabZdU4c/FcAY48
MWORqp7ml8do5wP3+T979WVVU10rVcUtiumFcpShHTk9vnuBLkfxKj5f5Ga1HLx5N6W+/wCJVKmErI
2OO5x7PYtpruilZHeuwGWVfXCuPXHrhN6bXYmvpoqrdtFvTL0Sl3Lo01xq8pR9j4PkzWFjqW/L/q2B3jTlZ
jwnL3muTzK+q2fdZr2PJxU4uMltPhgYYH1Ov8fzZl4d71/3v8ldcI1wUILUV2R5XTCpycI66nt8gSXNUeIwsl
xCS1s0zb61jSSkm5LSSZRZXCyPTOKkvtMoYdFcuqMOV229gS5EHX4ZXGXfe/zLl+4X3f0PbaoXR6bFtd+5
1pdPT6a0BJ4X9Wf3n+SOYfxWz7v6IqqqhTHprjpb33CpgrXao+21pvYHZCpKHis3JpJx438kXGVuNVdJSn
HbXrvQE+V9dx/n+pr4h9Tn+H5msqa5ShJx5h7vPY9srjbBwmtxfoBDlQcvDqZJe6lv+h1CjClWp70vXcuxb
GEYwUEvZS1oweDjuW/L/uwPcRUqpujfS36nniEHPFlr05N4xUYqMUkl6I9Awx8ip48ZOcVpaab7GHhzU
p3tdnJP8zd4WO5dXlr+r0Y+HJKzIS4XV/kDnw+ca/NhOSi1L1egrI2eKRcHtJa2vkymzEptl1Thz6tPWz2GN
TXNThBKSWk0wIp11rPnG/iMuYvejSdGFCUU3tyeklLZXbTXctWRUkcV4tNUuqEOfi3sCe+Sr8TrlJ6j09/6j
xK2uVMYRknLq3wyu2mu5JWR3rsZrCx1HXl/3YE/iEP+OmbTcY8SPfIwvL6+pdP3i1xTj0tJrtpmH0HH6t+X
/dgd4yrVEfK30PlbMfEoOVCklvpe2VpJLSWkg1taYGUcmqVXmdaS1zzyiXCTlHIt1pT3r+5Q8LHct+X/AHZ
soxUelJKOtaQEvhn1Z/eZzV/Fbfu/4K6qoUx6a1pd+54qYK12qPttab2B86FVX0u2vI429xe9G6pw4XQinu
bfGnsptoru15kE9ep5VjVUvcIafx7gakE7I1+KNzek1rf4F5nPHqnNzlBOTWnsCbxC6EqVXCSlKTXCezTJp6s
Hp/mhFNfgd14lNUuqEOfi3vR5k5UKE0+ZtbS0BLht5GSrJf8A04Jfj/uz6JPgVOrHXUtSlyygCG5qjxGFkuIS
WtmmbfWsaSUk3JaSTKLK4WR6ZxUl9plDDorl1Rhyu23sCXIg6/DK4y77T/MuX7hfd/Q9tqhbHpsW137n
Wl069NaA+fiwdnh1sV3bev6I5xKcW2r23qa77lovqqhTHprjpb33OLMSiyXVKHL9U9AZ4sMaNs/IbcktN7
4KIWQsTcJKWu+hXVCqPTCKijyqmulNVx6d8sDsAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAkl2SWwAAAAAAAAAAAAAAAAAAAAAAAAAAB44RbTcU2
uzaPQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAcSuqhJxlZFNejYjdVOXTGyLb9EwOwAABz1wU1B
yXU/TfJ0AAAAA8nJQg5S7JbYHoOa7I21qcHuLOgAOY2Qm2oyTce+n2OgAAAAHEroQtjW37Uu3AHYAA
AAAAAAAAAAAAAAAA5ssjVBym9JHFGTXe5KG+O+0BqAAAAAAAAAczshWtzkortywOgAAAAAAAAY3
ZVVMumcva+CR1TfXem623rvtAaAAAAAAAAAAAAAAAAAAAcxshNtRkm499PsB0Diq6FrkoPfS9PgVX
QuTcHvT0+AOwAAAOKroWuSg99L0+AOwAAAAAAzsuhXOEJPmb0uANAAABzOyENdUlHfC2z2cowi5S
eku7A9BjTlVXzcYN7S3yjYAAcQuhOyUIv2o9+AOwAAAAAAAAAAAAAAAAAAAAAAHFtsKYdU3pb0dp7Sa
7MAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAPnRjTZmXu9rpT0tvRVTj46krKkm16q
WzlYVErJTe59T3y+xioLF8QhGtvomuwF4AAis/i1f3f0Ztdl1029E0+29mNn8Wr+7+jPLUpeK1pra6f8gdxz
4OaUoTgn2bRWR+KL/x4v8A+RXD3I/ICLxC9qLripprppr2l2O3d5uFZ7Eo6h/Mu/A8T+q/8AsjS76jL7n6ATY
2XGrGhBQlOS3vSQyL3v+pnh8ZeQl23dM/D0M/D0liQaXL3v+pnh8ZeQl23oGuLKp23eXBxkn7Tfr3P7EbH
CEJWNd+lGeB+/yvfqzyEL8Oc+itWQk98dwKcfJhkJ9O013OTbdOcYOFv8yZUBNdmwrs8uMZTku
6iTu6N+fTKKa1w0/TuaeGJOFlj5k5csXpLd1z/AHAovyIY8U5LbfZIY8U572+yXdmCU1CcJVt9upeUepE98p7tYd
biuF+B7k/SMiCi8fT2nsCy++FEFKe+XwkPEOqb121+yvH+13h2BLc65MP2M7nk5M99FNWeuPU7bxM8K6cMPkR
McZXNPpeuPU1hJThsGS7SWyG7+FR+SK8f6vV9fxkB5j5ciLcFqM3RcSaal7ORU7gM2M6f5fYhKxkLbeCSK
/yOcOd1dP8Ax0dSb97fcSb97feAolkRjjZkRjRp0+qSGCxiiKlSCe6G6vLOeU46zPRePHkXZWrUt+0vb+/8Td
0PL8mFGlLbV4rV93/JI12ftCyKUzMRxSSssyT65B3U4eM3fpEy4V6RSIRzQ8iWf9ydDQt5H4YtddWHbDBR91kxs
qca+mSUZd9e4wvK9K9ZRPk6cGCBpN6MZU6Dq+4Pr/UzWbOaxXK/qLuWu4GT/IajZQDmaHQ2FLp1+QSmLbOZhh
t/3gO14hXKPswZ6m5f9UjXHyepoZEDaSczZLumYeFpeRKWuerW/wEOPFZ73/4gNR9omM+F2aTtk+CXr7+Q1K
E/hIxwV1ZGROXvdWvl3KLMeFlsLHtShHtSh216gRWZeRIcHtSh216gRWZeTeZXNQsSS9z1fcoyrKnRCd1cmm+F2aOL/4pT93/I8U/
cQ+9+gFVtkKa+q9x6D0kTrxCHDlXYyov+bRznyqmDMq5ncqSUqdTDkfJXZ5P/UR9Ztjxd+jPjkUk4k7krn9
bel2MceTfhlqfo3pIpYkKYWkbhSDQHoPUwk5cvkl8YXgC1EhlqbSpzS1b49fqksLvpqNVKx8W+r/oYNOQd
bk2t3GX/QN/i9pL2L+EHuz2o9Wr37K9fmVTwnPjlMqZt3VVPSZt00s/GJCOcR3WgNoMZjWrOUuJyJf2ISbuVSk
pOT44430l0La1kypjDUtbbS4ZhWn736sQ/i0gBDZ98V++PJ2vJeEy69JqEl+0aBK6wsHDTcsyI2+bsu09L9pU
9R2v7G+bQ8inps+Ms+iqOpPxTCemOfkUD5uDvf8+HYlPyDNX3Q5dwfvaJ77ZfgFNZcPx63pnM8WvYuq2Dpz+Pv0B+PQ8Uw+YN1/VAYfqnXZw2R9LBt

AwbnZXGDjPaXvPsze2xVVym02o86Rnh/Va/kayipRcX2a0wOa7Y2UqxcRa3yc4+RHIi5RTWnrTII2uvCtpf
vKXT/X/Wa0r6HkuEn7Mob/Ff6wKq8iNl064p+x3foZ2Z0IzcIQlY136UZYqlHCuu/mnt7M8Od1dP/HR1Jv
3t9wLaMiGRFuG9run3RpJ9MW9b0t6I8aFv0yVkquiMlytloGdF0b6+uKaW9aZzTkwunOMd7j8fUkrs+iv
Kr7a5j/v4o8og8a6iT7WR0/8Af6AW/SI/SfJSbkltv0Rxiyqdt3lwcZJ+0369zPC/5Lrr/i9L5f7o8wP3+T979W
A8O96/73+R4b7tv3h4d71/3v8AI8N9237wFpNZnQjNwhCVjXfpRtc3GmbXdRbRP4bFLG2u7b2Btj5ML0
+naa7p9yLFyI0SuTUpSlLhJfMtWPCN7uW1JrTXoTeHRXmXy1z1a2BvRlwum4dLjNejPb8mFGlLbk+0V3
MLePFatesf8mXXZ+0LJQr8yS4S32AqqzIWWeXKMoSfZSXc7vyIY8U572+yXdkeQsi9wf0fpcHtNM0muvx
SKl2jHgDuOdBzUZwnXvs5Lg4zfrWN979UU30wvh0T3re9oj8Qbrsx3Hlx7b9ewFV+TChxi05Sl2jHubJ7Sb
Wn8D5+BqWRN3b8/wC0+gBPlyqTr82Dlt8a9DHxK32PK6Zd0+r0PfEfeo+9/g68T+rL7yA0w7Iyr6Y1yh0p
Lldzu+3yob6JS3x7KO4e5H5I9l7r+QEHh1715clOTcve7pcHNd8aMy9y223pJLvya+F/V5ff/RHmKk8+9tcp
8Aa05kLbPLcZQl6Jnd+RDHinPe32S7snzOM3Ha77/UzvlP8AaW4w63FcL8AKK82EpqE4Srb7dSO8jJhjyip
p+16r0Jcn6RkQUXj6ae09nuYuq7FU13aTT/ACijKjf1NRcYxW9szfiENvprnKK7ySO89uOJLXrpGFFt8KIRhj
bjrvvuBZVbC6CnB7R2SYFdlbt64OCk9pFU+nol1e7rn5ATSz4dTVcJ2a7uK4NqL4ZEeqG+O6foTU3TcWsX
HXl77t9zzA6vpV/UlF+qXzA7fiFabXRJyT1o1syoVUxsntdS2o+pP4dFeZfLXPVrZxkyn+0Y9MOtxXEQN4Z0
HNRnCde+zkio+fk/SMivpePrnaey6frflQ6ve0t/MDy22FMOub0vzj14hDa6q5xi+0mDPxByeRTFR6l36
e3TyLqnW8bSf29gO/EnvFTXcFqK3Tdvy/MDy22FMOub0vzJ14hDa6q5xi+0mDPxByeRTFR6l36fjy
e3TyLqnW8bSf29gO/EnvFTXbqRv1xroU5vSSRHkRlHw2uM1qSetf1HiDfk0RS2n6Aa/tCHd1zUzP+2je2+N
dHm+9HjsTTsyJ1Ov6LqLWu5nOE6/C3GxNNS7fiBs/KlyjTGivota0tOC/Im8Li
lCx653oDfHyoXycUnGS9Ge35UKGovcpPtFGD48WWvWPP8AQyjOxZ104VeZJNrv2AqpzIW2eW4yhP4S
Xc9uy66beiafbeya5ZF1lcnR0uD7pnVqUvFa01tdP+QO458HNKUJwT7No48Qvai64qaaa9pdjrxRf+PF/w
DyHiH1KPzQG2PerKtuMoqK5cvUyfiENtxrnKK7ySN+jzMbo3rqhrf4ElcsjEh0Sp64L1iBZTdC6C6HXB7X5HZh
hul1uVCcU3yn6M3AAAAAAAAAAAAAAAAAAAAAAAAAAiVV+LZLyYqyuT3rfY6potsyfPvSi0tRiisAZJ3/SW
nFeTrh+pqABLOmx+IQtUfYS03v7GJ0zfiELVH2FHTe/mVACfOqndQo1rb6t9zeK1FJ+iPQBhmUyux3GPvb
2jiCvsxbIWVqMunUee5UAMcOuVWNCE1qS3tficY1M4ZN05R1GT4e+/JSAJcSmddl7nHSm+Oe/c4hDKx
nKMIq2De02+S0ASY1Fvnyvu1GUlrpRWABD5N+NdKVEVOEudP0EacmeXXbbFaXon7pcAJcnHsdsbqGv
Mj3T9TlyzbWo9Mal6y3ssAE2XjzscLKn/yQ/uYZMsqePLzK4wgu/Pc+gZ5FbuplWnpv1AxjV53h0YJ6bitGc
Ppsa1Uq4rS11t9kV0w8umMG99K1s7AlwaJ11ThbHXU/j3RnCvJxW41RVbe1t9i4AYY6yHOU72n4Hn4+nJ+9F+pWAJIvMtsjuKqint+uysACbNpna
q+iO9S2+TW+vzqZV71tGgAhr+mwrVSrpxwpNeFRZVCxWLTb457ylYAmwKp00dM4ak/UqFbwd9JgScqYx7pDIGjg2Eq1+rT5kbLeQwEbj15ZasyWp
db2UgCODdN1ORK3HSkp+9FfMqQoIjz6NOVxBzvh+4Oeg95KE1/+XoXJuAqn75sI6qMtThiAfWEoQlb7eGntaKLpiqPm51B6y2sv0iX9tuvYa7NpKA
dM8s4WGJiSNoC1LL6Qaz1nfIRS0jZfkYR1Xn5V7xt0AA7jEjSbtDLdoAQzs2StNAP8gVQ8HCDyGK9QqJ+A==
KcdKUtrnv3KgBNhUzqla5x11S2uRg0zqjZ5kdbltclIANJpp9mRQrycVyjVFWVt7W32LQBNj02+bK696k+F
FdkMKmdUrXOOuqW1yUgCWymcs+u1R9hR03v5nl9Fscjz8fTk/ei/UrAEkXmW2R3FVRT2/XZ1lUTlZG6lr
zI+j9SkARTjlZOoTiqob5afc7yKJyux3CO4wfL327FQAly8eU5Rup4th6o6AE2bTO2VXRHfpQTnHpl6o6AE2bTO2VXRHfTL
b5Pc6qd1CjWtvq33KAB5Fagk/RHr5QAEWJXkY83W4W4+5mmPTOGVdOUdRn2e+5SAJcSmddl7nHSm+Oe/c4hDKx
dxi+XvtyeZOPY7Y3UNeZHun6nlC3qj7SmmdmyTROM
ivdey0AZY0blFu6ae32S7I0yKcq5zhvoFfZ7oAOLYeaqJRS5FeNXPx6m60JtNyAAAAAAAAAAAAAAAAAAAAAA
bXIysec7I3UtKyPo/UpAEblm26j6Rq1KZYk5VWhluzM55Ff9zhntXAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
aa2+2zrIx/Px4x3qcdaZQAI1POUejyq2/wGVg8pCzVe7sdPfXVUAc+AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
W1yVACZ0zfiEbn2FW9nN1FteR5+Ppt+/0Mmk2kYnTPE0Kh8oCD/cy7eK1O9qqmI8wAV2VC38YHcmb8Qnem
FGtbfVvuMqmVO0R95aeigATVRvtx6Phj6rmovsAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAADK3IjXbCtptz7M1AAzyHaq90xUpb7M7jvpXV31yB6AAAPJS6YuT9Fs4ouV9fXFNLeu
QNAS4ds7Lb1OW1GWo/JcncqAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAyyMiCi5JvqeuDUAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA+fYrbPELK4WygtLfPppC6FmFKNkbJTg
3ppnvmRr8Vm5tJNJbfyR74jbGdcaoNSlKXZAe+ITmpUeXJpyb1p/I6UJYlVls7HY9evxOM1dNmKvg9fkUZ
sHPFmo8vuBLVTO+Hm2ZEoylykn2NcO6co2Qm+qVfr8THGpxLaU5PUl725aN8f6PBWyx05OK5+0CfHi8
vqnZfJS3xFPsVYqurU438xj7sm+6MYwxcqLn+7l686GHKU/Opc3OK4jIDmvqzZSnO1wgnqMU9HeNOd
WU8ec+uLW4tmGJTRPqhd7NkX6vRTRXixyEqnuaW+HtAY1xtvyb4K2UYKT3z9p1FTxcyFfmSlCfxOsL63k
/e/VjK+v44GWTR05VUfMm+t92+3PofQqh5dah1OWvV9yTNajl48m9Lff8AEtTTW0019gEviMpRxtxbT6
l2ZxnzlHFqcZNPa5T+w68T+q/+yM/EPqlXzX5Ae24tvlStlfPzEt6T4KMOx240ZS5l2Z3f+4s+6/yMfDvqi+bA
7y6vMqb65R6U3w+5P4dV/wAas65d2unfBZb+5n91k3hs4/R+nqXVt8AT49U7b74xscI9Xta7vlmlanjZsa
vMlKE16nWB+/yfvfqxkfxKj5f5AZNk7clY8J9CS3KRxapYcoWQuc4N6lFvZzkV1rxB+fvomuGaWUYVeup9/
hLYFy5RHmWWSuhj1S6XLltFiWkkuyIcl+T4hVbL3GtbA5ycezHoc4Xza9U2d5E5Lw2uSk09R52dZ91bxXF
TTctaSe/U4yE34XDXoogV1vdMX69KJcG1xxLLJty6W+7+w1hkVLGjJzj7vbfJNiQdnh90Vy23r+iA9qrnkx8
229x32SetGuHbPrspsl1OHaXxRhiU4ttXtvU133LRRiwxlbPyG3JLTe9oDzw2UpUycpNvq9X9g6pftPp6n0
9PbfHYz8OsjXCyE5KMlLfL0K7I2eKOUHtdOt/gB7bOeRlSpjZ5dcO7T7njcsS+vVrnXN6ab3ozdVSzrI38KT3
F70aypwoTjHe5N8alsC4jyLLLslY9cnBJbk0WEMpKjxJznxGxcP8A35AeXVW4kVbXbKST9pSOs+1vFrnBuP
U0+H9h1n3Q+juEZKUpa0lyY5sHDBpi+6a3/QCnHx51z8yy1yk1yvQ3n+7l8menNn7uXyYEGJVbk1bndN
QT0knyzTElOvKsx5zckltN/wC/adeGfVn95nNX8Vt+7/gDxuzMyZwjNwqhw9ep5PzMGyD8yU6pPTUvQY
slRlXV2Pp6ntNjPnG5101tSk5b49AOfEaulxn1yfVLs3wi6mryoOPXKfO9yZL4nxXX8FIr8yHQ59S6V67A6B
5GUZxUotNPs0egAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABB5cLPE7YzipLp9fki
mrFpql1Qhz8W9m2lvelv4gDiymFkoucduL2uex2ABhPDonLqdfL+D0a11wqj0wior7DoAYTw6LJdUocvv
p6Na64VR6YRUUdADK3Gpue5wTfxXB7VRXSv+OKW/U0AHEKYVzlKMdSm9t77iVMJ2RnKO5R7PZ2AOL
aa7opWR3rse11xqgoQWorsdADm2qFsemxbXfueWU12wUZx3Fdls7AHkkpRcXymtM8rrhVDpgtR+B0A
BlDFprs64Q1L5moR05Pb57iVMJWRscdzj2ezsAc2VFcuqMdS+J2OtqmeAbc4NmhAHNlcL
Y9M4qSOgBDl4tNOLOUIafUpgtaDRtbTWk1pGL5yo6YVMd2NVTuTZbdB6pnNsDAnBTGE5WYVuTrAR7vLBawHouSEFVuJvUXZNd3pZVSVHuVXUvs
GzEosl1Shy/VPRpXVCqPTXFRR0AMbMSiyXVKHL9U/tnsMaquanCCUktbr6J4agGiOarIqTzqRbMrtVUQqL1Qhz8
XyagAc2VQtj02wUo74aOABGvEpd1Rhz8W9ndtMLoqNkdpPfc7AAMbmqqFkemyKkuX7xaqFMumqYeO
bWm9nYAztoru15kd6NpgnGLJLLj4yJ0jEcvnVqftLzoYmgTs2UNG64prEVLiNFsZTw3O6G
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAABtJbbSX2nkZxmtwkpJccPZ8/ccu6U7rOmmL1FN62W0RqhDVOunfo9gaAADnrgpqDkup+m+RZZ
CqHVN6RJZ/Fq/u/ozPxC3rnCtwlqMuft+QFlGRXf1dG/Z77RqZ49ishtVuCT1prnRm3eXU4qMm5RFK9AK
Diq6Fybg96enwT4F7nXGtxm2k31Ps+SfEyY0KxNSlJy4UUB9MGGPlwvk46K1ZaCZL0bcn2iu4GWJ6s
yFlnlyjKEn2Ul3F+ZXRZ0TUu29oCgGNWTGyqVjVh5GPxMv2hDv5dnR/20BWDyE42QzVovcX2hy/pfR4x
D0Ej8Qhy41zlFd5JFFV0La+uD2gOwR/tKtriE2/gbX5MKEurblLtFdwNgTVZsJ2KEoyE9updykADO++FEdz
ffsl3ZjDOg5qM4Tr32ckBvbdCpxU3rcepEjYeqlwdkXiPvUfe/wAFV10KYdU3pfmB2CSOfDa64Tgn2k1wbZGRGiC
lJNpvXAGoJJeIQXKrm4f9tHUs6tcVxnxVs9Y9b9ldgKQZCZtVsJ2quuUJQb7dRhn5D6DPKnHpl37F37Ks6zjZ
NrrtlCSluPw9QKQTVZsJ2quUJQb7dRhn5D6lCKnHpl37F37Ks6zjZ
BTg9pnltiqqlNptL4AdgkfiENbjXOS9dLsb0XQvh1QfzT9ANARvxGvlKE3LetG1uTCquM573JbUfDYE1eb
CU1CcJVt9upGt98KI9U/Xsl3YGgJY58OpKcJwT7NrgotthVBzm9IDoEi8QhtOVc4xfaTRtdfGmpWNOUX20
BqCSXiEFzGE5RXeSXBpPJj5CshGU1LjhdgNwQeHXvXlyU5Ny97ulwXgcqyDm4qScl3W+x0T0yqeXaowas
S9p/E5ln1xnOLjLqi9aS7gVAnpzIW2eW4yhL4MoAHF10KYKU3pN67EORkt5NbUbIqL5X/Y7z5+Zhwn0u
O59n39QLlygRvxCEVxCcorhy1wVV2RsrU4v2WB0CSWfDb6ITnFd5JcFFN0LodcHx+QHYBLPOgpuMITs1
3cUBUCenMrut6IqW9b5O4ZEZ5EqUnuK3sDUGV2RGmUFJN9b0tHt98KIdU38ku7A0BLHPg5KM4Tr32c
lwa35EaXBSTfW9LQGoObJquuU2tqK3weVXRspVvuxa3z6AdgkfiENtxrnKK7ySKarI2wU4PaYHQM77fKh
volLfHsok8OvevLkpybl73dLgCyF0J2ShF+1HvwdnzqGRkt5NbUbIqL5X/Y7z5+ZhwlTAoBlfkQx4pz3t9

ku7M682EpqE4Srb7dSApBjfkwolBTT9r1XoZftCCabrmov+bQFYCakk09p9jK/IjR09SbUnra9ANQZ33Ror
c5ba+CPY2xlT5v8ALrYHYMqsiNlLt04xXxMf2hDuq7HBfzaArBzXONkFOD3FnOReqK+uUW1vXAGgOLLY
10ux+6lsU2xuqVkeE/iB1KSjFuTSS7tnMrYRq8zq3D4rkx+kV341knCThHhr4nNjg/DG649MdcJ/MCjzoKn
zd+xre9HUJqyCnHs+xHL+Ff8AqvzN8P6rX8gNgZ33Kivrkm1vXBg/EId41zlH1euAKwTPOq0uhSm2t6S7f
M7x8qF7aScZLumBsDC/KhRJRacpv+VHlWZCyfRKMoSfZSXcCgGTyIrJVLTTa2n6DIvjjwUpJvb1pAagyuyI
UVqU9pvtH1MoZ0HNRnCde+zkgKgD5/iF710RU4tS97snwBdZONcHOXZdz2E1ZBTj2fYlvt87BtfRKOuNS
RnTmwrohFQnNxXOlwgLwZ0Xwvh1Q38Gn6GVubCFjhGMrJLv0+gFIMsfJhen07Ul3T7mrek38AMrsiujX
XLl9kjuqyNtanHen8T5rvUs7zJVTaS4jrkssy4UqvqjJKa327AUAwx8qN/VqLio87Zm8+HU1Cuc0u7SArBnR
fC+HVB/NPujF59alKPTJyT1pLuBUCejLhdY4dMoT+DJbspvKrkoWJLvH4gfSBzXPzIKWnHfo+50AAAAAAA
AAPJb6Xrvo9AHzvD8au2uU7F1c6S+B1CP0bxFVw9ya3r/AH5HSpvxbJOiKnXLnpb7HWPRbLId9+lLWlFe
gFYMk7/pLTivJ1w/U1Ais/i1f3f0Y8Q9+j73+DudNj8Qhao+wlpvf2M9zKZ2zpcI76ZbfIFJllfVrfus1ObYeZV
OHbqTQGOB9Th+P5mXhqX/ACy1z1a2dYUb615VlaUFvUtnWDTOqNnmR1uW1yBnLjxaOvWPP9GZdd
n7QslCvzJLhLfYplTN+IQtUfYUdN7+ZzfRbHI8/H05P3ov1AxyFkXuD+j9Lg9ppndqUvFa01tdP+TuLzLbI7iq
op7frs9nTN+IQtUfYUdN7+YHPijaoil2cuTzzMjy/L+i+zrWtlGTQsilw3p90/tJ4yzYQ6PLjJrhS2BpgQsrocbIt
Pq42a3uCpk7eYa5QojZGpK2XVL4jJq86iUE9N9gJ6rbpVpUYyVfpuRz4b+5tX2iH02MFUoRSXClvsjvCpsp
hZGyOtvh77gc+FxXkSlrnq1v8DKUp/tGco1+ZKK4W+xTgVTpplGyOm5b7/YjnIosV6vo11+sX6gY5Kyb1H/
x+lxe00z6K7EieZbOKcVVFPl99lYHzsmU/wBox6YdbiuInuT9IyK+l4+udp7N8rHnOyN1LSsj6P1OHLNt1Ho
jV8ZbA4zN9GN1e9tb+fB5nuTy6oqPXpbUfjybZlNlrp6F1dL232OsvHld0zrerIdgMb5ZF1Tg8bW/XfY4y1K
ODSprUk9NGrnmzj0eXGD9ZbPc2i23HhGPtyT5fC9AN74r6NZHS0oPj8DHw1JYqaXLb2UWxcqZxXLcWkZ
YVc6sdRmtS2+AMcbjxC9LtowxLLVKycKfMlJ8vfYrppnHNtslHUZLh7M3Tfj3SnjpThLvFsDlRvszK7ZU9Gu
Hye1JPxS3a3qO1/Y1p+lTuU7NVwX8q9Tyqmcc+yxx9iUdJ7+QHHiH73Hfr1f4HifuV/eNMymds6XCO1G
W3z8hnUzuqXl8yi96A0yavOolBPTfYlV1tFXl30brS1uPwNnC7JxmrF5c98aZm5Zvl+W6ovjXVsCjG8ryU6e
IM5zfqlny/U9xKXRSoN7e9s8zfqlny/UD3DSWLXpd0YYiUM6+Efd7nNEsuGPFQrjOLXsvfY2w8eVSlOx7s
m+fsAy8Niv+WWuerWzi+U/2luMOtxXC/AowaZ1Rs8yOty2uTzJx7HbG6hrzI90/UDHJ+kZEFF4+mntPZrk
0WWxqshrzIc6fqeOWba1HpjUvWW9neTTbKULKZe3H0fZgZWZCklDMocVvv6HPiHv0QiuqPovidWxysl
KudcYR3tvZtk4vm1QUHqdfusDG2eRbVKt43DXx7HF8Zw8MhGa1JS/ya9ec49HlxT/7bOsqm2zEjBe3NN
bfYDaMYrGUUlrp7fgT+G/VJfef5FST8pR9enRhg0zqocbI6bk33A48L+ry+/8AoiwixK8jHm63BOtvfVstAix/
4jf8jzDSeZkPXKk9f1ZrTTOObbZKOoyXD2eYtM68i+Uo6jKW099+WBxk8eIUNd2Wk19M55dM4x3GPd7
KQIsv67j/AD/U98U+rR++vyZ1m02SnXbUuqUH2Ocqu7IxILy9WdW3HfbuBRKMVjOKS10dvwIa5NeFT1
8df3PoSTdTiu/TonxseSw5VWrW2/UDLHsvrojGGPuOt733O8GuyFlrnDojLlI5rWXjx8uMI2RXuvZRjRuSk
75JtviK9AO7240WNd1F/kYeHRSxU13k3sqklKLT7PhkNdeViuUK4KyDe1tgewSXis9LXs/ohR/E7vu/4OMf
zP2jLzddfTzo0uqurynfTFS6lpoDzxD97j/e/wAHGW5PPrSh19K2o/EWVZV1Ic5wSUX7qfY3y8edko21PVk
P7gY5DyL6uh42udp77HmV1KGKpe8u/wDY0cs2xKPRGv4y2d5mPO6qPS9zhz8NgaZf1Wz7pHNteEx16v
T/AKndn026twdcYrXL33Na8dywVTYul6/pyBlTZfCmMY4246+Pc78PrsrVinBxTe0jmH0yiHlquM0uFLZRj
xtjB+dLqk3vXwA1l7r+RH4X9Xl9/wDRFj5RFiV5GPN1uCdbe+rYDFSefe2uU+BmcZuO133+ppj0zhlXTlHU
ZdnvuMmmdmTROMdxi+XvtyBPfKf7S3GHW4rhfge5P0jIgovH009p7NsnHsdsbqGvMj3T9TlyzbWo9Ma
l6y3sDjMXVdiqa7tJp/gb56X0Of2a/M5yabLL6JRW1B7k9/ajXLhKzGnCC3J9l+IDE+q1/dRxn1+Ziy+MfaRp
jxcKIRktNLk0aTTT7MD585/Svo1W+GuqX+/1OFa68G2lv2oy6f6/6zbBxZ02zlNfZHnuLsSU82M0v+NtOXI
HOZF04Fda45Sf5nUbMhVKCxV0613KcilX0uDevVP4MmhLNrgq/LjLXClsDvw+uyqqUbIuPtbWzXKr8zHn
H11tHuPGyNerpdUt7+RoB812O7FopT5lLT+SPYT+j15NW/d5j+PH+DTGxJV5cpyWoLfTyMzFndfCUFw+J
c9gCr8vwtr1cdv8Tz/7T+H6lWRBzx5wgttrSRj5Nn7O8rp9vXbf2gcS/hX/AKr8zfD+q1/I4dM/2f5XT7fTrW
zXGhKGPCMlppcoDHxL6q/mjaqK+jQjpa6Vx+Bxm1ztx3GC29rjZrWnGqMX3UUgJPC0vJm9c9WhHjxW
WvWP6GmBTOmmUbI6blvuPJn+0Hb0+x063sDLDXXmZE5e8npf1PfE4pVQmuJKWkzq2i2rId2Pp9XvRZ
y6b8qyLvioVx50n3AZyahVkJe1BrYtayc2qC5hFdT/AN/oV21qyqUH6rRNgY86VOVi1J8LnfAGOTKf7Rj0w
63FcRPcn6RkV9Lx9c7T2b5WPOdkbqWlZH0fqcOWbbqPRGr4y2BVVvyodXvaW/mSeKfuIfe/QtXbl7J86
mV9KUOZJ718QPc36pZ8hhRSxa9LW1tnElfdhzjOtKb4ST7m2NCUMeEZLTS5QEmE+n6V0/yvhf1M8Kd0
K5Ounr2+ZbKcSmdc7nOOlJ8c9+5nGrIxZy8mKsrk96b7AKYXPOVsquhNafJcT0LJlY53NRjrSgigCKP8Vl939
EM5KWTjpractNfijRUzXiEren2HHW9nmVTOzIolCO1GW299uUB1nNxw56+xf3OsOKji169VtmlsFbXKEu

0lojrWXjx8uMI2RXuvYHtKVfidkY8Ra3r+h5gJPIyHrlS4f4s1xMedcpW2vdk/7HmJTOu2+U46UpbXPfuBxf
x4nS16r/Iyf4hQaXUzlm1WKO4RXL38znMqtdtdtSUnD0ArBzU5yrTsj0y9UdAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADO+t20ygnpv1NABxTW6qYwb30r
WzsAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADBY7WY7+paa1o3AAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAACK6dmRlOiubhCK9po5thZhdNkLJT
hvUoyAqyrXTRKa7rsZYkL3022WtqS306OPEIKdHnKctaXs+jNcOny64z65S6orhvhAUGWPkRvUulNdL1y
dXVebDp65R53uLIfD6eqUpdcl0y7J8MDeq2cvELa3L2FHaX9Co+c4Ts8RtjXPo2uWvhwezjPDyKmrZSjN6
aYH0ASZl01ZCiuXTKfeXwRldVLGh5tV8pNd033A+gDmufmVxmv5lsnzrpwjCut6nY9bAqBDZi2VUynG+f
Ult7fDPa5y/Zbl1Pq0+d89wLQY4bcsWtttvXdmOHOTvyOqTaUuNvtywLAfPh1Zs5Sla4Vp6ST0d48505fk
Sn5kZLcWwLQR4cpSychOTaUuNvtyz3JlJZ1CUmk+62BWCPKtnPIjj1z6F3lIztjLD6bK7nOO9Si2B9AlybZw
y6YRlqMnyviVJ7Sa9SDPUnlUqD1J9n+IF4ILsa2it2wvnKUeXt9xk2TsxKsiEnFrvp/wC+oF4Jsq/WF1xenNJI
1xoyhRBSbctbewNATZ18qoRjX783pGbw7Yw6o3zdi578MC0EkMiV2BZN8Timm0ZY9FuRQpTumo/yp
MD6AI8GyalbVZLq8t8NmcOrNnKUrXCtPSSegPoAix5zpy/IlPzIyW4tloAHz5uWRmTrna64x7JPWzWmq+
m/p6nOlru32A2lkRjkxp09yW9mp8yePrOhX5k3tb6t8ruUZdsqKq6q5Prlx1MCsHz7KJU1uyGRJzjy+e51k
3SngQsi3FtremBcCL6JbOtTd8vMa2ueDTBulbU1P3oPTYFIObZquuU3/KtkNVc8mPm23uO+yT1oD6AJM
K2fXZTZLqcO0vijnybbZOWRa616RjIC0EWJZOOTOiU/MiltM83ZmZM4KbhVDjj1AuJYWzfiM63L2FHaX4I
5jC7GyIKMp2VS4e+dGU4Ts8SsjXPobS2/s0gPog+fbCeHbXKNspRk9NM+gBlkZEceClJN7euDq1zVUnUk5
+iZD4lV0pWdcn1S91vhcG9tbowrNWTk++2+UBRU5uuLsSU9cpHRHOcv2WpdT6tLnfPc5qxrLqYznfNNr
hJ8IC4HzsZXZScJ2yjGHD13Z3T14+aqHNzhJbWwLgRXWTvynRGzy4RXtNepxLqw7YONrnXJ6ab3oD6AB
5OPVBx21ta2vQDNZEXkyo09xW9mWTbOGXTCMtRk+V8SaOPvPnV5k+Fvq3z2RpnKX0mhQftdk2BeD5
+RRZj1+bG+bknztlM52zxYypS65JP5AbhvUW/gQTx5Qqc3ky81Lfvf2O4dWVhxlKcoyjvbjxsCjHvV9fXFNL
euTQg8Nq3BW9cuG10747F4GVcrndNWRSrXutepqSYspPMvTk2k+FsyTllZNkbLnWovSinrYH0AS49d9N
0oSbnV6Sb7FQAEEpSysmcFY66occPuIyli5UIeY7K58cvegLwR0SlXnWVSk2pcx2xKUrfEVCMmo1rb0/8Af
sAsBBKUsrJnBWOuqHHD7iMpYuVCHmOyufHL3oC8Hz7lbZnuuuxxTXPPbgZNU8amE4WSl0y55A+gCX
Mv6cRSg9OetNG9EHCmEZNuWud/EDzIuVFfXJNreuDpyk6nKC3JraTIvEqtQdnXLlpdO+DWNbpw7Gpyk
3DfL7cAbUOx1J3JKfwRoRVzk/DJS6n1afO+e5zRj2X0RnO+a32SYF4PnY6vvlKmV0lGt8td2dQU8XNhX5k
pwmvUC8EWTZO3JWPCfQktykcWqWHKFkLnODepRb2B9ABcoktlKrxCt9T6JrWt8b/3QFYPn51845CVb
eq0nLT+3/8Aw1zrW6a41yadjWtP0A9ybZwy6YRlqMnyviVHz82MlkY8YP2taTYyKLMevzY3zck+dsD6AJc
jKcMSE4+/NLX2GTxpqtz+ky81LfvcfIC8EUr5W+Gyntqa0m180eVY1t9MZzvkm17KXoBcCXw+2c6pRm9
uD1sqAAi8i21uWTc6+eIxlwe4Vk1fZRKfWoraYFgPnVxtvyb4K2UYKT3z9p1FTxcyFfmSlCfxAvAI8q2c8iOP
XPoXeUgLCXKtnDIojGWlKWmvjyjG2MsPpsruc471KLZ3mPeTitesl+aAtBJmWzdsKK5dLly5fBGV1c8WKt
qvctPlN9wPoAhzrJOFEq5OPVyufkeX41ldTtV83OPL5AvBDVVbl1q2y6Ud9lHsdYNk921WS6vLfdgWA+fD
qzZylK1wrT0kno7x5zpy/IlPzIyW4tgWg+dGNt2XdCNsoxT55/I61PEy64qyUoT40wKrsiNM64tN9b0tGp8
3Np6b6/bk+uT7vtyuxfVX5UOnqlL7ZdwOwAAAAAAAAAAAEMJLH8Rs8zhWLhs98QujKpVQalKTXC5Kr
aq7o6sipI4qxaapdUIc/F8gZZkejw9Rf8AKoo3x/q9X3F+R1ZXG2DhNbi/Q9jFRiopaSWkB6Q+HSinbFtKXV
2ZcY/RafN83o9re+/qBhT/ABS77v8AgeIfvMf73+CqNMI2uxR9uS03sWUwtcXOO3F7XIEWfCKyq52J+W1
p6Op0YUIdTlx9kt7LZwjZFxnFNP0ZjHCx4y2q/wCrbA1qUVVFQ93XBJ4h7FtFutxjLktPJRjOLjJJp+jAwyb6v
os2pxfVFpafcxqTfhTS+D/M7txKK6bJRhyovW3vXB14f9Th+P5geYd1axI9U4rp4e2ZYDVlmS12k/8AJQ8L
Hcury/w29GldNdTk649Ll3A+di00Sc67+LE/V6KKa8SGSo1vdi5WntG9uNVc9zgm/j2PaqK6f3cUt+oEuLO
NeZkRm1Ft7W/meXWQs8Qp6JKWu7RVbjU3S6pw2/jvR5HFohKMo1pOPZ7YEmVCCz07k/Lmu53ZRhV
pOT7/AAlsssrhbHpnFSRlDDorl1Rhyu23sDaKSikuyXBFnTUMuiUuy5f9S44sprskpTipOPbYGOXkVrGnqcZ
OS0kmMeneAq5cdUX/AHOo4ePGXUq1v5s7uuhRFSm9J8LSA+bT1XWVY8lxXJtn1iPBi5zsyGtdb9n5FgE
XiKcXTbrahLkolk1Rq8zrTWuOe5q0pJqSTT7pk6wsdS35f4bYE+NBx8Ouk/5k2v6FGB9Tr/H82byhGUHBr
2WtaPK641QUILUV2QEeKt5eSvi3+Zji00Sc67+LE/V6PowphCcpxjqUu733Obcaq57nBN/HsBhTXiQyVGt
7sXK09osM6qK6f3cUt+poBHJ4+TbOFsOiceNt62Z0Pyc2NVVjnW1yt70V249Vz3OCb+PY9qx6qf3cEm/U

CW6Sj4pU5NJdPd/ieeIxTdVjW4b09Fd2PXfrzI712Z0q4KtV9KcUtaYEbowlX19S1948ylWvD4+Vvo6trZus
HHUt+X/dnHiSSxUlwupAawyKlQp9a0l8eTHw1Nxssa0py4O44dE4xk6+dLs2iiMVGKjFJJdkgOMmDsx5x
XdrghxKcW2r23qa77lo+kY2YlFkuqUOX6p6Azxo48bJvH25RWnztE2LGm7rnkzTnvtKWuD6NdUKo9NcV
FGU8Oic+qUOX309AS4nl/tCfle508f2O8WSoyrq7Go9T2tlUceqFinGCUktcC2iq7XmQT16gcTyoq+FUF1
uXfT7GNf8Vs+7+iKaqKqf3cEm/U9VMFa7VH22tN7Al8S7VfeLTiymFuuuO9Pa5OwIvFP3Efvfoa5cozwrH
GSa16M2srjbBxmtpnEcaqNUq1H2Zd1tgTWfwlfJfmU4v1ar7qOnTW6fKcfY+GzqMVCKjFaSWkBH4b/9b
7ws/itX3f0ZVXTCrfRHXU9vkOmDtVrj7aWk9gQTrrWfON/EZcxe9Gk6MKEopvbk9JKWyu2mu5asipI4rxa
apdUIc/FvYGxz5sPN8vq9vW9HRNXVN5s7prUUtR+0DJSUPFZuTSTjxv5I9yvruP8AP9Si3GqukpTjtr13o6l
TXKUJOPMPd57AZeIfU5/h+ZjkTnDw6robW0k2vkWWVxtg4TW4v0Hlw8vy+lOGtaYEMq8OGO5bUpa4
9rlv5GuDzgv8TSOFjxe1X/VtmlVUKY9Na0t77gTeGTj5Dj1Lq6nwWGUMWmFvmRhqXzNQIsT69kfP9Rv
Gy5yVkeiyPHL0yqFMITlOMdSl3e+5zbjU3Pc4bfxXAEuLJ15jphY7K9fHei84qorpX/HFLZ2B8uFVX0u2vI42
9xe9G6pw4XQinubfGnsptoru15kE9ep5VjVUvcIafx7gYZydc6shfyPT+R74fFuM7pd7Jf2KbIRsg4TW4vuj2
EYwioxWkuEgPmQqq+l215HG3uL3o3VOHC6EU9zb409lNtFd2vMgnr1PKsaql7hDT+PcCdfxV/d/QqurVt
UoP1Q8mHm+b0+3rW9nl18KIpzb57aQHzsfqvtppkuKm2z6pJgQbdl8lp2Pj5FYEviX1X/2R3KcZ4U+mSf/
ABvt8jacI2QcZrcX3RnXjVVwlCMeJ8PnuBLV/CZfJ/mUYX1Sv5fqaKitVeUo+w/TZ1CEa4KEFqK7ICPB+s5P
3v1Yyf4hQVQphXKUoR05Pb57iVMJ2RslHco9nsCDIrrXiD8/fRNcM0sowq9dT7/CWyyyqFsemyKkjOvEor
l1Rhyu23sDZLSSXZEviMW6VYu8JbKjycYzi4yW0+6Aixa/pFd9klza9L7DLEcrsiqMlxSn/v5H0a4RrgoQWor
sjyFNdc5TjHUpd2BLl/Xcf5/qa+IfU5/h+ZrOmE5xnKO5R7PfY9srjbBwmtxfoBBlQbwKJJbUUt/0O40YUq+
vqSX2yLVCMYKCXspa0YPBx3Lfl/3YGNiqXhtjp30tp8/NFOL9Wq+6jqVUJVeW4ro+C4OoxUIqMVpJaQEf
h3e77xZNtQk1y0uDmumFXV0R11Pb5OwPm4saLoysyJpz3/NLR1hOH0+zyuIdPH9imeHROXU6+X309H
cMeqFnXCCUta4AmwvreT979WMr6/jlUKYVzlKMdSm9t77iVMJ2RnKO5R7PYHZ87KhBZ6dyflzXc+ic2V
wtj0zipICOyjCrScn3+EtjLSWRipdk1r+qN4YdFcuqMOV229mk6YWTjKcduD2nvsBFmwisyuVqflyWmzqyj
Crh1N8fZLeyycI2R6ZxUl8GZRwqIS6lXz9r2BPmqMY4yh7qfHy4Ksv6rZ91nVlMLXFzjvpe1ydTipxcZLafDA
xwPqdf4/mYYi3lZS+Lf5ssrhGuChBaiuyPIUwrnKcY6lLlvfcD52LTRJzrv4sT9XooprxIZKjW92Llae0b241Vz3
OCb+PY9qorp/dxS36gTYn17I+f6jM+uY33v1RVCmEJynGOpS7vfcTphZOM5R3KPZ77AS+IvVuO32Unt/0
LVJSW4tNfYcW1Quj02R2j2qqFUOiC0gOgAAAAAAAAAAAAAAAAAAAAAAAAAAAB8rkJJLSWkAAAAA
AAAAAAAAAADyUYyWpRT+aPQAAAAAAAAAAAAAAAAAAANJrTSfzAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB5KMZe9FP5o9AAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABz5kOvo6l1d9b5OgAAAAAAAAAAAAA
AAAAAAAAADyc1CDlLsltnldkbYKcHuL7AdAGDzKVZ5fU2964QG4AAAAAAAAAAAAHFt0KnFTeup6XAHYA
AAAADyUlCLk+yW2cwuhZV5ifs/F8Adg8jKM4qUWmn6o9AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAnUqvpzXQ/N173oe35cKZ9GnOfwiYr+Kv7v6HttV1WU76Yqa
ktNMDSnLhbPoalCfwkUEddtV2RHzapV3LtssAnuza6bXCSltLfB1XkxlQ7ZJwin6k/SpeKva3qO/7DxRvVU
Utpt8Ad/tCHd1zUP8Ato3tvjXR5vvR47E07MidTr+i6i1ruZzhOvwtxsTTUu34gbPxCGtxrnJer12Onn1dMX
BSnJ/ypco0xor6LWtLTgvyJvC4pQseud6A3x8qF8nFJxkvRnt+VChqL3KT7RRg+PFlr1jz/Q8xl1598pd4vSA
1qzYTsUJRlXJ9lJGjyIrJVLTTa2n6GHicV5EZ/wA0ZcM8zU/LpyF70NbAoyL448FKSb29aQuyIUVqU9pvtH1J
7WsnNqguYRXU/wDf6GeTKf7Rj0w63FcRA3hnQc1GcJ177OSN7roUQ65vj8yLJ+kZFfS8fXO09jITndiwsX
otp/EDq7NhZROLhOPVF6bXDNsD6nX+P5s6y0ni2bXaJzgfU6/x/Ngd5NvlVN9Mpb44XYj8OsjFdHlybk/e
1wX2fup/Jk/hv1X/ANmAsz64TlBxluL18zR5MI46tnuKfp6k+LFPPvbXKfAy1159EJe7rYHa8Qhtddc4xfaTR
vbdGul2+9Fa7eoyIRnRNS7dLIoyb8Jlv0ev7oDZ+IQ1uNc5L1aXY6efV0xcVKUn/KlyjTFSWLWklzEm8Nik7
mlzvQG+Plwvk4acZL0Zu+ERT48Vr16x5/oy0D5sspvNhPosUUvc9X39DvPmnHHm00m96fp2OrP4rV939
GPEkm6U+zkB0/EIJ78ufR/20UqcXX1prp1vf2GeUl9EsWuUnwfXJrwmGvBHvYEsuJrwmwcGvF5m/cf5EtH8Kl8pAUY86liq
cF0Vrb5Mn4hDlqubiv5tDHr83w1Q3rafP4mcLL8avy7aOqC43H4AUTyo+SrIRlNS49ldifw6968uSnJuXvd
0uCjF8nyG6FqL7p+jMvC/q8vv8A6ICwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAABMqZ/tB29PsdOt7OZ1ZFN8rKfbjLvFvsVgCKNV9+TC26KgodkvUtAAmVM/2g7en2OnW9

neXj/SKulPUk9pmwAjU85R6PLjvt1bOrqbZYXlt9dnH2epUAOKYuNNcZLTUUn/QwwKZ0wmrI6bltclQA
mdM34hG3p9hR1vZzdRbDI8/H02/ei/UrAEMqsjKnFXRVdcXvSfcrtrVlUoejWjsASYGPOlTlYtSfC53wdZW
POdkbqWlZH0fqUgCNyzbdR6I1fGWzTLx5XQjKD/5Icp/EoAENn026pwdcYrXL33KMSEq8aEJrUlva/E2AH
k03CSXdpmGDVOqjpsWntvWygATY9M4ZV05R1GXZ77nuXjyuUZ1vVkOUUACKf0y6HluuME+HLZpbju
OC6a11Pj8eSkAcURcKIRktNRSZjhUzq8zrjrqltclIAlnTN+IV2qPsKOm9/MqAAjyqrlkwvpipNLWj3JqtujQ1
D2k9yW+xWAM8iLnROMVttaRlXjuWCqbF0vT/AA5KQBFD6ZRDy1XGaXClsox42xg/Ol1Sb3r4GoA4ui5U
2RittxaX9DCqmyOBKpx9tp8bKgBLXjzeCqZNwnz69uTiMs2EFX5UZa4UtloAww8d0UuMntye3oxxK8jH
m63BOtvfVstAGUHf9ImpxXla9lmoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAH//2Q=="></figure><p>Constraints:</p><p>0<size <100</p><p>0<data<1000</p><p>Input format:</p><p>First line indicates the size of the queue</p><p>Second line indicates the elements of the queue.</p><p>Output Format:</p><p>every lines indicates the enqueue of each elements</p><p>last line indicates the fine queued elements.</p>

answer

```
 #include <stdio.h>

#include <stdlib.h>

struct node

{

   int data;

   struct node* next;

}*front=NULL,*rear=NULL;

void linkedListTraversal(struct node *ptr)

{

   while (ptr != NULL)

   {

      printf("%d ", ptr->data);

      ptr = ptr->next;

   }

}

void enqueue(int data)

{

   struct node *n;
```

```
    n = (struct node*)malloc(sizeof(struct node));
        n->data = data;
        n->next = NULL;
        if(front==NULL){
            front=rear=n;
        }
        else{
            rear->next = n;
            rear=n;
        }
}

void dequeue()
{
    struct node* t;
    t = front;
        front = front->next;
        free(t);
}
int main()
{
    int n,i,data;
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&data);
        enqueue(data);
    }
    printf("Dequeuing elements:");
    for(i=0;i<n;i++){
        dequeue();
```

```
        printf("\n");
        linkedListTraversal(front);
    }
    return 0;
    printf("for(i=front;i<=rear;i++)");
}
```

question

Question description A positive integer <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi></math> has been stolen. But luckily, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> hints are available, each described by two integers <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>a</mi><mi>i</mi></msub></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>d</mi><mi>i</mi></msub></math>, meaning that <math xmlns="http://www.w3.org/1998/Math/MathML"><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><mi>X</mi><mo>−</mo><msub><mi>a</mi><mi>i</mi></msub><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><mo>=</mo><msub><mi>d</mi><mi>i</mi></msub></math>. The hints are numbered <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math> through <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math>. While some of those hints are helpful, some might be just a lie. Therefore, we are going to investigate the number <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi></math> under different possible scenarios. Initially, we neither trust nor distrust any hint. That is, each hint may be either true or false. Then, in each of the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> stages, we will either: 1 id Entrust the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math>-th hint (<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>i</mi><mi>d</mi><mo>≤</mo><mi>N</mi></math>). That is, from now on, the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math>-th hint must be true unless declared otherwise in the future 2 id Distrust the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math>-th hint (<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>i</mi><mi>d</mi><mo>≤</mo><mi>N</mi></math>). That is, from now on, the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math>-th hint must be false unless declared otherwise in the future. 3 id Neutralize

the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math>-th hint (<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>i</mi><mi>d</mi><mo>≤</mo><mi>N</mi></math>). That is, from now on, the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math>-th hint may be either true or false unless declared otherwise in the future. After each stage, you should determine the number of possible positive values <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi></math> and report such values in increasing order. If there are infinitely many such values, print <math xmlns="http://www.w3.org/1998/Math/MathML"><mo>−</mo><mn>1</mn></math> instead. Constraints <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>,</mo><mi>Q</mi><mo>≤</mo><mn>200</mn><mspace width="thinmathspace"> </mspace><mn>000</mn></math> <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>0</mn><mo>≤</mo><msub><mi>a</mi><mi>i</mi></msub><mo>,</mo><msub><mi>d</mi><mi>i</mi></msub><mo>≤</mo><msup><mn>10</mn><mn>9</mn></msup></math> <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>t</mi><mo>≤</mo><mn>3</mn></math> for every stage (update). <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>i</mi><mi>d</mi><mo>≤</mo><mi>N</mi></math> for every stage. In tests worth 74 points in total, <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>a</mi><mi>i</mi></msub><mo>,</mo><msub><mi>d</mi><mi>i</mi></msub><mo>≤</mo><mn>500</mn><mspace width="thinmathspace"> </mspace><mn>000</mn></math>. Input The first line contains two space-separated integers <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math>. The <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math>-th of the following <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> lines contains two space-separated integers <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>a</mi><mi>i</mi></msub></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>d</mi><mi>i</mi></msub></math>, describing the <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math>-th hint. It is guaranteed that no two hints are identical. That is, for every two different <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math>, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>j</mi></math>, it is guaranteed that <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>a</mi><mi>i</mi></msub><mo>≠</mo><msub><mi>a</mi><mi>j</mi></msub></math> or <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>d</mi><mi>i</mi></msub><mo>≠</mo><msub><mi>d</mi><mi>j</mi></msub></math>. Then, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> lines follow, each containing two integers <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>t</mi></math> and <math

xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math> — the type of an update and the index of an affected hint. Output After each stage, print the number of possible values of <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi></math> (in case there are infinitely many of them, print <math xmlns="http://www.w3.org/1998/Math/MathML"><mo>−</mo><mn>1</mn></math>). If the number of possible values is finite and non-zero, in the same line, continue to print those values in increasing order. Explanation for sample test case 1: In the sample test, we are given <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi><mo>=</mo><mn>3</mn></math> hints and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi><mo>=</mo><mn>10</mn></math> stages.
The first stage is described by a pair "1 1", which represents entrusting hint <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math>.
After this stage, <math xmlns="http://www.w3.org/1998/Math/MathML"><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><mi>X</mi><mo>−</mo><mn>3</mn><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><mo>=</mo><mn>0</mn></math> must be true, so <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi></math> must be equal to <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>3</mn></math>. We report <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math> possible value: <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>3</mn></math>. Then, the information that <math xmlns="http://www.w3.org/1998/Math/MathML"><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><mi>X</mi><mo>−</mo><mn>3</mn><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><mo>=</mo><mn>0</mn></math> is neutralized at stage <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn></math>. At this point, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi></math> could be any positive integer, so we print <math xmlns="http://www.w3.org/1998/Math/MathML"><mo>−</mo><mn>1</mn></math> in the second line.

answer

```
 #include <stdio.h>

#include <stdlib.h>

int compare(const void *a, const void *b);
```

```
int search(int* arr,int value,int start,int end);


int main()
{
    int i,n, q, t, id;
    int one = 0, len = 0, qu_rear = 0, front = 0, b;
    int idx[3][200002], sol[400001] = {0}, queue[200002], answer[2], **sol2;
    scanf("%d %d", &n, &q);
    for (i = 0; i < n; i++){
        idx[1][i] = 0;
        idx[2][i] = 3;
    }
    for(i = 0;i < n;i++){
        int a, d;
        scanf("%d %d", &a, &d);
        idx[0][i] = a + d;
        sol[len] = a + d;
        len++;
        if (d != 0 && a - d > 0){
            idx[1][i] = a - d;
            sol[len] = a - d;
            len++;
        }
    }
    qsort(sol, len, sizeof(int), compare);
    for (i = 0; i < 2 * n; i++){
        if (sol[i] != 0 && sol[i] == sol[i + 1])
            sol[i] = 0;
    }
    len = 0;
    for (i = 0; i < 2 * n; i++) {
```

```
    if (sol[i] != 0) {
        sol[len] = sol[i];
        len++;
    }
}
sol[len] = 0;

sol2 = calloc(2, sizeof(int *));
for (i = 0; i < 2; i++){
    sol2[i] = calloc(1 + len, sizeof(int));
}

for (i = 0; i < n; i++) {
    if (idx[0][i] == 0){
        idx[0][i] = len;
    }
    else {
        idx[0][i] = search(sol, 0, len - 1, idx[0][i]);
    }
}
for (i = 0; i < n; i++) {
    if (idx[1][i] == 0){
        idx[1][i] = len;
    }
    else{
        idx[1][i] = search(sol, 0, len - 1, idx[1][i]);
    }
}

for (i = 0; i < q; i++){
    scanf("%d %d", &t, &id);
```

```
id = id - 1;


if (t == 1 && idx[2][id] != t){
    if (idx[2][id] == 2){
        sol2[1][idx[0][id]]--;
        sol2[1][idx[1][id]]--;
    }
    one++;
    queue[qu_rear++] = id;
    sol2[0][idx[0][id]]++;
    sol2[0][idx[1][id]]++;
}
else if (t == 2 && idx[2][id] != t) {
    if (idx[2][id] == 1){
        sol2[0][idx[0][id]]--;
        sol2[0][idx[1][id]]--;
        one--;
    }
    sol2[1][idx[0][id]]++;
    sol2[1][idx[1][id]]++;
}
else if (t == 3){
    if (idx[2][id] == 1){
        one--;
        sol2[0][idx[0][id]]--;
        sol2[0][idx[1][id]]--;
    }
    else if (idx[2][id] == 2){
        sol2[1][idx[0][id]]--;
        sol2[1][idx[1][id]]--;
    }
```

```
}

idx[2][id] = t;
if (one == 0){
    printf("-1\n");
}
else{
    answer[0] = -1;
    answer[1] = -1;
    b = 0;
    while (idx[2][queue[front]] != 1)    {
        front++;
    }

    if (idx[1][queue[front]] != len){
        if (sol2[0][idx[1][queue[front]]] == one && sol2[1][idx[1][queue[front]]] == 0){
            answer[b++] = sol[idx[1][queue[front]]];
        }
    }
    if (idx[0][queue[front]] != len){
        if (sol2[0][idx[0][queue[front]]] == one && sol2[1][idx[0][queue[front]]] == 0)
        {
            answer[b++] = sol[idx[0][queue[front]]];
        }
    }
    printf("%d ", b);

    if (b > 0){
        printf("%d ", answer[0]);
        if (b == 2)
            printf("%d", answer[1]);
```

```
            }
            printf("\n");
        }
    }
    return 0;}
int compare(const void *a, const void *b){
    return *(int *)a - *(int *)b;}

int search(int *arr, int value, int start, int end){
    int mid = (start + value) / 2;
    while (arr[mid] != end){
        mid = (start + value) / 2;
        if (end < arr[mid])
            start = mid - 1;
        if (end > arr[mid])
            value = mid + 1;

        if (value > start)
            return -1;
    }
    return mid;}
```

question

Question description Anderson is a Placement trainer. he is working as CDC trainer in reputed institution that during training the youngsters are struggling in queue concept. Anderson usually gives interesting problems to the students to make them love the DS. One such day Joe Anderson provided to the final year students to solve the task such that, Circular Queue using Linked List. there is no memory waste while using Circular Queue, it is preferable than using a regular queue. Because linked lists allow for dynamic memory allocation, they are simple to build. Circular Queue implementation using linked list is identical to circular linked list except that circular Queue has two pointers front and back whereas circular linked list only has one pointer head. Final Year students were lacking the idea to solve the problem. Being an exciting youngster can you

solve it? Function Description <figure class="image"><img
src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAMiIlLCUfMiwp
LDg1MjtLfVFLRUVLmW1zWn21n767sp+vrMjh//PI1P/XrK/6//3///////B8P/////////////bAEMBNTg4S0
JLk1FRk//Or87//AABEIBCYGdAMBIgA
CEQEDEQH/xAAaAAEBAQEBAQEAAAAAAAAAAAAABAIDBQEG/8QAPxABAAIBAgEICAQFBAIDAQEBAA
ECAwQRkRITFCExUlNUBRUyM0FRceEiYXKhNDWBksFCYqKx0fAjc4IkQ/H/xAAYAQEBAQEBAAAAAAAA
AAAAAAAAAQMCBP/EACARAQABBAMBAQEBAAAAAAAAAAABAhESMQMTYVEhMkH/2gAMAwEAAh
EDEQA/APUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAfLXiu2+/X8ofWbe8p/UDnI7tv7Z
Ocju2/tloBnnI7tv7ZOcju2/tloBnnI7tv7ZOcju2/tloBnnI7tv7ZOcju2/tloBnnI7tv7ZOcju2/tloBy6Tjj4zwOk
4/nPBJb27fV8RlNcxNlnScfzngdJx/OeCME7JWdJx/OeB0nH854IwOyVnScfzngdJx/OeCMDslZ0nH854HS
cfzngjA7JWdJx/OeDq86ex6MdkK7pqyAB2AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AM295T+rTNveU/qDQAAAAAAAAPPt7dvq+Ptvbt9XxHmq2ACAAAAAAE9j0Y7IedPY9GOyFa8YANQA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACZ2iZ+TGLNTNXelt/8NvK9IYr6XN0jDMxFu3b5g
9Vm3vKf1RaT0lTLtTL+C/z+Erbe8p/UGgAHy960rveYiPnL6zkx0y15N68qvyBjpODxa8XWJ3jeOx42WNN
OtinJjHjpO0zt2y9DVaiNPiiKdd7dVIgHeMlJvNItHKj4NItLFNNmjDfe2fJHKm3w+n7LQAAefb27fV8fbe3b
6viPNVsBvF7yv1Ehnk2+Uvtaza20KM2aaX2iIfZna1LRG3K7R3jCa9LVttMPnJn5S75ckzl5HVtFofdRkms8iI
jaY6wmmP1MAOCex6MdkPOnsdfWemjqmbbx+SteNYI/Wmm+dv7T1ppu9b+0arBH6003et/aetNN3rf
2gsEfrTTd639p6003et/aCwR+tNN3rf2nrTTd639oLBDf0rgrXekWtPy22W1nlVi3zjcH0AAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAHnayuTJ6Qx4seW1N6/Cfq+9A1Pm78Zay/zjD+if8rgef0DU+bvxk6BqfN34y9AB5/QNT5
u/GToGp83fjL0AHn9A1Pm78ZOganzd+MvQAef0DU+bvxlzz6PLTDa2XVWmsR1xMy9R5Wtvk1mbmcM
TNKz1z8NweY9f0bGoia87vze08nftdNJ6Ox4drX/Hf9oVW95T+oNgAOWpvkx4+Vipy7b9n5OoDzdRmy6
rHzUae0TPxn4OtvR1b8m1st4tWIjqlaA8q2htGtpTl5Jrt13364epWvJrFd5naNt5fQAAHn29u31fH23t2+r
4jzVbG8XvK/VgEhVly0i+1qb7ON802yRbbs7IYmd+18HU1TLtfNW208jr37WMt4yX3iNmASapkAEJ7F0Y
qbR+CvBD8Hox2QrXjZ5rH3K8DmsfcrwaBqzzWPuV4HNY+5Xg0AzzWPuV4HNY+5Xg0AzzWPuV4HNY+
5Xg0Ai9KUpXRzMViJ3jshXi91T6Ql9K/wAFb6wqxe6p9IBoAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AHy0xWszPwc8dslsXKjaZmeqJ+QOo4XyZsdd7RR1xzaaRNo2mfgDQ55cnI2rWN7T2Q318nt2kH0c8F5v

WeV7VZ2l0AAAAAABDl/nGH9E/5XIcv84w/on/ACuAAAAAAAAAYvMRekzMR29rZtuDPOU79eJzlO/Xi1t
Hyg2j5QDPOU79eJzlO/Xi1tHyg2j5QDPOU79eJzlO/Xi1tHyg2j5QDPOU79eJzlO/Xi1tHyg2j5QDPOU79eJ
zlO/Xi1tHyg2j5QDz7THLt1/F83j5vQiKz2bPu0fKBnPHebvO3j5m8fN6O0fKDaPlAnW87ePmbx83o7R8o
No+UB1vO3j5m8fN6O0fKDaPlAdbzt4+ZvHzejtHyg2j5QHW83ePm9KOyDaPlAO6acQAdAAAAAAI/Sv8F
b6wqxe6p9IS+lf4K31hVi91T6QDQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAACfPeJy8i0zFYjr2+Khyy47cuMlNpnsmJ+IOM2pTa2KbRPy27W9Rvace3VMtRbLa
Yjm4rHx3fc1LWvSYjeInrBm+nrWk2rM8qI333LZbdGraPanqdrxM0tEdsw5VxTbTRSeqQI01ZrvM25Xz3f
NNExfJEzvMT2vsWzRHJ5ETPz3Z0sTF8kT279YNaftyfqkyfxON85OTFktNK8qtp3fOTktnpe1doj9gfMl4tm
mt5mKV+EfFnlUpeJxTbt64mHW9L1yTekRO/bD7WclrRvSKx8dwfL/AMTT6Nan3FnzNS02renXNfgxl57L
SY5HJj69oGW80wUiJ23iOtiYwxG9bXi3z2dr4ptirHZasQzysu23Nxv8wfLXm2kmZ7eyXbHG2Ov0Zy1m2C
Ynt2+D5W1rYKzj23/MGctMk5OXtFq17Ib5+vM85+zM3zTG0Y9p+e77jw1jFFbRv8QYxzERbJa0cuf2dM
OWMlI3mOV8i2DHyZ2pG+zODFFKRNq7WAxdWoyx9JdnHB+LLkv8N9odgAAAAAAQ5f5xh/RP+VyHL/O
MP6J/yuAAAAAAAAAAAATxkzXyXrTk7Vn4qEuOMk5svNzEdfXvAO2PneV/8nJ2/JrnaTt+KOvsfMcZI35y0
T8tocdJSvN8qY3ncHe1609q0Q+1tFo3rMTDhhrGS17Xjed9uv4PsRGPVRWvVFo3mAdZy0iN5tHbs+xes1
5UTEx80+nx1tbJa0bzypjrapWK6m9Y9ma77A+01FLWmJmI2naOvtbzW5GK0/H4OWCledy9UdVuprL+
PNSnwj8Ugza18XN46bbzHxfbXzY45VorasduzOoma58UxG/5GTNe0c3zfJm/VvMgoi8TSLb9Uxu+VyUt
O1bRMp80TSuLHETaPjHzLxeYjk4OTMdkxIKbXrT2rRBW9bxvWYn6ON7V5yI5vl326/wAmcW8aqfwcjev
YDvOSkb72jq7Wo64TY6VtqMs2jfaVIAAAAAAAAAAAAAAI/Sv8Fb6wqxe6p9IS+lf4K31hVi91T6QDQAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADNMcU
taYmfxTvLQAAAAAAAAAAzjxxjiYiZ2md2gAAAmN427ABnHSMdYrDQAAAAAAAAhy/wA4w/on/K5Dl/nGH
9E/5XAAAAAAAAAAAAAAMY8XIve2+/KlsAYw4+apyd9+tsBythnlzbHeaTPa+48XImbTM2tPxl0AYxY+b5XXvy
p3Ix7ZZvv2xts2A5Thty5tS8137Y2MMTN73tExMztG/ydQGL4uVkpffbk/DYy44yU232n4T8mwGL4+XSK
2nrj4wzGK+8crLMxHw22dQHK+GZvy6X5Nttp6tymGa5OXN5tO207uoDFMfJyXtvvymwAAAAAAAAAAAA
AABH6V/grfWFWL3VPpCX0r/AAVvrCrF7qn0gGgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQayuXHrcefHjm8RXb
aH3puo8rZcAh6bqPK2Om6jytlwCHpuo8rY6bqPK2XAIem6jytjpuo8rZcAh6bqPK2Om6jytlwCHpuo8rY6
bqPK2XAIem6jytjpuo8rZcAh6bqPK2Om6jytlwCHpuo8rY6Xq/KzxXAIel6vys8Tper8rPFcAh6Xq/KzxOl6v
yv7rgEPS9X5X9zper8rPFcAh6Xq/KzxOl6vys8VwCHper8rPE6Xq/KzxXAIel6vys8Tper8rPFcAh6Xq/KzxOl
6vys8VwCHper8rPE6Xq/KzxXAIel6vys8Tper8rPFcAh6Xq/KzxOl6vys8VwCHper8rPE6Xq/KzxXAIelayezS
8ZOka7yscVwDzNROs1OPmraeKxM9u70qRyaVifhGz6AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
A+WvWvtTtuD6M87TvHO07wNDPO07xztO8DQzztO8c7TvA0M87TvHO07wNDPO07xztO8DQxz+Pvw
c/j78BdsY5/H34Ofx9+BLtjHP4+/Bz+PvwLdsY5/H34Ofx9+Au2Mc/j78HP4+/AXbGOfx9+GwAAAAAAAA
AABnnKRO3K7Dnad4Ghnnad452neBoZ52neOdp3gaGedp3jnad4Ghnnad452neBoY52neg5/H34BsY5/
H34Ofx9+BLtjHP4+/Bz+PvwLdsY5/H34Ofx9+Au2Mc/j78HP4+/AXbGOfx9+H2t639m0SDQAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AADFve0/q2xb3tP6g2AAAAAAAAAAADz7e3b6y+Ptvbt9ZfEearYAIAAAAAAT2S9Cvsx9Hnz2S9Cvsx9Fa8b6
ANQAAAAAAAAAGMXZb9U/8AbbGLst+qf+2wAAAAAAAAAAc9R7i/0RR2LdR7i/0RR2Iy5AAZAAAAAACjSe1Z
Oo0ntWHfHtSArcAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAACZ2iZn4M48lcleVS0TANMW97T+rbFve0/qDYAAM3vXHXlXtFY+cg0OXSs
Hi04usTExvHYAMxkpN5pFom0dsNAAA8+3t2+svj7b27fWXxHmq2ACAPsRNpiI7ZB8H21ZraYnth8AAAn
sl6FfZj6PPnsl6FfZj6K1430AagAAAAOOqyZMWPnMccqK+1HzgHYcdPqsWorvS3X8YntdgYxezb9U/9tsY
vZt+qf+2wAAByt qcNbTFstYmO2JkjU4ZiZjJXaO2dwdRy6Vg8WnF1iYmImJ3iQAAc9R7i/wBEUdi3Ue4v9
EUdiMuQAGQAANTSYrFp7JZAAAUaT2rJ1Gk9qw749qQFbgAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA8/V2z219MWHJyN67/l8Wuj67zVeALhD0f
X+arwOj6/zVeALhD0fX+arwOj6/zVeALhD0fX+arwOj6/zVeALnlayuTQ5+ewTtS/bHw3d+j6/wA1Xg5ajT
armbc9qaTSI694BRpPSGPUbVt+C/yn4qLe9p/V+a7J6nr+jsme8152JmkRPJme0HogAM5MdMteTesWj5
S05ai2WmPfDXlW37PyB5mWNPOtik0jHipO0zt2yv1WpjBiiuPrvbqpEJs+XNqsfNRp5rM9sz8HW3o6l+T
Nsl4tERHVIPml5GmzVw33tmyRyrW+H0/Zc8q2gnptKxbJNOT1337HqVryaxXffaNusH0AHn29u31l8fbe
3b6y+I81WxvF7yv1Ybxe8r9RI275csY77RWJ+b71VtW1Y6rvmW2Ll7Xjrhyvm5WSsxHVXshWszZvLk3ycj
aOq0db7nyRWZrFY647Wb5MVpi0RPK3YzXi994RJnbmAMyeyXaPSOliIicnZ/tlxnslVXSYJrE81Xs+SteNj1l
pfF/4yestL4v/ABl06Jp/CrwOiafwq8Bq5+stL4v/ABk9ZaXxf+MunRNP4VeB0TT+FXgDn6y0vi/8ZPWWl8X
/AIy6dE0/hV4HRNP4VeAOfrLS+L/xkn0jpJjacn/GXTomn8KvA6Jp/CrwB4me1Meom+mv+Htjbq2XaT0p
E7Uz9U95P6P6SthjJzWlY9Zid4m0/8Abo5aXHGLDyK9lZmP3dQAAQ6
+mHFim/M1te07ROzPo/Fp76eaN+WW1fX+PVvTDpfU58WS0W0082zK9lZmP3dQAAQ6
Rjx0n8UxHbL16cnkxyfZ26tnn582bU45w9Gm/Z2fAw4Fx1AGu5x+wGDSUrAc9R7i/wBEUdi3Ue4v9EUdiM
uQAGSqLRTT1tNd5K2rlpNuTEWqRatdxPXlxEsWy0ik6Wy1mbN3y7Y625ZMdZS0U08MWmN+4MZZn+cMc44
reJ6mbZKzg5Edu6Jl/rF7cq02223LZtnK5jrhy0wxe8r9ybTEZ45y7fjjdq5/5sp4i8m1K6/PUxFHhvwy85
mn8KvA6Jp/CrwVu5+sdL4v/GT1l8/4y6dE0/hV4HRNP4VeAOfrHS+L/xkn0jpJjacn/GXTomn8KvA6Jp/CrwB4me1Meom+mv+Htjbq2XaT0pE7
V4A5+sdL4v/ABk9ZaXxf+MunRNP4VeB0TT+FXgDn6y0vi/8ZPWWl8X/AIy6dE0/hV4HRNP4VeCX0fStdXq
doiNp2j8gegAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAC
HL/ADjD+if8rkOX+cYf0T/lcAAAAAAA8v0hlvqcvRsETaI9rb5vUmN4mN5m5NK7fP8wRaT0ZT
HtbN+K3y+ELbRtkpEfm2xb3tP6g2AAAAAAAAADz7e3b6y+Ptv bt9ZfEearY+xMxO8Pgl+zM2neZlPnZ3l8AAA
AACex6FfZj6PPnsehX2Y+iteN9AGoAAAA46u+SuLk4+XaZl4ScFMvLSvq3Pfn2Yt0AAAAALGst+qf+22M
XZb9U/8AbYAAAAAAAOeo9o/6E/Fuo9o/6E/Fuo9o7EZzcgAMn2bTNdpnqj9YjNdpnqj9jNdpnqj9kBAr
wAAAAABDoP4vVfqfiyRdoxMesn72u/6Oo6l7VX9QkPgbidYLOWvx/Q6DmeEkZYXpqW+Ar4YeBALAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAQ5f5xh/RP+VyHL/OMP6J/yuAAAAAAAAYt72n9V2Le9p/V+W72nBwAAAAAAefb27f
WXx9t7dvrL4+1Wx9iJntjCMvAAAAAAAADGLst+qf8AbbAAAAAAAOeo9o/6E/Fuo9o/6E/Fuo9o7BlyAAZP
vVfqBcAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACbPffLyLTMViObvb/Lbtb1G9px7
dUyCjeBwvgrFJtWZ5URvvrvvMmiYvkiZ3mfyY7tW26vxqmzUbfjXV2J3lvNl1y5ZKeDn8A7K8vyR+xlX
Jcl4tmmt5c/7xr/srxdXz2w1m0q8t1y4r0/xqq29/FLVVnT1kS3FdjMB6UxnN1pTIDGKMdXzKTnWRpnT
c8dsbMNXKbhXbcZKxNqNqm+2trl8Xs1tWTZBfydFvxqjmqGj+sDZ/i8fa/wBlfnV/E/ALrwV+6xq0aSA/f/G
H9E/5XIcv84w/on/ACuDLtbVHqTpPZvWO5PeWtb1VvaVLpbdhuZm0ItzWrbPcb/ZTM4ZQ48DRJ2LrJJnqOO/
nsMaG/cHDPlf0mK/5c7uOuZuzt3A/wDPDlf0mKhSQzuLh9uIXIBe/wAc8pP8Cmd/iEi/wCZ3mVzt3ZFs/m
NU/7ZfIXnFT/jAQAAAAAAAAAGNr1YVs9d6s/Yk1t2vQvDhH/vcr+qf8AjZw0cZMH/AI/8fG5ON/wD+i9f/fHJw0
+JyXfz/2+AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAOWXHblxkxkxk7b/GJ+LqA4xcSDQDHwLlWkzxb4G4aJ8GJvEpFdmrqvnZ
w5VxTbS3VMIDxuNX3wDcGqY6HbGfZgpm5ZpetBWyHhGA/xMbmttFGqHzGYJz/b+mTs1oOTG4Xkd0spM/ZQ

A43x3rknJj2nftiX2s5bWjekVj4uoDlmpabVvTtr8GMkZstJjkRWPr2qAHK+Kb4qx2WrEPnKzbbc3G/z3dgH
PJWbYJie3b4PlZtbBWce2/5urOPHGOJiN9pncHOb5pjbm4ifnuVx3x44ikxM77zv8XYBwtbNaOTyIrv8d3
28Ri00x+WzszfHGTbffaJ32B8w15OKsfk2AAAAAAAIcv84w/on/K5Dl/nGH9E/wCVwAAAAAAAAAAAADOb
3Vvo0zm91b6Anw0wTirNuTv8d5dbWrhwTbHEbbs4MWO2GszSJnb5PuprFdNMVjaOr/ALB9pktMTe1
YrTbf82Yy5bRyq445P5z1y3krN8E1jtmGMeekY4i08mYjrgGue3w2vWOuO2JZjLltWLVxxt+csRE9Hy3mN
uXMzEO+L3VfoDF8lr6ebUj4Tvv8DTTkmleVEcnbqn4s4Y3014jtndrTZKzjrTf8W3YDsAAAAAAAAAAAA
AAAAAAAAAAAAA+X9i30fXy/sW+gOOlrWcETMRPa+Y9q6m0U9nbr2Zwael8MTaJ3n828ERXlYpjafn84
Ai+TLvNLVrX4b/FrHktaLVmI5deDhFMWPeuak7/ADjfrdK1jmb2xUmsz2bz2g+2tmpWbTas7fDZ9y5ZjBG
Svx2cJjFOOYilpvt19reWJ6FX+gO+PnJ677bT2RHwbAAAAAAAAAAAAAAAAABDoP4vVfqXIdB/F6r9QLgA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQ5f5xh/RP+Vzz
9ZN8Ovx5ox2vWKzH4WvWNvLZeALhD6xt5bLwPWNvLZeALhD6xt5bLwPWNvLZeALhD6xt5bLwPWNv
LZeALhD6xt5bLwPWNvLZeALhD6xt5bLwPWNvLZeALievtQ+sbeWy8D1jby2XgC6IiI2iNoJiJjaY3hD6xt5b
LwPWNvLZeALnya1md5rEz9EXrG3lsvA9Y28tl4AumImNpjeDs7EPrG3lsvA9Y28tl4AuiIjsjZ8itYneIjf57Iv
WNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt
5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWN
vLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bL
wBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZ
eB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBc
IfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBcIfWNvLZeB6xt5bLwBch0H8Xqv1Hr
G3lsvBn0Zbl59Rbaa7zE7T8AegAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAbRHwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAGb2tFqxXbr+YNDO2T514G2T514A0M7ZPnXgbZPnXgDQztk+deBtk+d
eANDO2T514G2T514A0M7ZPnXgbZPnXgDQlnU5ItMbV6p2fOlX+VRzNUQrEnSr/ACqdKv8AKoZwrEnSr
/Kp0q/yqGcKxJ0q/wAqnSr/ACqGcKxJ0q/yqdKv8qhnCsSdKv8AKquJ3iJFiYnQAKAAAAAAAAAADnWb23mJ
rERMx2NbZPnXgDQztk+deBtk+deANDO2T514G2T514A0M7ZPnXgbZPnXgDQztk+deBtk+deANDnktkp
Sbb1nb8nDpV/lUSZiNqxJ0q/wAqnSr/ACqJnCsSdKv8qnSr/KomcKxJ0q/yqdKv8qhnCsSdKv8AKp0q/wAq
hnCsSdKv8quuDNbJMxaIjb5CxVEuwA6AAAAAAAAAABnJaa13jbffbrNsnzrwBoZ2yfOvA2yfOvAGhnbJ868
DbJ868AaGdsnzrwNsnzrwBoZ2yfOvA2yfOvAGhnbJ868HDJqMlMk12rO35CTNlIk6Vf5VOlX+VRM4ViTp
V/lU6Vf5VDOFYk6Vf5VOlX+VQzhWJOlX+VTpV/lUM4ViTpWT5VOlX+VQzhWEdgOgAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABi3vaf1bYt72n9QbAAAAAAAAAAB59vbt9ZfH23t2+svi
PNVsAHIAAAAABPZL0K+zH0efPZL0K+zH0Vtx6fQBqAAAAAAAAxi9mf1T/22xi9mf1T/wBtgAAAAAAAAA5
6j3F/oijsW6j3F/oijsRlyAAyAAAAAAFGk9qydRpPasO+PakBW4AAAAAAADGX2Y+sf9tsZfZj6x/22AAAAA
AAAAi1H8Rb6QtRaj+It9IHNenMBHnABAAAAAAWNvRjsgI7IFeoAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAACd9p27fgmw6yl7ziyf/Hlidpifj9AUsW97T+rbFve0/qDYAAOefLOHHyopN/
ygHQQ29JcmYiEz2brYnesTPV1bg+jhg1MZ8lqrerZXI5OR5r+aHBkx8cBAGJ+1bHzgDUNbuZJkmKI0bJf3d2z0rm
+Dt0a/zgpGtN0PltPasR+qzp5tZLQKh3rc9F6+J4ZbEiTbx0YWp8g4iONiu44S2hPJKySy3eqi8Rt3k36HazU6Q4
mXYWWsDPwHrNAuTt7zP1V6ndhGKtVPtytXapKwhwBdIaMfAvjPEw7eIcoxdTj/a0vHVlE1uqq9HyEW8P64wdIyj

AAADeBFr6ZMU9IwTMTHtR84fdJ6Qx59q2/Bf5T8QU5fZj6x/22xl9mPrH/bYAAA56jPXBj5Vt5+UQnj0hE
WiMuK+OsztFpBYJc2tjHkmlMdslo7dm9Nqq6iZrtNb17ayDuAAi1H8Rb6QtRaj+It9IHNenMBHmB9rWb
WiI+Lr0a/zgdREy4jtjwTN9rdkPmTDNbRtMbTO0BjO3IdbYLVrMzMdTkJMTAAEbejHZAR2QK9QAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADnk1GLFaK5LxWZ69pY6Zp/Fql1WKub0pipeN6zTs4u/q
7TeH+4N9M0/i1OmafxasertN4f7nq7TeH+4N9M0/i1OmafxasertN4f7nq7TeH+4N9M0/i1Omafxasert
N4f7nq7TeH+4N9M0/i1Qek5wZYjLiyVm8dsR8Vnq7TeH+7hrNNpNNgm3N/inqrG/xBPpPSV8W1cv4q/
P4w9OmWma1LY7RMdbwMWK+a8Vx1mZl62i0U6W9Ztbe1oneI7IBeAADlqMU5sfJreaTvvEwCHU1z4t
ROpyUrelZ6o37Ia1uti1a4qW5PLiJtPyhudHqMv4c2o5VPlt2qowY4iI5FZ2jbrgEGPW4sGbHix2jmOT+Kdu
vfr+z0q2i1YtHZMbwmto4nWVy7V5ERtNdu1VEbRtAAAPPt7dvrL4+29u31l8R5qtjeL3lfqw1jmIvEz2RIkb
d89ck3ia7zH5NXna2KJn8Xxcsme3LnkW6nKbTM7zPWNJqiHbJW3Pb9e28Gpmec236tnOct5jabPlrTed7
TvI5mqP8ZAHJPZJGo123Vp42J7HoV9mPorXj0g6Rr/LwdI1/l4egDV5/SNf5eDpGv8vD0AHn9I1/l4Oka/y
8PQAef0jX+Xg6Rr/Lw9AtMViZmdogHga2uab85lxciZ6vq+aXWZdNP4Z3r3ZdNRfLr9TtjrM1jqj8luk9G0x
bWy7Xv8vhAKdJk53DF9pjlTM7T9XZjF7M/qn/ALbAABw1tM2TDyMO289u8/Bx9H5OTytPakVtj+Xxfb6
bURe1sWomItO+0vldBPIvM5rc7ad5tAOWqpnpqJ1N6Vvjp7Mb/B6GK8ZMdbx2WjdHbR6jJHIy6jfH8tu
1bSkY6RWvZEbQD6ADnqPcX+iKOxbqPcX+iKOxGXIADJVtadNEV7XzHFq4bcvfb4Mzl2w1itvxR2uV8lr+1
O40mqIdrxa2GnJ3k3mukjbqnf8Ay5Vy3rG0T1Pk3tNeTM9QmUMgDgbpbJSl5xV5V+raGFGk9qw749uH
SNf5eDpGv8vD0BW7z+ka/wAvB0jX+Xh6ADz+ka/y8HSNf5eHoAPP6Rr/AC8HSNf5eHoAPOnPrpiYnTxM
S8vNS+PLMXrybdu3ye/qs9dPhm89vwj5y8fDps2tyTfsiZ67SDppNdkia4r73jeNvm9pJj0eLTU7jeNvbeN7T
9VYAAExEx1xG35vP1Vum5K4MUb1rbe1vkq1WLJmxTTHfkTPPbKXFotVhryceesR+kFeXJj0+Ob22jfnKfR
Y73zX1N45PL7I/JjPoc+bJW1s0fhiNomHfBh1NMkTzBGDeH3tX
XNXJOXeu+3wcsUxXJEz2Q3fPblzyLdQ7iYt+utp/+bHG/X8XOa2jPvMTtynHlTyt9+v5tTlvO289gZRLeptP
Obb9Ti+2tNp3ntfBzM3kJ/LtASNt8n0h8L49jk+ke/jXR2QK9SHk+ke/jOT6R7+NcAh5PpHv4zk+ke/jXAIeT
6R7+M5PpHv41wCHk+ke/jOT6R7+NcAh5PpHv43ya+kNvbxryeyQR+jc2TNhvOW3KmLbLEPoj3GT/wCy
f+oXAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAhy/zjD+if8AK5Dl/nGL9E/5XAAAAAAATMREzPZDzL6fLr8/Lv
vTFHVX5vTAc8GDHgpycddo/wC323vaf1bYt72n9QbAAAAAAAAAAB59vbt9ZfH23t2+sviPNVsAEAAAAAA
J7HoV9mPo8+ex6FfZj6K1430AagAAADlqcM56RTlcmsz+Lbtl1AYw4aYHWIhsAYxezb9U/9tsYvZt+qf+
2wAAAAAAAAAc9R7i/0RR2LdR7i/0RR2Iy5AAZAAAAAACjSe1ZOo0ntWHfHtSArcAAAAABPl0sZ80WyzvS
vZVRWIrEREbRHwAGMvsx9Y/wC22Mvsx9Y/7bAAAAAAAAAAARaj+It9IWotR/EW+kDivTmAjAAAAAAAAC
NvRjsgI7IFeoAAAAAAAAJ7JCeyQQ+iPcZP8A7J/6hch9Ee4yf/ZP/ULgAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAmdomZ+DlXNPN
cu0dUz1RAOo4zmvEbzink/Vqc1YxxeImd+zYHQcZzXr12xTFfnu1kyTGOL02mPj9AdAiYmImOyQAAAAA
AEOX+cYv0T/lchy/zjF+if8rgAAAAAAAAGLe9p/Vti8Ty6zEb7g2M8q3hzxzg5VvDnjnDPKt4c8Ybw54
wDQzyreHPGDlW8OeMA0M8q3hzxzg5VvDnjANDPKt4c8Ybw54wCG3t2+svjrODLNpnk9s79Ps6Gnx7zXGu/
ujCqmbuY6cxl7v7nMZe7+4mEuY6cxl7v7nMZe7+4YS5jpzGXu/ucxl7v7hhLmOnMZe7+5zGXu/uGEuc9j
0K+zH0Rzp8u3s/usr1VhWlETG30AaAAAAAAAMYvZt+qf8Attzryq2yq7xyme2GuVbw54wDQzyreHPGDlW8OeMA0M8q3hzxzg5VvDnjANDPKt4c8YOVbw54wDQzyreHPGDlW8OeMAzqPcX+iKOxZl5d8dqdqxqHAdOYy939zmMvd/c5jL3f3DCXMdOYy939zmMvd/c5jL3f3DCXMdOYy939zmMvd/c5jL3f3DCXMdOYy93
9wwlzUaT2rOfMZe7+7tpsd6Tabcq7lyasCBWwAAAAAADGX2Y+sf9ts5Ima9UbxMT1tclHoZ5Vv
DnjByreHPGAaGeVbw54wcq3hzxzgHnlW8OeMHKt4c8YBoZ5VvDnjByreHPGAaRaj+It9IV8q3hzxhPlxZL
5ZtFeqfzgc1ReHAdOYy939zmMvd/dGOEuY6cxl7v7nMZe7+4YS5jpzGXu/ucxl7v7hhLmOnMZe7+5zGX
u/uGEuY6cxl7v7nMZd/Z/cIpm62OyAjsFegAAAAAAJ7JCeyQQ+iPcZP/sn/AKhch9Ee4yf/AGT/ANQuA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAT5sn/
AMkU5XJjbrlQ45aWjJGSteV1bTAOU3rj2tTLNvnzPkzTzHzm0PuasfHtE7RPmHwGy+G
a0m0ZLcqOt9nNPR4v/AKp6nS/XjtEfKXKuObaWK7bWj5g+xgtMzBzkzcnIvP8Aqfmk0jHDCt9yzuZ
+Xk5Xbv1g3gmZnJvP8AqfMkz0jHG/UzHLwLw5bTxFE+P6Yfn1n/AMl9RS00mIB9yZIt+kk35FY/dnlxjItHIyTaJn
riW70tTLN615cT2w+1tNrRth2j4zMA+3mek0jfqfq2fdRMxhtMM5q2i9cll327YYYy3yZccXXHaI+O4NZMk0w

02nabRHWxPIiN4zTynS+Ob4abe1WI6pZ5c7bcxPK+nUD7bJN9JNvj2S6442x1j8mMtZnTzG0RO2+0EXm
NPFq1m07dkA+5skVrt22nqiH3DTkYorPa4Y7TW3KvjyWtPx2dbZbc3M1x23322mAM9/w8iOu1vg+8jk6
eaz3XLHaafitjyTae2dnW198NrTE16uyQNPO+Cro54I2w1j8nQAAAAAAEOX+cYv0T/lchy/zjF+if8rgAAAA
AAAAAAHOc+OszE3iJh0SUvjpmyc58+rqBRTLS87VtEy254747zPN7bx+TngvlyxE7xERPXO3aCgcOXky3t
GOYrWvVvMdrWPJeMnN5Nt9t4mPiDqJ6XzZJvFZiIraY3mGsd7za2O8xyojeJB2iYnsmOoS4a5OdvteOq3
X1dqoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHLNlvS
1a0iJm3zB1HCcmakcq9KzWO3ky7VtFqxaOyQfR8i9bTtW0TP5STaKxvaYj6yD6PkWi0b1mJ+kk3rG+9o6
u3rB9CJ3jeAAAAAAAAAAAAAAAAAnskJ7JBD6I9xk/wDsn/qFyH0R7jJ/9k/9QuAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAYpj5Fr2335U7tgAA
AAAAAADGPHzcTG+8b9X5NgAADGWnOViu+0b9bYB2AAAAAAAAhy/zjF+if8rkOX+cYv0T/AJXAAAAAA
AAAAAOGCP8A5cu8fF3ANohx0kbYev5y7AJ62nBa0WrM1md4mH2nKy5uc2mKxG0b/F3AcdNExzm8f6
5Ij/8AqtP+12AT1tzWbJyon8U7xtCgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAE+pnk5cc7TO3whQ55MU3yUtvtyQc8mebRyIx2ibdUcpnNE0x48fXMfHb4u+XHG
Sm3ZPbE/J8vj5eOItP4o+MA4XjqjkYLVtHZMQ6ZLUm1eVjm99ux9jHl3jlZOqPlD7kxTOTl0tybbbA5Yp21
MRFJpEx1xL7XHW+qvyo3iPg3XDaMsXtfeYhuuOa5b339oG4jaNo7AAAAAAAAAAAAAAAAACeyQnskE
Poj3GT/7J/6hch9Ee4yf/ZP/AFC4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEOX+cYv0T/lck1WkyZc9MuLJyL
Vjbdjo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zU
cAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zU
cDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcA
XCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcD
o2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXC
Ho2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2
t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo
2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t8
1HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t
81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81H
AFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81
HA6NrfNRwBcIeja3zUcDo2t81HAFwh6NrfNRwOja3zUcAXCHo2t81HA6NrfNRwBcT2Sh6NrfNRwOja3z
UcAPRHuMn/2T/wBQuT6LTTpsU1tblTM7zKgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB8tatfamI+
r6zf26fUDncffrxOdx9+vFoBnncffrxOdx9+vFoBnncffrxOdx9+vFoBnncffrxOdx9+vFoBnncffrxOdx9+vFoB
jncffrxOdx+JXijye8t9WRnPJabLudx+JXic7j8SvFCInZ4u53H4leJzuPxK8UIHZ4u53H4leJzuPxK8UIHZ4u53
H4leJzuPxK8UIHYu53H4leLfa86ex6FPYr9Fd01ZPoA6AAAAAAAAZnJSJ2m9Yn6nO4+/XiY/9X6paBnncffr
xOdx9+vFoBnncffrxOdx9+vFoBnncffrxOdx9+vFoBnncffrxOdx9+vFoBnncffrxfOdx9+vEy+6t9EMdg5qqx
Xc7j8SvE53H4leKERx2eLudx+JXic7j8SvFCB2eLudx+JXic7j8SvFCB2eLudx+JXic7j8SvFCB2eLudx+JXi1W9
bezaJ+kvPUaT2rfRVprvNlIA0AAAAAAAAJmIjeZ2hnncffrxMvsNAzzuPv14nO4+/Xi0AzzuPv14nO4+/Xi0A
zzuPv14nO4+/Xi0AzzuPv14nO4+/Xi0j1Pvv6CTNoup53H4leJzuPxK8UIjPs8Xc7j8SvE
53H4leKEDs8Xc7j8SvE53H4leKEDs8Xc7j8SvE53H4leKEDs8Xc7j8SvE53H368UIp2PRCOyAagAAAAAAAD
5a1axvaYj6vrN/ap9QOdx9+vE53H368WgGedx9+vE53H368WgGedx9+vE53H368WgGedx9+vE53H368
WgGedx9+vE53H368WgGOdx9+vE53H368Uuo99LmM6q7TZdzuPv14nO4+/XihETsXc7j79eJzuPv14oQ
OxdzuPv14nO4+/XihA7F3O4+/Xic7j8SvFCB2Ludx9+vFvtedPY9CnsV+iuqasn0AdgAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADN/
bp9Wmb+3T6g0AAAAAAAAACDJ7y31Zaye8t9WUeerYAOQAAAAACex6FPYr9Hnz2PQp7FfoNeN9AVq
AAAAAAAAzj/wBX6paZx/6v1S0AAAAAAADGX3VvohjsXZfdW+iGOxGXIADIAAAAAAUaT2rfROo0ntW+
g7o2pAVuAAAAAAAAzl9hpnL7DQAAAAAAAACPU++/osR6n339BzXpyAR5wAAAAAAAI29GOyAjsgV6g
AAAAAAABm/t0+rTN/bp9QaAAAAAAAAABFqPfS5umo99Lmjz17AByAAAAAAT2PQp7Ffo8+ex6FPYr9B
rxvoCtQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAADsfItFo3id4fXm6ucuhy87i68dp66/CJB6TNvbp9XHS63FqI2idrfKXa3t0+oNAAG+
3aMZsUZsc0tMxE/IGuVHzh9eNlw4+l1w48lo29qbWenmzU0uDeevaNoj5g7bxuItJHJy8vPf8A+bL1xX5Q
tAABBk95b6stZPeW+rKPNXsB9p7dfqI+PsRMztHapy5K47bciJNq1vW9Y6rDvD1Nas1nae18U5rxOSK8m
O2Ot9z2rTesVjeY7RJphKAOSex6FPYr9Hnz2Kq6vBFYictd4j5jXjdxx6Zp/FrxOmafxa8Vauw49M0/i14nTN
P4teIOw49M0/i14nTNP4teIOznnyWxY+XWvK27Y/Jnpmn8WvEnV6eY97XiDWHPjz05WO2/5fJ0eFqLR
p9Ty9NkjaevqldpPSVMm1cu1bfP4SC3H/q/VLTGPr5X6mwAAN4+cG8fOEGtwYaVtmva+8/CJ+LOg01c
mntNskzN/lPYD0OVHzh9eNnw441VcGPJaJ3/ABTa3Y9ilYpSKx1xEA+gAxl91b6IYXZfdW+iGOxGXIADIFO
Oa108WmsSRyc1Jnk7TUdYuE0tFYtMdUsq7ZIjFW3Jjafg+UmtdPyprEi4x9SjV7cq0zEbbsjgUaT2rfRO7aa
9acu1piIiO2R3RtWOPTNP4teJ0zT+LXird2HHpmn8WvE6Zp/FrxB2HHpmn8WvE6Zp/FrxB2HHpmn8Wv
E6Zp/FrxB8z6jmLxy4/BP+r5O1bVvWLVmJifjCbNn0ubFNLZa7T+bysWpyaTLMUtyq79nwkHu5fYaSYtbj
1GPaJ5Nu7KsAAAiYnsmHPUYrZcfJreaT84efq9PXSY65MeS/Ocrsme0HqTMR2zsRO/YkvpJz35eXJbaYj8
MdWzlo98etyYqXm2OI+M9gPQAAR6n339FiPU++/oOa9OQCPMDWP3lfqoyZKY77ciJ+Y6im8XTVibTtH
aWiaztPaqiK0yVmsdV/2ZyXi2WK8mOq3aOsfxMKc9613rFY3mO1MOZi0gHYJG3ox2QOEavT7R/8teL70
zT+LXir1Ow49M0/i14nTNP4teIOw49M0/i14nTNP4teIOw49M0/i14nTNP4teIPuoyzhrF+TvWO3b4NYst
M1OVjtEw521WmtExOWkxLx73nS6mZ0+SJr2xsD32b+3T6pNJ6Rpm2rf8ADf8AaVV/ap9QbAActRqKaa
kXvvtM7dTq558FNRSK5N9onfqBPPpPDEbzW8f0db6vFTHW9p25XZHxT63bNqMemrEdu9vo45bWj0jy
aY+XNa7VrPZALcGsx578isWifzhQjxaq9dRXFqMUUtb2ZhYAACLUe+lzdNR76XNHnr/oBvD15a/Ucx+sPt
azado7VOTJSl+TyIn5vtYrjyxtHVf9h3h+7SzExMxPbD4otaLZoryY6pM96xvSK9fzExTgDknsehT2K/R589i2
mXHFI/8Akr2fMa8boMc9j8SnGDnsfiU4wrVsY57H4lOMHPY/EpxgGxjnsfiU4wc9j8SnGAbGOex+JTjBz2
PxKcYBsY57H4lOMPtcuO07VvWZ+USDQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAINZl1EazHiwWiOVXsns+JyfSXfxf8Av9H3UfzbT/pn/K4EHJ9Jd/F/7/Q5
PpLv4v8A3+i8BByfSXfxf+/0OT6S7+L/AN/ovAQcn0l38X/v9Dk+ku/i/wDf6LwEHJ9Jd/F/7/RjNi1+TFauS+
KazHX/AO7PSed6Sz2vMabDEzafa2B5MTNbb1nrj4w9b0fqM+aaxkrvWOyz5pPRkV2vn657q+YitqREbR
ANgAOeozcxi5fJm23wh0AeVqc2mz45jHinnbf7dp3dLaLPljHac0RNax1TG+0vQitYneKxv9H0Hk3xaiNfjp
Ofe/J3i23Y9WkTFIi07zEdcvu0b77RuAAAgye8t9WWsnvLfVlHnq2NU9uv1ZfYnad4HKrLOLlbX33cr5om1
eTH4auVrTad7TvL4Opqd73xWmLdfK3hjPeL33r2bOYJNVwAQ+CiNFp7REzirMz1p3oU9iv0GvG4dB03h
VOg6bwqqBWqfoOm8Kp0HTeFVQAn6DpvCqdB03hVUAJ+g6bwqnQdN4VVDnnyTjxzNY3tPVEA8r0lGD
FMY8WOIt2zMfBx0uhy6iYnbk070vQ0/o6JvzuonlXnr2XxERG0RtAOOlxxixzSJmYie2XZjH/q/VLYAAIs2s
wcu2PPittWereu8SnwZObzZc+LFbmdtuT85epNYt2xE/V9iNuwHl6nPpc+O3IxTOW3+3ad1+lreumx1ye
1EdbpFaxO8Vjf6PoAAMZfdW+iGOxdl91b6IYRlyAAyVY5rGmjl9jE5KUpNccdvxcuXbk8nfq+TI6mp3rkx2x
xW+/U+TkpzHIjt3/y4gZSADkdtPSuTl1vG9ZjrhxUaT2rfQd0bfeg6bwqnQdN4VVArdP0HTeFU6DpvCqoA
T9B03hVOg6bwqqAE/QdN4VToOm8KqgBPOh00RvOKrxtXOO2eYw0itY6o2+L2NZzmSOZxR127Z+UPm
l0OPT/AItuVf5yCDTejsm0ZMu9I+EfF7MdUbM5fYaAABy1Oeunwze0TPyiHm4NTivm5/VWtN49mu3VV
68xE9sbvnIr3Y4A87WekImIx4ZmOVHXbbsh00ObTV2xYptNrdszHat5Fe7HAitYneIjgD6AAj1Pvv6LEep9
9/Qc16cgEedrH7yv1UZZxcv8cdcJomYneO2C1ptO9p3kdRVaHW2bfJWYj8Nfg+2vim0WjffeJcAMpdM1
ovk3r2OYDmZv+hPX1AEbUxodNMe6qdB03hVUR2QK9SfoOm8Kp0HTeFVQAn6DpvCqdB03hVUAJ+g6
bwqnQdN4VVACfoOm8Kry/SPMUyxjw0iOT7Uw9bU3vXHtjje9uqE+l9HVpPOZvx3nr/KAQaT0fkzzFrfgp
8/m9iKRjjHSJmYj5urN/ap9QaAAABLpdPembJmzbcu89W077QajTZJzRnwWiMkRtMT2SqARY9Pnyaiub
UzX8Hs1qtAAAEWo99Lm6aj30uaPPX/Q3h97X6sPsTNZ3jtgcxtTknDy/wAcTvDnOffLFtuqPg5WtNp3md
5fB1NSib4uXFo33363LNaL5JmOxgEmq4AIT2OtfR2ntETNZ3nr9qXKex6FPYr9BrxpfVmm7tv7pPVmm7t
v7pVitUnqzTd2390nqzTd2390qwEnqzTd2390nqzTd2390qwEnqzTd2390nqzTd2390qwEnqzTd2390uG
mw0w+lLUpExEV6ut6SGn83v+gFwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAIdR/NtP+mf8AK5DqP5tp/wBM/wCVwAAAAAAE9jniw0xTM1j8U9sz2y6AD
N/bo0zf26A0AAAAAAAAACDJ7y31Zaye8t9WUeavYAIAAAAAAPQp7Ffo896FPYr9BrxvoCtQAAAAAAAG
cf8Aq/VLTOP/AFfqloAAAAAAAAGMvurfRDC7L7q30Qx2Iy5AAZAAAAAAACjSe1b6J1Gk9q30HdG1ICtwA
AAAAAAGcvsNM5fYaAAAAAAAAAR6n339FiPU++/oOa9OQCPOAAAAAAAABG3ox2QEdkCvUAAAAAAAA
M39qn1aZv7VPqDQAAAAAAAAItR76XN01Hvpc0eev+gAcgAAAAAE9j0KexX6PPnsehT2K/RWvG+gDU
AAAAAAQ0/m9/wBC5DT+b3/QC4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAE2a++bkW35M
R2R8QUxMT2TEkzEdspLTWvXireLR+Teo3tOP4bg78qN9t4fXDJgpGOZjeLRG++75bJbo1bfGercHflRvtvD
7vEuMaenJ695n57vmmiYvkiZ32kHfeJ7DeN9t3HT+1k/VL5l/icYO8zt2kTE9k7pcl4tmtF4tNa/CHzetbVnF
W8dfXEwCveN9iZ27XG/8TT6Nan3FgdDeN9t4T5bzXDjiN+uI7GJjFyequSJ+ewKrTyazM/Bw53JXDy9+uZ
6t/gWta2kmbdvY7UiObrG3wBz/wDn233pwMOW1sdr5Nto7Nn3U2mKRSvbbqZz15Gm5MdkbA+xfNk
66REV+G7U85OKd+q/w2Yrz1qRNeTWNuqG8OSbxMWja1e0H3FfnMcW+PxbcdL1VvHytLsAAAAAACH
UfzbT/pn/ACuQ6j+baf8ATP8AlcAAAAAAAAAzf26NPlqxbtjcH0Z5qnyOap8gaGeap8jmqfIGhnmqfI5qny
BoZ5qnyOap8gaGeap8jmqfIEWT3lvqyu5nH3IOYx9yEZTx3m6EXcxj7kHMY+5AnX6hF3MY+5BzGPuQH
X6hF3MY+5BzGPuQHX6hF3MY+5BzGPuQHX6hehT2K/RnmMfchvshXdNOIAOwAAAAAAAGcf8Aq/VLT
M46TO8x1yc1Tug0M81TunNU7oNDPNU7pzVO6DQzzVO6c1Tug0M81TunNU7oPmX3VvohjsXc1Tun
MY+5A5qpyQi7mMfcg5jH3IRn1+oRdzGPuQcxj7kB1+oRdzGPuQcxj7kB1+oVGk9q3
0duYx9yH2tK09msQai03aAGg8AAAAAAADOX2GiY0bTG8M81TunNU7oNDPNU7pzVO6DQzzVO6c1Tu
QzzVPkc1T5A0M81T5HNU+QNDPNU+QNI9T77+irmqfJ8jkq60Qu5jH3IGdVF5uhF3MY+5BzGPuQHX6hF3
MY+5BzGPuQHX6hF3MY+5BzGPuQHX6hF3MY+5BzGPuQHX6hF3MY+5BzGPuQHX6hehT2K/RnmMfchvshXdNOIAOwAAAAAAAGcf8Aq/VLT
M46TO8x1yc1Tug0M81TunNU7oNDPNU7pzVO6DQzzVO6c1Tug0M81TunNU7oPmX3VvohjsXc1Tun
MY+5A5qpyQi7mMfcg5jH3IRn1+oRdzGPuQcxj7kB1+oRdzGPuQcxj7kB1+oVGk9q3
0duYx9yH2tK09msQrqmi03aAGgAAAAAADOX2GiYi0bTG8M81TunNU7oNDPNU7pzVO6DQzzVO6c1Tug0M81TunNU7oPmX3VvohjsXc1Tun
QzzVPkc1T5A0M81T5HNU+QNDPNU+QNI9T77+irmqfJ8jkq60Qu5jH3IGdVF5uhF
3MY+5BzGPuQjnr9Qi7mMfcg5jH3IDr9Qi7mMfcg5jH3IDr9Qz2PQp7FfozzzGPuQ3H
VGyu6acQAdgAAAAACGn83v+hchp/N7/oBcAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA5Zcd
uXGTHMcqPhPxdQHKJzWmN61rHx+Jlpa16TEdUT1uoD5eJmlojtmHKuGZ08Ut4Tyaz/u3fN
NExfJEzvO/aoZrjilrTHbad5By5GTHktOOVuvaXyMeW2t7fFMpXmcZ6pQ5VAqrDn7cL+/Pt
Y+Px3dQHLNjtNq3p7V5nW+2O5lVZK7bVtsmqKgY7Pmx1S7kNz/tuxRWFvtRfV0kTvUDyGiUqUJtq+1oN+QtlBtV8JjtXfZjcZvUiGbXpTHB/2R+ypMXxdsD9k/yZXDwx4BxbxdKKV/D2/jHSDLVRv/ELSh5TzuuO0z
TubmqdJkBSmasu7xWfN95WnL8yxIuidPePq8rhpLQSJfNx8ULyDrRwXYc2swR7LOZdHSzrrKuhjKL3
zRWJjBSfAR6N8zMdGGDpn3PsEdwOa7KfPFX63cywOvNHSmZpCmSdRHaZvhdUP0Z7rNoAtaSYYVI7ztwQ
cbsFuYmIn1ej+SUPhB+YqoKbA+fHcrZuoRyc3+9Y95NFBuJJZAcxhrJe9BTMvJbfR29cZvXcAeiuyu2efpS
XhiUOBmsGfNqyDBtz1UxqPGX5R2NwOObN8o+dwuu/aCHHW+pM9yy3W1CmYn5XJ+f8CNmWuWgxmr2dG4lZlh
6K1rZ4r6kE7bRrTBvhCaU1tUmF7YB0m1cQ+F8A6Zb7SN6xjmy+xjpRJQ7yDQXH5jpVZ1oG3eJ28bUY7P9+9
/Xt0Tv5QF9Pv8jKv5M/TIiMD9LRnOdaCdjNLCtVyFmtwKwuM7fAELgS4cEXCMijDGD7c2O6I76Y61l3PfMvd2BKNRI6sEUv0EtV5
Gl5Y0U9UcpaUmhxqxBztx1oJW6dWFpR2+Zi2LqB7IYHRBrM4FWYjcQqrGg9nZvAOi6QF/8ATGGVc+H6EdjxUx
jH1HWlRxiRLmWjoh6Xk9d47bzKx9yWXOP5WWobiDrL5LGLbpyVGP80lphGPL+tdHGRXqpv8XymFcQ23OO
NGOgBFtVDx8qy7VZ6pE6qd0Z/GMexhbPO5BedAqc9BOvKzFc6rBIEgRT+QdZiCYhmATrPuSxZmuuaf37m
ueh+TWqsGkIUjxtPUNtq0EJMbcYY2HdCOUhhcPrRnhebnWRpbpBNBOhbYk7Cbg5jEvJkz5b9nvZ9eBN2Ttw+uuSsZZJoPP5d0eMYoLuzLxHSqTDX
EbKi8Y05tH9T55mPhqT4nmO7jdIqpJo3xo7UJ1nB4A4QO2hkKEK/pO2Y0JcMvIh0MJCoN+Xu/NsFvY4cQpiQHFpcg5Zy5bpCnfvK
qLm5xazR9l7K/Dwb3KdhJmjW4+HJUT1UEtIqjBN9VLO4cK1mppTOtyc8lsI5aqjRPdyeENwKkRkLvEOtXr3KYI0Zq/cNNH3kgR
fQVH7hcYFK9QU6EnJ2DqM2Ib8l3R5CQDmndnTPt2ZLbP4iiw7mFUXrMoobVSG3kXbiLqS4rIbOZM2ZuHUY3EEtl2Xvp9bZn6p1

wPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm788D1lpu/PBRzOLw6
8DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm7
88D1lpu/PBRzOLw68DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8Ov
A5nF4deAJ/WWm788D1lpu/PBRzOLw68DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/
PA9ZabvzwUczi8OvA5nF4deAJ/WWm788D1lpu/PBRzOLw68DmcXh14An9ZabvzwPWWm788FHM4v
DrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm788D1lpu/PBRzOLw68DmcXh14An9Zabvz
wPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm788D1lpu/PBRzOLw6
8DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm7
88D1lpu/PBRzOLw68DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8Ov
A5nF4deAJ/WWm788D1lpu/PBRzOLw68DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/
PA9ZabvzwUczi8OvA5nF4deAJ/WWm788D1lpu/PBRzOLw68DmcXh14An9ZabvzwPWWm788FHM4v
DrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm788D1lpu/PBRzOLw68DmcXh14An9Zabvz
wPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm788D1lpu/PBRzOLw6
8DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8OvA5nF4deAJ/WWm7
88D1lpu/PBRzOLw68DmcXh14An9ZabvzwPWWm788FHM4vDrwOZxeHXgCf1lpu/PA9ZabvzwUczi8Ov
A5nF4deAJ/WWm788Hz1ppu9bgp5nF4deD7zdO5XgCX1ppu9bgetNN3rcFXN07leBzdO5XgCX1ppu9bg
etNN3rcFXN07leBzdO5XgCX1ppu9bgetNN3rcFXN07leBzdO5XgCX1ppu9bgetNN3rcFXN07leBzdO5XgC
X1ppu9bgetNN3rcFXN07leBzdO5XgCX1ppu9bgetNN3rcFXN07leBzdO5XgCX1ppu9bgetNN3rcFXN07le
BzdO5XgCX1ppu9bgetNN3rcFXN07leBzdO5XgCX1ppu9bgetNN3rcFXN07leBzdO5XgCX1ppu9bgetNN3
rcFXN07leBzdO5XgCX1ppu9bgetNN3rcFXN07leBzdO5XgCX1ppu9bg4abPTN6UtekztNep6PN07leD7FK
xO8ViJ/KAfQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAGL771iJmN5+DbN/bp9QORPfscie/ZoBnkT37HInv2aAZ5E9+xyJ79mgGeRPfscie/ZoBnkT
37HInv24tAI75MlbzEXnaGeeyd+XzL7231ZRhVVN2+eyd+TnsnflgHOU/W+eyd+TnsnflgDKfrfPZO/Jz2Tv
ywBlP1vnsnfk57J35YAyn63z2TvytpO9Imfk896FPYr9Fa8czO30AaAAAAAAAAOdYm02/FaNp26muRPfs
Y/8AX+poGeRPfscie/ZoBnkT37HInv2aAZ5E9+xyJ79mgGeRPfscie/ZoBzvWa0mYvbqS89k78rMvurfRBH
YM+SZjTfPZO/Jz2TvywIyyn63z2Tvyc9k78sAZT9b57J35Oeyd+WAMp+t87k78nPZO/LAGU/W+eyd+XbT
Xta1otaZ6kzvpPbt9B3RVMyqAVsAAAAAAAAzkmYpO07HInv2MvsS0DPInv2ORPfs0AzyJ79jkT37NAM8i
e/Y5E9+zQDPInv2ORPfs0AzyJ79k+a96ZOTF522VI9V77+g5q0zz2Tvyc9k78sCMMpb57J35Oeyd+WAM
p+t89k78nPZO/LAGU/W+eyd+TnsnflgDKfrfPZO/Jz2TvywCxVL0Y7IHyvsw+q9AAAAAAAA5xE2vb8Vo2
+TozT27/UDkT37cTkT37cWgGeRPftxORPftxaAZ5E9+3E5E9+3FoBnkT37cTkT37cWgGeRPftxfLUmKzPL
s2+X9ifoCLnsnfk53J35YgR55qm7fO5O/JzuTvywCZT9b57J35Odyd+WAMp+t87k78nO5O/LAGU/W+dy
d+TncnflgDKfrvgyXtliJtMwqRaf31VqtqJvAAOwAAAAAAHOIm1rfitG07dTXInv2fMfbf9TYM8ie/Y5E9+z
QDPInv2ORPfs0AzyJ79jkT37NAM8ie/Y5E9+zQDlli1Mc2i9t4Tc9k78qtR7myIZ8kzGm+eyd+TnsnflgRllP1
vnsnfk57J35YAyn63z2Tvyc9k78sAZT9b57J35Oeyd+WAMp+t89k78u2mva1pi1pnqTO+k9u30HdFUzKo
BWwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAzf26fVpm/t0+oNAAA
AAAAAAgye9t9WWsnvbfVlHnr2ADkAAAAAAehT2K/R570KexX6K1430AagAAAAPlrVrtyp236gfQAZx/
wCv9TTOP/X+poAAAAAAAAGcvurfRBHYvy+6t9EEdiMuTQAMgAAAAB30nt2+jg76VAK3AA
AAAAB85deVyd+v5PoM5fYlpnL7EtAAAAAAAAAI9V77+ixHqvff0HNenIBHnAAAAAAAAjb0K+zD6+V9m
H1XqAAAAAAAmdo3fK2i0b1neAfWae3f6tM09u/1BoAAAAAAAB8v7E/R9fL+xP0B58BAjyzsAAAAAAAB
00/vqrUWn99Varfj0ADsAAAAB8i9ZtNYnrj4A+gAzj7b/qaZx9t/wBTQAAAxbPirO1slYmPhMlcuO/s3rO3
ykGxz5/FNuTzld/lu6AAA56j3NkS3Ue5siRlyAAyAAAAHfSe3b6OODvpPbt9B1RtUAAr0AAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAn1Otx6bJWl4tMWWl4tMzG/VDl6zwfK/8AaC0Res8Hyv8A2nrPB8

r/ANoLRF6zwfK/9p6zwfK/9oLRF6zwfK/9p6zwfK/9oLZ7EfTOYzzh1HV3b/CXz1ng+V/7U2u1On1WLaItF
47J2B60TExvE7xPyZv7dPq8LS63Lpp2ieVTuy9bDqsepmk0nr364BSAAxn5yMc8zETf4btvlrVrG9piI+cg87
LqtZivWlq45tbsiOt6EW5OPlX2jaN5QarBfFe+qpmjlfCJhjVZ8ufHjrGO/ImIm3JjtBXpc+TUZLWiIjDE7V+cq
Xk9LtXU4opiyVpWu3N/N6tLcqkW2mN432kH0AEGT3tvqy1k97b6so89ex9rG9oj5y+NU9uv1HLtbBSvtX
2K4YrljlT1fBrNhnJaJiY/q+XtWs46b77T1yrW0R/j5mx05XVPXM7bPmTBSkTM269uqGsuOeci+8bbwxqv
eR9ESYj9mziAMx6FPYr9Hnz2NRi18xvXUUivwjb7DXjXiHmfSPmMfD7HM+kfMY+H2VquEPM+kfMY+H2
OZ9I+Yx8PsC4Q8z6R8xj4fY5n0j5jHw+wLmMuOMuOaW7JScz6R8xj4fY5n0j5jHw+wJ6azLo804c29qx2T
8dnp4s1M1OVjtEw8j0hhz1it9RlpeeyNo6/+kuny5ceSJwzPK+UfEH6LH/r/U046W17YpnJXk2meuHYAAE
me+spa00jHzcfGZY02o1WfDe8VpvHVX83fU441OOcdcvJ2nr260+ky2w5r6fJes0pG8W7NgYzarWYZrF6
497TtER1y9DHypx1m8RFtuvZDrNPaMltVTN11jeI2V6bLObT0vaNpmOsHUAGcvurfRBHYvy+6t9EEdiM
uTQAMneMFeRFrW23gthrG1otvX4y3bHOTBSInr2fJjmsE1md5kaWj4+5sePkR17dXV+bEYK8iLTbbdvJS
clKzWY6oZy/w9BZiN2cJiImYjsfAGQ76T27fRwdMUZLVvGK0Vvt1TI7o2tEPM+kfMY+H2OZ9I+Yx8PsrdcI
eZ9I+Yx8Pscz6R8xj4fYFwh5n0j5jHw+xzPpHzGPh9gXCHmfSPmMfD7HM+kfMY+H2B11uGcmPlY52yV6
4mE2l9J9fI1HVPZynTmfSPmMfD7PL1eO+PNMZL1taeuZqD38kxbFvExMT8YbeDos2es8jHvak9sT2Pejs
6wAAc9Rkvjx746Te3ySX1Or08RkzUpNJnadp7F1rRWs2tO0R8Xn2tb0jk5Nfw6es9c94HbLm1N78nBjjk7
b8q3ZJptTktmthzViLxG8THxdNTqaaakfG09Vax8XPR4LRe2fNP8A8l/h8oBWAAj1Xvv6LEeq99/Qc16cgE
ed9rHKtEfOXecGOs7TfZxx+8r9VGbDN77xMDumPxmmGK5drT9PzfMuOnORET1zO0x8m7XrGXHXfs7Z
ZvjmM0X6tuVA6tFrQ+ZMNKRP4uv4Q4Oup97/AEchnVa/4ACRt6FfZh9TaimqtNZ0+WtK7dcTHx4OXM+
kfMY+H2V6lwh5n0j5jHw+xzPpHzGPh9gXCHmfSPmMfD7HM+kfMY+H2BcIeZ9I+Yx8Pscz6R8xj4fYFzzd
Zzmiy89i93aeuvw3dOZ9I+Yx8Ps55tPrbYrRl1GOabde8fYFOl12PURtvyb92Xent3+r8zvMW6p64+MPZ9
GZc+SJ5yu9e9PaC8ABHqtRnpqqYcMVmbRv1rHy20RNp26viCDpOrpqKYr1x72+XydL6rLly2x6asTye20s
aKJ1GfLqbdns0cNLhyZcuanPWxxFuyvbIPQ0855i0Z4rG3ZNfi7ItHkyV1OTT3vOSKxvFpWgPl/Yn6Pr5f2J+
gPPgIEeWdjvTBWccWm2zgqik301Yidh1TF3O2CNotW29fi6Xx4+brvbb5T83zbmcNotMbz8Cac5hpyZjq
V3aPjNMFZxxabbONoiLTETvDvfq0sR+adHFVoABy6af31VrzorltaIw2it/hMt8z6R8xj4fZW/HpcIeZ9I+Yx8
Pscz6R8xj4fYdrhDzPpHzGPh9jmfSPmMfD7AuEPM+kfMY+H2OZ9I+Yx8PsC5Jr8N5pz2GZjJT5fGGOZ9I+
Yx8Pscz6R8xj4fYGdJ6Trfamb8Nvn8HoRO8bw/N6ik481q2tW1t+ua9iv0bn1EXilYm+P47/AHr4+2/wCpp
nH22/U0AACLX48GPHbLfFFrT/25ej8OG+C9Yv8Ajt7W3VMO+XVZcWS0W09rU+E1ccUZIzZdVOGaxydo
pHbIPus02mwaafw7X/0z8ZlXpYvGmxxk9rbrebjyZJzzm1GDJe3+mIjqh6eDLOXHFppNPykHQAHPUe5si
W6j3NkSMuQAGLvjx0nFy7k4sd6zOOZ6vg1jrFtNETOxHJwUt+LeZVraLNW5vmq778n4MUxY5xcuz7WK
5MEV5URMPk7RpprvE9f+UX1xvyeXPI7GQGQ76T27fRwfazqIt/8AzxWZ+PKHVG3oCHleke7iOV6R7uJX
oXCHleke7iOV6R7uIFwh5XpHu4jleke7iBcIeV6R7uI5XpHu4gXCHleke7iOV6R7uIFwh5XpHu4n3Q58+XP
kpm5P4PlHxBaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACDURE+ldPExv8Ahn/K7kU7teCLUfzbT/pn
/K4GeRTu14HIp3a8GgGeRTu14HIp3a8GgGeRTu14HIp3a8GgGeRTu14JtdmppsMzFa8u3VWNlc9iLoc6j
Nzuo7P9NAeZp9Jl1Nt4javxtL1sGkx6aacnrtPbKmtYrERWIiI+EM39un1BsABjNipmpNLxvDYCOPRuKLbz
a9o+Uz1LIiIiIiNogAcraettTXPMzyqxs6gAACDJ7231Zaye9t9WUeevY+1na0T8pfAcumXJy7bxvDmATN33
efnJMzPa+AAAD0KexX6PPehT2K/RWvG+gDUAAAAfL2iIJtadoh9Yy465axW/XXfs+YPJtizekc83iNscdky9
LTaPFp6/hje3zl3iIrG0RtADGPtt+ptnH/q/U0AACXLoMWS83ibVtPbtLVNDhrjtTaZ5XbMz1qAEfq3Fv12v
Ne7urrWK1itY2iOyH0AABnL7q30QR2L8vurfRBHYMuTQAjF1vl5WOtY3iY+LnMzPa+Aszd93n5ybzttu+A
AADvpPbt9HB30nt2+g7o2qAVuAAAAAA46vPGnwzb/VPVEPN0/o/JqL85nmaxM7/nL1bYaWyRe0bzHZ
v8GwceZphw8nHWIh2Zy+xLQAAOeow1z4+RaZiPyTR6Mx1jaMmSI/KVoCO/o3Fe0Wm994jbtbw6KmLJ
F4yXmY+EypAAAEeq99/RYj1Xvv6DmvTkAjzvtZ5Non5S1kyTe8zG8MAXH3efnL4ATO/aAAAEbehX2YfXy
vsw+q9QAAAAAA830jmvmv0bBEzP8Aq2elPYxjxUxRPIjrntn4yCLSejK49rZfxW+Xwhbjja1oj5ts09u/1Bo
ABnLjjLjtSZmIt1Ts0AxhxVw4q469kOWbRY8t+Xvalp7ZrOygBy0+mx6eJ5ETvPbM9suoAPl/Yn6Pr5f2J+g
PPgIEeWdjpOX/AOKKRvEx8XMCJs+zMz2m8x2TL4A+7zttv1PgAAA6af31VqLT++qtVvx6AB2AAAAJtdqe
YxbV68luqsKWOZpzvOTG9vhM/AHmaX0bfJPOZ94ievb4y9THjpjrFaViIaAZx9t/1NM4+2/6mgAAAAAA
Ac9R7myJbqPc2RIy5AAYt85PNcjb+rAC3AAABB30nt2+jg76T27fQd0bVAK9AAAAAAAAAh0X8bqfquQ6
L+N1P1BcAADN71pG9p2Boc656WnaJmJ/Nq+SuOY5XxBocukY9+2frs67xtvv1ADl0jHv2z9dm6ZK3mYr
O+wNDNLxffb4TtJN4reKz2z2A0M3yVx+1P9Hymal52iev5SDYzN4i8U+Mvt7RSs2nsgH0fOVHJ5W+0bbu

cajHM9s/XYHUZvbk45t8oc8WPfDETMxNuuZgHYTZacmYrW1pvPw3dZx3jFFa36/jIOhu4XxTSk2rktvHz
Mm+TTxf8A1R1g7j5S3KpFvnD6AAAAAACHUfzbT/pn/K5DqP5tp/0z/lcAAAAAAAAAxf26fVtm/t0+oNAA
AAAAAAAgye9t9WWsnvbfVlHnr2ADkAAAAAAehT2K/R570KexX6DXjfQFagAAAAAAAM4/wDV+ppnH
/q/U0AAAAAAAAADOX3VvogjsX5fdW+iCOxGXJoAGQAAAAAA76T27fRwd9J7dvoO6NqgFbgAAAAAAA
M5fYlpnL7EtAAAAAAAAAI9V77+ixHqvff0HNenIBHnAAAAAAAAjb0K+zD6+V9mH1XqAAAAAAAAAGae3f
6tM09u/1BoAAAAAAAB8v7E/R9fL+xP0B58BAjyzsAAAAAAAB00/vqrUWn99Varfj0ADsAAAAAAABnH2
2/U0zj7bfqaAAAAAAAABz1HubIluo9zZEjLkABiACgAAADvpPbt9HB30nt2+g6o2qAV6AAAAAAAABDov4
3U/Vch0X8bqfqC4ABNk5U6nqiJ2jq3UsZMUZNp3mJjsmAcslM142mtI/ODPG9sUW+fW3GGd45WS1oj
4NZMfLtWd9uTIGWsTitG3wcLTPRK8JU2jlVmPnGzNcURijHPXAPta15ERERts5aaIi+SK9m77zEx1RltFfk+
aaIrfLEdkSD7p/ayfqfMn8Tjath3vNqXmkz27PldPtkrebzMx8/iDH451N+TFZmPm+2plvNZmtImJ7YdMm
GLzFoma2j4w+VwzFom2S1tuwHy/8TT6Pupn/AOCzWTFGTbrmJjsmGLaebRtbJM/IGM2/M4ojsnbtam
Ms12mmPZ1nHE44pPXEQ58xbbbnbcn5AzatqaSa27Ydsfu6/SHy9N8U0j5bMY4jLgiszMTHV1A+5METa
bxaYt8zBeb4t7dsfF8nBaY2nLaY+TV8MTijHWeTAOdpnPPbk16qR2z83W8RGK0fCKsVw3rG0ZZiPoZd6YZ
rNuVa3VANaf3FXR8x15FK1+UPoAAAAAAIdR/NtP+mf8rkOo/m2n/TP+VwAAAAAAAADN6zO01mImP
m0Aztk71eH3Nsnerw+75kzUxzEWnrn8mY1OOZ23ngDe2TvV4fc2yd6vD7tAM7ZO9Xh9zbJ3q8q8Pu0Aztk
71eH3Nsnerw+7T5yoi3J3jefgD5tk71eH3fNsnerw+7YCe2mta0zy46/yPlE3ky0VX4xyfFLmvD/cFI
GFKbotu/HA6LbvxwUgYUpuiT344KKxtWI+UPoLERGGgAUA
AAAAAABjk3iZ5M12md+uH3bJ3q8q8Pu0Aztk71eH3Nsnerw+7QDO2TvV4fc2yd6vD7tAM7ZO9Xh9zbJ3q
8Pu0Aztk71eH3Nsnerw+7QDFq5LVmJtXr/ACceiT344KQSYidpui278cDotu/HBSCYQm6LbvxwOi278cFI
GEJui278cDotu/HBSBhCbotu/HA6LbvxwUgYQm6LbvxwdMOGcUzM233233/ACdQIpiAAdAAAAAAAAPl68
qsxHUztk+deH3bAZ2yfOvD7m2T514fdoBnbJ868PubZPnXh92gGdsnzrw+5tk+deH3aAZ2yfOvD7m2T5
14fdoBnbJ868Pu5ZMFsluVN43+ifo7gTF03RZ78cDos78cFKbotu/HA6LbvxwUgYUpuiT344PlcOSJ3m/
BhSm6LbvxwOi278cFIGFKbotu/HA6LPfjgpAwgjqgAdAAAAAAAADHJvFpms16/nDYDO2T514fc2yfOvD7
tAM7ZPnXh9zbJ868Pu0Aztk+deH3Nsnzrw+7QDO2T514fc2yfOvD7tAM7ZPnXh93yYyTExyq9f5fdsBN0
Se/HA6LPfjgpBzjbos9+OCkFTvxwOi2b78cFIGFKbosd+OCkGs9+OB0Se/HBSBhSm6JPfjgdFnvxwUgYQ
BhS4YtPNMkWm0Tt+TuAsRYAAAAAAAAY5E7zMXmN+vsh95Fvnzh9zsw+vsh95FvnzhDQt4Ax2xLeJPC
DkW8SeENAM8i3iTwg5FvEnhDQt4k8k8IORbxJ4Q0mxzmy8qYy7i7RE7dkA62xau1u9IY6JXv8Xv2fa3vTLF
MkxbfsmG75aY52tPX8jiV79jole/xe/Z9rfaole/xe/Z9rfaOnzWKcrpVvkpe/xe/Z9rfZ0SePHHEzY0O/zdAW0QACgAAAAAAACHRfxup+q
5Dov43U/UFwAAAAAAABERHZHaAAAAAAAAAABERHZAAAAExE9sAAAAAAAAACHUfzbT/pn/ACuQ6j+b
af8ATP8AlcAAAAAAAAAACbPaK6jHM9kRLpXNS1oiK23n8nzJWZ1OOdp2ip1OPdp2iJ3l2BPNslRtwava83rip
PXtvNpKVmmNTedp2i6zJW1MsZaRyurayEw+Wm+eYryZrG+8zLqVmK9R+5fOqI7Ts+pExO8dcuF8jdhQw0
TPS99p25PKZaD5vkxZGcxoY/KradutjS+TW0+Wm+YryLVrE7zNmu
S3KrvtMflL6AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAkwW
yRy+RWJjlfGVbGPHGPfaZned+sHLFysuabX2iadXJYx85OTJNa1md+veVHNxznLiZifj+b5fDW9uzg
HOlb1y2vaKxG3XES+RbJfHHM0t1ryaxwTaPhEkYaxinHPVHvO0t
1ryaxWPhGwPoAAAAAAAAAAAAAACHRfxup+q5Dov43U/UFwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAIdR/NtP+mf8AK5DqP5tp/wBM/wCVwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACHRfxup+q
5Dov43U/UFwAAAAAAAAAAAAAAAAAAAAAJNVo7589ctM3NzWNo2hjoWp87fa87

Wp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHo
Wp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp8
7fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp8
7fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh9
1wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9
zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wC
HoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoW
p87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoW
p87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87f
h91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87f
h9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91w
CHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zo
Wp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHo
Wp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp8
7fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp8
7fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh9
1wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9
zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wCHoWp87fh9zoWp87fh91wC
HoWp87fh93TSaS2nyXvbLzk27epUAAAAAAAAAAAAAAAAAAAAAAAAAAbxIAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAACP0lqcmmpSce28z17wseb6a91j+sg4V1+ttG9axMflV9j0jq69dqR
MfnVf6N/gsaiaxaNpiJgEmj9IU1E8i0cm/wAvmseFraRptdvj6ttrR+T25vEU5dp2jbeQaHl5vSeS95rpqb/n
Mb7sR6Q1eKYnLT8P512B6446bU01OPlU7Y7Y+Sf0lqsmm5HNzHX27wC4eVPpPNesVxU3tt1zsxHpHVY
rRztI2+UxsD2B5+b0rSMdearyrz8J+CedbrY/FNdo/SD2BHodfGpnkXjk3j91kzFYmZnaIAHl5/SlpvyNPTf85
+LnGv1lOu9Or867A9gTaPWU1VZ2/DeO2Esa7Ni1vNZpjk77dmwPTBNr9TOmwcqvtzO0A7Zq2thvWk7
WmOpB6O0ufDnta+8V22nr7XfSZs2TS2y5Jjf/AE7Q5ejtXl1OS0ZJjaI+EA9AR+ktTk02Ok49t5nr3hLHpTN
bHWtKROT4zsD1h486/WY53vWNvzq9HR6quqx7x1WjtgHcYzZaYcc3vO0Q8y/pPPlttgptH03kHrDyaekd
TimJzY96/TZ9xekcuTWVpWY5u1to6uvYHqgT1ROwJ9Xq6aWK8utp5XydcOSM2KuSsTEWjfreFq8+fNye
erMRHZ+HZ1w6vWUxVrjpM0iOr8G4PYz9WDJMd2Xmeh73tnvFrWn8Pxn83oWm1tHabe1NOvg8bRan
ot725PKma7RAPfHkTrdbP4optX9LvovSPO5Ix5oitp7JgHoA87Vek+ReceCvKtHVuD0R4/TtbX8VqdX51Xa
PXV1UcmY5N4+AKgABLrNdXTfhiOVefgh6drb/AIq06vyqD2Gc3Vhv+mUGl9J8u8Y89eTbs3XZvc3/AEyDz
PRF721F4ta0/h+MvWeP6G/iL/pernzUwY5vedogGx5FvSWoy2nmMfV9N32npPPitEZ8fV9NpB6wziyVz
Y4vSd6y87V6/Lg1fIiY5Ebb9QPTHk39I6nLMzgx7V+m5i9J5aZOTqKxt8erbYHrDzc/pOZtyNNXlT83GdfrM
c75KRt+ddgewOGk1VdVj5UdUx2w6Zs1MGOb3naIBseRb0lqMtp5jH1fTd9p6Sz4rRGfH1T+W0g9YZxZK
5ccXpO8S87Xa7Ng1M0pMcmIjtgHpjyr+kc+W09Hx/hj8t5Zp6S1GK+2akTH02B66fV62mlmsXraeV8nSu
WMmDnMfXExvDwtXmz5prOeu0x2fh2B7+O8ZMdbx2Wjdp4uPWayuOtaUmaxHV+B6mXUVwafnMny
7PnIOw8ifSOqyzM4sfV+UblfSOqxzHO494n5xsD1x8pM2pEzG0zG+3yfZ7ATa/UTp9Pyq+1M7Q87Fp9X
qqc7y+rfq3ntc9bnz5fw5azFYnq/Ds1p9Vq8eGtMVJmkdk8ncHraOmTHgiM1pm7s56e1r4KWvG1pjreZT
0plrmnnNprG/VEA9ceZh1uqzZq/giuOZ6+pbqtTTTU5V+uZ7I+YOw8efSOqybziptX8o3bx+k8uO/J1FOr6
bTAPVGcd65KRek7xLztXr8uDV8iJjkRtv1A9MeTf0hqcszODHtX6bmL0nlpk5OorG3ziNtgesPNz+k7Tfkaa
vK/3S49P1eOYnJWNvzjYHsDjpNTXU4+VXqmO2Pk7AOefPTT05V52/L5ujw81r67XciJ/DvtH5QDtf0plvb
bFj6uLEa/WU6713j86vVwafHgpFaViPz+MukxExtMbwCLS+kaZ55F45Fv2la8n0no64ts2KNo364hZ6Ozz
n00Tad7V6pBUPNvrc1fSHMxMcjlRHY9IAS+kdRfTYK3x7bzbbrj8mtDmvn00Xvtypn4AoEus1tNLG23KvP
ZCHp2tv+KtOr8qg9gedpfSfKvGPPXkzPVuvyWmuK1o7YiZgGh53o7WZdTmtXJMbRXfqh6IA87X6zNg1E
UxzERMR2w9GJ/DEgDztV6T5N5x4K8qY6t3Dp2tp+K1Or86g9gS6PW01Ubbcm8dsKgAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAHm+mvdY/rL0nm+mvdY/rIGi1+DDpqUvaYtHb1Ot/SuniPw8q0/LbZx0egwZtNS94typ

+Uvms9G0phm+HfevXMSCalcnpDWcqY2rv1z8oXel8k009aR1cqet89FaitsfMzERavZ+bXpbFOTTxaI3ms
7gn0Gp0unxfimecntnkqMnpDR5KTW0zMT/ALU/o/ouXHyMtK85Hz+K22m0lY3tSkR9Qeb6OyRTXcmk
70tvDv6a/wD8v6qdLXSZJm+Gld6ym9Nf/wCX9QU+jcNcelrbb8VuuZa1+Ot9LfeN5iN4b0n8Lj/TD7q/4XJ
+mQeX6Jw1yZrWtG/JjsezMRMbTHU8n0L7zL9IesDxIrzPpSK17OUt9LZJpporH+udpR5v5vH6oXelMU5N
NvWN5rO4MeisFK4IyzETa3x+ULrVi0bWiJh53orVU5rmbzFbVnq3+K++WmOs2taIiPzB49o6J6Sjk9Ucr9
pUel8PVXNWOzqlPWZ1vpGLRH4eVv8A0exnxRlw2xz8YBy0Obn9NW3xjql5+uvOq1tcNeys7Oel1M6O2
Wlvl1fV39E4Zvktnt9IB6FqRj000jsiuzzfQ3vb/R6mb3N/pLy/Q3vb/QHb017rH9Zb9FYK100ZNom1vix6a9
1j+su/oz+Ax/1/7kHfNjrkxWraN94eT6Kmaa21PnEvZnsl43o3+Y2+kg6emckzfHj36evJiN5jeZ
S+mME2rXLWPZ6pa0XpDHOKKZbRW1Y23n4Y23n4KTW8bxLw8NOb9JVpH+nJs9Tr9JpH+nJs9TNr8GOkzF4tPwiHk
6W031+O09tr7yD9AADy/TXZi/qt0H8Fi/Sj9NRPJxTaN6eb5fxJ7Kmaa21PnEvZnsl43o3+Y2+kg6emckzfHj36evJiN5jeZ
o3isb7PYzTE6e8xO8TWf+nl+hff5P0/5B7G0bbbPE9l44w62Jr1RO09T23j+mP4qn6QehrMs00dJ+VMRhNs00drx2zDzP
R+bT4bWvmmeV8Ord6epxTl0U0jt5PU8vQTgi849RSN57JkHoT6S0sxtNPvNpmP0vNnJjprq5ME/h3+T+Gx0WmI/S82aV02umM1OVTf4/J
6lNPo71i1aUmJB53pDLp801vhn8fx6tt3o4Mk5PR3Knt5EwxkroceStLUpyrKL0rj0160IrFZ6oB5nob+Iv8
ApffTOSZzUx7/AIYjd89DfxF/0temMU85TLHZttIqO2n1ukwYq0rMxO3X+FnV6zSajDau8hx8rbqnk/FrSV0efF
E8inKiOuJdcuHRYqTa9KRAJvQ2Sd8mOPv1+ez1dLTT8mMuCsRyo7eh9eZH1gHr4cVcWO
taxttCD0xjrzVbvbSjsQ+mP4WP1AzQ6zwH6ezHoitZZjzdt65cNq2jsQv1MfwVh1YnerEnDP2aErzF/EzPUnMt/RR2hQ
8Nn30xkmc1MfwiN2fRP8ZbkQ3Oez7LGNoXhnA3J8XiH3TZDpC2wqJEqeA3V6FxOBX0jE/EZ91evjOQR7edCh+uhLu4JsXVWjt1KuRvtZKdvT31mYjb
TzdIvEdcS65cOixUm16UiATehsk//Jj36u2HD0pG+vmPnEPU0tMHJ5zBWs7XuHjyeX6U/jp+kA9fT4a4sVaRERtDhJhCO+dK4qjrtxHIoaT
q603pTFW2lm20b1+K1N6QD/gsn0x9MyWa8DeqoeDBlo1PzM7DMxJTQe7QXo91d49R8WOnvkuyqbW2K
bfyXa7KlyZx4mbFq02cnDBtCnVbP6b2lpQHLSA7NXIqLKstX+W0IlFs5czPBQI/YrrrxPpb1eZHhY9z/1NMMC9ji7xpkDnpNDLQUZLXvZYN9TKg1Ne
j6M/gqf1U5PdW+kg8n0PaY1+Y2Hjeh/wCJt+l7IM5Pd22+UvH9W4mqx30Ws5ysfhd4m/lmvx9D6Ya1mleQ
B7Yn0+sxZ6xMt+WjLIg+GJdrZKVje14PS+jjSyDLOuMqelQtLGZU3i8zNYMwU1AJHGYfyNXXKnLiv/Vfs/I8G4t50vDE5YsHS11+oC+ss
/WQeZzlL66b6ifwcqd/i9OPSWliNotMR1+l5tqq10+umM1eVTfr/OHqU0+jyVi1i1aUmJB53pDLp801vhmeV
8erZ6OnyTl9G6Oq2Cmhx+3rS1KbypH3U7NeodJmJyz9+CCYDpqT3C7OoLyanlOAnfdqtPQN8tYuKVCagzrVUrCaxKjQXGo5bKT4J1YWcMhkezPPy3Q
anJSK7zaIj57g8j0t+G1/+kPQ1uScWn1Zr23UtRkwGwAvTxXn20jGN1G0TbeL9dTAsdoLgswWmFqAHZmkrWuZmRIgB37m1qTmnsXHfsQ7oAbPEpuYrA
8+k9LMbTaZj9KD0fOn5dseorG89ky9Pouk235umwPJ5ylNfGTTz+Hldj3kWKmhyZppSlJtVaAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAl1+ltqqUitojkzv1qgHLSYpwaeuO0xMx8nUAedk9HXrqedwXivXvES9CI3rtaI6463
0B52f0VW9uVityfyco9FZZZ6r5Y2esA4aXS001ZivMYTaI5PzWAMYKTjw0pPXNY2M1JyYb0j
qm0bNgIvR+ivpbXm1omRLRHYtAEF9Be2t5+LRLRtvvsmvMN42kAefqPRdPNRIbiWqo+LqZrTUbTXTlKA/ZbTsmAfRwjnUPAjGEisw5EhHbX/XHdKKJvuMT5N1F4ZNcJHKkA8Ndn8aWfHpdOS7wj0oulUcfUgTZPJ01TtdCPJgsG0Ph6IvUgj8UNvWqzqaMZ
2nfrdNJhnBpqY7TEzXfguj1U5bWiYnfqheAzktStJnJMPPsHsvX5DCv0lRWA7MUtoV6wG+sNvNiu0A1u5EjlnvwIdLb42KdnhWiaYk3rPuq4Y
Ry5r8XndE12CdsVpmP8AbYFFPReDF+PJabRX19UI9HWMnpKJpH4eVNobnTa/P8Ahvytv91l+i0VdLW
ZmeVee2QVAA5anT11GKaT9ppa1ocknDhjwTj0nM8reeTkuGg0N9Lkta1txuDhVpX9eoneNupdaxCi2+qiJFwsVbWtkXXqzSozcn5DiaC5UTYSkjGzXY2+9o5HypSBamNAIpl7tpcZNfEUHRq99a0SP67mTS9wPuwSfZGocSmaLSHMsmknl0DrthHTymyDFY2h4q4XnVl5b1d9Lhk7H5VkVCHFOxWEPfT1ld8MFTL4mzbk1WA2xMn0jvLmL5IpW3QOCS6D05cYTmEQYnuWBFr2EqSwXbgF57xXKaDQtbWh70dgSajNLyjTiBrAAMlBRDXsj9KrDKszlSQ3BEi2JE0SWF7dzmjn1AapUXZCT31I4VNWQbgHKLB7bzGq/K2Kk9psW8M9tnLY4HBY6Eq4dwOX68k3c1zaUjFe0O9OxFeHB+Ww7c2ckXZT47OTnBmDaMjOBRs76sJVs0xC8xbZMOXcxZr6/tYGTW6ZDHTxsWjsQQ/Ss0Hr3XFwc7WWdp0Qq4fF7VZLaEoqcKHHyvN+z4VM9gvR2Ryuyl9WU9aqTzRTcIlKIuFzx6dIiYVXC6YaDiKw2tI+WEKoTlC0D/EkKQT2XdTJW2PK9KMawCzZLmGsB8L4z16+fxU09RjDD10Qi8hs57eFwYdcHWn07Ra1ZP5p4iHuDmhQHJTAeQVWiwz6hn2P76ibZGmrwxSBprGRrZ3WwImb+HwVpjdvUxjJDF+7Nh08yaA37oT7nnSsKV+FlsXhrbQJO8tmPpYGeNeDAQG5Xt0Tr7S1JxhRoD8ok3fiFDdTzMe8vt9+/UMGzyAPa9hkavKEg/Ed6G13vgh5m6+W9kkxDG8kiJ2mB22AxHlobWZa3lbo7CVWbG7Uce6fdCySpxaVdA3L3afNyGMJdDw9omkgfj0OkfNBO24VEzktfGTJ5d9i13gdTnSTkW4K1oUlBt8AZ3J7/dyZLIHr9IlEKfcVmrMhYJhO2yN06U0WVyvYPFNlvGMmKnmSxz3nrVfY+2/Sq2E38n+7wHV6Xw7DCm39ZEl+6oPP9x/85Cfr8zutvKeNIIpT2i4eMTMX0DI3TB3UzoNqXxb15Ey2BbRVkRTvt7j1VjEqEV8lVbFKJyrm8ST8/CAyRsbfBEZyIdpmpUhqu0w6TCx7xDbpJjN5X15YDwBRnGQRzgJY0r0kVAjSaG+BCkEBSs8dWtebcwzJbQsHDa6pEm17C6DyeLmJmS2Kd9Vbf5kRo9aW8PJ5Z3vX6ULGfDNaQuIfNDTUYjqyJ7JzqtXn73oJPtpmYuHQY20AYsxa4wXmybFkOMBtE8E6Q91ZB39VnwUNKQkiX1grUk3fi7Tzbv9SSxjNbYIhCvDBrE1lGZyV2iHXnDAFcVKXixKMbt4WYUxr0IZ6m2htsv4x2K9T4rLmRF0zq0+vQeuvNj6r0FXDb5O2CnhtY1E1sjz+Z8lUrjCn0XsBrCCR4wjmfAXJyEd7uRVzsnTzbdAzcMHJ7XDGzJsDJM+hfrqXvW5z7U5K1m4OFpIPYtzw5lH1Pi9ILMknFXAkwi3jbW5rl2D9OwqNGqffs9r/UBqK1sxgZ8tvDynK8dRyp3cTDgsXuU4UMrRUg2aLSbCCKC4N8vrkHeNbW0O+wEBbIxTPUOjlTtBwYL6rCYr+hp5N6W7xGHRaNhlYCL+y5XhY6ofHUq9k1Pxn6Sg4CnkPH0WpMbYjmWTnWbbrJTmuqXNYrJeHX0s7RBPtpP0A1sKOQ4hrVEmW2+FuIcOxjAOwo6+6JJtoyAIT2eYaPUe0VE7qZKLCqZHfLlpV1Vp4xsVLY1xDtlk6vA3T9YXn3TfHU8AIGTg6z5+hd4aZ5zmTyUSyhAwKXcvRc/6Zt1eqQm+8j4K3VH+4E3GEQaKIqTrHdnqgLn4xzymXTBcg5Q0BZspdGnAZEtTe7dIVi8jmk/U3pHxpuvkOBN8jTb9MHS7eaOpAYgKL8bi2/CvYC85X/VWKqqBtvBC0LHTDdDkkgpAJ+9aUJKLkZ9pjn59RsCIbpjMmtUzDdcUYmKSrdrRH5/Wq/tKhD9mzQaR5CXuiwiQzaI+k5E08wPgUDE5fHA5iEUKEsIvsOSuOpTWmK1NPxjGVpP/ItklWFzGuAjRK9QV2YkC9JdIRAb1A+fa9E/7hP9ZTZNwD/APS2EcexD3eu0ssMH1Kg7ewoZcEaARhw3KUSbWGgq/sCUUSk6Us6KITtkOQa1F2CJ5c0GhPJBubfT8cDK92ccIgxgmyaR/31iLk49t2nb8SzD/U2qMAP2qRUbqSYTDFmKLKTE9S9AA+A33HEx+73DGVDZs4o/4ZNsaGTUcRmvnvdoXMyPlyk54d4p0bmuMTu68bEEuAHzftqsWcZ44Lk0F+IXB/l/0Hcm6sL6E9Pa34d3ut40pXHIL58fvR/lH4jS/ns2Ea7Q+0H9vLPAmCyOawtmn9a3BuacVzbXrwSLbTPx6Qg5kvyYqVctObfWf/Gw0OQm1Y8fuYMK4hDT7Kg6zvy4iiRQfZaz/4NhM8+/Ek8lFcPsP/KYD2CwF8nfy3bEbPyr/MxzZSjIdM49XVxQBqm6Un2B9KuHR8slnCl/rHeVaDhF2cFpZ/x6WEXmmZO8EM6efJM2z8c6S7cIHLwAGbRYtMrbadX9LNn1TVaMy/USCuYRrpcg2oNDKXrhdmlR9XBIKklMkMKk7G31sKqAvr7aJ8Z+qGDgUeeDZ+38dmyHMh1dDh5NBO9LWAH4wbRshPgubzs8aX1qjyL4RPfs7QUjOkL56qdV5nrXYZCrN04I3qmVbPX3Fwr+gWfuTq1uybTP2njrRm+wsi5XjMe2H+UDVUMZQvTHHTtnTIjrd4J5wbt6/hMJfVKPFs5nqq9HqWvZVrLbIqyhQO+Kd6mvFXSq5h8Xll1LDL4gyV8m3D+H5bOuc8iGg7F4mUaf+OfR5CGlxHPRTM/JVXrcIRLKm7OL2QzmKNwp4Tw4pOjNs3DbZrs7z8OgvqvKLbbjQfR7pvhJD6O0/wgyIiNaxAaBUKUtm3/FUbKyRFmqrsldDBG0VdNj9fItRp3FvQxCk0A1pJDbkOcnOCJVotrQG/kxQyR9sTzBjM+UxZ2BOqNHSe8s0EG7L1sWx1hhGI6cb51/7xtpUaeZqa1eNgtOFfTKHb6v2Dd3dZeo/iGSgt4/YDOrmUvMmlSnS79+WEYeqUjzqwjRuJ2VwPZYjC2ubaspzlnsMWO2JMeM9RHq8dWjrJUCxyTj61MIhTEH0w7xGaYF1ajv1AKyq0m4dpZn9lStaZTwNbRslM3LK7xX3+5Az/Q0uHxZrUi+EMH5yrHMuLH01SwW9ROoO92pUrB3YZuuGv4R/g0MOLp6ASMZlp5gjYxu+GqwW/TqsZk3VqTcQHpwaBmTqHBYg+KkzbrLqf2yTJ4h/1OhxG6WU5nKl99HSLRQwQc/jl9VHNQJH57pTk8Qp8AaKqWmgxK54DxXr3qj8B8mDMMP/dGT5YPsSXR8dDkG/jXNN9vDLrjnMh3FrjbxRtSaX8BXWg2LLAfUg+Q3SxtN+Ie2ESTDuFoPLFvrsG8sDqXV2N+CpNCW+fu8FxBn6WUpeFAsmQ67bT9c2rXKq9xjvxe+sP3Ha4ANj9dSz1NH5W2n6pR3uozHrudtgiNC5SZAgKsqzYC2nLgTddvDt2DhK/QHtvvEUjlM7jWEFjWnJ7mTIfbiXE/SDbFkh7ImKlfT/5kN7xq1YnivFKuqGSVmnr33lhpzHxxcTJyA/GwWmhXqEOpY76hHf6qN/B8cZS7uJfZ9tU5TzgtQxlG2Evfc7YQqkBOGJfIT3iBx9HOOsa1DH5t19QAIP0pD8pL2PTr9Edsk5k8iY9iJ2GDHL3sv9Of+UvSgMDW75Xe1XPqZ+U57tGgQGQBP/eXx3dZuEx9hqVlcNBE7VVL2Rq/HHzGT3E7uoyH8RkDwzvLaD3Xs3/JZW7IJfTxLSu1Vzs6eEqpmbyNGWzyiOB8wZ0ojQVZsZWGbrh7fDvWnKF5nOQtYcJgw+iUi0YvFrYFqcN8fIqwpP7sb0uKJO5EgkX7z1j2CvHh+pFXA0N7uqZnpp/Vg3Y8sY+HhvEjUcTrhrsL1Xh3LW3LiOoZL7ENcpvuQfy4ML1L/G8BRkxDNbU8EiOSHOCmtvkv8pBRZGTOI9OCOs7SEZvMHbJj9JD+UnOvmjH8nqcu2iU8kpDU+OP1jpLoGhCpBYdENzXCBvrS2bGiMUamqp6y6RYt8kSMxbULnvL1JE2PnrMOmM5sMW4ySYijkFO5rHQDgZ8unVnftDJC+ApPD2jdqHrWv6WS6dH7Qy37vgRzJlDdkNKJTLuHw29dLbsJSnUnDhnk5Qaq4TkJLtU/XiuCYg9nFMt7GbJbsgm1VK3Y3ukD/6xlAedrtSQKLltUwPU/8Nuv7PejYxmFPQbrNIjWtt3EC6s4upQYxQvfpzPKsYE7nXTL82lTTzo+nUPxtQkrf/2OLsWeTBo6I7f8gdP4iD2ZB/1bqt7q8i8VHp3mWlTUHJ6NDWmHTvSqVlQ8UcjWG2e7MN3iXxvuPHd6jjUH/8yAeOlmQn8UzZ7JEgz5dO9kNWtavB2NY7jAUnaEZt6yh/K0s+zgOnbIuCFMcV5D4QIQEvxrkuUbmfbN6WknoVdsiIFuoRQHcvP0xHfs6TK+Lv+6SYsGmMUpqNVyFkMYCYxKLrjU90AS0v1JEa8oHtojJtrN+NaUsGXM4rprTHgF+UZdRPSlcH41v8BqAEnw4yhk40xeL5r+bqoSKYDxgZQPyFYLWIUsSzs2qcS6AflI80e3QyqGcIvqCeWySQfHIRytDPnQn7CjIyUFS7XutQbjzt1FOR6VEbv4Lf4SQO49R0pEAW5mP3JSItE1ML3FUUt8nR+vGUo1VRr07gCGHXk8zjcQGTJ7UJzXtVsp1+7SobL3MbrWWUqpeqPqFmAm4Eia0lD3lEBbPxCQoNF3S9oGcvrq7k8ZAAXnU4Ec4JAT81+HEr5OqNYDc8N+mHFzwZ1X5Ejbkv5uXh8wR8F7gkHZdPtx3ahVC7+jTZC1lHqWsC1w1pIeCyvAtMNmyvIdqGQbGyW+eq+zf1vBWDyeB8Ydb0CBuXW49v+kUaPE3A4lHMcrAm+Y5T2NTlEXNBKcs5ga90BDBLrY5FcvwAtQmqrXTFLfkNHq+vECu2EqdIQ3WMJgbbjb/0G8J5YKrLtq0PzvqnVEzOxsobAmgWOQ9/Blhkpd99Z9RlOG3SgdEuEkDoNj+l8Ye0h4mNvW2kvPZPdMVPAl/5sFBmaQJX9c1l0bf4tH6MJBHcmcWfQZGBYiqn+hRDA2P6ob/BJXlqH7F/3GOo3/1b8oKWXdrJ3apzA5DN5BZxcIA+4Eiw4VStQ/kvF6L1ouE4D2fVhIFG3TvEcvuGaJkRbPlhHMhi8jc+ahd6J3JdIkx/vWTCbyq6GpT7CeBXcWYJ4yVdwTnNRJ8J9+KLu0xzpsj6u/L2K7NNoBvQqnVGvODqE2sYEqkpzEh8M3jVk0RiBsm2ls2NphkybSvO9HPaeh7UhGXdHu03yzHuTNPQtyVQ4yC/7b4pjAnfFNgfLXgM09KVgTVCqH0P6rGoGYC0aQNQh05rHwnUIrzqdmgSk8bMy6+lWLcx1oNPjxcNbIvJeVXUn6p7WBbaJWiT6Um92HqXIpNGk8hHGH4/P1xuR2f5W7bWo5pPVIeA2ng3cg4nCe/UHXgjNKmvNfEO8QS0dC70B+F8h1u5G6z8wFvbuqZcLDf/uXbTbGefvBPhqf43Ai2XFYGUR5FlQ9jEZsxHntQUHyOiqHvWRYRXQW85aoDIyRHNlRZ0uaWqeIIcQhs/GZKuuMoHUHn4xPt7QfRtVrPlS0c2yWcWtJhvb/74SwHjK2F+Zd4/BKpFuNv4zqXgRKRAoaCPm3YmN71KzjGEPZA4/hJuDhVNnc3AJL+m2fmF/VWtDC1yCmsl3CBKI7mHG5tgcbCbPE1sKpbH9gSySJJ8Ca3KNzPbcD2pJrF3ieWf5Wex0d0DWKrlEtpSvlQ7SVjsrh0QBvHKbGHw5gOjbkL+Ibc3WTRUgKUWBQgRm6IW29wfdRyz96WhWNGLh5tNugJ5e7CpoLnm5l9zGjDT5ORlCJ8rtE5b6DRPa8IQC9GX6e8KnXxmS75f8NKgEbXjUuwwn1/EIxB6TGqoOCSIdRs4jAJfa5uz7oQ3TX+ZMybnITsWknDYo6UYEC21Gyt4ZGOshq3b+q9LTsixzEC2K8vTCqjAiFq8BHRrJ7tnCRbZGQTnENrl6Nxd6R+A5T+bzfOBt6bW4u0kfMZlO0cBwnWsfbcUYoAewE6EW09CbbnIPoe1rzcTgaoK7uhixDA28K6z8m31vd99oK4aYWAJ6qjgeZZaqtwOVSRm7CG7+oXHZP8B8/QpT1BN7WQFO+ngSYKuoJAHq6yH3WItC+qSzHlQdA12XX6OVCN6eaZe27DZ6AFnvfqOybn9jO+fXqTh+JPevO6KFLlvBVWq3H7ldFg5+lWJzNB1fQ+ADexWpPwyZ1dvVGmBrx2lOmWrO9rGFyVaKWXMX3kdoXklt6nOmtPjLP6FN2I9ydNdLBEg7nC5TPSLLRzCRLqVaG/LsXBm+3V9QWFDBjhnPBvmFJGmeRJzPRQnXHTOMRlMU8uT8EwJPHVvBg6Yuhx2Pc+XNETFFzCG/hLuSdfwA4tyJIiG4B1Ygq7uCzCmhKaCfeX0O9H6i75C0b1jaz3TT6bjxSz4JSqxSS0sAf0QbRCPRsD7AyxpBsJ6e+XIbY4d/w8VLA4ZtU6NDUMRgJOrQOEEb1VmUYvLk02DVZzXKuHHuRqXRRMTtJKFv9TF8QgiuaSzKB/gY7BVtfJ+bnX9rY5NQmuMmfG+uWFuuqH4uuW33SuOI+4JZLmmJQgPqLfVg1F5GMoVH9kZ5jJXZp8m8mfVyoDr4bGvA5kdoMRmnA4QtvVp7Md6qIPbXpAeYOa5Ky7l1R8NYZq1+jlh2Tz13Z4vZ4Z+HVpo8OIV7QgbkudL9rOzAgaC8+hWwS/LbkFKUy+K3AjWrsUjNyJH9Xo5qH7nwwevPPmvAB6X9gTQw/xYS1BQJl3lJb7HqPdU1fVtCKdJRGk0lyFYKVcM9O58FjPdnmGxCQa8gcVI6Z6VBXg9IxUUSSuhj4C+bOyCNyI2sX9IM1kBmZMHypBQJhKUpJ1O9GLd0TkOBgBbsNFHrSsXoT2yV30t6Fj31P0uJtNH8DRZP0vqkvZxfbx9FgCw+GbX+ypUD8/qNyeDmrNeQL6YGm54JGd6WNdBLK0kvRlRKEzC8/pmPIIsXTtHFt2A1hRNhtvlGxB3Jn5OnJkFqzBBiAXglBVUmVqQ6xYcM6TgYhkMYv2LVSBbknx6BfYLkWglkGa9zmEu1H+YUFp7tJ1V3xT0P6bSOQKH0gXpq9oJl2pwc7H5LhA1y+hW7bL8RU7ZQdr3pwRMXrH1hsa5WYyK/dbtd11Pi9wYDn7s5bIQc8zbMhiS2vtJo3DTHWD1IErYf5iWPZo7n14C5hNMIxK77CJbg6+7LzSnRgjn0eJfUfiPLE88cr8ncXpzlKn+W31H85cNu+0XvmKfqBDvr8FfchIwHX5mpsfXojuFFSUjv5emoFY4FNuSHZdVDmcHBzLpOAbEOuJ6K1pDnD7tKUwSRkDRMK0ypPmzgFkcL0etSsw29Y1KfGCZSFkh8lEa6hUl2TNMS0NbuFmwE06G3GJjcAW6EOMQqbs8Xa12cpqt9Hm+5jXNKNFPc3s8klu40aeT5D3cJ4pt6Vn9edp9gsmsx5f6NYYb8kZcOSAn33nEJgdqaZRTbOq5Rk4+F8XKXkRbjR5XV7RGdGmbyRh+Zf9WS3TRjrDXsD1UW4yHIcO1Ddf3dl1VKwkPQsRzfxg8+gFTvF35jXuw9wEPb4vRVU0LjEnM6QL2c7L2nrH9vDHNN3ZnUbE4Pa9v0yqa6AOhrxLrZb0eBZA43C7qkb5Ho3J+oV3bsfUxeV2vS6QTrjWGH7E6YHDsnMTCFTmVjrlm2gGg0pgfPlp6tkV10Ozng2p1FyW+mhhD1ffmJyfI2I0drMYpFn4VCT+qaVjfnYKJt2GNDT4A1Ci17Y+VhNdP1J7VNGEeAg6IXbNf3P9VLlYpnXS0VPVU5TX0Msd8ghVCAc2c+KjWprqXNOt9cnwFG5kqw4XgoQR05QtbSjdk2f1RYNqjK9/5tlMGZp3oePjnT8jtBwK2xbwFT5VshJy+25rt69jcqJYaNazLUMHbEkj86U6ajDOcy02VPJoTKJuV6RlJy6EprXhfyIAUmW07PqhoO1yC+iJnkYOxWSw8X/Z6mtrcJ1TiwuJhNoU9pF+EkbgTKY/xfaeJtIEOrq6ldCyUC18Ad27PEQsA+/B1X+kSJS9Lp3pPq3pFGJNiMiFh1S7ZWc6Rp8ijd7V5iUIj+YqqnUzTyBdPHvxNQGjSjlfUMB5ELF64uR/NPT8gwBPkzsNGoO+RYBw2k98Rd8Rf+GUPfuiBaXVrEXfm3pxuAYEndkhCKAkYBfkFxtY8mojrKFVQFYK+u/0/OQOr3smj7qGh93yfGbC9vC8qBLZG6z+hh3/DVg47c+0AiI6+v26A9CJgOTkyjvrlN0CSsJJWH6aWzoN1IkCGwVhG5bKb5xBtoAgvzlm+nTwZ20mLw+kqXvMGP27j1HzTxd8SGpjNbK+kqyrsRvzBgYRMoZDaiXlUEX4ERMz7uhwdvR0pOv82cXiL7swqrH1Z8B7HaYOLzB08A7XgdFWw3zhn2c4QnFKaLJ1NwuRtsnbLPZYgTLL7LFE+K49I5hR0tpj7mSpjfnHRHrxDn9VYUs4bCZ0g5pfvz9gJLeaHyLVZe8bBYwPtH3o3HkGXP3fEQ2dF+oJdUQGT6KCCkC5Jh6wwEfl5KkxqJ+0A2Zw1zFhbAGFtDQAbQpbH0eBmATiYXuZyaq9jfGQbcD8PH9qGtb5vQHkg5EeAeV6VbsbLj9aRqGtwLeYaOFlV6pJXFD0eNQ1FwupRhS6jBPcNzakBgTVnb9rNHFJDJSpnEWh0M8RMgbs5XUtqL0Jmx8qM2Kar6FT/0qg2fNwEJ4aRk9Az6ugzHj/fJVeP3tlqkJ1nDiz4/uRt1ZTtyl3NvF4JTZqqJB3dJQg66w7Pk+Sa3JcJOUYpX8ju1DASuQmRJsY/d8XWLAwXzd7Gj6dhTg9GK4EpHnJg3Nt7LlVVNGQxkJ/1ecK5dMVglB1KvZBcgiJdKDqGnILxDiHRrSovYzWgC/KrhNqhbh1b0FjKQGeQEsafYoBkhEuGfizfkz0lzGchXq6xDQHA7x+LbvfLhUlqTunljWs32OGhmRy16IHTQIsmPNpQRjX1pRP+qaGJlVcOcpjhUW6lscyDGlLiqVRzaTMGh7fHjH9cKnnZ31WxdHX3aDQX0i8A6wW/YGmGYlgnnXJDWrtM1z4o8Y3eBXUrNEiaWTv1VJ8nK+6RuaQMSxB9u1aTLjyA/P4MZBpkQiJm0KZ2+ncwAcCgKvlPCPukEZ0V6a5BncmWB8nLw+fx7sgeMjEHKzBnrLBEdnW5H3eF9QPmfE5vWtVHTRuxj4xZJGpdTOUvqFvbrQOxYrD/8W5XiHD0DtWIa9g6vvwNeYTjhgTBBEe/q9KVv1XUd7/ch5IN6bAyMVTlmbohUiZxgo4CMTPT+lWOBtv2cGtOsQDfK8Xwpd7YgzLzk6Zcv9sTvcDBPBX+AJXDUA4SMu8yaqh8P3SUmwN2Z+9KDq/NeHzwkJgNTWmJWQ7pXdqiB/jCvyt1JvL47KXG7OSEtldXxkOKjx7IAPvDkQBcy89shK9DLcEGebsgdgrtxU/K8WI2hnXzH2YJ21vuEPr63u1bIS5uQTVOa1vwbcK9yxMjW27YYzAJHIOsVWVBDF+UK4Ej3PZz6R5AJGC9+Z2uszE0OPiJzDi3JCpxM1iLMgfabY0WBD2LN+Zn2qDyVXdSrX/3WcgTgvsbN3H6gCb5D7u11tuyYHWobR4SI0s3bl0ziqyi0sQk0nf5WdYrQzWtqmrA8hgHj8+5RLGAMGQOKhgAZVp1Tl8hHT3YNGVEWYaBDytE5xIGlS4fu9zvqBRLplr3XD0zOKFWqKzTKNBvTFIGR0brqaUOFVS+/XsCIePMyUHbPyefZnO8KqbOiOapVZEgOdrAwj8M4ixIovSoVQNahXRFHHSg2e4Wfdqi8g5vxJhkf30p1bJqmoYv/r1Jd3b1lT+h4tLn63mWtAbXkRbd6hU0FXQswhmQgBmqswWiIEIptsz4AbRQmc9uy60lZKukvArQYnQUjRBtwNyd8NCe8MBT9Ajx5Q9ZUXSbZ4MnUfxYzfXnIABdiMs6XrvQB5BBNyflW42Yr9wlbrfgASVhgapaWzoTMdtDKgWa6bRGkHVBkuZzcUIAB84zh9jwPPF4T08oC3pt8i7gm55D7sIUWG6FIkpFh1ljqJZ77RaBvWj7VkGoJ6qsTAFUx6Ayc9Q3nJZ5Qrr2d0vcgIl7+GAKqtVXydE7qpYKQ89/c7zu+zbEbZMTwjYuOCVFc0EuHMEtQEcbtALQ8zZPoCRKdPWSBAV2xiOn5Aj8Jyk8kaVvRxb2tLPvUwWhbTQ7hdhU3nwO3nDuIyGTnWqIBFRIVthKZMHLSeSthjYkMZSEeeNKvSv7cCkxgROhmMyX5Ne7Tu3JYLxMS9cgDjvYkM0RFzOxU8y15NH/SvRbTQVy/Hjn1WS0I4nXK6jV0f+4W9Is5tRD2GWJOWv8qCBPDrGmSGzeq9rgRtFFzMjsMd+gz4bAHR8c0ciNxp4c4ETv3fDXeOr0lQxw1Mg+k7sUnMUyRJc6JzLtmhvW7RGQ9Bl8oUBujoKJnj1fomPSaO/0jPApsTGYUOIKNAa4BOJuMcGG7tb0lkL+1MJn0mV+NmZpxqR1dvxSlo6CGadWy23dRjLWZhbnOXLHjYPkoKsvsG0MY9ibzZpjMnqb8E3SUR7Xgz0C6kNrzEhm7/Wf+YoF7mnI8BhtDBZwVAehszJv6LNUEGRLdRIWnWHYp1kz4aP4DpMWgHAP2Ym4ysQWpKjNUrxx1ruJqBIlq5rBn5eqWOdwDO9rJm1swtjHmZRrTt4hfu6jBf7AGTCjBiIBlPqPqNYZV35/K2n4Dmjev0FLs6UVpJQZyt6ZGMsSP+/1t0E8mcjv4oDm9CRHLHTBU4rfnB5rdcHUTqq4pkHd3Phh/dZ5v6wRlQ5x9dJR+vPsa7bqxlX1dlg5Zy9o6ymAGdTr8oyPKJbjZqy9x7WrmL0oiVnl9e/NvygSQ7Ruxd3XADCaKHdemQHHbg2dnz/NcHrEG5D9j+Ll+zR0s4qNHd8ZnlFWjsWp7OaxsHuvcnpq4zCcfqcmUpPJ9gzgHx9EAFNZUNXvDNRNJbNvKbU9JomCuRvndb7sHm5JOhu2JJ9DPFYsYz6PNvgwB/vkZgfgUuzqGOYQvI7Eg8GmB23Fwl5jB9O3JmTfytgzVmavL6VBrsW9G3twiwmiBJuCsVZbNEnPLuVRwsm0s03WP4uL07YcrZUtxcrx4F3pVfNlQ3ifxFQhCAUjkrbqPQvRsEywM/gqbYamHLV8J4Ot61PiDfyT3nDdrPe8IUdf2rh4SdfbO31ckwoaS9IoCUKmJPTwnQ7fMbE8HIvg+1rrPpmGkHpo0A5EZCCdwQeQLJtAnkZ0JsGiTG9FQJTHLjm5SgAxBjjTeBX5DzSDo8t9ERl0JINpPpQmBfUS1Y9lC9zTgNwI91GRYSD4qsf/6kzqGmYxLjdxZ1bQ3ZagmvZhxzArsrttjNAbzFZGZ2LnuW6WxDE1LU/4mlL6YcW4IsFNiFBjuqtfmqovOaSkw+Gli0PZiUjWcQf6dsOAYvHmT1ZlPzwpnfSeSJ8xUgdE7VD7BfQwGwjDMAsDcwoC6A9J2V6I4tPz4b8ENO6rXL3LqTYJzCbAqEN9YjFKnn6ZdXwvyu4LBJRKRW6AlYgD+UW5T8n3xkHNgDXR3Lv3YPw4WEvtSmR7ORW/PbdyW5Fp3XJFeRcUwtxh99p2fb7yKDcH6UAV0p1oPFF4HMFZp4xYBeMltUs7Ex6YY4fnbQWyNXgSvNR8tAMFFuZ1qT1LfeM5lH/j8Mu06iv1IYjDjj5dhjT6r7QsOnX6NzA0nffkr/HmSGZJa1iCkjpRhSWWX+WcjwgShO56Y0O9dMmwdpLkZ9Tg7cbb4mPUlhftaTsimZNlrY4zAVOHdFBw3PInbvKmpdltnJ4SFHVqr3hMDdLaV8WGKhNehe+Eov4E0ZTdvthiUZoMoIrgxNz3E7Nm+6qa6ffmlMk6LQXPW/DWgE53n+wLLmGxjsuyNe4UEJNr6qHq8pQKdeUxjyCA5hIdmF0kIaLd/gvdTnlO6wHEZ4eOsIgFkdItZeBQSDA5O5fNi3XRbQJv5g9HW0Z+ItJTbtt6QzGPuL6HhqTMXoo2SzDF8DQVhG8/gVLP3ENzwwG6J1HmXRtYRZbkRp4cmzGUd6N7VtW5Tf84DxVxhEkc7G3gcqsvtr6y6a6t7dNamcQyNTYXJqaCHbmbE+WjcE+QHhzj0K1hO+GMiYLTUEKJbxBrhVtHvZXo5Kig73ohPlfYjqPZN6eIRwhcHhp4xVnhHP5gHf08JYJkLvvkqe2sCD5rE7D0YFkfMtU+kLRRPA7Cs0Gdyq0Cyn5qcDX9FFBrCCMwPK/C1E6y3dbzsS48w9KBgU8cBQm7o7Qu5Rl7DSXvC72oGPzyZIjBkmEzBvz7ZGOEFz/n8mjH5/J4Or/7W2LpVPv/fkgPcXpZmrVqeDFCp8lXHl2uepvdt6VRArWbz5sWaLoRs5yv6x9kKFXfT5wMXkc8FFs9MHfXaMwHT6n/SpfQKJJ4pKe/nKIaA9bN4RM7RHLrfOI4bAhsLFSUH/4CXy6xUW5F4zdRk6pQVJ68sW/8uHXcpeqEkHG6QaExc98oOhhvTugX2g4nvO4oS1T3b8kSSu7BkTo+M/VD9qb7qtRJeMoAzGzn3/ZT0VeLYwubjSHkmhDfPzOjmAwsuLnhPyRC8cowpuVOQ6M+NvACGVhMXfVDcX9g9h82N8BAsbbZt3Sm9QKAf9iqlz53dnqEArCo+Tv4oz33yJBhkh13ivHqHmw1PtpKPvZ9nh8lTiB2l5BDbqT1Zq+T19oNtNx6B4ySm8Ne5z/BjuDfmxpYKyCXPGYSoqb3CzvXsy3+8mRdtr1VJttzQxBhpibMRrDnoMd8exeKPP8pELgu+p1mFuUyBr/D0x3jkN3gHRfqJOdWc/+nfG/uPZR6x72KvvmZXFwzMCtFCYxf7UnIjsN0IXQ+4YGXqbZCgzbr/7OFTe1k1ZRIKCoNgMs7yNKHOlmlEG+aw5P+vyvKvs+s6qwuC5iNXC5VeU8A/yEMv4kYjkG7XnrUJl2Zo0qXE0aGtH5PKvSNq8ZT5EwpUoaFDJxhsIHXCcb3zv3e8eNPH1X9vMMsO/tqMn3UE5KYI2Jgr4+nYBuvj3aOHtAm4zGtFY/YZpc1kMtfQ4WMIRVVeaEZTLyaHcRkWSeeoBrX28mnRmPnqWYkPDfCLVoQ+gL7ylWFh3+KB3+nyE4nQHP0xTSNtbC0h1Z3CbwNqWUd7F5NMHT/Dhb9tpmuekXCD+Eddr89vAhjTY4Xb6lv2bKiFPTG0TFhCeEMkZJW0YuqT9aJB1cxFJDC2txvVLbpjtPJxt1LnTGPHw9VuGrpmyxgmWpbzrDvoWb33D1VAXY8kGzF5jQGvPgdIkeh/NA9gh7c1pPZiAkkKkHDy1Bm8gWdJDWd3ouw4KaS8YIW5F3Jd7gVwfgnOv1IfYzCp9ab2iVd9gSg8lqRcCfDOyfr0BRy+jaz5jHgZ7l/y8a8TSCmvwZP5KC/VmnWDsaPmDwQvtcw+N4RdvBpzAY/rIJR0XKe5d6ZgMxapG4/2eaudUj2VE4s2tmO3J/f4Hb3UPNHWN6e0G1XF0qrU5RjX8Gnn8c4dwPXsnMPgaAdM6YBDv3hTsyCSBaYq1RXS12Es/60tykIIaRqrvZbUcAjTxAm1ZJ9N+tYv81dklSlHplEzcxgdKuh0H3zffPgvWHwCsOJEEfY1Mn57tWhskG1J3onzkrRZhgJtiz8QmaZp8PXgPSDJWXMNBe3p0CUK3yUwp8e0QpPaMkCtOmGuNMmhjHHaw3KCcpHT1Gk0uQxv61yEGqwhP3qJ1r3cv9rZ3jGeK5Oym2dm3e4IZHvqEt6YUGiDRAGyeiGMY6W92jH21Z9pj0wzEHufDAoxZGpdRYJEDq3gSWCTaPf6rUdqtBfUOvtUaGBiFsLazsSa0x1WzGL0h2pNn6XsEi7IiIN5o7RFj8fYQlrbBN5C0eCHZK5AeLzYVqOG9C7YCbymJeaImKiW9yKS6uiwmOdyL92Hqa2NWVDu7zzg+PM1y1QR8Qi2MA9I0NwFz11WGeTT3h4AdvbOtXY0tsPHKUfbmhCNF8t68ti2Z9HAF0anGuMjaiq6yW18PMjAL8CgmGTeqdhxPaVDOzxQNWz3u6KZjxcIW3aIf6M9QUJj7lTd4dpQb7vk8umWg84IAZw2ojxHeEa3Bt8D4UEy+hKNQ4Ilcv5EvGO5CjL2lkMnIXFhkmIXPKVLmrd3h43AU3rQBqmDYfeIUYM7LgEYzhN4rWMPKxMHqjlbV3BBDsWkhkFVmr8Xi6Kr1I/aV+IjvLMDlcsmh+jC6H4PKZNfTazVH0MkOhVh7bS6Yq5qkmeMS8C5k84SK1bS9vIazHfpyjT/QPbHzrA+qWbsUp4CTK6EJgWU2qBXldtjtWbMy2JNNLpJRLzk5Q/P8XRkR0lnmLmvnnGTtRwKkoTWfgCLHdlJp8D2BYC44F0LkMtR3Cq7+Pwdm5lTAUi5BLp5tz6o0RJc31WgZrHvHH9Xyu1SlsHi7n6ypK/U4XcHEz8bxLIJVpJRIOyzvT58zKoxWDoifLqqUvBRoY9J3vwWJF3Jl3mNb+CWiW7ajA/yuOORe8CfPz+AAO1bY93ECaBnHv5eOpNcwR/9brKMM2O4IXthhxDhr/mj5grWcQIWCWDk5OL4VNjkc/XXvMm4MPvXfkNrh8zPP73Ku8+m8aImlG+v8u2R/RFXAVMGw0n/hSXd6J3P0mRSWDnsHUKPPvVTiEjiXXktgHnJW2fwTJ7QfPNShuQB6bmK26JEx6WzKwe7tb+QFUdSIqH4wvXGBQ4/88Fc2ATKKe3KMz8hS8e6O09dg7A0s3p/nE6nPQt1HwJXTGbaHgmyy6e5MtyhgnN2VXFYJMgpc2x/OcxC9XcmEMQaVrf3/n8kC9x6aXMYsxBJyoxxhOUTyB6snjmiRPW7OCMPqWVutYsdKFqQRZ0jAK+AWrfuVc0PCYxZsRzxE0iKpN1S9rhY6BZKuPUQ1VFhDqcBcjtJ9rHKjMIGVGXSw6klrsT6B3NbnhnsHDp3B6VR8bsU62hHkXl6qtVd/WbJz5tB9rpdNfADjZ6eIRUyvATpUsgmn6P5SVNpeA/J5oHMwvv/OYLp3Bkhbk/ByMR2VCnDlbhPBGcGF1wiCWSXm4GUr5VA1B2ujq+Z8g1T6GcuGMYF2GaR9Oxv+d5jk26Lbql3QUpkcJ0SPmbAqUSNFwUu32W0whEgytCpBtv2d/ayBS/gQxe3zAt4kuWgXr7srdKqQbnBBOMUcQFnYJFKJFqIsGUmfvD+qFt6K1n2N1knMpZs/kOmXEWGaCMfZ8ByAqS6Fr6EoxxPkT6+A24M5ofuWl4jaCUkM9ML/ALFnt0NdxbFU/DTF7ShuEzIYaC7gj+d36DxhcOd3ZYUgbxUkXvbzn2e3EgxvgpM2Xcyf7bVcZpsHpUuzzRDbS/wEcx1pZ+WEzRhUmOy1utnm5P6MdOnlFlJpCFCgBBCfrVCqGjwsHq7H6uvJsPGfDvvO2hZtkF9ylXsEI+pFfOcH/a5Dh4ezLAixL9OkJnXAAGM0jOzc7uaZR5p4eCtcbZl+13XbAVRq4YudGGsjSMg4rm38dZYcAJ6vlnhmmZ6L4ltRj8JVLGwd0/xuBkthxt2PEE5ojWLlMvf4WcTEQd3dBGMibH1KWmXJmqebMD+7xsAJ6LfkRAbn6LPSt3L3ggRbP5XYyd82Gw+ssbgf+zBr4hWjr62mxgWbXvtvl56t5cVtMRDDA/w2e+0KZF2cILxL0qWeDeQOSpXH/P2ixsmy61ag3trh9/jqxFJFLCN1RHj2B8DH2+X+h9O6Y/WhX3kYmkFTeDyKBlI7u3C5qYZd/8ubCdl75tfP5dcfYIoaIbCm8oGqgJqbRl60HFUmLNP5AtSwLMkFGbFZ/MQmZD5jGYiTB1hLaHz28JWJOa6dJBSJIa4X+ecj3ouwrrUBDgr+dn2V3vQEP/Tr7fbmNB4xnX4S8J7Akb/HxR8ZWtmXt91eJGS2fSXrXQpevTXmhuTK13nAv2v9tFHOzs2HQcPWWIaPYt8t0zjIGPUYs4emZu9gJ8RMxgo1tfGDt2kKzsaaobJCH+mBMUgmtcFCaR/fjRyfxANjFaVX6SdXdZdl2Q5wZTnM6dmo3xPvI9OcxFIIq8JpJZAcP3VM7Hw5e4lgNA4n1DtWeTu3R9PpZvxb3UzYvIr5kk/8ljYqJlUfFbCtP12IvZ9SMdCcLBHVBNubYeT6B2RxdMP5Vh6Z0GFEfJ5NbfyqThFxj7UZRL4RnA+LWpwF+62n3HCPHYvaakUQfgmVmzabK+2SJMpVdHXKkJxjmh7mj9xrvApvRSaO3QEaqpuFX27jUbcgSdCaYfjr5YoK4vHzUCGHg4Vl9IazjqLGeofnfuk0wf0Wdnj3aHlAXqDEm1Pv8EUXmT72TbBxU/dPSIBDtyPvIwSW4Ab2Q3ti9oy5rrzUDW6MNA5NhjvbBjIMk3fL6mqzqv3YdcGNNobYIafE1sTQOT9VDYZtkRZ3wdG91IZr6M4ckKwqMaZwJPBrn6FtnWM8+sRv8vRVS5oLMAjkeZpu3jBFwRwbhsmlJyvFgrPfSYbJO6LqYXf3bmJ+bGGOafRMgzBYrhIcp0Bnm0V/dDVqQ53KMlVAlQu3V/Vq+lbWY/Oxj6kmx0xXdNtR/pKMi3fW2v2+qlxNqF5N8Q+yrk6Pbz0IvuIOr7cUjWv73X4QM+ChuS1jAAAAAA

```
odDfTZZva0TExt1LwAYy4qZqTS8bxLYDy8voj8W+LJtHylBmw5MOTkZd/q/RuebBjz15OSu4ItHXR4a85G
Ss2+c/BTp9bi1GW1Kb7x2T800+iMfK3jJaI+WyvTaXFpq7Y4657ZntB81ekpqa/i6rR2S8/1VlidoyRyXrgPM
n0Tth2i+95ntlZo8E6fTxjtMTMT8HcBw1Olx6mu142mOyYQW9FZaz+DJGz1gHnYPRVa25WW3K/J6MRt
G0ACLV+jqZ7zek8m09v5pq+issztbJHJesA8zL6KnevNWiIjt3+L04jaIgARar0bjzWm1J5FpTeqs2+3OxyXrA
JNJoKaaeVM8q/zVgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAADE2tNpitYnb5zt/h93yd2v8Ad9int3aBnfJ3a/3fY3yd2v8Ad9mgGd8ndr/d9jfJ3a/3fZoBnfJ3a/3f
Y3yd2v8Ad9mgGd8ndr/d9jfJ3a/3fZoBibZIiZ5Ner/d9nHpc9z/AJKL+xP0efA4rmYj8UdLnuR/cdLnuR/cnE
Z9kqOlz3I/uOlz3I/uTgdkqOlz3I/uOlz3I/uTgdkqOlz3I/uOlz3I/uTgdkqOlz3I/ubxZ+cttydv6pHbS+9n6Dq
muZmysBWoAAAAAAAD5e01rvEbs75O7X+77PuX2f6w0DO+Tu1/u+xvk7tf7vs0Azvk7tf7vsb5O7X+77N
AM75O7X+77G+Tu1/u+zQDO+Tu1/u+xvk7tf7vs0Azvk7tf7vs5X1FqW5M0jf8rfZ3R6n30/Qc1TaLt9Lnw/
8AkdLnw/8AknEZdkqOlz4f/I6XPh/8k4HZKjpc+H/yOlz4f/JOB2So6XPh/wDI6XPh/wDJOB2So6XPh/8AI6
VO/sf8k5HbH1CK5eiEdgrcAAAAAAAAYi17dcVrt+dvs2zi9j+oG+Tu1/u+xvk7tf7vs0Azvk7tf7vsb5O7X+77
NAM75O7X+77G+Tu1/u+zQDO+Tu1/u+xvk7tf7vs0Azvk7tf7vsze96Vm00rtHyt9nRz1HubA5dLnw4/uOl
z4cf3fZP8AARh2So6XPhx/d9jpc+HH932TgdkqOlz4cf3fY6XPhx/d9k4HZKjpc+HH932Olz4cf3fZOB2So6X
Phx/d9jpc+HH932TgdkrMOactpjk7bR83VLpPbt9FStaZvAAOgAAAAABm9prEbRvMzs0xk7K/qgH3fJ3a/
wB32N8ndr/d9mgGd8ndr/d9jfJ3a/3fZoBnfJ3a/wB32N8ndr/d9mgGd8ndr/d9jfJ3a/3fZoBnfJ3a/wB32
N8ndr/d9mgHC+pmluTNI3j/AHfZnpc9yP7vs56j31nNGVVcxNlHS57kf3HS57kf3Jwc9kqOlz3P+X2Olz4f/
L7JwOyVHS58OP7vsdLnuf8AL7JwOyVHS57n/L7HS57n/L7JwOyVHS57n7qYneIn5vOehT2I+itKKpnb6A
OwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAGae3dpmnt3aAAAAAAAAB8v7E/R58PQv7E/R58DPk0AIxAAAAAAAHbS+9n6OL
tpfez9B1R/SsBXoAAAAAAAAiYnskBnL7P9YaZy+zH1hoAAAAAAAAABHqffT9FiPU++n6DmvTkAjzgAAAAAB
HbAR2wEbejHYEdgr1AAAAAAAAbxvtv1gDOL2P6tM4vY/qDQAAAAAAADnqPc2dHPUe5sCIBHlAAAAAAA
AAd9J7dvoqS6T27fRUr0Ua0UaAB0AAAAAARMT2SAzk7K/qhpnJ2V/VANAAAAAAAAAAi1HvrOjnqPc2BEAjz
9AA5AAAAAAAAAHoU9iPo896FPYj6K1430AagAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAOefLOGvL5O9Y7dvgDoM48tMteVS0TDQ
M09u7TNPbu0AACfNrceHJNLVtMx8x8oZx+kMWTlcjeep11PLjacVYtx+f4JPR9oxZLYMmOeas7TG8buvVeB
ufSeGO2Lx/wDlXiyRlxxeuvu+0/NBr65bZYnzmuVhp1zt8VunyprYUy4a3pG0zvEw5Ax+HyB0AB8v7E/R58PQv7E/R58D
Pk0AIxbriteN6x1FsV6xvfMdTTth36Pbk9u5SFMM9Hnm9/9Xyc64a3232jsnZ23t0jsnbe0JnfcxzNc
F5+O6raHC1ZpO09rL7Mzad5neXXxGY7aX3s/RxIy5MV4nHnjJMx2QOqP6emanytjpmp8rYYXwh6emanytjpmp8rYFwh6ZqfK2OmanytgZ1dsujy87j68dvarPz6U6bV4tRny45
pbSTtLyYm1L71mazAzKrv2TmEg0gAAYzZacryOdpPGxC0wXo7WbBAdhW3YnCdtaoYsxbDesuOrrn5ptFWcmoyank
mluqsfMFwACPU++n6LEep99P0RK9FGtFGgAdAAAAAAAAGsvmepSQbAAAAAAAAACLUe+s6OeK9zYEQCPP9UA
3a+3w2rbaOyZ2h3tO2anzmOtynIdIjffbfbDqaYsxbDesMM1Q5u+pvXAjZ9l5H6AxFb9p+eHbXp2S0zmUpsT
Rt6Udgg6bqfK2K2fem6lMMQSVzMEzNXW8LMXM3nQSrS9JrxpIXmuBAGeVcUzNHuuI47iebdepNqzvNkS
Dv8FMbO9+p9cHB9qvQvCgc1y/ul21T09TN28D0wrtjdLhoMWOeyhkk8xWuRjh6oaYXmr6r0hDUD4SLOHn9vsYp
vau8R1eM6MhR7LTxa5hLsVm4fHmkxkfQ0dRHpOpV8FZVo8xnaH1RdpkTH4urkCcXW6VCRNZtI0Ep/mo1
vau8R1MKcfKjTTye0WmLuNsV6RvMdTpOnnkRMe01hm00tzm+35lptOnrMHKuG9o3iGLVmmOGOeO6IBG
Xyv7SKZwhtsDGbyaotfKC6zrKnbq4BXvUqjQ2TjW5zuMnHzrHkEnb1BQd4W6DNWhAtXFALqzIcfkI6WEdAxG31PT1t8xJzBfXXBhlpAqAWRlmF2Tm
BQoZVUW5A0DCDHB0uTUrN1tmyYzEyO6QaAi+JSJFFtZ3FBgHx7ESE2MOmC3yBupeWIsXvJnzDv1sbmCSmxVpS
DCbK9FeJ1o5rxTuvORRVcFY1C7SU4vdCDoPqePSG6w91KLg4sVmGatPNg9H7lZsNEeUlZIM09VRB3O4lfH0ZKSxRxfaRSZkbsXtEKZUhbyOfzCPX
tXWedsazqHtqEYu59xpXzLnrHRKstEaZdYMKCaqHTbj2kVUWtC4AW1rZ9L1TA2Yn6bEaIE8nA4qLQHhwBxzvI0fvxBXFDe0FcoBtqXwIIVIPDBe8xCb
Srg7OXngmKTdkvrTkeSH1TQHqlz9cQxn5lqdDG9l8tmkC0jWKbtQSXbpbDDk0d4p1j4ylC29CGlxVXodJBXxPBnU98FN0TrVSl5AJBFvT7F8hwFVDoHd
N2Sc4Fwq2Jw7xtrSLsiGmiuQ2EsYEGBpFN+CX4VPJjZBrc9Ii5UMnCKPTQjy4RfAJJNa6AkHVWNGG3ZrMgF3d9
cjLRb4grKHAb8aYoKAv4mf9jXiylIpaCDCuJbdCx7oOyXSTTJW9vKrDtUK8XstSmpqhG9gPm5HRcOEBwjHZj
MXkzgq7AHyoeMDp6Gs0NvqGscEUoEYLcrQpMWu9zRjHLWb5YTvZBiZ2JrhgHbrMXy7GbBN0JUA4ehkYkErXRSgW
```

HSZjpFY369usaYxdxvgmLxETHX2Pl8FqVm0zHU3Wto1ETMdW8saiZnLaN+oSYi13IAcD0KexH0ee9CnsR
9Fa8b6ANQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAABz6Rh8Wn90N39i30eZ6P0mHPp+XkrvblTHaD0OkYfFp/dB0jD4tP7ocfV2m8P9z1dpvD
/AHB26Rh8Wn90HSMPi0/uhx9Xabw/3PV2m8P9wdukYfFp/dB0jD4tP7ocfV2m8P8Ac9Xabw/3B26Rh8
Wn90Pls+C0TE5aTE/7ocvV2m8P9z1dpvD/AHB5mW9tHqZnBkiaz1xtO8PR0npDHn2rf8F/2l5/pGmDFe
MeGu1o7Z3Z0uhy6iYn2afOQe3T27tuWCnNxNN5ttt1y6gAAlzU1cZZvhyVms/6bfBzrpM98ls2XJWMnJ2
ryfguAQ2x6+1ZxzfHyZ6uV8VWmwxgw1xxO+3xdAAAHy/sT9Hnw9C/sT9HnwM+TQAjF3xXimC20xFt+xi
+a9o26oj8nMFym1nSmW1I2js/MnLaa2jq6+1zAvIAIO2l97P0cXbS+9n6Dqj+lYCvQAAAAAAj9I6nmsfN0
9u/y+CTSejLZNr59617vxl6NNNSMs5LfivPxn4OwOU46YscVpWKxEx2OrOX2Y+rQAAOOrrmvhmuCYi0/
GfkjwafW6es1pGLrneZntl6QDzdRptZnyVm3NzWv+nfqUaeNXF4jLGOMf8AtVAAACPU++n6LEep99P0H
NenIBHnaxzEZKzPV1u2TPat5isxMJwWKpiGuXab8rfrbnPeduzqcgLy1e03tvPayAgR2wEdsBG3ox2AK9Q
AAAAADnnzVwYrXt2R+7x8emz6/NOS3VWZ67T/AIevl09c14nJ11r2VdYiIjaI2gHDT6TFp67Ujr+Np7XXF7
H9WmcXsf1BoAByzaemaazeJnkzvEbuoBMRMbfBLb0dp7W5XImPyiepUA+VrFKxWsbRHZD6ADnqPc2
dHPUe5sCIBHlFFMnIwdUxyt+xOCxNnS+a942mdo/IpmtSNo22/NzAvLc5bTSazttLACTNwAR30nt2+ipLp
Pbt9FSvTRoAHQAAAAj9I6nmcXIp7y/VH5LHGumpGWctvxn4z8AebpPRlskxfPvWvy+MvT5umKla0rER
Fo7HVnJ2V/VANAAPlq1vXKX0BMFbcquKuMs9T9H3JpcOS2S2Tn0kXZmzdbn3pNUtTLACAIs/vrIzDlz
4NdPyV6ui5J26t/8A2GXoU9iPorXjQxQ+sMnlMn/v8AQ9YZPKZP/0XgIPWGTymT/3+h6wyeUyf+/0XgIP
WGTymT/3+h6wyeUyf+/0XgIPWGTymT/3+h6wyeUyf+/0XgIPWGTymT/3+hPpG9YmbaXJER2zP/8Axe
46z+Dy/pkHTHkrlxxek71nsaT+j/4LF9FAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADGTJyLVjb
flTs58vJlvMY5itY+MsZOcjJjjJtP4uqYBUOWXJaLxSm3Kn4z8GZtlxbWtaL1+P5A7jllyzE1rSN7W7Gbc/jrNp
tFo+MbA7jjGS3RuXv+LbtdMczbHWZ7ZgGhyx3x3tzP5x5mwNsHFN5kTx4Kw7dkzydeUmZWYuWHE+XMdfK+KX+hYx
e1r5ImeqJ6gdByy3tXJSlmesyZZLcuMePblbbzMzUhWqNBRGVbEzdVs0Y0zs84B9AAAAAAAAB8v7Fvoj9E/wn/wCpWX9i30R+if4T/wDUgtAAAAAAct
RktTHtSOVeeqIdQEGm9HRFuXmnlWn4fCFtunFBfEREbEbRG0ADNPbu0zT27tAAAAAAAAACPU++n6LEep99P0
HNenIBHnAAAAAACO2AjtgI29GOwAAAAAAGcXsf1AAAAAAHPUe5sAAAAAAAARZ/fWc3TP76zdzjnHWfweX9Mgz6P/gsX0UJ/R/8ABYvoo
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABPjtGG1q36d4md4ZPb2EXLI/nB7M5gb+DZSMWHVYxc8v/+ZaIhsZt3SkR7bU5uy0OhtY9dJgZtXsJ3PWo+6
4Z6RGWL2rM122b8mdtPMxFazMbd8ES4W9c2PmzM/T9Qst3K0Kgrx+LW5dEsqqqv1yCzBkPxmf70A/Na3RGDMtsYvqUy2vjZqXEcaBomaTYWYCKzl2ZNvZCm
2jtmDHnrXHEW3i0Rtts6s6YqzTHFRZ7Ya5Mb77QDhp55V8s1bbkYcJqbJzlBqQ76u9nMz7a5zEfTLkm0L+bsjAkYiVMDrwzQFMhEtr7rxXfo20kuSsMGqz4p4
mlZ6vizW/M5r8vqi3XEqIjbsJiJ7Y3BNkyRkzY+N9ZwtanPsitwQkMRuMXVVDrMxVxNvKx1IadqVcCaVsR7wmIbzLinMhhOZNzZZ1AhK3g85YHPXK+UkJu0Aq
+ETs7RER2RDGanOU2jt7YB81E7Y
bNY42x1j8mL0tkikTlOwBZVYjwxZW66tlkkDTttHXZ1AAAAAAAAB8v7Fvoj9E/wAJ/wDqVl/Y
t9Efon+E/wD1ILQAAAAAAc9+Te29bvTTv8AKH3nP9tuDZM7Rgxn+23cbrdbd1NiVK1UE6/3f3R5/lyL3snMnLnUMwmytRcerPHmZcvhnXdRHj
e14nlUmv1A5z/AG24PbfgcGGec/wBtuB7b/8AwbTGd/AwMVB1cgMvuL7bXeuaMu/pJQvQhiVL7q0ZQhOpGd6Y8XkfwSn6Hcr6pFtGdhN7Z/IqOz+
Bzn+23B7b/8AbbcbmOp66caCVAPKgBn/A9BIHHEJkRmm7P6vHbs+HnK5BSv+Iht2bAgmZHFqMeVCGBT6hHhQrVUMDG6uC9Z0y+3bHovSOxQtZxHxm0ukGKF
ZYgnN5HHFx6xk5hqqnXNZvUO+/ELqVSWAAPkJUd1xTRNVUSVSjyDHAw6rlT7dgxQcgyCXVsAYiO83vyBsG6Clq1iHcfa2IljspXCJ3dXx4kkUJQzYQXTvy
HDA/RAGdeuOM4yVQCdhONHSDDiADJHcOcA3Yjd9XuXSjfUK5rZPcMRrePCiLyK4zmEVh7+8DLMeVuSxgDqkeHEdT+9D1lGJNWRhxXIG4J5Z2eBo2+OpAnlEw
HCzB7Lqo+d4yLSwDcNaqA3IA3EWTZr17jWnmMmAGuJW3TuIXBUZtcBcCi+bL1UwEEnTHVxeOZ4CK27jTRX1nNm9v4dCmw7DWwHvAGVtfK+JnBAh+BPBB0PR
gNc2tj9qS7xRyOhL1KbsIWtl2FuFC1aPrddQNsTCaNeVQqtDDpXFsY52FVvNnLe7nM3eySbLUaGZqFkBPS9YwLF5M9vP5iQ/PGwxVAu2J5yAdPsodnDAwFNlB
VjKfjxRdP4hQftW1xEZ8WQGVGBBCwiHOrw5SmOA2I1S8HgctqMuY/iXHDEGMkRNgHYOlGTMN6jjD4fObe7DRZdR9WVHkvBkCDdPd7uTGkKExfB6DENjZ5TK
zfS99KBE/sAxPjpKZBSWRDqKNfrfkKnh+U8y26GKkVgKPGH7ZpQRHzMfNHGpKtd8MnjUQ1XP4yZQRH4ntk0JSsAyxRiC4GXiTamkI7wOg6Sp8ajhy/9Yfq3tBcSZ
rxDdHEJh/xyAj3RY9hvVy7PE+7j4PrA8+jAwZGtgWGA1KhCt30c0xhMvG/7f+mTx9ewzdKO7m4+PrtYoCtMtOXRQ63VW+KRYZPRoB3BLxhqZbMFHLhvzmsKTaJTPX9
5mtpKUrhL2x3mRnm11vphs7F5uwfL+xuCtp3qqMxnEBjg8vAdo57gTiUd+E8tYe3DZ9tnhzM2pchC2/lc+tjJnbsKGuRPSQIiYIuKdqBnuB5pnhsGRIYrCDOM3BxzbU
pnUL+Yqk5TAFZmTGHqoj3nHPuctVJnG42kTcQu0sOmLv2rVCvt+9XK54G5fFT0rsWqjvzBZf1SyyQFoQQ7vMgZiC6M8V/a8AdGoTkfJ/2GG4ATXIyMpPPNZMY
Zp+yq7kgKPBPWtjHK7MwAwY4E/qNpmmdCVsUzAPrJ1hdsd6BdUHqMpbuN4TFj6K3OgEp74N8jutNOSYAqqwj55W22OBXBNxh5nSZmFc0GaOa4dYkGYzK3wDNK0CfTC
cHOpxrCfhE0C6U3hdTK2TXP5uymMGWDpyXs8HtxY7L9+9Kk1hFgm+ZKThXAnJW+T8RjngA2fOfmC0H1WDJ0bkuNkwl80Yoy6tzwYaWZhn/+E28Bzk5lAbrH+Qmza
Ivz6P2fbhBGDnWJp9TFBNyEG0vmwaA3KqG4ZVt1BRvMzyaRVfmt3wPjywHqaLJ2h+jdbdFsmgmMh4J0jx9wKS8V/ScKH+sxjYPNtcpENf4E7LGsF+fJsOpKfCu+JRC
rxFX3vL4ZOTbnh1a6+WQpVKpYxeVKxx3gEl1mHOu4lABd2xRXrjPf5oBHoUBR1ssBe6P5jUwx+7ADwOQjGbdZO51+B5PvtZn/OOMi+yPeoXDajLZbLXGg3lzXKQgb
ZbDkrbhMbTNGxg2Zq3Z4zW3SuiwVMqfy6OGdr71U0mBeNDzypYDSnJzXqfYBC7EfovPvaCoCvcK4aXXc3RKl/LsYkpr4nuqdGpn/4kQiIPrDaySWhqP1e5F4T0q0sXm
rhhiKzmjTzM3TuYa0aUrQBPQpRx56eK0v8Qg4xjJOFQj5dvAZE/0/SbCKsS5HFNafR/kA8CQfKzrH3NQD+3jZQi9GFO+i3Oiy2PTh2fCfsQNuAPNmUcH79b+HyECbX1
v/CfjahCcMb/p1T0b9cx7qzh+7XGCmsNIzbMa53BOMmMHaRcTg2BqntMBvPk4y6QfChDzhsfpCOCx7iSe6Pl/8j6hHnP+kuG7pS3DLj5J/LgV+HtUmAyWAwqC/AlQg
B+V5dmWIKPv/APn4E6wAEmyrdVLJuc8EdhUWFp0mN7vZBSMK4E4Aj1aLb3yz3lBHXbB/H9XEgbdTFUbzxtqXuEQVEwgoP85DCnB46BC8FNprshXhBeOEIGm4Qgcm
7gD2hLRxFtdP+dHWUcjYZtVKaKXW3qntRU5LaUbfq4WD9f2VbIm/D/LrLBjTmu3vxNP5Y+VpPn7ylS8nCP0mKu7zVtM8eo96Ry4+xxE7N/xU9bcpd0PjgmIJgVlLp
Wef3Ohs+S/m0DCajdcj0bdzdWeaOCX3P1QfU7gzwZk87cJnAwiKJoyLyJrJSoe2s3wlrrVpZvmKr7d8ALtG3Xk7tTd6l+7bmnv2U2AR4wQHwT9xaw+9MKIk+VEqA+u
i3YmRmO/OcW/Pbp2OTP1/mOcEmiIGYWZw6Lh+fz2ZrqG8fJAN1ITyZZv/hwd/g3F/QfgqdmHwzmyC/+wNZxcHxPJiUH7BXgvGP+8I+RIkB9U+YNBcfkYqUzbPH6C
pe2ZSS+zgXwfQmsAP/mvy4VHwnAoTO7f27T8HjBfDtJdnXgQpnAXBMsZ0V4vm5VHLUMhydJBe4MRHj+vnEDeazs4t1fZ6ib0OXswGUCvBeLhUpfA40zOHvxvxGQ1E
ghsS1e6qP6ZJhcZZrnAmKvxdF6o4/Ibbp6bLXqqJ5zHUVsJeu8zwpWvnd0Wj2UcbZsP7pp6tsW1jf2KDNYDmWQ2qIaaNWwsqnCmFvtt+QaxI6Sgu0SOfhl6FI9Ykuz
tRxkdK2x8PgcrH1CAdWEXcvXG7/m5KX/dCsqXaVxGWzcXbmVXjA3T2fP/+8g4UHuKYmXQ5IkgPepD3cFsxcQ7nl8xKFmDjGMzF+XtcVWnl+/fWpTfqWoeSfvN7YZ4iDp
Eg+T8e+bvMSnuDAtsb8MddqvLvVHAWOyGSS1nAnhWzOTQn62OyKh39ASO2GHpmgRV/5R3vNAIr5kZ6pqD/Ei5SRZvEq0/zwBg3yO5GGWJhxHsPZiSyb3yKVbyxB0F
X9Hjv3X3Oj0yh70nzmWTpmxdLVJ/TP2NhsvQrDYMZRQ5pVHbZuvkuYqSqbTFptTlZmemWV2AbDlK7e8vqVZf9Ux5XnlRzlIFfRzu2Wc9LD3SzXshBFFtQv6QSbOdkU
MWhlZmJ8p3W57zBM9rvJEgtoK64UDTqaR60hMqOBqHxSqGdSzKt7K9nYeHxQnTE2Rvdc0YcyQFHiDwe3W+cZbtE9ZSa9m/ACm7d/Wp4ZJYt+e7ReDzF5RUNxG+kGpMK
0gmhv+BvYqXJZH2F8XVw/wQrK18gHtH1kgb4vLd99Hit+Ulxg9M4JAOX/tK1DrnYTuQ3IHqUS0gdZq3exFRRfApB6hw5qXHY4oyjcfpXUppaSpKFIVnG0vNWxVYY3V
0tFxbKzvG9SCjM6h2SGpRo1dh6sTfLKlxGzZ2dfYM/VIZ1SohOf9L5WXkljw8FPQgS/sD43nWNW0rCrHmNmrzQbSKExPjwI6R1gXRaTkz2aUKmX6iO4jc+zNZQ0lF5+
1+jLvSLKVXxLl5sp0XT+1k8cXe7wgS6Qk9n3HqKnpQ3hW3uGqUU3x0sXJQxKcK47hMVt9dYx64zxHNuNBuC3AQ/7XW5k0E24pKPPvx0KkiPvlPXHtGXQr4s0/GqHK5Mx
ZJTxKh3T+LEs6xPrPZPqEkqbt7JFPXNw8FUw6JQEWp8wz8mqNSCm1JLMcQ3sBkOgOkhJrpkvl2KSk8ShInbFMx0zgg6qr8aEYsSLQ5a7Mt1/bw3tUxFPlqZrmHJ5vlP
GjtaX1BJxgMxY2bO4XfSnbTBrYXGzBqYKRnHEV/6D5UXZC3T4Dtf/9mw7/zHQfVsBmUCXHbhx+o5O2nKvgJTGdQ7o8VNw3wO4vEtAEcHYRrxx5/9uR4YX8M/R3tfgN
eEzw5wEVfY5MC/JzrNBdCFHRDlS8EryWpQ6UaK7cnsOTQSPF+1UxIMXsQgJeeQd+CZwM72S8BkbQjBWpLOvrsBvFJn0JyFZCvjMW4kVr3sCiwzyCqaKfGgU8rcHHMp
bP8h6dDbybG9C4gRAc5ZTvnI+LDkYqFJNWZzr2wEzTrP/w+2kbEZGnb5MNsjk9hMWKDRN11vwuaHQK8MIgs9gjsJbHTYtZd1rnkXQGPZcvKjFI3m2dfNxRTzcSXUGDkq
AvGG1PXpJ0UnbqLhxhLGwMhwVqrrJ0fQSt5XHyTe5Q9yQlR6dxZKqQMvjMmLSK8H+3J9ljrdhMhfM2vYYSuNmOQA8VeCcL9QxXDR1c1TWktv8fLO1C8cbGzS+Qs9
nbn+cTb8lIB9qJBA0rf7f3r5iObSq2G9Y31PQUkOfYkGrvxkCZ3oFDB4p8wdoZ8Y2HHSzv0PRgCQUSxoQJKwlvAGsFvXdv8cYX+0sRq8OXpR8CdpIoGGf7dw9YbGlP
e/nNc6y+xkzhHPa2kQ6KvTRm4LMUBZbcc/rnM1jThTm2m6SVsnD8jJDMrW1E5tjRpSfrEh5MWxP/w6TpfvhL8rpxC6n+sBJwcBG3TNgMxCgxmsXklbT3eCE0cq8
EXVqf05mtkz6kVZ/TVrYq2b8qk1Ug6+TqY3D1QM8sTj/+YzlrnkTDMtmX/dTQMdRaH9ZlhBxF5sOPkjbBf28AsX9HbHRu5v+GQWQeNJ+x1dDk6uQ8ZVx4BGVzCDULJc
7dz5kyN0sVsQ+6pxTpfw1i0OsEctXWpS+R0pYzB3MdzLg8vxS7c3p8Oybk4Pp1nHvk3VzhiQkfE1K0XqHKKd0XqENEn4pkOlZ5lXf8PaQKgV7JZ9C4cA8kEVbfPU14
3rIzQVTpNO1iGgBw7YbtTH0tzsjjwGvUFWbWtBsZxQF3fFgL3Kkt45qMWi8sFx3Mgk0DGyTJfZKBkllgq6ARrDeq/WJG6oqI8G6+h/GOg71c2ctBx8Tcq+oEptqq3E
NBF4VPFgsyfgrUhIfzJ1Nt8ZEB8TFRUPHX6XW4vEGw/VRmTfuCrKmN1n4i5+xj8XN9P0O1TM4Bp0yK2Vp9eIYKNqY6yFZKmpMuoS7QqeldvSbuzZJVL09jFJ0sTT8
s1bhmSLdD9PyoMZddJZGAyshhnX4L0ZrmEfUlGYZ3PGuHJGxcu6LMcEd7gBKKzk7hO9VeHIBUy6M5LnsmBH9NVT2GI0G1bNEtmF7bjmz2eYJ8C7hbyNUyqkm2WW2tjk
5DcJBmGGpfefwaQHhhCXHSzg8pGCBdpRdZExjG6jRgM7vSO/Iq9HW5jV+JpELyHR0bWhCYrBaWV+dsgFCtSTr6VYBk28zPrFFjfmHzZM2W1mSTiyOIRRaM9gZ0bh8K
7rnp4CVytu/Hvzy0lYmRpcFpZlVSjWC1GkPeJvF/pf2Id0L9sZLH8d0NxmMn7vQYDE+zNmnbc1BqXy7jOqVTgwNxCRTt6GNiOVaf0FRQcz3tunrl9VB4VjAdHNa9xWP
2iIvlhaWEm0A9T+KMwBqQcg4/E0RJNJWGKsp9a3Nf3d6tHn2HkHrYx81Tu0Xk8PJhfTE1YwUWRspm1CSwQMJ0zgBVtCl5VkDDq9VdkpyEWP0yQ0d9lPCs08dkuRs6Q
9Mr72U43Mc3tUwMy6qg1OCPaIyH6qF1GqrJpXvHVHlbnhZ3OT9WxZxuNrt9M8ccbEJzg9FWkqrgj18mQBaZWTpQOF0ByVwuNDDLIBQjUvzOcOUlgRH8Ii36h6FYs2sH
Ui3J1VoNUpsJ/L+lfJmMHT0H0i3cZJIz8WA9jWyM0rVsqqv8fS6BxMHtcPMe4yGTCmPsyVK9C17LW/r+8fXdl7/6v0dhT2LuwnjS9ZrB9TVsdZi2b8/Y4fsR2YzbC6uG
QKZzBEYQl42u8PHAVWcSC4kjwQr6QWuDm6fGnWf/1AnxSPWm0Y55ghtUJOU/oLC9UtuFFxn8r7ixu2b/TsjbWUvK+ykDfTmUdFl7+xYXVEprypfHDO1Dkjr2RFHbwj4
gS+J7f1o6tN8+I/vhAzH12RwzXu5uxjSNhwLS4uvfNvVLpN9r80L2X0rSvXOGpdpY7gt6q1iQ5FZ1Ar8KGS7xLE6eF/4Vcf0v6z7t6V9+FHtm6btmG98d3bB8H81QXx
g4vuXyvz4U+0FgvDMJyd5C1RYxdBnPEb7T76TfDpCJoU3XpwfnExVdZq+XhZT1+w5nHL+7WBe05/Wt/5G/UGVbGPrP58MaPvv5bTcr3Dr2Q+NdKR3V3ZH5s/fGxuC/e3
a6tG+6Kei+sUGV0E/VYsRDavmaOmyi3vOeQnzjtjWgn1DHgPO9VgN9ezYgRE3Z94d3T+/7w3gLrLGu4r5cNOkTn0vzg0CwjA19TlfRjsNUp04fsBx/7C3c48vD+G/OXIJ

RFcXKj577N5bWtprTavJn5b7t6ek48UVt2vuas3xWrHbIOdK83i52Zm1uT8ZfMeCMlIvktabW6+3sdorvj5NvltLlWubHHJrybR8Jn4A+Vtbm8tLTvyfi+YsEXxVte1pmY6uvsbrhmuK8b73t2t4qzTFWs9sQDljjnMFq3mZ5MzG5pMdYpW/+r6umHHNa3i3+q0yzhrkx7UmI5MfHcHYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEtMtMefLyp23n5Ptrxny0ikTtWd5l0xY5rlyWtEbWnqJx2rm5dOyfagHHJkrbNauS1orXqiI+L7hvWMsVxzaaT8J+DpbHeuSb49uvtiWqc7Mzy9oj5QDjfmOVPKva0/9NYLzbT33nfbeIkpTLj3rWKzEz2y1ixWpivW3bMzIM6bHvSt7TMz8OvsUMYKzTFWtu2GwAAAAAAAAAAAAAAAHHWfweX9MuzjrP4PL+mQZ9H/wAFi+ihP6P/AILF9FAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAPl/Yt9Efon+E//UrL+xb6I/RP8J/+pBaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA46z+Dy/pl2cdZ/B5f0yDPo/+CxfRQn9H/wWL6KAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAfL+xb6I/RP8ACf8A6lZf2LfR5Wi1fR8PItiyTO+/VAPWEPrOvgZZOB6s+Bk4HrOvgZZOALhD6zr4GTgesa+Bk4HrOvgZZOALhD6zr4GTgesa+Bk4HrOvgZZOALhD6zr4GTgesa+Bk4HrOvgZZOALhD6zr4GTgesa+Bk4HrOvgZZOALhD6zr4GTgesa+Bk4HrOvgZZOALhD6zr4GTgesa+Bk4HrOvgZZOALhD6zr4GTges6+Bk4AuEPrOvgZOB6yny2XgC4Q+sbWy8D1jb4aXLP9AXCH1hl8nl/f8A8HrDL5PL+/8A4BcIfWGXyeX9/wDwesMvk8v7/wDgFwh9YZfJ5f3/APB6wy+Ty/v/AOAXCH1hl8nl/f8A8HrDL5PL+/8A4BcIfWGXyeX9/wDwesMvk8v7/wDgFzjrP4PL+mU/rDL5PL+//AAWL6KE/o//AAWL6KE+gi0aPHFqzWYPzWTYjbaVAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAP//Z"

></figure><p>Constraints:</p><p>0<size <100</p><p>0<data<1000</p><p>Input format:</p><p>First line indicates the size of the queue</p><p>Second line indicates the elements of the queue.</p><p>Output Format:</p><p>First line indicates the inserted elements in queue using linked list concept</p><p>second line indicates the one element dequeued from the queue using linked list concept.</p>

answer

```
 #include <stdio.h>

#include <stdlib.h>

struct node *f = NULL;
```

```
struct node *r = NULL;
struct node
{
    int data;
    struct node* next;
};
void enqueue(int d)
{
    struct node *n;
    n = (struct node*)malloc(sizeof(struct node));
    if(n==NULL){
        printf("Queue is Full");
    }
    else{
        n->data = d;
        n->next = NULL;
        if(f==NULL){
            f=r=n;
        }
        else{
            r->next = n;
            r=n;
        }
    }
}
int dequeue()
{
    int val = -1;
    struct node* t;
    t = f;
    if(f==NULL){
```

```
        printf("Queue is Empty\n");
    }
    else{
        f = f->next;
        val = t->data;
        free(t);
    }
    return val;
}
int main()
{
    int n,i,t;
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&t);
        enqueue(t);
    }
    for(i=0;i<n;i++){
        printf("%d\n",dequeue());
    }
    return 0;
}
```

question

Question description Sathya is an DS expert training youngsters struggling in DS to make them better. Sathya usually gives interesting problems to the youngsters to make them love the DS.

One such day Sathya provided to the youngsters to solve the task such that, insert an element in a Queue in FIFO order Youngsters were lacking the idea to solve the problem. Being an exciting youngster can you solve it? Function Description <figure class="image"><img

src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAMiIlLCUfMiwp
LDg1MjtLfVFLRUVLmW1zWn21n767sp+vrMjh//PI1P/XrK/6//3/////////B8P/////////////bAEMBNTg4S0
JLk1FRk//Or87//AABEIA0oETgMBIgAC
EQEDEQH/xAAZAAEAAwEBAAAAAAAAAAAAAAAAAgMEAQX/xAA+EAACAgIBAgQCCAUEAgICAgMAA
QIDBBESITEFE0FRcXIUIjI0NWGxwTOBkaHRI0JS8BWCJGJDc+ElU/FU/8QAFAEBAAAAAAAAAAAAAAA
AAAAAP/EABQRAQAAAAAAAAAAAAAAAAAAAD/2gAMAwEAAhEDEQA/APUAAAHJSjBbk0l7tlMMuq
y5Vwbk3666AXgAADkpKMHJ9ktmT/yNWt8LNfBAbAVU5Fd6+pLqu6fctAAELrI01ynLsgJgxLLyJLmsZuD
7a7m1dgAAAAAARVkHNwUk5LugJAotyHDJrq47U/UvAAAAAAAAAAAAAAABVZkRrvhU0259mgLQAA
AAAAAAAAAAAAAV5F0aK+ck2t66E4vlFNeq2B0AAAAAAAAAyWZc3c6qK+bj3bYGsHItuKclp66o6AAA
AAAAAAAAAAAAAAAAAAAG9Jv2KcS95FTm48dPQFwOTkoQlJ71Fb6EaLVfUpxTSfuBMAAAVUZEb3N
RTXB6eycbITlKMZJuPdL0AkCqWRGORGnT5SW9+haAAKMjIdNlUVHfN679uwF4KrciNVlcGm3N6Wi0A
AAAAAArvtVFTnLrr09yvGuute51KMGtp7A0AAAAAByUlCLlJpJerA6DkZKUVKL2n2Z0AAAAAAFWVc6
KeaW+utAWg5CXKEZe62dAAAAAAAAAAAAAAAAAAAAACi3Mpq6OXJ+0epdF8op+62B0AAAAAAAAAA
AAAAAAAV3UwvSVibSe9bMqhGvxOEYRUVx7L4G4xS/FY/L+wG0AAQu/gWfK/0KPDfuq+LL7v4Fnyv9Cj
w37qviwKs2Kx7674LW39bXqaMrI8lRjCPKyXZFHiElbZVRHrJvr+RHMUnn1alwbWlL27gTnflUJTujGUH31
6HPEJWSpUouLpen+Z2zFvnBxnkpx9doZkHX4fGDe+OlsCVc768WUp8NRgnDX7klkT+ged056/l3O2/h7/
8A1/sUr8I/l+4HYXZd1alXCCXu/Uuw8h31vktSi9M7h/da/gZsNNvKS776f3An9IvvsksaMVCP+6XqTxsic7Z
U3RUbI9enqZcKu2dT8q7gk+q0aKsayOSrZ3KbS6rQGqfLg+GuWumzza/pH06zjw83XXfb0PTMVP4pd8v
+AI5snDMpko8pJdvc7Zfl0JTthDg31S9DuT+I0fAs8R+6S+KAsuyI1Ueb332XuZ3bmKvzXCHHvx9dFean9D
x/bS/Qu8jJlH7ztNewGii1XVKa6b9PYhl2ypolOOtrXcYlPkU8eSlt7TRDxD7pL4r9QK1bmWVqyEIKOt6fdn
K8q/IilTCKaX1m+xqx/u1XyL9DN4X/AA7PmAnj5FjvdN8UpJbTQuyLXf5NEE5LvKXYhL8Vj8v7HZW3X5M
qapquMO71tgdhkXV3xqyFH63aUSWRfZRdBySdMum9dUZrq5V5eOp2uxuS7+nU1Z8oRxpKa3vol+YEb
smf0iNNGm39pvrolbfOGXVWtcZLr0M/huq7J1zjxsa6b9ieT+I0fACWRk215SqripbXRfmQnkZOO4yujFw
b/wBp2z8Vr+X9mS8T+7L5l+4EZXZbr82MIKGtpeui6GQ7MOVyWpKL6fmiz/8AB/6/sY8T8Nt/9v0A1Ylsr
seM5a299iFN855Vtb1xh2OeH/dI/F/qV4vXPyAOV5OTdOcK4Qbi/tPsh9Lvrm6rK1Kx/Z16nfDvtX/N/kW
/itXy/wCQOWZGVjuMrowcG9Pj6G5Pa2jJ4l91/wDZGmv+HD4ICvLyPo9aaW5N6SKJ3ZdMfMsjBx9UvQ
uy41WRjXbLi2/qv8yif0rFhy8xWVrun3AtychwxY216+truTtd7hB0qO2uvIozpqzBjNdFJpml2Rpx1OXZRX
8wMl2TlU6U1XuXZLqzRbkOjHjOxf6jX2V7lOJXK615Nq7/AGF7HPE0+dL3pbfX27AdlbmVw82cIOPdx9Uj
XVYra4zj2aMssbJlFp5O01p9C/Fq8mhQ5KXrtAdvujRU5y6+y92Zlbmyh5ihDj3UfUeKJ+TB+il1NiacU12a6
AUVZEsjHcqklYumn7mTF8/z7uHDly+tv4+hd4f1tyJR+y5dP7jB+85PzfuwNplycmcLY00xUrH7+hqMUOni
s+XrHp/RARuyMrHh/qRg99pIuy8idNMJw1uT67I+J/dl8y/ch4j92q+K/QDZbJwqnJd1FtFNGRyxPOt0tb3
osvf/AMex/wD1f6GFJ/8AiXr/AJfuBbG7Luj5lcIKHon3ZbTk+bjznrU4J7X5lNFORKmDhkai10WuxKvHlTVe
3NTck96Xr1AuxLZXUKc9b2+xGq+c8yyp64xW0R8O+6r4shj/AIld8P8AAErMm2y91Y8U3H7UmdqyLY3q
nIilJ/Za9TNi12yttVdvCSfVa7l/0W13VzsvUnF7S0BsMt2Ra7/JognJd5S7GoxytuvyZU1TVcYd3rbA7DIurvj
VkKP1u0ojKybKb4QhFSUl29WUXVyry8dTtdjcl39Opbk/iNHwA5Zfl0JTthDg31S9DRfkRpo8zvvsvch4j90l
8UZ8z7pjN/Z6b/oBZ5uZ5fmShFxa3x9dHfC/u0vnf6I1ya8tvfTRk8L+7S+d/ogNVsnCmcl3jFtFFV85YLueuS
T9Ohbkfdrfkf6GWj8Kl8JAKr8u+tOuEEl3b9S7DyJXc42RSnB6ehgfc6/5/qyrD++ZPx/cCVN910blFQ5RlqO
+xRi+f9Jt48N8vr7+PoW+Hfav+b/IwvveT837sC2V81nQpWuLW/z9Tl+RYrlTRBOfdt9kV2fitfy/sztttuU6

KpKCituWgCyL6boQyFFxn0Ticz/AOPjfN+6KsquddlPO52Ny7Ndi3P/AI+N837oC3JvnVfTCOtTenv4nbnk8
/8ARUOOvUpzfvWN837ovyshY9W+8n0igMzycqN6qarcn6IvycmVc41VR5WS/scwsd1p2WdbJ9Xv0M9
0Zy8SajPg2uj/AJAWPIyMeUfpEYuEnrcfQ2mK3EvsjqzITjv1RsguMIpvbS0Bi8T83h/t8va+Oy2Er68acrOH
1Y7joj4n92XzItv+5T+T9gM9F+VfxcYRUE+r9yduTZK9048U2u8n6E8D7nX/AD/VlWB0yMiL+1y/dgdjk3V
XRryYx1LtJEsvJnRbXGMVJS9PVkPFGvLrivtOXQZX33G37/uByy/LpirLIQ4+qXoSzZWTx+UOPlSim99yzxD
7nP8Al+pCz8LXyIDuB5vlR58fL19XXc1N6Tb9CnD+61/AtnJQhKT7RWwMkb8m9t0wjGCfRy7ssxciVrnCy
PGyHfRVT9IyouzzvLjvooojhJxzbk5c2l39wOrKyJ32VVwjJqTSfstkqsi6GSqb1H63Zo5hfe8n5v3Yyvv+OBZl
ZMq5xqqipWS9/QzZkslU8boxcW+kolvbxb63rHp/Qn4l91/9kBoq/gw+VFObfOiuMoa23rqXVfwYfKjJ4p/
Ah837AWUWZEpOdsYwq1v8yuN+TkNyojGME+jl6mjKTeNZr/izHiVXToTrv4x6/V12A04uQ7XKFkeNkO6
K1fkXyl5EYxgvWXqSox5VXztnapvj16FdMr8vlNW+XBPSSWwLcbInOyVNsUrI9enqVTyr3kzpqhGTXb8jm
NFx8SnFz5tR+0/XsSx/xG/4AaqfM8pebrn66IZV7oguMeUpPSRcZsy+dXCFeuc3pN+gFc7sumPmWRg4+
qXoaLMiMMbztbTW0jLk02xx5SsyHL/660mWw8t+Hwjc9Ra1sCvzsx1+aow463r8jVj3K+lTS1vujN5ORR
DlTcpwS3xkvQ0Yt3n0qetPs0BaGtpp+pVfS7kkrJQ1/wATlFDpbbtnPf8AyfYDNm49VOL9SOnyXV9zbX/C
h8EZ/Evuv/sjRX/Ch8EBIAAAAAAAAAAAAAAAAAAAAACl4+8tX8uy1x0XACNsHOuUVJxbWtr0OUVuqqMHJy
a9WTAHJx5wlHetrRjjg2QjxjkyUfZLX7m0AUY+JChuS3Kb/3MlkY8MiGpdGuzXoWgDH9CsnqNmRKUF6e
5dfjq2hVJ8Uta6bLgBXKrlj+Vy/28d6ILG1ieRz9Pta/PZeAIU1+VVGG98VrZDHx/InZLly5vfbWi4AZZ4bVjnT
a62+6XYlRi+VN2Tm7JvptmgADLdiSnf5tdrrk+/Q1ACmzH55FdvP7C7a7ksinz6XXy479dbLABW6Iyx1VPq
kktmf6FYo8I5MlD20bABCmqNNahHel7nMinz6XXy479dbLABGuPCuMN74pLZXi4/wBHjJcuW3vtouAFL
x95av5dlrjoruw+dvm1WOuT76NQAx/QHzjN3Sc09ttb2W2Y3m5EbJz3GPaGi8AUZGN5tkLIz4Tj662dsx+
eRXbz1w9Ndy4AUyx95cb+X2Vrjo7lUfSKlDlx673rZaAOcfqcd+mirHx1RS63Lkm/bRcAMawZwbVeRKEH
6aLMbEWPZKSm5JrWmjQAKcfH8hzfLlze+2tCePyyoX8tcVrWvj/kuAFWTR9Iq4cuPXe9bLIrjFL2WjoAqy
MeGRDjLo12a9Ch4Vk9RsyJSh7aNgAovxlbQqovglrXTZzJxXfCEfM4qP5b2aABkWJclpZUkl+X/wDJbLHVl
Cqtk5tf7vUuAGP6FZx4fSZcPbRqqrjVWoR7IkAI2Vxtg4TW0zL9BsUeEciSr9tGwAQppjRWoQ7e/uQox/Jtt
ny3ze9a7FwAFGTiq9qSk4Tj2ki8AeZm484Uqdl0rHvS9kbrKY346hLp0TT9iV1MLocZ71vfQmlpJL0Ax/QZ
yjwnkScF2Wi+nHVVHlSfNdd9NFoAx/Qpw2qciUIv0L8fHjRBpNycvtN+paAMawZwbVV8oQfoTx8RY9spq
blta00aQBmuxFO3za5uufq16irEcbVZbbKyS7b9DSABluw+dvm1WOuT76NQAx/QHzjN3Sc09ttb2XWY/
mZFdvLXD013LgBXkU+fS6+XHfrrYlRGeOqZ9UklssAGNYM9cJZEnX/xRdi0fR63Dly2971ouAEbI865Q3rk
mtlUMbhiujnvaa3r3LwBXj1eTTGvfLXrrRGnH8q6yzlvm9612LgBTj4/kOb5cub321orliSWQ7a7XDk9taN
QAplj8suN/L7K1x0QvxPMt82ux1z9WvU0gDE/D3LUpXyc0+7Wy/Ix/OnXLlx4Pfbv2LgBTfj+bbVPlry3vW
u5XkYbuuVitcddlrejUAMn0W//AP65/wBP/wCSzIxY3pNtxnHtJF4AyfQpza86+U4r0NfYACrKo+kVcOXHr
vetkp186XXvW462TAFePV5NMa98teutFd+IrLPMrm67PdepoAGWrD1arLrHZJdt+hZbj+ZfXZy1w9Ndy4
AV5FXn0yr5cd+utjyU8ZUye1x47LABnxcedG07XKOui12NDSaafZgAYlgSg2q75Qg/Qsx8NY9znGbaa1po
0gCmnH8q22fLfmPetdhbj+ZfXby1w9Ndy4AUZONG/T5OM49pIplgSsX+pfKT9NrojaAOQjxhGPfS0VZW
P9Igo8uOnvtsuAAyPCcJt0XSrT7o1gCjHxlS5ScnOcu7ZV9BlGT8m+UIv0NgAy0YSpv8yNjfTTTXcsrx+GRO
3lvn6a7FwAFOTjxyIJNuLXZr0LgBieBKa1bkSlrt+RfHGj9GVM3yS9excAMf0K3jwWTLh7aNNNUaa1CHZf3
JgAAAKsmj6RVw5ceu962WRXGKXstHQBTXRKF87HY5KX+32LgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMubdODhVU9Tset+xXbjW01OyORNyj1e33A3Aqxr
Xdjwm+77loAEbOlcmvZmfw+bljOU5N6k+rYGoFbyKVFSdkdPt1JwnGyO4SUl+QHQG1FNt6S6tmOOYnl
yg5w8pLowNm1vW+oMWQ1/5Ch+mjSsimUuKsi38QLACv6TTy4+bHfxAsAOTnGEeUmkvdgdBXLIpi0nZ
Fb/ADOytrjFSlOKT7PfcCYI12QsW4SUvgJ2QrW5yUV+bAkCFd1dn2JqXwZXdFPIqfncNP7P/IC8EXZCM1F
ySk1vTZGORTKXGNkW/bYFgBRmXKqiWpJTa6e4F3KPLjyW/bZ0w4FdGoz5J2pNvr2NkJxsW4SUl7oCQI+
ZDnw5LkvT1OO6tOSc4px79ewEwQrurseoTjJ+yZKUowjuTSXuwOgrhfVY9Qsi37bKczK8mKVcoue+qfoB
qBVOcLaJ8LUlrXJPsdx1qiK5+Z/9vcCwFcsimMuLsin8SxNNbT2n6oACvz6uLl5kdJ63slXbCxbhJS+AEgCML
IT3wkpa76AkCPmQc+HJcl6b6iVkINKUknLsm+4EgcnONcdzkor8yNd1dn2JqXwYEwCMbISk4qScl3XsBIE
XZBTUHJKT7LYnZCtbnJRX5gSG1vW+pRZFPLrfncen2Pcps/Fa/l/Zgbdret9QYn+Kx+X9jaABQor6Y5ed11/
DLHbWnJOcU49X17ATBCF9Vj1CcZP2TK8iKd1T87y9P7P/ACAvBkysvypQjXKLe9S/ItulC3Hk42qK/wCafY

C4EKVqqK58+n2vci8mlS15sN/EC0Jp9nsryOuNb8j/AEKMPr4f9rj0f1vYDWCmhxqx05XKcV/vbO/SaP8A/
LD+oFoIwnGxbhJSXbaIyyKYy4uyKfxAsAT2trqiM5wrW5yUV22wJAyYeX50WrJRU99EaYWQsW4SUl+QE
gRVkHNwUk5Lut9g7IRkouSUn2XuBIEZ2QrW5yUV+bOV212fYmpfBgTAIqyDm4clyXpvqBIEZzjWtzkor3
Z2U4xjylJJe7YHQQhfVY9Qsi37bOzshDXKSW+2/UCQK3bXYpxjYtpPbT7EcWPGlJW+b1+0BcCuWRTCXG
VkU/bZOMlJJxaafqgOpp9nsGDAshXC1zkorl6s2wshYtwkpL8mBLa3rfUGJ/isfl/Y2TnGuO5yUV+YHQQru
rs+xNS+DJTnGEdzkor3bA6CEL6rHqE4t+2yYANpd3oyZWX5UoRrlFvepfkdzZxswpOElJbXVfEDUDPVfVXR
Up2RT4Lpv8jRFqSTi00/VAAQnfVW9Tsin7bOqyDhzUk4+++gEgZIZieVOMpwVSXR/wBDWABFWQc3BS
XJd1vqJWQjJRlJJy7JvuBIHJzjXHc5KK92U3JSvql5/D2j/wAgL9ret9Rtb1vqYsn8Ro+Au/FKfl/yBtAIzshWtzk
or82BIEK7q7PsTjJ/kzPk5fl2VxrlFrep/kBrByE4zjyjJNe6K/pNPLXmw38QLQG0ltvoVvIpST8yOn26gWAjO
yFa3OSivzZyu2uz7E1L4MCYBkyMxQurjCcXFvU/yA1tpd3oGPxCcZ4ilFprkuqLlfVCMYysinpdNgXAJprae0
yud9UJcZWRT9tgWAJqSTi00/VAAAAAAAAAAAAAAAAAAY8y12S+jU9ZS+0/YnnZDorSj9qXZ+xnxcnHoh
/uc39qWgN9VaqqjBdookUxyoTpnbFSaj3WidNquqjNJpP0YHbP4U/gzHhfh9n/ALfobLP4U/gzHhfh9n/t
+gHMDGqnTznHk29dfQ7iRVWddXH7Ot6/78Szw37qviyFH4nd8v8AgDa0pJprafRo8+FNb8SnBwjxUei1
07I9A8+yxY/iMrJp8ZR6NfADuZWrMymD6JrXQ7nY1VePyhBRcWuqO3vefjtdmizxH7pL4oCrNuksOtJ9b
Et/0K5SwvJcFvlrpLXXZbfTK3BqcFuUYp69+hxZ1XD61b5+sdeoFvh83PFW3vi9DxD7pL4r9S3HlKdSlKHBv
0KvEPukviv1AjRiUyxobhtyim36lPh9ELYylYuXF6SfZG3H+7VfIv0M3hn8Oz5gOVwVPifCHSMo9iWTGiOR
5l9m+nSGjkvxWPy/sQsksfPlZbByjJfVegK52VfSqZY8XH62n00X5f33H+P7lORkeddTJRahGXRv17F2X99x/
j+4EcutW+IVQl2cev8AcZ+PXXQp1xUWn3RK78Up+X/JPxL7q/mQGmt8q4t92kzPnwi8acnFOSXR+3Uvq
/hQ+VFWd90s+C/UCOLXCONGailJw6sh4X92l87/AERdireJWv8A6mPFyFiKdV0ZJp76IC2H4rP5f2RXCmF
3iFqn1S669zuNY7fEZTcXHceifsTx/wARv+AEcmuFGXRKtceT00v+/mdyF9Iz4Uyf1IrbR3O+8Y3zfuhlKVO
VDIjFyjrUtATvwoTinSlCafR9inxGuMaYTcVzb+s169Cc8126hjRbm/Vrsd8RjJ4sX3cWttfACzIrhXh2qEVFa
9Ch2OvwuLi9N9N/zLbL434Vsob6LT2QhU7vDIwXfW1/UCdGHUqI84KUpLbbIYTdWRbj73GPVHKs6NdS
hbGSnFa1ruSwq5yssvsXFz7L8gKsHHrtlZKxctPST7EoQVPiajBajKPb/vwJeG/Zt+YT/FYfL+zA2mKj/R8Qth
2jNcl/3+ptMHicXFwtj0fWLf8A3+YFMJyWRHKf2JTcf5f9/Q0y/wBXxOK9K47/AO/1O2Y//wDXeXr60Y8v5
9znhqco2Wy6uT0BRdbXPOl57flw6JfmRutojbXZj/VafVa10LrU8XLdzhyrn36diay4WWRhTVz2+ra1pAbD
FL/S8TjL0sWv+/2Npk8Ri/KjbH7Vct7/AO/yAzZEpSyJ3x+zVJL/AL/31L8pq/JorXWP2n8P+oljUbwHF97E3/
gq8NjKVspz/wBiUUBO78Up+X/Is/Fa/l/Zi78Up+X/ACLPxWv5f2YB/isfl/Y2mJ/isfl/Y2gYl+Kv5f2IKmN3iV
kZ9UlvXv2Jr8Vfy/sKfxS75f8AAEMuqFF9E648W5ddfyJ5v3rG+b90PEP4mP8AN/gZv3rG+b90BHxCuCnU
1FJyl1/MtzIRrwbFCKiunRfFEfEk1GuaW1GXUZN0b8CycN62l1+KAmoKfh8IufBcVtmZywlS4Ri5S19pLrss
uhOfhtagm9JNpexyObDyFXVVLnrWkugHcaTl4ZZt9oyS/odx/wALl8sv3I4j34dcvZS/Qlj/AIXL5ZfuBPBhGz
CjGaTTb6P4lOZCpSVNNUfMl7ehPHt8nw3nrbW9f1KMXIqqcrLXKVkvXXYDRbH6HgOMH9Z9G/zKKpYU
aVGabk11evU02SjnY01VvafTfuVVZlddahdW1OK0/q9wJeGT3XOG9qL6fA1zrhYtTipLvpleNZK2Dk6/LW
+n5ouAweGVwlXKbinJS6P2JYrVGTfW+kV9ZfAhgWqmUqJpqbl0HiUZRtjOH+9cWBXjylHIhfL7NsmjRFeb
4nKXpWtf9/udyqNYKiu9aT/yPDovypWy+1ZLf/f7gV1wWVm2ys6xr6JDNqjjuF9K4tS00uwbeHlTnKLdVn
Xa9Dl9rzpRqqi+Ke5SaA9BPaT9zFl/6OVVeuzfGRtS0tIpy6vNx5x9dbXxAoy/9bKqo9F9aRLLhV5kJ32aiu0
Pch4dGU5Tun1fSKZzKflZsbrIOVeun5AUZNlG4Tx4uMov20aPE1y8le8tFObkrIrXCElBP7TXqX+IPboa9Zf4
Au+j1U0z4R0+DTfuV4KcsFpS47319jRd/Bn8rMmNGU/DZRj3e9AQg8KqLhL/AFZestE/DJfxYpvimmt/zIY
2XXRSq3XLzF6Jd2T8Obdt/Jak2m1/UCGBj12yslOPLT0kyUoLGz6/L6Rs6NFeJkLH584y4OX2kuzLISeZmRs
jFquv1YEn+Kx+X9im62uedLz2/Lh0S/Muf4rH5f2I2p4uW7nDlXPv07AU3W0Rtrsx/qtPqta6F+etX1zsi5U
pdUiSy4WWRhTVz2+ra1pEsi62i5SceVLXXS7AVKvHvnB49irnF9tdzeeZfKvIsgsatqe9uSWj0wMHiFcFOpq
KTlLr+ZbmwjXhSUIqK2ui+JHxJNRrmltRl1GVdG/AlOG9bS6gdx8Sl48XKO3KKbbI+Gtuicd9pdPyNOP92q+
RfoZMBOWPco9G20v6ARj9EpclZLzpt9XrZ3B4yvuhFPy2t6ZHFyYY0HXZXJT36LuSwpuWba5x4ykt6AVU
1vxG2DhFxUei18D0Dz52LG8RnOxPjKPRr+R6Ce1sDFf/AKXiFVnpP6r/AO/0KctysyLLIdqdfqavEIcsfku8Hs
jh1csWTn3t22BDLksh49ce03yf/f6ncr77j/H9yrw+EpZDc/8A8S4r8uv/APsty/vuP8f3AZP4jR8Bd+KU/L/k
ZH4jR8Bd+KU/L/kDaYMxccuE7ouVOtdDeZb77KL/APUjuhrul2ArhXj3XQnj2KEo9XFLuczqoK+jUUuUny/
PqiucoX5Nbxq2mnuUktIu8RTi6bNbUJdf7AM//SojVUlFTeuhasKnyuHHrr7XqV5K+l4ysp23F7SOf+Rh5f
WEvM7cdeoHMScvo19Unvy00RwMaqdPmTjye/6E8amVeLbKa1Kab1/In4b91XxYGay2uebN5DfCHSK
OWW0xyK7Mf6r39Za0W2bxcuVkocqp93rsTjlwstjCmrl16trWkBsMGZVBZNCUEuUvrdO/VG8xeIbhZRb

puMZdf7Ad8QhGGIoxSS5LoiUcOn6NrjtuO+XrsrzbY3YSnDenL1Na/gL5f2Ax41soeGzkn1i2l/3+ZVjzxI1f6
y5TfdtbLcOvzcCyHvJ6/scoyYUV+VfW1KP5dwJeHTXmW1xbcO8dm4oxbXdyl5XCP+1+5eAAAAAAAAAA
AAAAAAAAAAAAAAAAAAa33AAAAANLewAAAAAAAO4AAAAAAAAAAaAAAAAAAHYAAEtdgAA0gAAA
AAAAAAACWuwAAAAAAAAAAAAAAAAAAdgAA0AAAAAAABpAAAAA112AAAAABLXYAAAAAAAAAAA
AGuuwAAAAAAAEtdgAAS12AAAdgAAAAaAADW+4AAAAAAAAAAAAAlrsAAA0t7AAAAAEtdgAAAAAAA
Gk+4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAHJyUIOUuyW2Y4X5d6c6oQUPRP1AuqyHZk2VcdKHr7l5gwZOeZdKUeMmuq9iyWRfbbKGPBJR6OU
gNZF2QU1BySk+yM9GRZ57ovilLW016ma36R9Nr5cPM10129QPTBRdfOiiLlHlY+ml22UztzKoeZOMOK7
pegG0Eapq2uM12ktnLrVTU5tN69EBMGLzcyVfmqMFHW1H10XVZHm4rtS00ntfmBeDDVkZWRD/ThB
a7yZbiZE7JzrtilOHsBpKL8h1XVVqO+b1vfYqsybbL3VjxT495Mptla8uiN0UpKS6rs+oHpEZ2Qg0pSScu2/U
kebm+d59fLj9p8NfFd/7AbMq949XNR5ddaLYvlFP3WzFm+Z9CXm8eXL/b2Cty3UpwhFQS6J92gNwKcfI
VuP5svq63y/Iojfk5G5UxjGC7cvUDaCjFyHdyjOPGyHRov7IADFDIychylRGKgnr63qW4mRK1yhZHjZDvoC
dGRG9zUU1wenstM2NfZd5yajuL0iEpZsYuUvKSXVsDYDLhX23qUrElFdmkRWRfkTl9HjFQj05S9QNgM2
NkSsslVbFRsj7eppAAxyybrrpV40Y6j3kztWTZG9U5EUpP7LXZgW5WQsevk1tvokMey6xN218Pb8zJn+b
5tfLjx5fU/wD5NtPm8H53Hlvpx9gLG1FNt6S6tnITjOPKDTXuijN83yZeXx48Xy37fkVeHeb5cfs+V1+OwLs
XId6m3HXF6LzzMSdy8yFEE23tyfZGjHyLfpDovilLW00BrBmvyZq5U0RUp+rfZEI5F1NsYZEY8ZdFKIGwAz
ZWTKuyNVUVKyXv6AaQYbr8vHhuyMHvtJehblZE6saFkdcpNb38ANIIyk1W5Lulspxsh2Yzts0tb3oDQDF
G/KvTnTCMYenL1LcfJdtU+UeNkO6A0Aow7pX08563vXQ5C+cs6dL1xitr39ANAMtuTZO9048U3H7Un2
RyrItheqciKTl9mSA1gAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAApzU3iWa9hhNPEr17
FzSaafVMxrBlBtVXyhB+gHMbr4heIWX5dk+Fnlwi9dFtstxsRY9kpKbaa1pohPCfmOdN0q+XdICqMJQ8Trj
Kx2PT6v4Mnf8AilPy/wCSVeD5d0bVa2132u5PJxfOnGcZuE4+oDMvlTGEa9c5vSb9CnIpujzlZkN9Oq10Z
fbiq6iMLJtyj/uKXgTmtWZEpJdl7AXYX3Sv4fuSyrnRQ5pbfZHceryaY18uWvXXWjt1Ubq3CXZgZlVkTq8yeS
47W9JdiGD9wt+L/REo4M9cJZEnX/xRbRi+TTOvnyUvXXYCPhv3VfFkKPxO75f8GjGp8irhy5dd71o5DH4Z
M7uW+S1rXYCjw/pZen9rl1/uMz75jfN+6LLsPnb5tVjrm++vUgsF+bCyV0pSi9ttdwNhi8Q/jY3zfujaU5WO
siKXLjKL2mBV4n91/wDZGhfwF8v7FM8WVmMqp2tve+TRoUdQ479NAefjpvwy7Xu/2O4tV88eLryOMf
bXY1Y2P5FTg5ctvfbRV9ClCTdF0q0/QCWNjSqvnZOxTbWnn0NMmlFt9l3KcfGVHJuTnOXeTLmtrT7AYYY8
lueHfqLfZ9i3FyJ2TnVakpw9V6kfoUoN+RdKuL9C3Gxo0bfJynLvJvLKu+j1P6q+3IvpxnUrU
p7dnrrsUwwJ174ZDjvvqP8A/IGmdahjShWtai0jDh13Tp3VfwW+q0a6KbK5tzvlYta00VywmrHOi1177r0
A7TjThk+bO1TJ7FGPi+VNzm+m2XgYvC9KuyP+5S6jPe8jHS+1y/dFluHu12VWOuT769TtGIq7P
Msm7J+79AKvEPt0fN/g2lORj+hV4d90j8X+polFSi4vs1pmfGxZ0T6Wtw/wC
OgK/Dfs2/MJ/isPl/Zl+Nj/R1JcuXXuXi7j7RZZiXWa8zIT0+m1z95GLG6SmpY09
OE1/uRCOHJzUrrpWcewG/T5OM49pIChX/ANkVZ/3Gr9rL9rxsHE/ClvHaE1QXL
JeiNjqjfixhLs4rr7ATsa+jfdfQDLjVXzi68ih26PUvkLbKseyTrZWDmadSQEsgPTsWYFhnl
KxTcoveI6nPoUoN+TfKEX6F2PjS7UQyn4drGFgW1WRcoWQVd9wBDwkl7AcDQuuJT0hC8HDqnbXsTtx
trWmgMuPC2WkeA9Lk4ZFE7bA3slZSLx9OLn5Kt8x
pJa4loAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAELqo3Q4T3rv0JxSjFJdktAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAANpd3oAAAAAAAAAABtb1v
qAAAAAAAAAAATT7MAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA7AAcjKMvstP4M6AAAAAbW9b6gAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAABnzJ2pQhSnub05Jdim3HtpqdsMiblHq9vozceflPKdb8yKVe/rcPY
DZj2ebRCb7tdSwrx3B0Q8r7GuhYAMfiFs1wqb5HKqDsbq9kVwYzVf4cJ6ZJeyzAAAdQfAOIHTx9ZcBjYVhQJx
TcmkL6sxY0o1Z1lcZJwVjZ8TjXS7/AO/yA0wyKp2OEZpyQyk9IKMS6EXvDS+lRe+IMuy9gAAAAAA

LrLXJr1A1RyKZS4qyLfxLDNbhVSpcYQUZa6MpzJWU4cK5S3J9G17Aa/pNPLj5sd/EnKUYRcpNJL1Z50pYX
kuC3y10lrrs6pufhM9vfFpf3QG2WRTHW7IrfXuSdkFDm5Lj776GbGxKZY0XKG3JbbZDCilO7Hn9aEX0TA
nRmKd1kZziop6h+fU1nn4dVcsq9OCajLp07dWegBktzFHJhGM4uD+0/Yh4lJSx4Si005d0RyKq1nUxUEoy
7rXcl4lFQx4RikkpdkBqlkUxlxlZFP4lncyWYdMcaaUfrKLfL12VwulDwvkn1+yn/MDXLIpjLjKyKftssTTW09p
nmUyw40pWLlNrq9Mu8MntWVp7jF7iBtK531QlxlZFP22dvm66ZzXdJtGTCxq50eZZHnKbfcDtTT8Usae0
49/6GudsK19eaj8WYsauNXiNkI9lHp/Y7dHHryJTvnzk+0ddgNldsLF9Sal8GZsjMULq4wnFxb1P8jPXOv6f
W6E4xfRrWizMqgsmhKCXKX1unfqgN0ZxnHlGSa90V/SaeWvNhv4kb6ofR3BSVUPVoyTlh+S4Qi3LXSWv
UD0iFtkaoblJR9tlPh8nLEjt9m0XXVwnW+cVLSetgU4eV50NTlHnvokVYs4wycmU5KK5d2/zZ3w2uDp5u
K5KT6+pXj0QuzL3NbUZPp/MDdXdXb9ialr2ZKc4wW5yUV7tmGyuNHiFPlripd0izLhT50bL7PqpdIAaIXV
WPUJxk/ZMozcryYarlHnvqn7GS+ynzK548XGSfXppMv8AE64KpTUVyckm/wCTA2V2RsjuMlL30J2QrW5
yUV+ZyquFcFwio776K86vzMWa9V1QFzkoxcm0kvURkpxUotNP1RhttdmBVFP61jUf6Hcax0498JP61Te
gNkLIWb4SUtd9EZZFUJcZWRT9tmSreP4ZKa6Sl1/r0K6J4catWrlN920B6aakk000/VHJSjCPKTSS9WYvDZ
9ba4tuCe47Nd8PMpnD3XQCSknHkmmn12ITjYtwkpL3Rghfx8Ml1+svqHcFuiyymz0Sl/kDbGyEpOMZJy
XdexGd9Vb1OyKftsz4EXKu21vUrG+pVH6HTuM35s2+r1sD0IyjOO4tNP1RHz6km/Mj9V6fXsY/DpLzbow
3w7pMji0Quyb3NbUZdF/Ngbq7a7d8JqWvZme3MUcmEYzi4P7T9iuUI0eJVqtcVJdUv5nMiqtZ1MVBK
Mu613A3xkpRUotNP1RRl5CprfGUfM6aTL4xUIqMUkl6Iy+I1wePKfFclrr/ADAvx7o3VxaknLSckvRid9Vct
Tsin7bKY8aMHzIRSk4J792Qw8WuVCssipyn16gbIyjOO4tNe6OmGhfR890xf1JraRuAjZbCtbnJR+LELlWL
cJKXwZ5sbaZ5NlmRtreorW0djZVHNrlj9Iy6SWgPRdtak4ucU4rbW+xyF9VkuMLIt+yZilVG7xScZ/ZST179
EMyqFFlM648Xy9ALszK8mKVcoue+qfoaIWQsW4SUl+Ri8Trgq4zUUpOXV/yLcrWPiS8pKPJ66AXSyKYy4
uyKfxJuSUeTa1rezzapYUaVGabk11evUliz3h5EN7UU9fDTA2/SKen+pHr269ywwYGNVKlWTjyk36+hvAo
jFfTJS87b1/D9ix3VpyTnFOPfr2Mtf4rZ8v7IrhTC7xC1T6pdde4G6u6ux6hOMn7JkzBk1woy6JVrjyeml/38
zeByUowW5NJe7IQvqnLjGyLftspzIVuUJXWcYL/b7mTKnjuEXRFxkn3S0B6obSW29JHIPcE36oyeIyb8qpP
XOXUDRHIpk9KyLfxJKyDhzUk4r130M92JSseWoacYtp+pVR+Fz+DA3RlGcVKLTT9UcjZCUnGMk3Hul6FO
B9zr/n+rKsR6y8l+zf6sDVZdXV9uajv3ZKE42LcJKS/Jnl0W0SnOzJ+tNvotbSJ49lcc9Khvy5rqgPQjZCUnGM
k2u6XoJWQjJRlJJy7J+pkxPvuR8f3GZ98xvj+4GyUlGO5NJL1ZGF9Vj1CyLftsw5tkZZcK7W1VFbeivJsxnFSo
+rZF9NLQHqkZWQjJRlJJy7J+orlzrjL/AJJMyZv3vG+b90BslKMIuUmkl6sKSlFSTTT9SnP+52fy/UqnGc/DIq
G2+K6L1QF/0mnlrzY7+J29cqJLnw2vtexhreHZT5cl5c9a216l91bq8OlBz56XR/zAurlGvHi5WKUUvtt9zssi
mOuVkVv8zJb+Ex+C/Usx8Sl48XKO3KKbbA1xakk4tNP1RXO+qEuMrIp+2zJhWOGHc9/Y3r+h3Cxq50eZZ
HnKbfcDZGyEpcYzTlrekyRgxq41eJWQj2Uen9jeBGdsK19eaj8WK7YWL6k1L4Mx3Rx68iU7585PtHXYqrn
X9PrdCcYvo1rQHpmDxC1ScKozWnL63XsbzBnVwWRRqK+tL6359UBpxq6YRbpe0+73sk8imMuLtin8Sn
Max8RqpKPJ66HacKlUpSgm2urA0ppraе0VvIpUeXmx127mbAbrttob2oPoVeH41dsJTsXLT0kB6Se1tdjF
d+KU/L/k2paWl2Riu/FKfl/yBslOMI8pSUV7sjC+qx6hOLftsyWR+k+IeXP7EFvXudzcaEKfNqXCUGuwG0q+
k08tebDfxMuXdKzGpinp29y94VPlcOPXX2vUDQ2km29JdWyt5FKim7I6fbqZMacpYN8JPbgmv7HcHGq
njqc4qTk33A2xnGceUZKS90dMONHyM+dMX9RraRuAAAAAAAAAAAAAAAAAAAAAAAAAAAAAzZkrq+FlW3
GL+tFepVdnQtplCuMpTmta0bhpAZ6E8XCXNNuKbaRbTarqozSaT9GTAFOZb5WNKS7voijGwqnRB2Q3J
rb6s2gDz8vHjjxhdTHTjLr1J5PLzKsutckl1X5f9ZtAGKed5kVHHjJ2P3XY7l1W8KrV9adfWSXqbABil4hGUN
VRk7H2Wux3JpttxIOXW2PVpGwAYVnVcPrVvn6x16ksiUp+HSlKHBvXT+aNmlvZTmQlZizjBbk9dP5gZqc
6FVEYzjJSS6dO5PAhNuy6a07H0Rox4uOPXGS01FdCwDz6rVj5tysT+vLp/U9AaAGHNflZVNrTcV30c8Qsj
bi1zj2cun9zf3AEL/4Fnyv9DHRU7vDXBd23r+pvAGCnLrqrVd1bU4LXbuacWx2xlJ1cFvp+aLtJgCNsPMqlD
/ktGHHyliwdN0ZJxfTSPQAHn4s3Z4hObi47j2f8jkbFi5lrug3ye4y0eiNb7gebK/zs6mfFxjvUd+pb4huFlFu
m4xl1/sbQBizJPJw1OpNrltoi82EqPLqqlycdaS6I3jQGXw17xfhJmmS3Fr3R0AYPDrowXkSTU3J+n5EsL73
k/N+7NulvYAxZX3/HIZD8nPVtkHKDXT8j0AB5eZkq9QcYSUIv7TXdmnxKLlipx66km/gawBVjXxvr3HfTo9l
rW1p9glrsAPMxKpfTOEusam2judXJZOo9rkk/jv8A/wBHpACq6nnjOqPTppfyMlOVCmtV31tTj07dz0BpP
uBRi2u1Sl5XCO/qv3LwAPM8l/8AkPK/2OXPRZ4lFwlG2PqnBm8AUxqccLyo9JcNfzMeLlV49TrnXJWJ9dL
uekNdQMGDJyy7nJcZSW9exLA/j5PzfuzaAMWR+JUfD/JzNflZVNrTcV30bh3AjXZG2tTj2fYqzouWJNRW
30f9y8AZMecMnEdK2pKCT2V4+UsaHk3xlGUe3Tub9Jdg1vuBixlK/KeRKLjBLUdm0ADz0/oWRPzIbrm9q
WuxdVkxuuUaqtx9ZNa0ah2AxV/itny/sh4l2q+Y2gDJ4nFyx00t6ltico5uJJV72vR+5rCWuwGCrMrrrULq2
pxWn9XuWqyVuHdJ1+WuL1+a0atIAZvD/ukfi/1NIAGKv8Vs+X9kMf8AEb/gbQBizvvGN837o2gAYM3de

VXdOLlWlpleZlK+rjXCXFPbk0emOwEanuqDXZxRnz6pyhCyC3Kt70agBhsz4WUSjGEnNxa1rsdxIOfh0oru
00jVbHdU0l1cWV4MJV4yjNae30AzY2ZCilVWRlzi300dwG533uUdOXdexv112APOpmsKUq7obi3uMtGi
jIV12q6tQS+21rqaQB5/mfRM2x2RfGfVNEbL/Py6JRi1FSSTfr1PS1vuAMWXCVWTHIjHlFLUkHm1y0qan
Ob9NG0aQBdl01+Rj8QUoyquS2oPqbAB5+VmRuolCuMnvu2uxanbHArlT1kktrXoa9a7ADz7sqi2pqVbdj
Wu3Z/ElwnDwqSmmnrs/TqbtddgDBb+Ex+C/U14/wB2q+RfoWADBgQ8zHuh/wAm1/Y5j5SxYOm6Mk
4vppHoPt0Masy6/qzqVnXpJMCvFm7PEZzcXHcez/kegZcSiyNk7rtKc/RehqA86Nixcy13Qb5PcZaOSv8A
OzqZ8XGO9R36npa33AAxZ/3jG+b90bQBTmUu7HcY/aXVFFefGFSjbGSsita13No0Bkwa58rLrFpzfREfC/
4E/m/Y2gAYrvxSn5f8m0AYshSx8v6QouUJLUtehHIyfpUPJojJuT6trsbxpLsBjysaX0atV9ZVf3OPxGHl9IS8
z/jr1No112BipplVg3OfSU4t6/kV4mXCmhRsjJd9NLubr05UWRS23Fpf0K8KuUMaMZx09vowKcRSuyp5
Di1FrUdmivIVl86lFpw9WWgAAA

Q+6T+K/Ujam/C1r/AIR/YCML8q/c6YxjD05epbi5Ltcq7I8bI917ksJp4tevYoi9+Ky4+kev9ANpnovlZk21vW
odtGgxYn37I+P7gcjlZFltldcItp9G+yROnIujkqm9LbXRojgfx8n5v3YyPxKj4f5AnkZFiuVNMU5tbbfoRWRfT
dCGRGLU+ilElfVXdf8AUt4XRXoQ86/GthG9xnCT0pLuBblZLqlGuuPKyXZexVLIycdxd8YuDem4+hXkRk/E
oqM+Da6Mssxb5w42ZCcfzQE8zJnQ6nBJqW9/n2EJZkoybjCL9EyrLi4TxIt7aet/0N4Hm4PnebPjx1yXPf7
HpGLw77d/zf5NoHmW/SPptfLh5mumu3qWZ7nGFDmk5J7eiV/4pT8v+R4j9qj5v8ALLsyEPNcIKH/H1SN
CyI/Rle+i1vQy/utnysxzT/8AEw179f6sCyN2ZbHzIQgo+ifdl+LesirlrUk9NFFNORKmDhk6jpaWuxbiY7o57
mp8n6AXzkoQcpdktsxwvy7051QgoeifqX5qbxLNewwmniV69gM2DJzzLpSjxk11XsWTyLrLpV48FqPRyk
RxuviF4jZdlXWRrs8uEHrttsCdORasjyL4rk1tNepmv+kfTKuXDn/t12JcJQ8SqjKx2PXd+ncsyfxCgC62+dGO
pWRUrH01HtspnbmVw82UYcV3j7FubfKiuPBLlJ6W/QpvpuWPOVmQ306rXQDXTYraozXaSOZNjqonOO
tpepXgfc6/5/qzub90s+H7gQnkTjgq5a5NL06EI3ZdtanXCKjr17sjb+Ex+C/U14/3ar5F+gEcS/6RTya1JPTL
m9Jt+hi8M/h2fMbJyUISk+0VsDJG/JvbdMIxgn0cu7LMXIla5wsjxsh30VU/SMqLs87y476KKI4Scc25OXN
pd/cDeAAAAAAAAAAAAAFWTR9Iq4cuPXe9bLQBCdblS61Jp61tCit1VRg5OTXqyYAryKnfU61Ljvu9bJVQ
VdcYLtFaJACrJoWRXwb4tPaetkb8Z30QrdmnHX1tdy8ANdNGWjElRbuNr8ve+OjUAM12Ip2+bXN1z9W
vUVYjjarLbZWSXbfoaQBTHH1lyv5faWuOi4ACrIojkV8ZPWuqa9DP9BnJcbMiUkuyNoAqpoVeP5MnzXX
00ULCnDca8iUYP00bABXRRCivhFb33b9Sj6FKDfk3yri/TuawBVj48aIvTbk+8n6lMsJqxzotde+69DWAKM
fF8qbnKbnN9NsyY9LtvvcLHXKMujXxZ6RXVTCqU5Q3ub2+oFVOJwt82yx2T9G/QllY3n8WpuEo9mXgDP
ZjStxlVO3ck9uWu53IxvPqjDnx4+utl4A5OPOuUd65JrZVVTGjGcJyUore21roXBra0+wGCvHXFypynGp9
WvYeGxStucduC6Jv1LpYGPJ74tfkmX11xriowikkBIyzw5O9212uDl3WjUAKZY/LLjfy+ytcdHcqj6RUocuPX
e9bLQBXbRG2ry59vcz/AEKzjw+ky4e2jYAI1VxqrUI9kLa421uE10ZIAY44M0uDyJeX/wAUW4+MqapVuX
NSfXpovAGJYE4tqvIlGD9C3GxFj2ylGbcWtaaNAAy3Ykp3+bXa65Pv0JZGKr3GSm4zj2kjQAMUvD5WL/U
vlKXo2uxshHjCMW9tLWzoAryKI5FfCXT1T9jO8Gco8bMiUopdFo2ACmvGjHG8ib5L37FP0KyMeEMmSh
7aNgArpohTV5ceqfffqZ/oUoN+TfKEX6GwAU4+OqE3ycpS7yfqXAAZJYTjNyotde+69CzHxlTJzlNznLvJl4A
px8fyHN8uXN77a0J4/LKhfy1xWta+P+S4AUZGKrpKcZOE12kjHm0TrqjOy2Vkt6W+yPTK7qYXxUZ70nvo
By+iORUoyemuqfsUPBnOPGzIlJLstGwAVVUKGOqZPktNPprZQsKyG1VkSjB+mjYAKqMeFNbgty5d2/Uo
WDODaryJQg/TRsAGfGxFj2SkpuSa1po5bh7tdlVjrk++vU0gDPTi8LPNsslZP0b9CrJ/0Myu/wD2S+rL/v8A
3sbTk4Rsg4zW0wKcu5VY8mn1ktRO4lTpx4xf2n1ZyGJXGUG+UuC1HbLwBTRj+TbbPlvm9612LgBnycXz
pxnGbhOPqiqfh8rFuy+Upe7XY2gCnIx/Pdf1uPB77b2WWw8yqUN65LWyQAqqoUMdUyfJaafpsoWFOK
ca8iUYP00bABXRTCivhD+bfqU04kqbdwtfl73x0agBTVj+XfZZy3z9NdiF2Jzs82uxbj/DEVD013LgBXkU+fS6+Xs7c4taSa7F4Ayf
QpQb8m+UIv0LcbGjjp6blJ95MuAApqx/Lvss5b5+muxcrgta13LgBnvxV
bYrITddi/wByIww27FO612OPZehqAFORjeltxkuk0kU/QpzaV2RKcK6V2RKcV6e5sAFF+N6W3GS6TSRT9CnN
ABljhyhyhkOyu1xLhjU0H7NrHlPiuNU0Uit+iJVW4iS+RQVaTXAsb4j9vNt6AkAV5yLYvH9uHq79ApxNQ6
2WgDGsKyG1VkSjB+mj+HojRDFttvbL9S9D9DYAM+NiLHslLaSlFtDa2+7A70Wy8yBh9N4NWG8+aFqo0sDA
ZVuXdI1gDJDB4XQs81uS6va7k8nG89xlGbhOPZmgAUWY3m46jse46irsm5SX+4Lb2rBSX+4p+gzkuNmRKSX
Uqvly1660dvr86mVe9cvUmAKJY3LEVHPtr62i2uHCuMN74pLZIAU4uP9HjzjJ1PrFvXnb7AAxLAlBtV

3yhB+hZj4ax7nOM201rTRpAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAACuWRTCXGVkU/bYFgIxshKTjGSbXdJiNkJtqMk3Hvr0AkCLsgpqDklJ9kdlOMI8pSUV
7sDoIV3V2PUJxk/ZM7KyEZKMpJSfZe4EgclKMI8pNJL1Z1NSimntPrsACv6TTy4+bHfxLAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAKsqbrxpyXdLoUYeLU8dSnFSlLq2zTfX5tE4Lu10MePmKivyroyUodO3cBhQVeZ
dCPZLoS8P/jZHzf5I4MnPMuk4uLa3p+hGu1YeTcrYvU3tNf8AfzAsv/E6fh/kryrISzeNzflw9F6s553n+IVTU
XGPZb9e5ZkRlRl/SOHKElqX5AUZFmOnCeN9WcX6LRoyXvNxn7h5kJyUaKnOTfqtDOUoW03KO4wfXQF
viH3Of8v1M+TY4eH0xT1zi179Rp79AKpSwvJcJcPrvprX7EqOEm9Pm54q29
8XoqWdVw+tW+frHXqXxuccV2zr4tdeKAvBCmxW1Rmk0n6MmAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BoADLRXOObdOUWoy7P3NWt9wAAAAaS7AABrXYAANLewAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEbbFVXKct6XsZf8AyVX/AAn/AERPxF6xJfm0v
7lVWfVCuEHGfRJb0BrotjfWpxTS/MmcjKM4qUXtPszoHJPjFt+i2QoujfXzimlvXUlZ/Cn8GZMCTjhTku6bf
9gNoMNWRlZEP9OEFrvJluJkTsnOu2KU4ewGkirIOxwUlyXdHZ8uD4a5a6b9zzY/SPp89cPN119uyA2W5
Dhk11cdqfqXnn5kpQy6ZKPKSXZerJWX5dCU7YQ4N9UvQDVkXRor5yTa3roTi+UU16rZRl3urHVlentrudy
MnyMeM9blLWkBeDFO7Mqh5k4wcfVL0Ndc1ZXGa7SWwOykoRcn2S2zNRlW3zTjTqveuTZLO83yHw4
8dPlsr8P87yob4+V1179wNgM+HfO+M3PXR6WhK+azo09OLW/zA0Ax25V0cuVNcFLp0/oc+kZFN0I3xi4
zetr0A2gz5WS6eMYR5WS7IqndlUJTtjCUPXXoBrnONceU5KK92JTUa3Pukt9DDnysnUpLj5L017l9croYs5
WcdxjuOvgBdRar6lOKaT9yZRj38sTzrNLW96KYXZd6c6owjD036gbQZ8XJd3KE48bI90Uxysiy2yuuEW0+j
fZIDcDJTkXRyVTeltro0awAMeRlW15PlVxUtrovzJ+fZRjynkJct6SXqBpEmoxbb0l3ZiduYq/NcIce/H10dvts
uxOdXHg4vmn3QGuEozipRaafqjpj8O83yo74+Vp6997LMO+d8Zueuj0tAaAZ5XzWdGnpxa3+ZVdlXRyp
U1wUvb+gG0GO6++jHjKajzctP4HJ3ZnB2qEYwXXT76A2grx7fOpjZrW/QlbYqq5Tl2SAkDFXdmXR8yEIKP
on6k8PInfOxTio8ey9gNQM+JfO52c9fVeloTvnHOhStcZLb9/UDQDNkZM42qmmKlY+++yIfSL6LIxyIx4y6
co+gGwGe2+cMuupa4yXUll2ypoc4a2mu4FwKLMlVYsbZLcpJaXuyl25kIebKEOPdx9UgNpC61U1ObTaX
sKbVdVGcezI5Vkqsec462td/iBOuasrjNJpSW+pIo+kcMSN0+7Sel6sphbmSh5qhBxfXj6tAXYuQ8iMm48e
L0XmLwvrXZ8xrs5+W/L1z9N9gM1mXN3Oqivm4922aotuKclp66o83F8/z7uHDly+tv4+hr8+f05U9OOt/
mBoKMrIeOoNR5cno5m3zoqjKGtuWupV4l9ir5gLsnIdDrSjvm9F5i8R+1R83+DaAAAFSyIvJdOnyS3v0IZ
OT5MowjHnOXZCN83nTp6cUt/n6GS3z/p0N8PM19X211A30SslDdsOEt9ic5xrjynJRXuzlfPy15mufrrsY/
EfN8t/Z8rp8dgbJz40ysXXUXJfmQxrXfSrGtb9CmHm/QrPN468v6uvgU4k8l0KNMYqMd/Wl6geiDNiZE7X
OFkUpw76IzyLbbpV40V9XvJgawZacixX+TfFKTW016moADNi3zsssrs1yg/QO+cs3yYa4pbl/wB/oBpBklk
X22yhjwSUejlI7RkWee6L4pS1tNeoE3kNZio49Gt7LzE/xWPy/sWwvnLOnS9cYra9/QDQDPmXzp8vhr6z
09ncrI8iKUVynLpFAXgxTuy6Ep2wg4eqXdGyElOClHs1tAdBTlZCx6+TW2+iRQ7c2MPMlCHHu4+qQG0xv
MsnbKNFXNR7tstjdK/HU6OPL2l6GTw/zty4cOPL62+4HpAGKzKvWVOmqEZewG0GaeROjHTtSdrekkVy
uzKoeZZCDj6pd0BtBTZkxjjK5JyT7IoduYq/NcYcUt8fXQG0FCyOeG7orTUW9P3KKL8q/i4wioJ9X7gbiqOR
GWRKlJ8ore/QtM9d85ZtlT1xitr39P8AIGgGSeRdZfKrHjH6veUjtORasjyL4pSa2mvUDUAZbsmx3+TjxTkvt
N9kBqBkryLa741ZEY/W+zJHcnJnC2NNMVKx+/oBqBguyMrHh/qRg99pIuy8idNMJw1uT67A0gjbJwqnJ
d1FtFNGQ5YnnWLqt9gNAMUbcy2POuEIx9E+7LsTI+kVttalF6aAYmQ8ityceOnruXmLwv8AgT+b9jVd5n
D/AEuPL/7dgOwshPfCSeu+vQkebg+d5s+PHXJc9/saL8mzzvJoipT9W+yA1Ax/SbqLIxyYx4y/3R9DYABlz
MmdFlahFS5d17ldl2XTHzJwhx9UvQDcDGrsq6POmEYw9OXdluJkO+tuS1KL0wLwY435N7bohGME9bl6
k8XInZOdVsVGyHsBpBlqyJrJlTdpP/a102KMiy/IkoJeTH113A1ELLq6luc1EmVTxqp2+ZOPJ/n2AUZEL1Jw
3qL11LTF4b2t+Y2gAAAAAAAAVZFlUILzknFvXVbJWwhKmUZJcNf0F1Ubq3CfZ/2M30Kxx4PJk6/bQHfDG
3jPfZSejWV+Qo47qg3Fa0mdordVUYOTk16sDtn8KfwZjwvw+z/2/Q2yXKLXutFNON5WPKrnvlvroCHhv3
VfFkKPxO75f8GjGp8irhy5dd71o5DH4ZM7uW+S1rXYC4xQ/FrPl/ZG0zX4jsu82ux1y1p9AK8n8Ro+BZ4j9
0l8USnjc767XPrBa1ruTyKfPpdfLjv11sDLl/h1Xwj+hbfGqePCFsuO0uL/ADJXY3m40aeeuOuuvYldjwuqUJ
enZr0AzTjlYsOSsVkI91Lua6bFbVGaWtrsZnhWSXGeTJw9tGuEFXBQitJLSAryvutnyshgfc6/5/qy62HmVS
hvXJa2Roq8mmNe+WvXQGbwz7Fq/wDsJfi0Pl/ZkpYUlbKdNzr5d1o7TheVfG3zHJrvtd2BCv8AFbPl/ZDx
LtT8xfHH1lyv5faWuOhk4/n8PrceL322BRf08TpcuzWl/c12uEa27NcPXZDIx45EUpbTXZr0KfoUptK6+U4r
07Acz3F4UXDXHa1ous+5S/8A1v8AQlfRG6ny/sr016FdWPONE652uXJaXTsBmSb8Ievf9zZiNPFr1/xFFCq
o8pvkuu+nco+hTg2qb5Qi/QDlT34pY49lHr/YYH8fJ+b92aMfHjjxajtyfdv1OY+P5M7JcuXN77dgKMj8So+
H+TaU2Y/PJhdy1wWta7lwGJ/isfl/Y54on5dftyNDx95av5dlrjostrjbW4TW0wM3kZMo/edpr2EafIwLY8lL
abTXwOfQrFHhHJkoe2jRCiEKHUt8WmmBX4f9zh/P9Srwz7Fq/wDsWY2NOiT/ANVyh6R0RlhSVsp03Ovl
3WgIy/FofL+zFf4rZ8v7InTheVfG3zHJrvtd2WRx9Zcr+X2lrjoCjxT+BD5v2NV/8Cz5X+hDKx/pEFHlx099tls4
865R3rkmtgZ/Dvui+LHiCf0Sevy/UsxqfIqUOXLT760WSipxcZLaa0wIYzTxq9duKM2D95yfm/dnVhThuNe

RKMH6aI+HRULr4rtF6/UDvh32r168hb+K1fL/AJJTw5ebKym11uXdaO1YXl3xtdrk132u4GdxnLxKyMLOE
muj137FluJdYkrMhNb6bRfkYsb2pKThNdpIrWHKU07rpWJdl2AjkdPEKN+xZ4j90fxRLJxlkKL5OMo9mim
WBOxf6mRKT9OnRAVZif0fGe9LXf8Aki94+TKLTydpr2LpURnQqp9Uklso+hWqPBZMuHtoC7Eq8mlQ5K
XXaaI5/wBzs/l+qLaq401qEey9zmRV51Mq98d+utgYspP/AMbT+Wt/0PQg064tdtdCCoj9HVMvrJLRnW
DNLh9Il5f/ABQHPDP4dnzG0pxcf6PGUVLlt77aLgMWD95yfm/dh9PFV+cf2L6MfybbZ8t83vWuxHJxfOn
GcZuE4+qAq8U/gQ+b9h4l9ir5hPw+Vi3ZfKUvdrsX5OP58YLlx4vfbYFHiP2qPm/wbSnIx/PcHy48HvtvZcA
Bjs3Z4lBR3qC3I2AYq/xWz5f2Qu/FKvl/yXxx9Zcr+X2lrjoTx+WVC7lrita18f8AIFxl8R+6P4o1EL6ldU4N636
gVy+4P/8AV+xHw/7pH4v9TtWNOFM65WuSlHiun2SzHp8ilV8uWvXWgM2N+IXlGHXbJ2KF3Bp9Vrubq
sfy8iy3lvn6a7ELcPlb5tVjrm++vUCEcW3z4WWXKTj6a9DYZqcRwt822yVk1236GkDFe1j50LX0jNal/wB/o
S8Pi5Ky+XecunwLsmhZFXBvi09p6J1VqquMF2itAZIWX5dk+Fnlwi9dFtshGEoeJ1xlY7Hp9X8GWzwn5jnT
dKvl3SFeD5d0bVa2132u4EX+Kx+X9hX+K2fL+yL3j7y1fy7LXHRHIxPNsVkJuua9UBV4l3p+YhnKTzKdS47
Wk/Z7JvAcpKU75Skn3aNGRRDIhxl012a9AKJ42RKDU8lOL77RpordVMYN70u5meFZJKM8iUoe2jXCKhF
Rj2S0gMfiHS3HlL7Kl1/sbJNKLb7JdSN1Ubq3Cfb9DN9BsceDyJOv20Bzwvfkz9uXQeGfYs+Y111xqgoWk
ivGx/o8ZLly5PfbQFxip/FLvl/wbSmGPwyp3ct8lrWu3b/AABm8RT86h8uPXv7dicsbIlFqWTuLXXoaL6YX1
8J/wAmvQz/AEKxx4SyZOHtoBOcsLCik1KW9J+nucnTf5Mp2ZL+y20l0L541c8dU9VFdn7FCwJtcbMiUoL0
Ajj/AIXP4SL8D7nX/P8AVivF8vGnTz3y3112LMeryaY175a9daAsMVP4pd8v+DaUwx+GVO7lvkta127f4A
plTG26VmNdxsX2kjtN90MhUX8W2ukkSsw92uymx1yffXZijE8u3zbLHZP0b9ANJ5lULJZd6hb5ctt9u/U9
Mz34itsVkJuuz3QFUsS2U4SsvUuL2toQ6eKz5esen9ETrw35isutlY49l7E8nFV7UlJwnHtJAV+J/dl8y/ch4j9
2q+K/QzcecKVOy6Vj3peyN1lMb8dQl06Jp+wHb3/APHsf/1f6GbFt8nw52a3rf6nfoM5R4TyJOC7LRdVj
RhjOmT5J7660BTVDIvrVksjgpdUooj4Z/8Am67+t39zqwJpcFkSVf8AxRdi430Zz1PkemuwFPhf8CfzfsbS
nFx/o8HHly299zZfLw77d/zf5GL0z8hS+03tfDZOOHKGQ7asjsSbXlqip/FLvl/wbSmGPwyp3ct8lrWu3b/AA
DTXtVx330tmeGG3Yp3wUvgr6Sukn3eLxpwUvRNNKP1lFvl67A19yuWVTxGFmuoOXaMRkc5G1dTZgmvl2
Ajj/AIXP4SL8D7nX/P8AVivF8vGnTz3y3112LMeryaY175a9daAsMVP4pd8v+DaUwx+GVO7lvkta127f4A

VV+bNNONVR1hHrrTfuBK7+DP5WZcG2urFXOajtvuzVd/Bn8rMeBjVTp5zipNvXX0A2xshKHKMk4+6Yh
ONi3CSkvdGOmKo8QlVH7E1vQw/9DJtofbugNnmQ8zhyXP231JGLCXm5F177N6X/f6G0DDfZluE5JKuEf
6s04jcsattttruzmb90s+Aw/utfwAuAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABmz6ZWVRlDrKD2kVTz4zpcYw
l5slrWvU3DXUDNjxeLh7mntfWaRdTarqozSaT9GTAELv4Fnyv9DBh5ldFHCalve+iPSAGerNqusUIqW37o
qwvvWT837s2gDCsp1uVeXDfs+PRnMNc8udlUHGrWupv1vuAPPT+hZE/Mhuub2pa7F1WTG65Rqq3H1
k1rRqHYAYl+Kv5f2NoAxU/il3y/wCB4h/Ex/m/wbQBjzIThfXkQjy49JJexyed5kVHHjJ2P3XY2gDDnctY/N7l
vr8ehrvr82mcPddCYA8+jJhRX5V9bUo/l3L6rJZFdvGvgmtRfuaWk+4A83FyYY0HXZXJT36LuSwpuWba5x
4ykt6PQ0AMTyZVWThlR3H/AGtLoQxkrM7zKIOFSXXoeh3AHny3iZU5zhyqs6712LYZUbbYxpq5Lf1m1rR
rCWuwAxZ7dd9FrTcYvro2juBgz7I24lc4705ev8zZf/As+V/oTAGHHq87w1wXd71/UjTlQprVd9bU49O3c
9AaT7gUYtrtUpeVwjv6r9y8ADzMqqSynXDpG5pm3JrbxJV1rrrSRcAPPx8yFVKrVcvMXTSXdkvDG+VyktS
2m1/U3a67AGLw1bhav/sVYuQsRTqujJNPfRHpDQHn41jt8RlNxcdx6J+wlvEypznDlVZ13rsegAMkMqNts
Y01clv6za1o1hLXYAYbuWLmO/i5VyWpa9CORkfS4qmiLe31bR6AS12Ay5keHh7gv9qS/ui7G+7VfKiwAR
s/hy+DM3hn3b/2ZrAGC1PFy3c4cq59+nYmsuFlkYU1c9vq2taRsCWuwGDM/wBLMhdODlWkV5uSsipK
EJcE9uTXqemAMPiD3VS12bNwAELv4M/lZgw8tUU8bIy4t7jJLuekO4GHG5ZGXLIcXGCWo7OeJRcJwuj0
bTi3/wB/mbzC43Zd0FZX5dcHt79QNOJX5WPCPrrb+JaABTm/dLPgMP7rX8C4AVfSF9J8ji963v0LQAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACm/JjRJ
JwnLa39VFX/kIf/4rf6AawV498ciDnFNLeupYAAK77448OU9vb1pAWAjXNWVxnHtJbI5Fyoqc5Lfol7gWA
ox7rbNysrUIa2nsthONkdwkpL3QEgAAAAAHJzjCPKbSXuzkppVOceq1tfmBIFWLc76VNrXXWi0ACqrIjbZ
ZBJpwenstAAAACqvIjZdOtJpw7loAAAAAAAAAjCyE98JJ6769AJAEXZBTUHJKT7ICQKMnIdDrSjvm9F4
AAAAAAAKKsh2ZNlXHSh6+4F4AAAi7IRmoOSUn2XuSAAAAA2optvSXVs5CcZx5Qaa90B0FGNkO92Jx1w
ei8AAAAIzshXrnJR322SAAAAAAAAAAjbPy6pT1vit6I49ruojY1re+n8wLAAAAAAAAAAAAAAAAAAAAAA
AAACMbITbUZJuPfXoBIAAACrJudFLmlvXoBaCNcudcZ61ySZIAAAAAAAFDyGsxUcejW9gXgq+kR+k+Tp8
tb36FoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMufa40quP2rHp
Gow0//ACc6VneFfSIGuitU0xgvRdSYAAw5CWTnRp/2QW5f9/obZzUISm+0Vs83FeSnO6upT5vu2Bf4fJx
VlEu8H0K/E/M4rfHy9rXvvRHnbVmxtugoKfR6Zd4p93j8/wCzA6/pfkvfl9v7Ffhvm8Frj5W3v37G2X8J/KZ
vDPuz+ZgLMm2y91Y8U3H7UmdqyLY3qnIilJ/Za9TNi12yttVdvCSfVa7l/wBFtd1c7L1Jxe0tAcuyro5Uqa4
KXt/Q00eb5f8Ara579DNX+K2fL+yNoGLxHzfKlrj5XTfvvZKjzfocvM48fL+rrv2JeIfdJ/FfqSh9wX/6v2AyYc8
h08KYxST6ykacTInZOddsUpw9jnhv3VfFkKPxO75f8AW4187b7oS1qD0tfEr+kX32zjjxioxetyOYX3rJ+b92
FRznK3Ev47fVemwLMbInO2VN0Upx69PU0mXHvs8+VF6XNLe16moDPRfKzJtretQ7aKY5WRZbZXXCLaf
RvskdxPv2R8f3GB/Hyfm/dgSpyLo5KpvS210aO5GRYrlTTFObW236EMj8So+H+Sd9Vd1/1LeF0V6ARWR
fTdCGRGLU+ilEsysl1SjXXHlZLsvYq86/GthG9xnCT0pLuV5EZPxKKjPg2ujAslkZOO4u+MXBvTcfQnmZM6H
U4JNS3v8APsQsxb5w42ZCcfzRHLi4TxIt7aet/wBALYSzJRk3GEX6JmfB87zZ8eOuS57/AGPSMXh327/m/
wAgbTzLfpH02vlw8zXTXb1PTMV/4pT8v+QI57nGFDmk5J7eiVl2ZCHmuEFD/j6pDxH7VHzf4NGX91s+Vg
FkR+jK99FrejPG7Mtj5kIQUfRPuyuaf/iYa9+v9WWU05EqYOGTqOlpa7AX4t6yKuWtST00WzkoQcpdktso
xMd0c9zU+T9CWam8SzXsBRC/LvTnVCCh6J+pHBk55l0pR4ya6r2NOE08SvXsUY3XxC8CU8i6y6VePBaj0
cpHaci1ZHkXxXJraa9SEbLsq6yNdnlwg9dttlfCUPEqoysdj13fp3Ajf9I+mVcuHP8A267Gy2+dGOpWRUrH
01HtspyfxCgtzb5UVx4JcpPS36AVTtzK4ebKMOK7x9jVTYraozXaSMl9Nyx5ysyG+nVa6F2B9zr/AJ/qwGb
5vky8vjx4vlv2/Iq8O83y4/Z8rr8dmnI+7W/I/wBCrw77pH4v9QMuNO5WWwpgm3Lbb7I0UZFqyPIvS5N
bTRHw77V/zf5Fv4pV8v8AkCy/IsVypognPu2+yILIvpuhDIUXGfROIttttynRVJQUVty0U5Vc67Kedzsbl2a7
Ad8R87ceXHhy+rrubqfN4PzuPLf+32Mvif2K/mNoFGZbZTSp166PrsZGR5eN5sdbeuO/zLL6/NonD3XQ8
6mf0j6PT/wbcv5AbcfI8zF82WtpPlr8iFGTJ4srrtJJ9NIy2T+jrIo/5NOP8y/Krdfh0YL/AG62AjdmWx8yEIKP
on3Zfi3rIq5a1JPTRRTTkSpg4ZOo6WlrsW4mO6Oe5qfJ+gE8r7tb8rMeLPJljxjTCKjHf1pevU2ZX3a35WQ
wPudf8/1YDDyJXc42RSnB6eit5F91ko48IqMejlI5h/fMn4/ucrnflyk4WeXCL0tLbAsx8ix3ypuilNLaa9TuNf
Od1tVmuUO2vVFFUXDxJRlY5tR7v4E8n/Qza7u0ZfVl/wB/72AslfN5saYa4pbkyNmTbZe6seKbj9qTOYCc

5W3vvN6XwKMWu2Vtqrt4ST6rXcDTVkWxvVORFKT+y16mox/RbXdXOy9ScXtLRsAy3ZFrv8miCcl3lLsch
kXV3xqyFH63aUTkrbr8mVNU1XGHd62ym6uVeXjqdrsbku/p1A1ZWS6nGFceVkuyKp35WPqd0Yyg314
+gs6eK1uXZx6f3NVsq41t264eu0BVk5Sqri4LlKf2UVStzKY+ZZCDj6pehzJcfpWNJa4PWvY2WOKg3PXHXX
YFVuVCvHVq68uyKXbmQh5soQ493H1SIZ/GVdEq2lDb00uha8fJlFp5O017AStyf/h+fXrfToytW5llashCCj
rs+7I3VeT4dKHJS6ppr4mvG+7VfIv0AppyJ5FDdairYvTT7GfD8/z7ePD7S57+L7FuB1yMhx+y5dP6sYH8fJ
+b92BZfkzVypoipT9W+yIRyLqbYwyIx4y6KUSlQsl4haoWcJe+u6LLMS6zXmZCen02gLcrJlVKNdUeVkuy9j
NlyyVQ43Ri4t94+hbLp4rHl6x6f0LPEfukvigLcf7vV8i/QZFyoqc319EvdjH+71fIv0Mviifl1+3IDrtzFX5rhDj3
4+ui9ZMXi+fp612/Mq8jJlH7ztNexyKeFhyakpvfT2A4rcycPMjCCj3UfVl1GR52O7Nakt7X5lUKsi2tWTyXH
a3pLsQ8O+6W/F/oByrJy74/6cI9O8ib/FY/L+xLwz7s/mZF/isfl/YC1Xy+nOnpx47/ADI35NnneTRFOfq32R
Bfir+X9iqELJZ1yhZ5ctt9u6AvhkXVXRryIx1PtJHMnKtqyVXXFS2ui/M5PEunKLsyE9PptCz8Vr+X9mBox3c4
Pz0lLfTRaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABGyLnXKMZcW1rfsQ
xqFj1cE99dt+5aAI2wc65RUnFta2vQ5RW6qowcnJr1ZMAV5FTvqdalx33etkqoKuuMF2itEgBVk0LIr4N8
WntPWyN+M76IVuzTjr62u5eAGumjLRiSot3G1+XvfHRqAGa7EU7fNrm65+rXqKsRxtVltsrJLtv0NIApjj6y
5X8vtLXHRcABC6pXVSrl0TKqMaddc4Stck1pdOxoAFWNT5FXDly673rRyGPwyZ3ct8lrWuxcAKacfyrbZ8t
+Y9612K5YTjNyotde+69DUAM+PiqmbslNzsfqQABTVj+XfZZy3z3z9NdhJ4/kzsly5c3ct2LgTddi/3I0ADLDDbsU7rXY49l6FmRjRyEttxku0kXADH9CnNpXZEpnpx+N5sqmpcVW961vf
b/BeABljhyhkOyu1xi3tx13NQAGfJxfOnGcZuE4E4+poAGezFdsalKzrKzrB7b13LrYeZVKG9comzeWYDGtnKSm2mtaaNMW229tv1LWk00+qYAGNYMoaoaeALkseiNFZKSm2mtaaNMW2tv1LWk00+qYAGNYMoaoaeALkseiNFZKS

kXkUqPJ2R0/XYFgIwshYtwkpL8hZbCv7c1H4sCQIwthYm4TUku+mZpZiWXGCnDymurA1gz5KjZCDV/lre
00+5obSW29IACuORTKXFWRb+JOUlCLlJpJerA6CuWRTFJuyK31XUnGSnFSi00/VAdAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAGXxHy/I1PfLf1de5nu+mfR9Wr6n+7XfRfn1zfl2wjy8t7aI2Z9c6
nGEZOclrWgNON5fkQ8r7GuhYZsdPFwtzT2tyaRdTarqozSaT9GAu/gWfK/0PPqx/OwE4/bjJtHoXfwLPlf6
FHhv3VfMwKbcjz/D5b+3FpSJZFjr8OqUXpyil/Yq8Qodc3ZD7E/tL8y+2l3eH1qP2lFNL36ATrwqVSoygm9
dX6lWGm4340ntR2kdh4hGNerIyVi6Na7slgVTXO2xalY96Aqx7nXgWp9JQbS/mMDdFzrn/vipIhfU/p/lL7
NklJr/AL/Mu8Si4qFsOjW4/wBQO4C8yy69/wC6Wka7JcK5S9k2VYlfl40I+utstlHlFxfZrQGHBx4W1yttXO
Um+5yuqNXifGHRa3r+Qx8j6GpU3RktPaaQpsdviKnxcU49E/YCdf4rZ8v7Iru4150pZMHKD+y/Qsr/ABWz
5f2R2eTOm6UciO639lpAKKqZZCtx7Ekl1gkVyqjd4pOM1tJJ6/kjlXG3OjPHg4wX2nrSE7fJ8Ssm4uS0t69F
pASzqI0RhdUuElL0O575fR37vf6EMi76Y400p63ttos8QWnjpekv8AWZsISjF22cIJ9V7mTJniyq/wBGLUk+
jS0XeIJxtqtcXKuPdFeVlxvocKoS10cm12AnlqUsajIX2opNs7m2ebVTCHe1pl9cVbhQh6OCX9jHgQlPI3Pt
UtL8v+9QPShFQgorslpGLxOuCqU1FcnJJv8AkzcZfEouWNtLfGW2B26uFeFZwio7j10V4eLVLHjOcVJy9yU
7434NjjvpHT2WYX3Sv4fuBRgvypZMf9sHtf3KMe2hudmT9acn6raL8NcsjKT7OWv7shTYsNyqvh03tS0A
xbIRznGlvy5rt+Z6Jmx8hXWvhVqtL7T6dTSB5+NXHKyLbLVySekjmRTCnNocFpSktr+Z2M3hZFisi3XN7TR
G2/z8uiUYtQUkk369QNGZCpzhO+zUF/s9zJk2UOMZY8XGUX3S0XZn+lmQunByrSK83JWRUlCEuCe3Jr1
Au8RhHyPM4rm2upfXVGOPqGoOUerXwKs5OeEnHrrT/kI2rJw5xq3yUdNAUweFVFwl/qy9ZaJ+GS/ixTf
FNNb/AJkMbLropVbrl5i9Eu7J+HNu2/ktSbTa/qBtnLjBy9lsw4VMb1K65c5Sfqb2uSafZnn0WvCcqrovjvcZ
JdwOygsXOr8vpCzo0MyqCy6NRX15fW/PqIOWZlxsUWqq+zfqSzPveN837oDZCEa48YRUV7Ioza4Sx5yc
U5KPR+xoK8mLlj2JLbcQKcKuCxYzUUpOL2yjw7HrsrlOceT3T3pJ9i3BvjKlU9ecYvY8L+7y+d/ogI0RVXiVkIdIu
Pb+hVK2qeZOWQ24xeoouj+Kz+X9kQlvEypznDlVZ13rsBXK2mOVVPH6bepLWkeoZIZUbbYxpq5Lf1m1r
RrAwSisnxGUJ9YVrsR8RorrrjOEVF710J3OWLm+fxbrmtPXoVZ2VG+uMYRlxT25NAXel/Zp+Y05P3a35W
Z/EISlRCUVvg9shdnQtolCuEnKUevTsAhY6vClKPR9Uv6ldMsONKVi5Ta6vTLaand4YoLu96/qcpy66q1Xd
W1OC127gcw57x8itPcYpuP9y/w77oviztMpX02br8tS2o/mZsXKjjVuq2MlKL7JAWUfiV3w/wZ67qZ3zsy
Nvb+qtbRdhTc862Ti4tx7P+RGD+hXTjbDdcnuMtAcrsqjnQdD1CXRrR6RlpyVdco1VfUXeTWtGoDFR+J3f
L/geIfxcf5v8ABGyf0XPlZOL4TWtr/v5FeRkLIupcYyUFLu/VgW+IRfm1znFyqXdI55eNkOPkTVU0/YuybraL
Yy48qfXS6mXInXkSiset+ZvuloC7P/jY3zfuiOfBWZVEeG/2AlZf33H+P7gczcaqGK5Qu0l6jw
oy7Zf+Pr69ZpJ/0LvEPuc/5fqVW0yu8Or4rcopPXv0AsWDT5PBx66+167K/C/4E/m/YR8Qi60uEnb20l6jw
v8AgT+b9gNk03BpPi2u/sefB4dScZPzer1s3XxlOicY93FpGHGy68ery5bvdOO+hHj0mbj
B4fJvIv5LjJ9WvbqbwPPw6o5MrLrVybekmJ0wp8RqUFpro5TZ9BsnXbF8G9xaHnefn0zUXGPZb9QLLPx
Wv5f2ZDLSjmKd0XrJ2fitfy/syR/Njul9ml2AjXXRbfzCzHsRbr5cwPvz4BhRkrwlp2ZZ9f4f4Q
40HHi9yetIlkzdfiMJqLlqPZfzAlm41cKPMrwlBrsczbHPBqb7yab/ocycpZMFTRk12cvxwYXxXXy9f4A
2paikvQxXfV8Tqa9V/k11TU6oyXqtmSX+p4pHXauPX/AL/MDaYPEK4K2lqK3K3X8+xvMXiH26Pm/wAAS
za4V4clCKitp6Rfj/d6vkX6EM2t2Ys4xW330UVZ0Y0Rhwk7EuOku4HPD/u13xf6HPD8euypznzHk96W/Q7
4f92u+L/Qs8M+7f+zAhjRVXiFtcekdb1/T/JzGrjlX222rkk9JMlV+K2/L/gjCbwr7FZF+t+JMlNdkKd
Rj2lH3KZ01rxKuChHi49Vrp6l7teRZkXx8lNVL7Ta7leVLyc+u2Wp0SS0lLohnt2W1UJ6UntnM
+asqpnHs5bRPOrnyrurW3XM+qAldhUulqMFFFpdGUxsdnhc+T2Oys1NcZOysrXY46XT4ZOM
vtPqwJYmLTLHjKceTZKceTkurZzw36ruhvpGXQV/utfwKPD/4uR83+QNoKsfIV/PUWuL11LLQAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA0gAAAAAAAAAAGkAAAAAAAAAAAAAAAGWFc14jOxxfBx6P+S
NQAa0AAA1rsABTk221pKqtzb9fY5h0umr6/wBuT3IvAAAAEtdgAAHcAAAAHcAAAAACWuwADXXYAAD
v3AAAAAAAGkAAAAAJa7AAAZfEKp2URjXHbUt6XwNQALshrQAAaTAADQAAAAOwAAdwAAGtdgAAAA
ELlY6mqmlP0bJgDFK3LcXDyEpta5bL8SjyKVFvbb2y4ABrrsAAAAHcAAAAA1rsZZ1zfiNdii+Cj1f8magA0cl
FTi4yW01pnQBjqduLGdfBzUVuDXr+RZiUOuMp2dbJvcvyNAAAAANdQAAAAAAAAGt9wAAAAaW9gAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAhbbCmHOb0jPHxCttcozin2bXQj4h1ux1L7Dl1/sX5kYyxZqWtJbXxAuT2toFGD
JyxIb9OheByT4xbfotkKLo3184ppb11JWfwp/BmTAk44U5Lum3/AGA2gw1ZGVkQ/wBOEFrvJluJkTsnOu
2KU4ewGkirIOxwUlyXdHZ8uD4a5a6b9zzY/SPp89cPN119uyA2W5Dhk11cdfqfqXnn5kpKSPKPKS

WX5dCU7YQ4N9UvQDVkXRor5yTa3roTi+UU16rZRl3urHVlentrudyMnyMeM9blLWkBeDFO7Mqh5k4
wcfVL0Ndc1ZXGa7SWwOykoRcn2S2zNRlW3zTjTqveuTZLO83yHw48dPlsr8P87yob4+V1179wNgM+Hf
O+M3PXR6WhK+azo09OLW/zA0Ax25V0cuVNcFLp0/oc+kZFN0I3xi4zetr0A2gz5WS6eMYR5WS7Iqndl
UJTtjCUPXXoBrnONceU5KK92JTUa3Pukt9DDnysnUpLj5L017l9croYs5WcdxjuOvgBdRar6lOKaT9yZRj38
sTzrNLW96KYXZd6c6owjD036gbQZ8XJd3KE48bI90Uxysiy2yuuEW0+jfZIDcDJTkXRyVTeltro0awKpZEY5
EadPlJb36EcrJWPFfV5Sl2RyV81nQpWuLW/wA/Uy5nnfSq+XD7X1P6+oG2ids4t21qD9CyUlCLlJpJerI0+Z
w/1ePL/wCvYz5/m+VLjx8vX1t9wNSkpQ5Re1raZViXvIqc3Hjp6KsLzfIXPj5fH6uu5nwp3+U4URj9rblID0w
ZcbIsldKm6KU49egtybJ3unHim4/ak+yA1AyVZFsL1TkRScvsyRrAAzU3zeVZTZrp1jr2O23zWXXTXrr1lv2A
0Ay25Fsr3VRBbXeUuxyvIuhkKnIjH63aUQLJ5Dhlwp47Ulve/j/gvMV34pT8v+S3z5/TlT0463+YGgGfNvnR
VGUNbctdSWTkKitPW5S6JAXAxSuzKoeZZCDj6pd0a65qyuM49mtgSBXkXKipza36Je7M3m5rh5nCGu/H
10BtMk8ux3Sroq5uPdtlld8sjGc6UlPtqXoZMPz/ADreHDfL6+/59gPSW2lvoyjKyVjxX1eUpdkXnm5nnfSq
+XD7X1P6+oG2ids4t21qD9C0zXW3U4vOfDnv07HMe3JtnGcoRjU/6gagYvpN99kljRioR/3S9SdGTPzvJvi
oz9GuzA1Ax3ZVsMp1VwUunRfmceRkUWQ8+MXCT109ANc7IVrc5KK7bZI8/wAS83j14+VyWvfejXj+bw
/1uO/Tj7ATtmq65TabUVvocpsVtUZpNJ+5zIm66JzjraXqVwyNYSvmuuuy+IGgGFW5kq/NUYcdbS/I041y
vpU0tPs0BaDk5quDnLsltmON2XcnZVCCh6J+oG0FGNk+dXJyXGUPtIphkZORKTojFQT7y9QNoMlGTbPK
dVkVHS6onTfOeVbW9cY9gNAM+TfOq6mMdam9Pf8AIZORKuUa6ocrJdt9kBoKcu949Smo8tvRRPIycdx
ldGDg3p8fQ74m940Wv+a/RgbE9pMjOca48pyUV7s7H7K+Bh8S83g98fK2te+wN6e1tAz0zsroc7+KiltcfY
qhdl3pzqjCMPTfqBtBnxcl3coTjxsj3RoAELb66l9eaX5epMpni1SslZKPKT9+wEqL43xcoJ6T11LDH4X93l8/
7I2ARnZCGuckt9tkMm10UuxLevQx5/neZDlx48vqa/ctyvM+gy87jy2vs/EDVVLnVCetckmSMFVmXKiLqh
FRjFJb7vRoxcjz6eckk10YF4MUcjIyJSePGKgnrcvUtxciVkpV2xUbI90vUDQDDXlZFs5wrhFtPu+yROjItWR5
F6XJraaA1lX0iP0nydPlre/QtM3nz+nOnpx47AvdkFYoOS5PsiR5k/pH06G+Hm66e3qelDlwXPXLXXQHQZ
czInRKvgk+Te179iFl2ZVHzJwhx9V7AbQY1fk3x50wjGHpy7ssw8h3wlzWpxemBoBjeRdfZKONGPGPeUie
PkTlc6boqM11TXqBpBknkXWXSrx4LUejII7TkWrI8i+K5NbTXqBqBjvyra8ryoQUtrovzIzyMnHlF3xi4Sfp6
AbgUZeQ6IR4rlOT0kUzuy6IqdsYOHql6AbSjLyHj1qSjy29dy6ElOEZLs1tGPxT+BD5v2A0ZGRGhRck3yeuh
aZ8y+VEYOGur11O5eQ6ILiuU5PSQF5yc4wjym0l7sxzuy6IqdsYOHql6HM6Vk8flHj5Mkn+YG5NSSae0+q
ZyUuMXJ9kt9CjC83yo+Zx48Vx13LrJONcpLuk2BGi5X184ppb11LDPj5Dliu2zXTfYqhdl3Q8yuMFH0T9QN
oKcTI+kVttaknpouAAAAAAAAAAAAAAAAAqyKI5FfGXT2fsY8miVdD83IlJLpGPuz0JxcoSipcW1pP2M0M
GKmp2zla1/wAuwFmFBwxYJ99bLiNsHOuUVJxbWtr0OUVuqqMHJya9WB2z+FP4Mx4X4fZ/7fobZLlFr3
WimnG8rHlVz3y310BDw37qviyFH4nd8v8Ag0Y1PkVcOXLrvetHIY/DJndy3yWta7AXGKH4tZ8v7I2ma/E
dl3m12OuWtPoBXk/iNHwLPEfukviiU8bnfXa59YLWtdyeRT59Lr5cd+utgZcv8Oq+Ef0Lb41Tx4Qtlx2lxf5k
rsbzcaNPPXHXXXsSux4XVKEvTs16AZpxysWHJWKyEe6l3NdNitqjNLW12MzwrJLjPJk4e2jXCCrgoRWklp
AV5X3Wz5WQwPudf8/1ZdbDzKpQ3rktbI0VeTTGvfLXroDN4Z9i1f8A2EvxaHy/syUsKStlOm518u60dpw
vKvjb5jk132u7AhX+K2fL+yHiXan5i+OPrLlfy+0tcdDJx/P4fW48XvtsCi/p4nS5dmtL+5rtcI1t2a4euyGRjxyI
pS2muzXoU/QpTaV18pxXp2A5nuLwouGuO1rRdZ9yl/8Arf6Er6I3U+X9lemvQrqx5xonXO1y5LS6dgMyT
fhD17/ubMRp4tev+IooVVHlN8l1307lH0KcG1TfKEX6Acqe/FLHHso9f7DA/j5PzfuzRj48ceLUduT7t+pzHx
/JnZLly5vfbsBRkfiVHw/ybSmzH55MLuWuC1rXcuAxWfitfy/sxm/e8b5v3RfLH5Zcb+X2VrjoXY/m3V2ctc
HvWu4FxRm/dLPgXkbIKyuUJdpLQFWL9zh8pV4X93l87/RFmNjTp2na5R1pLXYli0fR63Dly2971oCiP4rP5
f2RTjwtlkXKFvlyUuvTv1Nqx9Zbv5d1rjojfiKyzzK5uufuvUCp4lsra5WXqTi9robTNXiNWqy22Vko9vyNIGP
M/wBLIpvXbfGX/f6jDXm5F177b4x/7/Q0X1K+pwb1v19hRUqKlBPevX3Azuy7IyZ11TVcYdG9bbKrK5V5t
Cna7Hv19C+3DcrXbVa65PvoisDVkLPOk5p7ba3sDl34pT8v+Q+niq/OP7F88fnlQu5a4rWtd+/+SOTi+dO
M4zcJx9UBV4p/Ah837EfEE/NofLj+ft2JT8PlYt2XylL3a7Gm6mF9fCf8mvQDPLGyJRalk7i116F+NV5NEYcl
LXqij6FY48JZMnD20aq4RrgoR7IDJ4n9itv7Kl1Nu1re+hG2uNtbhNbTMv0GzjwWRLy/bQHPDerua+y5dB
4f/GyPm/yaqao01qEF0X9yGPj+ROyXLlze+2tAXGLN+943zfujaU3Y/m3V2ctcHvWu4FfiP3V/FFsU3ipLu4
fsMmnz6nDlx673rZZCPCEY73paAy+GNfR2l3UupHKe8+hR+0u/w2TnhvzHOm11t90uxLHxFVN2Sk52P/
cwKl+Kv5f2Hin8KHzF6x//AJbv5d1rjoZWP9IhGPLjp77bAp8U+7R+dfozXH7K+BDIpV9Tg3r1T9jmPVOqD
jOxz9vyA5mfdbPgVU+X/wCOirXqDWn/AFNF1fm1ShvXJa2QjjxWMqZvkku/YChU5FMOVFynDW1F+xfi
X/SKeetNPTRT9CsUeEcmSh7aNNFMaK1CHb3fqBVnpvDnr8v1KMeq+dEHDI1HXRa7G9pSTTW0+jRj+hT
g35N8oRfoByONKqvIk7FOUovel6lnhzTxFrum9lmPjxoi9NylLu36lLwpRm3Rc60+60BGP4rP5f2QxvxC9fk

Qx6/K8RlDk5fV6t+pfdiOd3m12Oufrr1ArzvvON837onk3WefCipqMpLbkyKwH5kLJXSlKL2213LcnFV7jJS
cJx7NAZM2qyulOd7nt9mtFniH3Kv4r9Gdl4fKxf6l8pS9G12Lr8bzqIVOenHXXXfoBdH7K+Bl8T+6/wDsjWl
pJFeRSr6nBvW+z9gKctN+HvXsv2LcRp4tev8Aicx6JV1yhZZ5ifRJrsin6FODapvlCL9AOVPfiljj2Uev9jZJcotb
a2tbXoV4+PHHi1Hbk+7fqWgZo4cozUvPsenvTfc0y+ywH1QGPwv7vL5/2RsKcXH+j1uHLlt73rRcBi8R+1R
8xZ4h90l8V+pPKx1kQS5cWntMi8ac8Z1Ttcm3vk0BZj/dqvkX6GLETeHkJd+v6G+uHCuMN74pLZXi4/0e
Mly5cnvtoDJh1XToTrv4rfbXYvoxp15Dtnapya0+hyWE4zcqLXXvuvQsx8ZUyc5Tc5y7yYFPh32r/m/yLfxSr
5f8l+Pj+Q5vly5vfbWhPH5ZUL+WuK1rXx/yBcYl+Kv5f2NpSsf/AOW7+Xda46Aos/Favl/Zm0z5OK7rI2Qm
4Tj02XwTjCKlLk0ur9wMef8Ax8b5v3RfmfdLPgMjH86dcuXHg99u5O6vzapQ3rku4EML7pX8DPhbduVrv
v8Aya6a/KqjDe+K7kKMfybLJcuXN71rsBiwq7Z1y8u7hp9Vo0V4tiyY22XKUku2vQ7Zh/6rspsdcn316kqM
Xy7PMssdk+236AVRsuyrrI12eXCD1222V8JQ8SqjKx2PXd+ncuswn5rsptlW5d0hDB4XQs81uS6va7gRf4r
H5f2O+Kfd4/N+zLnj7y1fy7LXHQysf6RWocuOnvetgRyYVWxhXZPjJ/ZZTN5WJHk5q2td99zTfjwvglLaa7N
ehQ8KyelbkSlBemgNVc1ZXGa7SWzJ4p/Ah837GyMVGKjFaSWkVZWP9Igo8uOnvtsCjxP7NXzFuXCq1Rr
snxk39VksnH+kKK5ceL322dyMeGRFKW012a9AM03lYkeTmra1333J5k1ZgOa7S0zjwrJ6VuRKUF6aNFl
MZ0OrtHWlr0AY33ar5F+h27+DP5WV41E6E1K1zXotdi6ceUJR3ra0BkwnBYD8x6htpnIY91ceWLfuD6q
Mi+nGVeO6ZPmnvfTRT9CsgnGvIlGHtrsBbh5DvhLkkpRenovKseiOPDjHbb6tv1LQAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAArVMFe7uvNrXcsAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFc76oS4ysin7bOZVjqx5zXdLoYMeeJGr/WXKb7trYH
pqSceSaa77IPIpUeTsjp/mY8Ka/164tuGm47Hh+PVZS5zipPeuoG1W1uHNTjxXrsQvqseoTi37JmCvHg8+
dT35cfrcd/99yWZVCi2mdceL5ddAbpWQjJKUkm+yb7kVkUuXFWRb+Jl8Qip30RfaT1/dE8nDqWPJwgou
K2mgLcm9U1yfJKetpP1I4+TCylSnOKkluS9in6t3hvmTSc4xaTZPDprliJ8VucWm/UDVGUZxUotNP1RyNkJ
TcYyTku632MeDZ5VVtc+9TbJeHQflztl3mwNgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIX1+bTOHuuhioyY
UV+VfW1KP5dz0A0n3AzU2u6uyXlcI6+q/cj4Z91/9mawBiq/Fbfl/wADxHvT8xtAGHxBuN9EkttPaXv1Qvz
Y2VOuqMnOXTTXYsy65zyKJRi2oy6v26o1a6gZVTKHh7r1uXF9PzOeH3xlVGnrzinv+prOS+rGTjHb1vXuB
5ufCUMjcP8A8q18f+9DbKUcXGT02oJLoU1wtyMmNt0OEYdk/c2ARrmrK4zSaUlvqSAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIWW11Lc5qPxI0ZEL3LhvUfV
+oFoI2WRqg5zekjMvEK9rcJqL/3NAawE1JJp7T6pgADkpKMHJ9ktmVeI1vtXY/5AawZ6cyFtigoTTfq0doy
HbdbBx1wet77gXgAAAAABV9Ij9J8nT5a3v0AtAAAFVuRGqyuDTbm9LROdkIa5yS322BIEL7VTU5yTaXsS
hJThGa2lJb6gdAI2S4VTnrfFNgSBRVkc8XzpR1rb0iym1XVKcU0n7gTAAAAAAAAAAAAAGW7Lkr/Jpr5yXc0w
cnCLmtSa6r2A6AAAAAAoxsh3uxOOuD0XgAAAAAAAAAADHHLstsapq5QT05NgbAAAAAAAAAAAAABRlZKx
4r6vKUuyJUTtnFu2tQfoBaAAAAAAEZ2QrW5yUV+YEgA2optvSXVsADkJxnHlBpr3R0AAAAAAAAAAAAAA
AAAAAAAAAAoxMh5Fbk48dPReAAAAAqvyI0OHJN8nroBaAAAOSkoRcpNJL1YjJSipRe0+zA6ARjZCbajJN
x769AJAoWQ/pjo49Et7LwAAAAAAADknxi37LZVi3O+rm46660BcAAAIxshOUoxkm490vQkAAAAEZ2Qr
W5yUV+Zy+zyqZWJb16ATBCmzzaYza1yW9EwAAAAj5kPM4cly9iQAAAAAAAAAAAAAAAAAAAAAAAAAAF
U8aqy3zJx5PWuvYz+HLU70u3L/ACbTF4f9u/5v8gPEPrTorfaUuv8AY05EFLHnHS1xejN4h9W3Hm+yl1/s
ar5KNFjf/FgU+HycsSO/RtGkzeHLWJH822aQIXfwLPlf6GTAvqrxlGc4xe30bNd38Cz5X+hkwKKrMflOCk9v
qwNcLa7G1Calr2ZlwvveT837s1V011NuEFHffRlwvvWT837sDv0i++2cceMVGL1uRPGyJztlTdKcevT1K1
RznK3Ev47fVemyePfZ58qL0uaW9r1A5Zk22XurHim4t4ST6rXcv+i
2u6udl6k4vaWgNhnV8vpzp6ceO/zNBiX4q/l/YBPKuWVfFj4x+DxD+Jj/A

Df4Atyb51X0wjrU3p7+Jmz/ADvMhy48eX1NfuWZv3rG+b90PEftUfMBZbbdTiSnPhzT9OxKWR5eJG2S2
3FdF7sj4h90l8V+pGy94+BVKK3JxSX9AIu3MVfmuMOKW+ProtdquwZzS1uD2v5FU6b/ACZTsyX9ltpLoc
xvwyz4SAlizdfhzmu6TaLK7LrcOM4KPmP37dymj8Kl8JFuC1HCg29Jb2/5sCq67Lphzn5SRdVfJYruvSXr09i
iCedkc5fwYdl7lviKf0R67JoCEbsy2PmQhBR9E+7L8W9ZFXLWpJ6aKKaciVMHDJ1HS0tdi3Ex3Rz3NT5P0A
vnJQhKT7JbMcbcu5c64QjH0T7s1XWeVVKet8V2MtMMjIrVkr+Cl2UUBdiZDvhLktTi9NF0+XB8NctdNmL
w3pZf15dV19+5uA8yn6R9Lt4cPM/3b7GvJyXTxhCPK2XZFWN+IXi36vilbl2cen9wE7sqhKdsISh669C3IyH
DFVtWnvWtk8tpYtm/+JimmvCY79/3A0Y9uTbOM5QjGp/1ZqIU/wAGv5V+hMDF4d9q/wCb/I+k332SW
NGKhH/dL1I4Sbjkpd2/8lnhjX0dpd1LqB2jJn53k3xUZ+jXZncjJnG1U0xUrH7+hXlPefQo/aXf4bFXTxS1S7t
dP7AHkZGPKP0iMXB9Nx9CzMyJUQhKGnyY8RaWI0+7a0Z8vaxcbffp+gFk7cyMPNcIKK68fVI00Wq+pTX
TfdexKxpVyb7JPZl8LT+jPf8Ayev7AaMjzfL/ANLjv15exj8O87j9Xj5fL62+5vl9h/AyeF/dpfO/0QEZ5V7yZ01
QjJrt+RZZkTx8ePmJO6XRJEMf8Rv+AzNLMx5S+zv9wErcyqHmThBx9Uu6Lp5G8N3V+3qW2tKqbl209nn
1b/8AFWb9+n9gLYXZl1anCEEvz9TleXfeuFdcVNfab7I0Yf3Wv4FHh/8AFyPm/wAgSpyLVkeRfFba2mjWY
r/xOn5f8m0DzczzvpVfLh9r6n9fU0X3XUYqnPhz5a6dtEM373jfN+6JeJ/df/ZAdx7cmyanOMY1NfzILIvyJy
+jxioR/wB0vU0STeK0u7h0/oVeGtPF0u6b2AoyZu503xUbPTXZkbcq6OXKmuCl06f0OZD34lSo90uor/Fb
Pl/ZAPpGRTdCN8YuM3ra9CvxLzdLlx8vl9XXfsWeJdqfmHin8CHzfsBpo83g/O48t9OPsV5vm+TLy+PHi+
W/b8jQV5H3a35H+gGbw7zfLj9nyuvx2deTdfbKONGPGPeUiWDt4SS79SPhbXkTXqpdQJU5Niv8nIioyfZr
szUYs17zMdR+0nv+5tAx5GVbXk+VXFS2ui/Mn59lGPKeQly3pJepW/xWPy/sc8UT8uv25AdduYq/NcIce/
H10XrJi8Xz9PWu35lXkZMo/edpr2ORTwsOTUlN76ewHFbmTh5kYQUe6j6supyfOxpWJalFPa/MqhVkW
1qyeS47W9JdivB+5Xfz/QDtOTlXpeXCOk+sjeZfDfuq+LNQGa7It87yaIJyXdvsiEci6q+FeQotT6KURZbddlS
oqkoKK23rqU5Nc676FO12Ny9fTqgNGXkzotrjGKkpenqyuy/LpirLIQ4+qXody/vuP8f3LfEPuc/5fqBK3IjXj
q3umlpe+yiNuY4eZ5cHHvx9dFeXv/x9D9On6HoJpw2n01sDH4X/AAJfN+xrtk4Uzku8YtoyeF/wJfN+xpyP
u1vyP9AMtV+XfXuuEEl3b9S3EyZWylXbHjZH+48O+6R+L/Urh18Vnx9I9f6AR+lZE77KqoRbjJpP2RZffbTC
nko8pPUiOF97yfm/djxH7VHzf4AnmZM6J1qKTUu69yuy7Lqj5k4Q4+q9jub96xvm/dF2d9zs/l+oFWZOyz
G5w4+VKKb33JYHm+VHnx8vX1ddyMvwr/0Rdh/da/gBO3zPLflcefpvsYMPz/Pt48PtLnv4vsekYsD+Pk/N
+7ArunOHiTdceUmtJfyLHkZFFkPPjFwk9dPQL8Vfy/sPFP4UPmAuysjyIpRXKcukUUzuy6Ep2wg4eqXdEM
5SeZTqXHa0n7PZZPGyJQankpxffaA1wkpwUo9mtoryb1j1cmtt9EiVFbqpjBvel3MviXR0t/ZUuoHJWZiqc
5wi4tdUu6RLAkoYUpPsm2arWvKm99OLMeHZ5WBKet8W+gHY25dy51whGPon3ZOnIsvonwjFWxemn
2IUwyMitWSv4KXZRRzw3pZf15dV19+4FeL5/wBJt48N8vr7+PoacjIsjaqqYcpvrt9kV4X3vJ+b92SuutnlfR
6pKGlty0Bz6RfRbCOQouMum4+hsPMzKp1+Xzudm5dmtHpged4l5uly4+Xy+rrv2Lr/ADfoNvnceXpx9iPi
n8CHzfsXZv3Sz4AZseeVKiPlQioxWtv1NOHkO+t8lqUXpncP7rX8CjAerMl+0v8AIHVfkXyl5EYxgvWXqSoy
LJysqsilbFbXsyumV+XymrfLgnpJLZzGi4+JTi582o/afr2ArX0j6e/sebr+R6Ud8Vy7666Ma/FX8v7G0CjLyfIjF
RXKcuyKZ3ZlNbnOEGvy9BldM+hy+yX5n3Sz4AQlkTWB53TnpfDuXUzc6YTfeSTZjn+EL4L9TVi/da/lQFeJk
SuVjnpKL10K45GRkSk8eMVBPW5epXi7eNla9d/oMOq6dCdd/Fb7a7AacXIlZKVdsVGyPdL1GJfO52c9fV
eIojRjTryHbO1Tk1p9CHh32r168gLZ3zjnQpWuMlt+/qcyMmcbVTTFSsfffZFdv4rV8v8AkqcZy8SsjCzhJro
9d+wF30i+iyMciMeMunKPobDDbiXWJKzITW+m0bl2AAAAAAAAAFOPj+RKx8uXN77a0XACF9Mb63Cfb
39jN9BnJKE8iTrXpo2ACEql5Dqg3Ba0mvQUVuqqMHJya9WTAHJx5wlHetrRjhg2QjqGTKK9kv8A+TaAM
9WPbCxSlkSml/ta7/3JU4/lW2z5b8x71rsXADLLCcZuVFrr33XoSx8VUzdkpudj9WaABmuxFO3za5uufq16
irEcbVZbbKyS7b9DSABSsf8A+W7+Xda46LgBTDH4ZU7uW+S1rXbt/gZGP58q3y48HvtvZcAKb8fzbap8te
W9613GVjrIgly4tPaZcAM7xpzxnVO1ybe+TRKeNGzGjTJ/ZSSfwLgBiWBNrjZkSlBehdVjeXjTp575b667bL
wBRDG4Yro572mt69zjxX9EVCs1r/dr8zQAMcMK2EeMcqSS9FH/APkvrpaqlC2x2qXuWgDGsKyG1VkSjB
+mi/HojRDjFttvbb9S0AcnBTg4y6prTMawJx+rHIkq36G0AZ8bF+jym1Pal6a7GgACmrH8vIst5b5+mux3Ix
45EUpbTXZr0LQBj+hTm0rr5TivQvvoVtHlJ8Eta6FoA5CPCEY73paOgAU4+P5Dm+XLm99taK54b8xzptdb
fdLsagBnx8RVTdkpOdj/ANzO5GLG9qSk4TXaSLwBkWE5TUr7XYl2RHxNbhWv/sbSu6mFySnvo9rTAzyw
ZtcPpEvL9mjVXXGqtQitJEgAa2mvcpxaPo9bhy5be960XACmvH4ZE7eW+fprsSvohfXxn/Jr0LABjeFZJKM
8iUoL00X2URljumP1VrS9S0AQpr8qqMN74rWyGPj+RKyXLlze+2tFwApnj88qF3LXFa1ruXAAU3Y/m3V2
ctcHvWu53Jo+kVcOXHrvetloA5FcYpey0ZpYTVjnRa6+XdehqAFGPiqmTnKTnY/9zOxx9Zcr+X2lrjouAFOT
j+fw+tx4vfbZ3JoWRVwb112mWgCrHrnVDjOxzfo36FkoqUXF9mtM6AM2Nizon0tbh/x0csw92uymx1yf
fXZmoAZ6MRVWOyc3ZZ7v0NAAFLx95av5dlrjostrjbW4TW0yQAx/QrFHhHJkoe2i5Y1ax/J68ff1LgBiWD

PXB5EnX/xRbRi+TVZXz2p/l2NAAqxqfIq4cuXXe9aLQAM1+H5lvm12Oufrr1K3gNyjN3Sc097a2bQBTbj+Z
fXZy1w9NdyWRV59Mq+XHfrrZYAKvIi8dUz6pJLZRHBmlweRLy/+KNgApxcf6PBx5ctvfbRLI+7W/I/0LDk4
qcHF9mtMDz8TGnPHU67pQb3tehrxsaOOm9uUpd5MsqrjTBQh2RICmnH8q22fLfmPetdhkY/nuD5ceD
323suAFN+P5ttU+WvLe9a7k76vOplXvW/XRMAVKhfRvJk9rjrZDFx50bTtco66LXY0AAZfoco5DsrtcVJ7a
13NQApWP/8ALd/LutcdDKx/pEIx5cdPfbZcAKsiiGRDjLprs16FDwrJJRnkSlD20bAByEVCKjHslpEbqo3VuE
+z/sTAGNYM3HhLIk6/+JbTjKvHdMnyT3vpovAGJYE4/VjkSVb9C3Gxfo8ptT2pemuxoAGWWJJZDtrtcOT
21o7kYnm2KyE3XNeqNIAxS8Pc1ud8nP3a2bQAKsmhZFXBvXXaZBY03jSqna5OXZtdjQAIU1+VVGG98Vr
ZDHx/JlY+XLm99uxcAMf0GUZPyb5Qi/QlRhKm/wAyNjfTTTXc1ADNfiOy5W12OEktdjRFNRSb20u/udAF
WRjxyIcZdGuzXoZMjFsjjydl8pqK6I9AjZBWQcJdn3ApxoKzBhCXZx0VxwZpcFkSVfskaq4RrgoR7LsSAoxcb
6OpJS5KT9iuWE4zcqLXXvuvQ1gCjHxlTJzlNznLvJlc8OXmysptdbl3WjWAMtWF5d8bXa5Nd9ruTyMWN7
UlJwmu0kXgDIsOUpp3XSsS7LsawAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAI2da5fW49H9b2IYq40JKzzP8A7AWgreTTGXF2xT+JYmmtp7QDsE010Ksr7tb8rKc
O6uvDr5zjHv3f5sDWDkJxnHlCSkvdEbLa6vtzUfiwJja3rfUjCyFi3CSkvyZku/FKfl/yBtDaXd6OTnGuO5yUV
7tmLPtrtxfqTUtSXZgbgZsiKlRXu7yu3X3L5TjDSlJLfRb9QJAreRSpcXZHfxLAAK3kUqXHzI7+JKNkJtqMk3Hv
r0AkAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAACvI+7W/I/0M+MpS8Nah9pp6NGR92t+R/oZ8XzP/Hrytc+ut/ECjHeKq/L
uhxn2baNuJV5NPFWc4t7TRmeZTKrjkVtzXRriWeHQnCh8k0m9pMC7K+7W/KzNg41U8dTnHk5b7+hpy
vu1vyshgfc6/5/qwKcH/TvyK19mL6EcOuOTOy61cuukn6E8T75k/H9yFU3g2Trsi/Lb3GSA7dBYuXVOv6s
ZvTRK78Up+X/ACRcnm5NbjFqqt72yV34pT8v+QOWR+k+I+XP7EFvRHxDHrrqU4R4veuhLI5Y2Ysji5QktS
0VZuVG+pRrjLintyaAt8R+61fFfoPE1y8le8tDxH7rV8V+h3xH7VHzf4A7kYlMMWXGnFbT9SLtlHwuMk+
rXHf8zTlfdbPlZmhU7vC4xj9rq1/UCyjDp+jx4xbc2dbDK/4a253trTbW1
/UDcCrHyFfz1Fri9dS0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABoAAAAAAAAAAB2AAAAARsUnXJQaUtdGyQAx+bl
xjxdCc/wDlssw6JUVvm9zk9s0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAD//Z"></figure><p>Constraints:</p><p>0<size
<100</p><p>0<data<1000</p><p>Input format:</p><p>First line indicates the size of the queue</p><p>Second line indicates the elements of the queue.</p><p>Output Format:</p><p>every lines indicates the enqueue of each elements</p><p>last line indicates the fine queued elements.</p>

answer

```
 #include <stdio.h>

#define SIZE 100

void enqueue(int);

void display();

int items[SIZE], front = -1, rear = -1;
```

```
int main() {
  int n,data,i;
  scanf("%d",&n);
  for(i=0;i<n;i++)
  {
      scanf("%d",&data);
      enqueue(data);
      display();
  }
  return 0;
}
void enqueue(int data) {
  if (rear == SIZE - 1)
    printf("Queue is Full!!");
  else {
    if (front == -1)
      front = 0;
    rear++;
    items[rear] = data;
    printf("Enqueuing %d\n", data);
  }
}
void display() {
  if (rear == -1)
    printf("\nQueue is Empty!!!");
  else {
    int i;
    for(i=front;i<=rear;i++)
      printf("%d ", items[i]);
  }
}
```

question

Question description There is a bit string consisting of n bits. Then, there are some changes that invert one given bit. Your task is to report, after each change, the length of the longest substring whose each bit is the same.
Constraints 1≤n≤2·10^5
1≤m≤2·10^5
1≤xi≤n
Input Format

The first input line has a bit string consisting of n bits. The bits are numbered 1,2,…,n.
The next line contains an integer m: the number of changes.
The last line contains m integers x1,x2,…,xm describing the changes.

Output Format
After each change, print the length of the longest substring whose each bit is the same.

answer

```
#include <stdio.h>

#include <string.h>


#define N    200000

#define M    (1 << 18)    /* M = pow2(ceil(log2(N))) */


int max(int a, int b) { return a > b ? a : b; }


char cc[N + 1];

int pp[M * 2], qq[M * 2], tr[M * 2];


void pull(int k,int l,int r) {

    int m = (l + r) / 2, k1 = k * 2 + 1, k2 = k * 2 + 2;


    pp[k] = pp[k1];

    if (pp[k1] == m - l && cc[l] == cc[m])

        pp[k] += pp[k2];

    qq[k] = qq[k2];

    if (qq[k2] == r - m && cc[r - 1] == cc[m - 1])

        qq[k] += qq[k1];
```

```
    tr[k] = max(tr[k1], tr[k2]);
    if (cc[m - 1] == cc[m])
        tr[k] = max(tr[k], qq[k1] + pp[k2]);
}

void build(int k, int l, int r) {
    int m;

    if (r - l == 1) {
        pp[k] = qq[k] = tr[k] = 1;
        return;
    }
    m = (l + r) / 2;
    build(k * 2 + 1, l, m);
    build(k * 2 + 2, m, r);
    pull(k, l, r);
}

void update(int k, int l, int r, int i) {
    int m;

    if (r - l == 1) {
        cc[i] = cc[i] == '0' ? '1' : '0';
        return;
    }
    m = (l + r) / 2;
    if (i < m)
        update(k * 2 + 1, l, m, i);
    else
        update(k * 2 + 2, m, r, i);
    pull(k, l, r);
```

```
}

int main() {
    int n, m;

    scanf("%s%d", cc, &m);
    n = strlen(cc);
    build(0, 0, n);
    while (m--) {
        int i;

        scanf("%d", &i), i--;
        update(0, 0, n, i);
        printf("%d ", tr[0]);
    }
    printf("\n");
    return 0;
}
```

question

Question description Given an array of n integers, your task is to process q queries of the form: what is the sum of values in range [a,b]?
Constraints 1≤n,q≤2·10^5
1≤xi≤10^9
1≤a≤b≤n
Input
The first input line has two integers n and q: the number of values and queries.
The second line has n integers x1,x2,…,xn: the array values.
Finally, there are q lines describing the queries. Each line has two integers a and b: what is the sum of values in range [a,b]?

Output
Print the result of each query.

answer

```
#include<bits/stdc++.h>
using namespace std;
```

```
int main(){
   int n,q,i,a,b;
   cin>>n>>q;
   int x[n];
   for(i=0;i<n;i++)
   cin>>x[i];
   while(q--){
      int sum=0;
      cin>>a>>b;

      for(i=a;i<=b;i++)
      sum=sum+x[i-1];
      cout<<sum<<endl;
   }
}
```

question

Question description You are given an n×n grid representing the map of a forest. Each square is either empty or contains a tree. The upper-left square has coordinates (1,1), and the lower-right square has coordinates (n,n).

Your task is to process q queries of the form: how many trees are inside a given rectangle in the forest?

Constraints 1≤n≤1000
1≤q≤2·10^5
1≤y1≤y2≤n
1≤x1≤x2≤n
Input

The first input line has two integers n and q: the size of the forest and the number of queries.
Then, there are n lines describing the forest. Each line has n<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> characters: . is an empty square and * is a tree.
Finally, there are q lines describing the queries. Each line has four integers y1, x1, y2, x2 corresponding to the corners of a rectangle.

Output
Print the number of trees inside each rectangle.

answer

```
#include<bits/stdc++.h>
using namespace std;
```

```
#define rep(i,a,b) for (int i=a; i<b; ++i)
int dp[1005][1005];
int main(){
   int n,m; cin>>n>>m;
   rep(i,1,n+1){
      rep(j,1,n+1){
         char x; cin>>x;
         dp[i][j] = (dp[i-1][j] - dp[i-1][j-1]) + dp[i][j-1] + (x=='*');
      }
   }
   while(m--){
      int y1 , x1, y2, x2; cin>>y1>>x1>>y2>>x2;
      cout<<dp[y2][x2]+ dp[y1-1][x1-1] - dp[y2][x1-1] - dp[y1-1][x2]<<endl;
   }
   return 0;
   cout<<"for(i=1;i<=n;i++)";
}
```

question

Question description Given an array of n integers, your task is to process q queries of the form: what is the minimum value in range [a,b]? Constraints 1≤n,q≤2·10^5
1≤xi≤10^9
1≤a≤b≤n

Input

The first input line has two integers n and q: the number of values and queries.
The second line has n integers x1,x2,…,xn: the array values.
Finally, there are q lines describing the queries. Each line has two integers a and b: what is the minimum value in range [a,b]?

Output
Print the result of each query.

answer

```
#include <stdio.h>

#define N    200000
```

```
#define N_    (1 << 18)    /* N_ = pow2(ceil(log2(N))) */
#define INF    0x3f3f3f3f

int tt[N_ * 2];

void build(int *aa,int k,int l,int r) {
   int m, k1, k2;

   if (r - l == 1) {
      tt[k] = aa[l];
      return;
   }
   m = (l + r) / 2, k1 = k * 2 + 1, k2 = k * 2 + 2;
   build(aa, k1, l, m);
   build(aa, k2, m, r);
   tt[k] = tt[k1] < tt[k2] ? tt[k1] : tt[k2];
}

int query(int k,int l,int r,int ql,int qr) {
   int m, q1, q2;

   if (qr <= l || r <= ql)
      return INF;
   if (ql <= l && r <= qr)
      return tt[k];
   m = (l + r) / 2;
   q1 = query(k * 2 + 1, l, m, ql, qr);
   q2 = query(k * 2 + 2, m, r, ql, qr);
   return q1 < q2 ? q1 : q2;
}
```

```
int main() {
    static int aa[N];
    int n, q, i, j;

    scanf("%d%d", &n, &q);
    for (i = 0; i < n; i++)
        scanf("%d", &aa[i]);
    build(aa, 0, 0, n);
    while (q--) {
        scanf("%d%d", &i, &j), i--;
        printf("%d\n", query(0, 0, n, i, j));
    }
    return 0;
}
```

question

Question description There are n children, and each of them independently gets a random integer number of candies between 1 and k.

What is the expected maximum number of candies a child gets? Constraints 1≤n≤100
1≤k≤100
Input

The only input line contains two integers n and k.

Output

Print the expected number rounded to six decimal places.

answer

```
#include <bits/stdc++.h>

using namespace std;

int N, K;
double ans, a, b;
```

```
int main(){
   scanf("%d %d", &N, &K);
   for(int i = 1; i <= K; i++){
      a = b = 1.0;
      for(int j = 1; j <= N; j++){
         a *= (double) i / K;
         b *= (double) (i-1) / K;
      }
      ans += (a-b) * i;
   }
   printf("%.6f\n", ans);
   return 0;
   cout<<"double power(double a,int k)";
}
```

question

Question description A company has n employees, who form a tree hierarchy where each employee has a boss, except for the general director.
Your task is to process q queries of the form: who is the lowest common boss of employees a and b in the hierarchy?

Input

The first input line has two integers n and q: the number of employees and queries. The employees are numbered 1,2,…,n, and employee 1 is the general director.
The next line has n−1 integers e2,e3,…,en: for each employee 2,3,…,n their boss.
Finally, there are q lines describing the queries. Each line has two integers a and b: who is the lowest common boss of employees a and b?

Output
Print the answer for each query.

Constraints 1≤n,q≤2·10^5
1≤ei≤i−1
1≤a,b≤n

answer

```
#include <stdio.h>

#define N    200000
#define LN    17    /* LN = floor(log2(N - 1)) */
```

```
int main() {
    static int dd[N], pp[LN + 1][N];
    int n, q, p, i, j, k, tmp;

    scanf("%d%d", &n, &q);
    for (i = 1; i < n; i++) {
        scanf("%d", &p), p--;
        dd[i] = dd[p] + 1;
        pp[0][i] = p;
    }
    for (k = 1; k <= LN; k++)
        for (i = 0; i < n; i++)
            pp[k][i] = pp[k - 1][pp[k - 1][i]];
    while(q--) {
        scanf("%d%d", &i, &j), i--, j--;
        if (dd[i] < dd[j])
            tmp = i, i = j, j = tmp;
        if (dd[i] != dd[j])
            for (k = LN; k >= 0; k--)
                if (1 << k <= dd[i] - dd[j])
                    i = pp[k][i];
        if (i != j) {
            for (k = LN; k >= 0; k--)
                if (1 << k <= dd[i] && pp[k][i] != pp[k][j]) {
                    i = pp[k][i];
                    j = pp[k][j];
                }
            i = pp[0][i];
        }
        printf("%d\n", i + 1);
```

```
    }
    return 0;
}
```

question

Question description A forest is an undirected graph without cycles (not necessarily connected). Mohana and john are friends in kerala, both of them have a forest with nodes numbered from 1 to n, and they would like to add edges to their forests such that: After adding edges, both of their graphs are still forests.
They add the same edges. That is, if an edge (u,v) is added to Mohana's forest, then an edge (u,v) is added to john's forest, and vice versa. Mohana and johns want to know the maximum number of edges they can add, and which edges to add. Constraints: 1≤n≤105, 0≤m1 m2<n 1≤u,v≤n, u≠v Input The first line contains three integers n, m1 and m2 — the number of nodes and the number of initial edges in Mohana's forest and john's forest. Each of the next m1 lines contains two integers u and v — the edges in Mohana's forest. Each of the next m2 lines contains two integers u and v (1≤u,v≤n, u≠v) — the edges in Johns's forest. Output The first line contains only one integer h, the maximum number of edges Mohana and John can add. Each of the next h lines contains two integers u and v (1≤u,v≤n, u≠v) — the edge you add each time. If there are multiple correct answers, you can print any one of them.

answer

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
int fa[1005],fa2[1005],n,m1,m2;
int gf(int x,int *f){
        return f[x]==x?x:f[x]=gf(f[x],f);
}
int main(){
        cin>>n>>m1>>m2;
        for(int i=1;i<=n;i++)fa[i]=fa2[i]=i;
        for(int i=1,x,y;i<=m1;i++)cin>>x>>y,fa[gf(x,fa)]=gf(y,fa);
        for(int i=1,x,y;i<=m2;i++)cin>>x>>y,fa2[gf(x,fa2)]=gf(y,fa2);
```

```
    cout<<n-max(m1,m2)-1<<'\n';
    for(int i=1;i<=n;i++){
        for(int j=i+1;j<=n;j++){
            if(gf(i,fa)!=gf(j,fa)&&gf(i,fa2)!=gf(j,fa2)){
                cout<<i<<' '<<j<<'\n';
                fa[gf(i,fa)]=gf(j,fa);
                fa2[gf(i,fa2)]=gf(j,fa2);
            }
        }
    }
    return 0;
    cout<<"while(m1--)";
}
```

question

<p>Question description</p><p>Siva Sir students were chatting and playing quite loudly on the last day of the year, celebrating the end of the academic session. Siva sir was harshly chastised by the college's principal. But, instead of becoming enraged, he attempted to engage everyone in a different task.

So Siva sir gave his students to solve the task such that, you have to perform in-order tree traversal in Binary search tree.</p><figure class="image"><img

src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAMiIlLCUfMiwp
LDg1MjtLfVFLRUVLmW1zWn21n767sp+vrMjh//PI1P/XrK/6//3////////B8P////////////bAEMBNTg4S0
JLk1FRk//Or87//AABEIAgIHSAMBIgAC
EQEDEQH/xAAaAAEAAwEBAQAAAAAAAAAAAAAAAAAgMEAQUG/8QAQBAAAgIBAgIGCAUDBAICAgMBA
AECAxEEEiExBRNBUVJxFCIyNFNhkaEzcoGxwTVCRBUj0fAkgmLhQ5IlovFU/8QAFgEBAQEAAAAAAAAAAA
AAAAAAAAAEC/8QAIhEBAQEAAwEAAQQDAAAAAAAAAABEBAhIxIUEiYXGhQlGB/9oADAMBAAIRAxEAP
wD1AAABwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwA
dBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAd

BwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdBwAdB
wAdBwAdBwAdBwAdBwAdBwAdAXIAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAcAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAdXIBcgAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABTqtRHTVqcot5eOBbF7op96yB0AA
AAAAAAp1dz0+nlZFJtd4FwI1TdlUZvm1kkAAAAhfb1NTntcsdi5gTBgl0ooLMqLEvmSh0hukl6PasvGc
AbQAAAAAAAAAAAAAAAAAM2k1Mr5WqSS2SwsAaQCrU2ypplOEN7XYBaCFFjtpjOUXFtciYAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABwAAAAAKXOXpKhn1e4uMtkn
HU5Sy+4mt8MtainrJek7M+r3foc66z4TIQk5apNrD7v0FXOMtagUWWTlZ1dfB9rOSVtS3b9y7RUzg0Az2
WScIWReF2ondZtqzF8XyFTrq0FVE3Kt7nxjzI0SlOUpNvHYhTruVZdJxqbXNCmTlWm+bOaj8GRCLcdJlPD
/wDsflcy8f8Aq8GaCtthnfhL7k6JyblGfFoU3hFwM++d02oPbFdpyTspacpbosU6NIM+oscVBxeEzseshmy
yXDHIU6/KvKbZyjbCKfB8yEett9bdtXYRlvV0FPi01x7ybrXHjNawU3WSUlXD2n2kZRtrjuVm7HNFrGcWgF
FljdCnF4bYrVs9s3Lh3Cr1+XV4IW56uW14ZGmzNO6T4rmVmfKtBmotlKzEnz5EpSlLUKMW0lzJWt4bmx
eDPvndNqD2xXack7KWnKW6LFOjScfJlOom1GLi8ZOwjZndOXBrkKnX5XdPOU4tyeeJaZKbVXXLteeCLql
P2pt8ewZrXPj93VoMsHO5t9Zt7kX1qaj67TYrO8YWS2QciOnjtr3PnLiR1T9RL5lsvVrfyQ/K/wCP8qKbpSs
Sk8pmkxKOKVNc1I0W2f7G5f3Ima1z43fiuN8pXJZ9Vs0mNx2dU+18WaLI2Sa2S2rtLic8z5FgM0+spalv3L
5ktRNqEXFtZFTp/peDM1c4b9+OGcE42OWnlLtXaKdVwK6JOVSbeWQrlJ6icW3hZ4Cp19/ZeCl12ybzZhd
mDlU5xtdc3nuYp1+fNdU5ekuGfV7gpy9JcM+r3EV74/8AvYF74/L+CNzP6aAUUzkrJQm89wtnJ2xhBtd+
C1jr9g/9rUf/ABmXlGqXqRfcy5cVkYvL7ma6ACsAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAA6uQC5AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAADD0t7vH8yLdTbKnRb4Y3JLmVdLe7x/MiWv8A6c/JAQhbrNTBSqUK445y7
S3Q6mdznXakrIPjgt0vu1f5UZdJ/UdVgCU9TddfKrSqOI+1ORCWo1OknH0lRnW3jdHsO9FNLroPhNSyyz
pRxWjknzb4AS12olTplZW1xa59xK5cxVXZq9RKM4KMs9r1qa6LrUufA9CtJVxS7kBl1Oqs65tvkP
rfS4aaSu2Ti+2PYWaZqPSmoUucuRd0n7lP9AJ1WRq0UZy5Rgm/gmZRgz1Sb1hbPXPCsnNPX
qp0QlXqYqOOC28gL9JqZWuddsdtsOO5rb53201xaSaZ1rb53201xUpqWI9y9yXey7TaW327TpqY7v
5YGvTK5Vv0iScK9c9ncWgAYOmfdI/ny/nj+dfszdH2V5AYbtbbDDWzpFDcWzpFDcWzpFDcWzpFDcWzpF
OEsrHYUVpPpqz5Q/hGjXpvRWqPPPFE0GQMFPKFtxWqdmHPEVbFXHFl9cnnn9TaqqoWaqvqXrNW5K
/WqvyF8AIFr1M4a9UzskvFtaqaZZWPsRn6qcqVbnXnz9T26nn9aXyAvjbJnny
S7SCfSEodYnW3Yx5yYmfWuDaRPnW4SFb6rWKWSZ5EXIWmWVVW3Fqx5u1KN+IB1Pnt3rXjXcUCvak+SL9Pp
p6eu5mzpmOfHgR6SGbS0DqbzJWx0znN1YTu7sBDrtzq1FR5qk3K1Gua2q11RnBOG1DJ3KQv8zsn
480Bq1mp9HgsLdOXCKKV/qCjvbqfbsI6ySWu005ewb5+xLyAz6C+eoqlKzGVJrgjG9xJ/CwIgWHI/AD
s50Z+JqPzgbynV2Sp007IY3JcMlxm6R9xt8gEr5ro7ruG/amUVX6zVVp1KMElxLtfyJS/o/8A6Iv0PudX5QK
9FqbJ2TpvSVkO1dprMFP9Xu/KbwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAA4AAAAAGd++L/vYaCOyO/fj1u8mtcdiRn/zP+9xoI7I79+PW7xpx2VRnqtS3Lk+0ndbDq2k021jgWS
hGaxJZIxpri8qP1Ea7Zv3UIVt6Zp83xK6m7Jwi+UTWRjXGDbisNiGc/Wa1uuyaXKaL6Y7aku18SU64zxuWc
EhE3lcivUfgyK/8P/veXyipRw1lM5sjs2Y9XuEM5TJ+6Gn/AAY/qQp94s/X9y+MVGOIrCRyMIxk5JcXzEO3
qiiSrlKE+DO6iyMoqEXl57C2dcJ+0hCqEHlR494n4Xtl7KL1tjUn2L/gvui5VSS5nZwjPG5ZxyJCJvLxRp7I9Wo
tpNd5CyanfDbxSa4l0qa5PLqxO9VDh6vLkJq9uNqi+KV6cs7WSddKWd/DzUOfxXaorT
rZxWS+v8ADj5ISh GUdrXDuOpJJckzvK5HTFLMHOtdrNpF1xclJri0lizrBGjhcJkRmBcl+nzfAF1qN6uPFzqxw
XyipLDWUxEJ9O/wCmM9ElXKUJ8Gd1FkZRUIvLz2Fs64T9pCFUIPKiPFRreeb6plCqxb6hZsnpZRK9Nz6JvqTMMh
mcnCM8blnNIRneVzGOurrK5NPiucFr7HERReeb6plCqxboy2snppSlB7uOHz
JOitvO0nGKisJYQhvLNyKdUs1p9zJ2ldUW63CyPBPmB6HM35/BTHdp8PtyUQUpzjW+UP
IxUY4SwkcqIzYm5eqDEdXpMl5yZHUSzWn3MmM3cEI6Mkl5r2JcH0SJGD5QW+aw1O4ZSb92g2QHqBjK
VcZJJrKXIvnx2X4b8iipZD0fODPnPXtKfCv3zyiO2Ixhcb/rPm9D0gtUYhOlzm/lfQo0NYfEtBbOV
reXH7+7NWla27Jce7J2CitSlDl/9F0qa5PLjxOqqEZKSjhokXvipe+P/vYF74/+9hdsjv349bvGyO/fj1u8sTtn9
Kb1ssjYv1GnW+crH+hy+1TTrinnOC+uOyCj3E/K7s4/VWq4qMVzbLipPlhR5Fxcy5ZeZxvqgCsgAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA6uQC5AAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACnV6b0mtQctuHnOMi+jr
tP1W7HBLOC4ARqh1dcYZztWMlVWm6rU23bs9Z2Y5F4Ay36FWW9bXN1WdrXaRhoG7FPUWyta5J8jY
AKdXp1qaer3beOc4yWxWIpdyOgDNqtHHUSU1JwsXKSK/QHYmtRfKzhwXJI2vguJDr6l/8Akj9QIUUdXR
1U5OxcuPcZ1oLK2+o1EoRf9uMm2MlJZi018iM7q65xjOaUpck+0CGmolTGW+2VjbzlnNPplRbbZuz1jzjH
IvAAAAUazTelVKDltxLOcZL0sJIACiOmUdZLUbvajjbgvaTWHyAAw/6fKubenvlXF848yX+npShONs1Ynlyf
abAByUVKLi+T4GbSaGOlnKSk5N8FlcjRKyEXiUkn82c62vxx+oC2qN1bhNZizItDdBba9VJQ7scjbGcZezJPy
OgVVUuujq3Nzb/uZzSaf0anq1LdxznGC4AUanTSucZQtlXKPLBStBOck9RfKxReduMG0AVajTw1FWya5c
muwzx0Nnsz1M5V+HvNoAz6bSLTTm4TeyXKPcV2aGXXSsoudblzWDYAI1xcYRjKW5pcX3kdRV19Mq87
d3aWACl6bOj9H3f243YJ0VdTTGvOdqxkmAKIaZR1c792dyxjBeAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAHAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAABzCznHE6AAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAB1cgFyAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQv8Ad7Pyv9jz+jtJRdpVOytSll8cs9C/3ez
8r/Y83Qae6zTKUNQ4LL4JASlWtJr6o0N7Z845NOqWn9Ip61Nzb9XBljGeh1cZXYsVjwp9qLdf75pfzAartR
XTKEZvDm8Iol0lRGWPWaX9yXAr6Uip3aaL5Slh/Y12VQWmnWopR2vgBZCcbIKUXmL5MzXa+mqbhxlJc
1FZwUaOyUOipyXOO7Bb0ZVGOljPHrT4tgTr19NlkIRb3S7Mciy7UQplXGeczeEZNQkultPhJZXH7nOlo75
6aOcZk1+wF619MrerjuljPFLhwL6LoX1qcHlM7CqFcVGEUkvkYdL/AOLrrNO/Yn60ANl19dLgpvDm8IsM
D/8AJ6VS5wpX3N4Ge7RU32b7Ity8zBp9JTPX3VSi9kVwWT1zz9J/VNR5f8AaYVU6Kuc4pqOMvtIT6Qoh
GLy25LKSXEnrfc7fysq6MphHSxm0nKXHLA0TvhXUrJvbFrPEoj0lRKST3RT5NrgUdIWL02qM4uUIrO1dpK
/VwuplW9PZxXD1eQG9ySjubWOeTI+kqFLC3NeJLgZrpzj0VXGWU28PvwXV6yuFSgtNZtxj2eYG2uyNsF
ODzFkjB0ZlSuW2UYN5imbwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAADgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgO
gDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgD
gOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDgOgDg
OgDgOgDgOgDgOgAuQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAACFybpsSWW4vH0PO0l12mpVb01jw+eD1AB5zrv1t0JW19VXB5w+bLNZVOeq08ox
bUXxa7DaAMeuqnO/TOEXJRnltdnI1WLNckubTJADHoKJLRSqti47m47Oapm0oOgB5li1
MtRVqnTwi8bFzwXa+udl2mcItqM8vHZyNoAGLpKpuEb4NWq8+s88d7+cotRkuD7+RtAFWri56WyMU23F4SI6KMoaS
uMk1JLimXgDHrqLJThfTxsh2d5Fa29rC0s932NwAz6qh6nS7L
LK91lex55FgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAZZaSbbfpNizxxkz6SqzUVuT1FkcP
HNgek2lzeAYNdW6tHGLm5vfzfkzV6RVDEZWRT7sgWgJ5WVyK531QltlZFPuyBYAmpJOLTT7UAAys4zxI
TvqrltnZFPubMsGpdKSaaacea8kBtBRGK9MlLrsvH4fcWO6qLadkU48+PICYIxshKG+Mk495GOopnLbGy
LfdkCwAq9Jp3betjnzAtBVqLo01Se5KWG4p9pHSahXVR3Sj1nHKXmBeCjSxUVPF3W5f0LLLq63ic4xfc2B
MHITjNZhJSXemcdtak4ucU4rLWeQEgQhfVZLbCyLfcmTAANpLLaSXayuOopnLbGyLfdkCw45RTw5JPuy
RtsjVBylJLuyYNFCq177ZZtcspZA9IBtRTbeEuLZjjrE9XKDnDqkuDA2AjKyEWlKaTlyy+Yrtrsb2TUsc8ASBU9
TSpYdsM+ZOVkIR3SklHvbAkAmmsrkyMLIWZ2SUsc8MCQK5aimMtrsin5k3JKO5tYxnIHQV+kU8P9yPHl
x5kp2QrWZyUfNgSBCu2uz2JqXkyU5xgszkorvbA6G0ubIQuqseITjJ9yZl1/wCPpvzfygNoBj6RuUadkZLc3x
WeOANalGTwpJv5M6ZtHVRBZqkpSxiTTyWy1FUJbZWRT7sgWARkpJOLTT7UVu+pZzZHg8PiBYDkZKUV
KLynyZTqIp3VPrurw/Z8QF4IyshGSk5JSfJNkVqKZS2qyLfmBYAVek07tsdB0PmBaA2kst8Ct6ilJPlPrI4fLiBYG0
ubwRnZCtZnJRXzZi19kLI1OElJbuJLtZXHUVTltjZFvuyBYCM7IVrM5KK+bYCM7IVrM5KK+ZyNtcpuEZZpyXNJgTAfJ
8cFGkio1PF3W8efcBeCueoqhLbKyKfcmd6q3idkU+oqM6Wsq7q1OK2

8+PICYIV3V2PEJqTXczPqNYoXVxhOLi3ifyA1go1Djbp8xuUIt+0mXV8K4rdu4Lj3gdAK3qaVLa7Y58wLOQ
TTXAq1LzpbGvCyGg9zr/X92BoBXLUUwltlZFPuyWJqSTTTT7UABCy6ur25qPmzsLIWLMJKS+TAllN4yDFp
fftR5/wAmt2wU9jmt3dniBIEYWQsTcJKWO5hWQc3BSW5c1niBIEZWQjJRlJJy5JvmVQivS5y67c8fh9wF
4I2W11+3NR82K7YWLMJKXkwJA5Oca45nJRXzI13V2exNS8mBMAAAAAAAAAAAAAAAAAAyWStlqZwhY
4pJAawY9t/xhtv8AjBLjYDHtv+MNt/xgXGwGPbf8Ybb/AIwLjYDHtv8AjDbf8YFxsBj23/GG2/4wLjYDHtv+
MNt/xgXGwGPbf8Ybb/jAuNgMe2/4w23/ABgXGwGPbf8AGG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/AI
wLjYDHtv8AjDbf8YFxsBj23/GG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/jAuNgMe2/4w23/ABgXGwGPbf8
AGG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/AIwLjYDHtv8AjDbf8YFxsBj23/GG2/4wLjYDHtv+MNt/xgXGw
GPbf8Ybb/jAuNgMe2/4w23/ABgXGwGPbf8AGG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/AIwLjYDHtv8Aj
Dbf8YFxsBj23/GG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/jAuNgMe2/4w23/ABgXGwGPbf8AGG2/4wLjY
DHtv+MNt/xgXGwGPbf8Ybb/AIwLjYDHtv8AjDbf8YFxsBj23/GG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/j
AuNgMe2/4w23/ABgXGwGPbf8AGG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/AIwLjYDHtv8AjDbf8YFxsBj
23/GG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/jAuNgMe2/4w23/ABgXGwGPbf8AGG2/4wLjYDHtv+MNt/
xgXGwGPbf8Ybb/AIwLjYDHtv8AjDbf8YFxsBj23/GG2/4wLjYDHtv+MNt/xgXGwGPbf8Ybb/jAuNgMUlfG
LfW8lk0aaTnRGUnlvPH9Qq0AAAAAAAAAACrVTlDTTlD2ki0NZWGB4VVk4WqUW92fqe6VQ0tMJ741p
SLQAAAPkYui/wACf5v4Nr5GLov8Cf5v4A70p7tH86/Zkno6VpmtuZbc7u3JHpT3aP51+zNM/wACX5f4Ay
UWyh0ZKSfGOUinTz0kav8AeW6b5trJdpK+t6PlDvbwRo1MKK+qvralH5cwJdHTXWW1xbcOccmy2Wyq
c/CmyrS2u7dLqtkf7X3l1kd9cov+5NAYtFp4WVO21b5Sb5nKK41dJShDlt4HNPqfRIum+Mk4vg0NPY7ek
XNxccx4J9wE6/6rZ+X+EVwphd0hap8UuOO8sr/qtn5f4Q0/9Rv8gIa9xr6qhepXzIjzIXz0cqWqlia5PBo11U
267oR3OD4oi9dU4+pU3N8o4ArvunPQ0pPjY8N9+DS9FT1OxRWce125Iauqd2mhKMNs48dqOf6hF14U
JdZ4cdoCWncNDJW7ZShF7X3Euj64LTQmorc85fbzJJXT0U1bxslF8Cvo6+LrVOGpxz+4ENDLZTfLwtv7Hd
FRC6ErrVvlJ9o6PipVXRfJvBGi/wBD3U3xaSeYtLmB3atLr4KHCFi4o5KqN3Sk4z9lJPHfwRKvdq9ZG7a1XDl
ntO1/1Wz8v8ICGsqhRZTOuO17uw9AxdJcqvzG0DFrW7dRVp84i+LJ36Op0S2QUZRWU0c1tc1ZXfWtzhz
XyIW66NlThVGTnJYxjkBOhrUaHNiUnFNcSPRtcHSpuK3KT4ltFTp0bjL2sNsj0b7r/wCzA1NKSaaynwaPPh
TW+kpwcI7VHgscOSPQPPssWn6RlZNPbKPBryA7roKepohyT4cDRPTVwonGD6pNcZFOpedZpmu0t18JT
00lBNvKeF2gZd2ijU4JOTx7SjxyT0sXf0dOD4tNqP7ivWwVCrrqlvxjCXaT6Lf+xJdu7+EBynUY6Ocs+tBbf+C
CzpujcrhKfb5//RXbTL0x0L2LJKX/AH7m7V0u3TyhHmuKAxVS0UaVGabk1xeO0lpZ50eohnKinjywyVWsr
rrULq2pxWH6vMtVkrdHdJ19Wtrx81gCrQaaqVKsnHdJvt7DmpShrd98XKprh3Iv6P8AdI+b/cjbqJ0XtXRz
S+TSAjVXRZfG3T2KOOcUuZLVwp66Nl9nqpcIFCcLtZXLTQcVF+s8YRLUPqderbIOUGuHyAqvsp6yueni4y
T48MJmjX/j6b838oo1mpV6g4wkoRftNc2X6/8AG035v+ANph6UhFVRmorc5Yb/AENxj6U93j+f+GBO/b
ptLOVUVFtLkR0ukqenjKcVKU1ltl19XXaeUFza4GajWKitVXRlGcOHLmB3S5o1llCeYYyivS0Qu1N7sWVGX
Bfqy3SQnZqJ6mcdqlwimc0H4+p/N/LA2RioxUYrCXJGPW+9ab838o2mLW+9ab838oCOtgrNbTB8muJ3
XaaqvT7oQUXFrijuo/qNHkWdI+6S80BVrLZPSVRT42JZ+hctFT1Wzbxx7XaV30yt0NThxlGKaXfwOf6jDq+
MJdZy247QOaScvRr6pPPVpojoNNVOnrJx3PP0J6amVeltlNYlNN4/Qn0b7qvNgVRgtVr7Os4xr4JEddRXV
KuUI7cvDSJTk9HrJWSi3XZ2or1eoWodeyMtil7TXNgWdI2Lra6pNqHOWCrUT0kqcVLbNcmkaNdXNTrvh
Hds5r5HHrqnHFdTlN8o4Ar1k3ZoaZPm3x+hsp01VGHCPrYxnvM3SOXpa247Xniu7gbgOS9l+Rj6OTejsU
eEtzx9EbJey/IxdH7vQ7NmN254z34QFOnenrUoamDVmeLkjZo6VVGThZvhJ5WOwpWrhscNVW967Nv
M70dGS6yW1xrk/VTA2nnr0SmUlZLrpt8XjJvmm4SS4NrgedpdRDSxlCyuSnnsQEtBKPpdsa8qtrKTOVUw
u1129ZUXnB3STc+kLJSjtco8n+hPSe/ajz/kCN1caNbQ61t3PDSGsqgtTQlBLdL1uHPiiWs98035v5Q6QzCy
i3DcYy4/YCWvhGGjcYpJZXBEnqFTVUnCcswXsor1lsbtDvhnDfaaqfwa/wAq/YDLqdU5aSTjGUG5bePAsq
0VKoUZQTbXF9pLWUu7TuMfaXFFNevjGtRshJWRWMY5gdVM6NDbCclJYeMdxCFjq6KUo8HxS+pZm6
WhtIdwbi8LHJEaand0WoLm84+oEtNpKvR4ucFKUlltkdJmnVW0ZzFLcjlOtjTUq7oyU4LGMcyWjhOd1m
onHbu4JPuAhoqo6mVl1q3NvCT7C6GldWqVlTUa2vWiUU2egznXbF7G8xkkXU32ajUZrTVK5trmBllqPR
9TqGlmUnhGrRUKMeuk99k+Oe4qorhZrdQpxT5naJS0l/UWP1JP1WB3T/7OtvrfCL9Zf9/UoqnKOphqH7
Nk2v8Av/ews6Si4zhZHm04stv0+Oj1Bc4LP69v8gcx1vSfyqj/AN/c5T/VLvy/8EujotwnbL2pyI0/1S78v/AF
d22vXSlqYOUH7L7C2iqmWoVunsSSXGCQnqZ03SjqI5rfstIqq2266M9PBxgvaeMICN1tc9dLr2+rhwS+ZG
62iNtdmn9Vp8VjHAutT0urdzhurnz4cia1cLLIwpq35fFtYwgNgAAAAAAAAAAAAAAABk/zbPJfwazJ/m2e
S/gJvi0AGmQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACO6PiX1IJA4m

nyZ0oAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAjZ+HLyZLSe7Q/X9yNn4cvJktJ7tD9f3Jq4uABGgAAAAAAAAAAAAAAAAA
AAAADSfMAAAAGAAAAAAYWcgAB2gABhAABjPMAAAAAAADAAAAABjPMAAAAGAAAAADCAAAAAO
QAAAAAAAAwmAAAAAAAAAAGFnIAAAAAAAGEAAAAAAAMZAAAYAAAAAAAAAAAAAAAGEAAHMAA
AAAAAAAAAAACWOQAAAAAAAAAAAAAAAAAAAyf5tnkv4NZk/wA2zyX8BN8WgA0yAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGPU6+FLcYLdL7I50hquqj1cH6z5vuPKIuYut1V1vtTe
O5FO597AI0lC2yDzGbT8zbR0k1wuWV3owAEfQQnGcVKLymSPG0WpdFmH7D5/I9hPKyisbkdABQAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABFpynGKbWckupfxJEq5gB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L
+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIdS/iS
FOoB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIdS/iSFOoB1L
+JIdS/iSFOoB1L+JIdS/iSFOoB1L+JIik42Si23hLmKQs/Dl5MlpPdofr+5Gz8OXkyWk92h+v7jTFwAI0AAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAABk/zbPJfwazJ/m2eS/gJvi0AGmQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAOSe2Lb7DpTrHjS2P5EHjXWO22U32sgARt2EXOSjFZb5HraXouEY5v9aT7F2HkJtNNPDR63R
V1l11krJOTwgPLtSjbJLkmyJO/wDGn+ZkAB7HR1rs06TeXHgeOb+iX69i+SCa9MAGmQAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAHF+LD9S4pX4sP1LZ+xLyM63njqaaymmvkDDRc6OjlYlnD5fqTlqbobZ2VRVc
mlz4oitYM9t8+t6qmKlPGXl8EKdRN3Om6CjPGVjk0BoBkp1N179SuOFLEm3+xKWotnbKFEItQ4SlJ9vcBf
GcZOSi8uLw/kSMWkntepnYtuJZa7uBL0m9V9d1Ueq54z62ANYM12qcLK4Vw39ZHKLqnY4J2qKl3R5AT
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAKZfjS8kXFMvxpeSLib45Z+HLyZLSe7Q/X9yNn4cvJktJ7tD
9f3LrOLgARoAAAAAAAAAI2TjXBzk8JcwJAxV9JQlYoyg4p9uTaAAAGe7WQpt6txm3jPBEY9IUuWJKcfm0
Qf9Wj+X+C3XOv0aanjOPV8wNCaaTTymDDXbOno2M1jK5Z7smyqTnVCT5yimwOye2LfcslWlveoq3uO
3jjBXpr530WueMrK4eRHo6SjpJSbwk22BsBijfqdRmVMYxguW7tLtLqHdujOO2yHBoC8y3auSv6mmvfJc
zTPdsezG7HDJ5tPpHpduzZ1n92eQHpQcnCLmsSa4ruOmbU6mVTjXCO62XZ3Fc79Vp8SuhGUO3b2AaN
Tc6KXNLPHGCN+odWmjaopt44ZK9fJT0W6Lym00Q1v9Pr/APX9gNsXuin3rJ0psuVGmjNrPBJLvZR1utcO
s2Qxz29uANoMy1Ls0cra0lKPNPsJV6jdo+ufNRy/MC8GbRah6iEt+N0X2dw0+olbO5ywq4PgwNIMUb9T
qG5URjGCfBy7S3S6h2uULI7bIc0BoKJ6hx1cKdvCSznPn/wU+k332yjp4x2x/ufaVxlZLpGrrYqMkscO3mB6
JV6RH0nqcPdjOewtM6vl6c6eG3bn5gaAZLdRbLUOmiKylltivUWw1Cp1EVmXKSA1lUtRGOojTh7pLOew
tMVn9Vr/AC/wwNpC22FMN03hfuTMNy6/pGFcuMYrOPuBP/Ua+brsUe/BpjZGde+LzE64qUXFpNNYw
UafTy09dkXJOL4r5AVrpGt8q7H+iLadZVdLbFtS7mirov3eX5/4RDWOL1dPV4dmeOAN5C2yNNbnLkiZi1
3+5fTT2N5f/fqBbVdYtPK65YXNJdxbVYra1OKaT7yGs90s8jmh9zr/AF/cC8HJSUYuTeEllmON+p1GZUxjG
C5bu0DaZbtXJX9TTXvkuZ2jUTuhZHao3Q4YfIy0+kel27NnWf3Z5Ab1aowg7WoSl2Z7Sw83Xddvhv243er
g309Zs/3tu7P9oEzNbrqq57EpTkvCWaqbr09klzS4FXR9Shp1LHrS4tgTp1ld0tizGXdJF5n1Omd04Tg1GcX
z7xqtQ6dsIR3WS5IDQDFK/U6fEroxlBvjt7Ces1MqY1yhhqX7Aag2optvCXFsxWXayEOtcIKHh7UiV9lluk3
1KKjKD3Z/j7gaoTjOO6DTXejpi6O63q4+z1XHzydeouvslHTRjtjzlIDYDNp9ROVzpuiozXFNdpVLVaiWosqq
hGTT4fIDcV33Ror3yTazjgSq39XHrcb+3Bn6S91fmgOf6jXzcLMd+C+m+u+Oa3nHNdwpaWnhnGNq/Yya
Ta9fa6vw8dnIDVVqI22WQSacHh5LTPpr5233QljEHhY8yt6m666UNNGO2POTA2FC1D9MdG3glnJTHU3
rUwpshGLfPHaV3TnDpJuuO6TWEv0A9EjOyFazOSiuWWZHqNRRZDr4xcJPHDsIdJdbt47eq3LHfnAGvUS
sjVupw2uLT7USotjfUpx7ea7iOn63Z/vbc9m3uM+k/wBrWXUr2ea/7+oG0AAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAyf5tnkv4NZk/wA2zyX8BN8WgA0yAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAc5AdMWrudsZ00rdw9Z9iO2Wz1M3VQ8QXtT/4NFVMKq9kVw7f
mQeCC7V0um+Uex8UUkbEstLvPa6O0lmmcpTaakuGDxS30q/4s/qBbrtLOie+TWJt4wZSU7bLMb5yljvZ
EAeh0SuNj8jz0svCPUr0ThTCVcttq457/kE1uBno1HWPZYtli5rvNBWQAFAAAAAAAAAAAAAAAAAAAA
AAAAAAAABxfiw/Utn7EvIqX4sP1LjOt548/ZL/TFHa87uWPmX62Llpkkm3uXI0giscnLT6qVrhKULIpNpZw
0dg3qNXG1RlGEI4TksZZrAGbQRcaGmmnufMrrm9JdbGcJOM5boyism0AYKoStWqg4Sg7OMcorrjSq1C
2u/elhpZwz0wBllXt1tCjF7Ywaz3GoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAUy/Gl5IuKZfjS8kX
E3xyz8OXkyWk92h+v7kbPw5eTJaT3aH6/uXWcXAAjQAAAAAAAAV31ddTKvOMosAHlV9H3OxKaSjni8n
qgAAAB599UbuklCWcOPZ5ENTpoaWddiTlXn1kza9PnVq/dyWNuCy2tW1yhLk0Bn1zUtE3H2Xhou07/8

AGrf/AMF+xCvTbdM6Zz3xfLhjBUtDNLY9RLq/CkBDo/3a7zf7EaE30XbjvZq0+m6iqcN+7d245EtNR1FTrc
t2XnlgDJpar56eLr1G2PdjkX6bTSqvnZOxTbWHwI+hShJui6VafYW6fTKjc3JznLnJgXGLTf1C82lNWn6vU
WW7s7+zHICiWI9Kxcu2PD6F+taWlsz3DUaaOoSy3GS5SXYU+hSm1118pxXYBVZldEwz/wB4k9b/AE+v/
wBf2NOooV1PVp7Vw7CN2m63Txq342444Az6/wDAob9lNZ+hvysZzwITqjZT1c+Kxgzeg2bdi1Eur7sAR6
OSl1/gb4fczqUlVLSf3OzH6Hp01RprUILgvuVeix9L6/d/647QM9rWj1MnHhGdfDzX/fuTqrcOjJY5yi5f9/Q
u1WmWpjFbtri+eMlyilFRXJLAHn6Sq6dCdd+2PH1cci6nTTrunZK1Tk44fAPROE26LpVp80WafTKlyk5Oc5
c5MCrotrqJrtUuJy3+q1fl/wCSdmifWuym11uXNIV6LZfG12uTXPK5gajEv6q/y/wbSlaf/wAt37uaxtwBVZ
TC6+U6Ltlq9rByF91WojVftlu5SRO3Sbretqsdc3za7RVo9tqttsdklyz2AaTFZ/Va/wAv8M2ma/SStuVsbXB
pY4IDSYbH1XSkZS4Kaxkn6Jf/AP8AXP6f/Zbbpo3VRhY23FcJdoF3LmUU6haiFjjFpR4ZfaVehWtbZamTh3
YNNdMaqurgsIDz9FpK76nKe7KljgyekjHT6udU0tz9mRq0un9HrcN27LznGDmp03XyhJT2TjyeMgXmK/h
0nS3y24/c2rlxMnSEJbYXQ9qt5/QC3We62eRzQ+51/r+5KSjqdM1GWFNcyiOithFRjqpJLsS/+wLtYm9LZj
uMulqvnp4uvUbY92ORqoonW31lzsTWMNFXoUoSboulWn2AS02mlVfOydim2sPgV6b+oXl+n0yo3Nyc
5y5yYq0/V6iy3dnf2Y5AUdI+1R+Y0X0O7biyUMeF8zmq061EEt21p5TJUQnXXtnY5vPNgU2aZw0tsesnY2
s8fkS0E1PSxS5x4M0GSWixNzotdTfNLkBbqNSqHBbXKUnwSMuqjJ9IwUZ7G48Jd3Mvp0ahZ1lk3ZNcm+
ws1GnhqIpSypLk12AZ7NLfODjZqE49uUR1kHCGmg3na8Z+hP0KyeFbqJSguzvLb9MrerSltUH3ZAlq/dbP
yspp/pb/ACS/k02w6yqUM43LGSFVKr06qb3LDTfICrQZeiWOfEzaKu2dcuru2YfFYNWm0s6J8LW4eHByz
R/7rspsdcnzx2gcr0ti1MbbLlKSXLHYR0nv2o8/5LaNL1dnWWWOyfLL7CVWn6u+yzdnf2Y5Acu0zts3K6c
PkmU6yt1aHa5OXrc2bSrU09fVs3beOc4yBl9AhLTqUHLe4prjwyWdHTi6Nqioyi/W+ZqhHZCMc5wkimO
m2ap3QnhS5xxzAp0XvWp/N/LHReFXZF+0pcS+nT9VbbPdnrHnGORnlTXZqJyov6uafrJAd1H9So8v+Qv6
q/y/wUwrS6Qgo2OxrjKT7zatP/5bv3c1jbgCjpT8KH5jvSnu0fzr9mXarT+kQjHdtw88sktRSr6nBvHan3ATj
7K8jHDj0rPHZHj9EX0wlp6pdbbvS7X2Iq0MXOVmoksb3w8gNYAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAABk/zbPJfwazJ/m2eS/gJvi0AGmQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAjKUYRcpPCXaQdbSTbeEjHKU9ZLbXmNK5y7x6+tl2xoX1ka4xUIqMVhLsAVwjXBRgsJE
gAM+r061FeOUlyZ49lcq5uM1ho+gKrqK71icc/MLmvCBvt6MkuNc0/kyh6DUJ+xn9SLWcGyHRt0n6zjFf
U2UaGqlpv15d7BWfQaN5Vtq/KmekAVlTfp43LPszXKS7CunUShPqtRwn2S7JGoruphdDbNeT7gLAY4W
z00lXe8wfsz/AOTWnlZQHQAUAAAAAAAAAAAAAAAAAAAAAAAAAABxfiw/UuKV+LD9S4zreeAAIoAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABTL8aXki4pl+NLyRcTfHLPw5eTJaT3aH6/uRs/
Dl5MlpPdofr+5dZxcACNAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAaysMACummNKkoZw3nHcWAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAApt0lNst0
o+s+1PBcAK6dPXQn1ccN832lgAAAAQupjdFRnnGc8HzJpJJJLCXYAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAMn+bZ5L+DWZP82zyX8BN8WgA0yAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAABGUowi5SaSXayCRxyUebS8zzNR0jKTcafVXi7TFKcpvMpNv5sVY97rq846yH/7Ikmmsppr5Hz
xKFk63mEnHyYpH0APO03SOWo3f/sjbbdCqvfJ8OzHaEjtlkaoOU3hIyxhPWSU7E41L2Y952uqepmrb1iK
9mH/JrAJJLCWEjoBQAAAAAAAAAAAAAAAABGcIzi4yWUzJ6+ifHM6H9Ym040msPiiDkJRnFSi8p9pIxyrnp
ZOylbq37UO7yNFVsLoKUHlfsBYAYtVr41NwrxKXa+xAbCLtrXOyC/U8S2+215nNv5dhWKsfQRnGXsyT8
mSPnU2nlNo1Ua+2p4m98fnzFI9gFdN0LoboPK/YsCAAKAAAAAAAAAAAAAADi/Fh+pcUr8WH6lz5PsM63
nimWrojPa7Fnl8i7nyPLVrr01lSr3wy11mOB6GnSjRBRluSS495FWNpLL5I5CcZxUovKfJmfWzfVxqj7Vr2/
p2k6ZxVkqIxx1SXEC4pnq6IT2SsSfb8i58jz6LK69PKm6L6zjlbcuQHoJ5WVyI2WQqjunJRXeyno9uWirb8v
uXzUWszSwuPHsAhVqKrm1XNNrmiwx0/8Akazr4LFcI7U/EbAKbNXRVNwnYlJc1hlkJxsgpweYvkzI3ZRqr
7HRKcJ7cNY7EaqbIXVqcH6rAmQhbCyUowkpOPPHYdtgrK3BtpPuZl0cI16rUwgsRW3C/QDYRsshVHdZJ
RXzJHn75W6uyXUu3Y9sVnCQG6uyNsFKDzF9pIq090boNqLi4vDi+xlk474OLbWVzQEY2wlNwjJOUeaXY
Lba6Y7rJKKMumrjVrrIQWEoIlJb+kkpcVCGUBoqurujmuSkkT5GTCh0lHbw3wef0NYFC1mnc9qsWeReYd
bYnB09U4pvG9rgjbBYhFZzhcwOtpLLeEiNdkLY7oSUlyyiN1MbklPOE84T5lPRqxpml43+4Giy2FUd1klFf
MknlZRi6Qog6p2vLlwSy+CNkPYj5AcssjVBym8RXaUrXaZvHWr6M0GSKWp1UspdXU8Jd7A1kbLIVx3Tk
opdrJGTX0QnTOyWW4x4LPBAak1KKafBlL1mnU9vWrPL5fUso/Ar/KjNq3Fw9GqgnOfYl7K7wNhTL8aXk
i2EdkIxznCwVS/Gl5IuJvjln4cvJktJ7tD9f3I2fhy8mS0nu0P1/cus4uABGgAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMn+bZ5

L+DWZP82zyX8BN8WgA0yAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB4+u1Tus2xfqRf1PQ11nVaaTTw
3wR4pNXAAv0UqI3Z1CzHHDuyRpQD2aNVp9VZ1KowmuGUjzNbSqNVOuPsrigKTVoZxd8Y25ePZy+CZl
CbTTXNAfRAros62mE+9FhWAAFAAAAAAAAAAAAAAAAAAADLbp5Qn12n4S/uj2SNRzkQefqdfmnbBO
M3wlnsPOJ32dbdOfeyBGsASqip2xjKW1N4b7j0/RNBGXVSszPl7QV5QNOu0notiSeYS5NmYC3TXy09ql
Hl2rvPbhNTgpR5NZPnz0+i7N1cq2/ZeUXE1vABWQAAAAAAAAAAAAABxfiw/UssjvrlHOMrBWvxYfqXGdb
zxireoqo6n0fc0sKSksM0aWp00RhJ5a5loIqm/TQunGcpSi4rC2vBTRpJV6yybc9uFtblz8zYABkss1MlKCo
W58FPdwwawBCipU0xrXHainXwtsrjCuDknL1knjKNIAopstclCWmdcEue5PBeABnndfGUox07l4XuWGS
0lLooUZNOTbbwXADk24wbjHc1yWeZj0/Xx1Vk5adqNjXHcuGEbQAMm27TX2Srr6yFjzweGmawBRpKp
wU52YU7JZaXYXSbUW0svuOgDFX6QtXK16dpSSXtLgWaiuyN8b6o72liUc4yjSAM1MLLNQ77YbMLEY5y
aXnDxzAAx2z1F1Tq9H2uXByclhGquOyuMc5wsEgBC2c4RzCt2PuTwZ9Crq4uFlList7tyZrAGXXK6yt1V0u
Sf925IuolOVa31uDXDGclgAGfRVzrqkrFiTk2aAAM+s62VUq66nPcue5LBoAFOndnUYnU4SisJNp5M2n9I
pUm9K5Tk8ylvXE3gDkW3FNrD7V3FUvxpeSLimX40vJFxN8cs/Dl5MlpPdofr+5Gz8OXkyWk92h+v7l1nF
wAI0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAABk/wA2zyX8Gsyf5tnkv4Cb4tABpkAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAYOln/ALUF3s8w9XpSDenUl/bI8ozrWB2EJWTUILMnwSOHodDRj11k5YzFYWfmFaK4VdGUb7Hutkvr
8keTbZK62Vk+cnk9fU6COotdk9R5LHJHlamuNVzhCanFf3ICsAAev0a86RfJtGsz6CDhpIZ7eJoKxoACgAA
AAAAAAAAAAAAAAAABXe8UWPuiywjZHdXKPemiD58BrDaYI2Hqabo2uSrslbuylLajzqapXWxrjzkzV/pu
qhatqXB8JqQE+mZyd0IOLUYrg+8889XpmUVXVBvM85/Q8oAbOi3jUSXfExm7oqGbZz7EsBNeoADTIAA
AAAAAAAAAA4vxYfqXFK/Fh+pcZ1vPAAEUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAApl+NLyRcUy/Gl5IuJvjln4cvJktJ7tD9f3I2fhy8mS0nu0P1/cus4uABGgAAAAAAbS5vAAAAAAAAAATT5P
IAAAAAAAAAAAAAbS5vAbS5tIAAAAAAAAAAAAAAAAAAAAAAAAAAAAA2lzeAAAAAAAAAAAAAA
BxTjLlJPyZ0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABtLm8AAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAG8LiOYAAAAAAAAAAAAAAAMn+bZ5L+DWZP82zyX8BN8WgA0yAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAIXVq2qUH/AHI8GcHCbjJYaeGfQHm6qHplz6iOXFetLsZNXGACUXGTjJNNdjBGgAACdFTutj
CPbz+RGuErJqME22exo9KtPDjxm+bCbrRFKMVFcksHQDTIAAAAAAAAAAAAAAAAAAAAAAADxukKHVe
5JerPijMe7fTG+pwn+j7jxJw2SazuSeMrkzLWaim4tOLaa5NGldIapRx1r+iMwCuznKyTlOTlJ9rOAAD2dDS
6dOlL2pcWZOjtLGb62bTSfCPz+Z6hcZ3QAFQAAAAAAAAAAAAcX4sP1Lilfiw/UuM63ngACKAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAUy/Gl5IuKZfjS8kXE3xyz8OXkyWk92h+v7kbPw5eT
JaT3aH6/uXWcXAAjQAAMstJNtv0mxZ44yZ9JVZqK3J6iyOHjmz0nyMXRf4E/zfwBHXVurRxi5ub3835M1
ekVQxGVkU+7JR7p7Zno6Zdbc7u3IGizDql621OL9ZdznaL9ZdznzIaVbaElElZ1n/wAjPppN9GWZ7IyX2O
6ZSl0a1D2mngDQ9TTGW12xT8yxNNZTyzKTS7nukQjNfGLmlMJeeiSxbktbaikRF/sZdDpqp0Kc4qzqm
qvnK+fWTb5YzgDbCySzCSvkxOyFazisivmzvfqX7ol5uYkZhSNAuREvtIzMzoOaP9m5ZYPbUf8r8hbPqQ
rsinq4j5RTjqwv8ShCDRaqFfWcdwVfbWkHhDCxmuEaZyeUJDEF0NnJG+BfuPRQ5ylDmF2wnqCCr6Ps2OkZ
rsinq4PrtrS/D7ymuui2+FmnsUHHnFLmLv6pT+X/kDZOcYR8mvbvIxNTWJxpGpA/9DhjKBDT0p7tH86/
8A7ksv0ae2VG2ucXz7ANMpRpGpIGvx0HLV0G0a7bCAnebMeqWCl0/zr+gX7u8k5yJBiLaTnyVfOD8GuC
Zkekvq/SLD9v+ThILfNG3hl4ddTI/zrLyxbkOVZzOF6+q+KAbHS9Bhuq0dJWwdfPvuoQn9AGjLKxFM01
+n/cq1ATcdCS0+vSKuzJbGHxc6C2Ge1MJu5nF8/VTnPDjJNQ1tkzpMyLBsRRMP3qPzy9JS+l6iuNfvlZGO
MRmamy0/xIa8MEoTN2CZaRCLvpNKbifmJGQxlGqpyMQ3ghlb3WWmPG8nhuSNK7P8hIUVJJLMcFgvq0xk
yOTv3y6g41jEkHxaYVRspybFb4WS4QJzu0lK6klvRbklpYQMnX/pLHvJjOjuGx6nTKIFKE3tJFOcOG
WE0ajhQjLavVqpxcXmKflkzGhtlVKyjzK0XP7XVN65HaCsnmHU8DrKV4OiksBnsxuSc0OsM3JiIrNE2T
uWJhwMbXBZYKPtTeGunjfkJrFxO0nZaU4yTPcqyzhCtA5RRaFWgmGNRUPpxdbdhvBbXKQPsO7WxCtZw0
rZDr1sfTxW8Cb7PjdcvzbkcnJSUm0uoBKsRxrW6jhJrnHyJcttUGksNgTjbPxx6n1ZgS0ZUqKm2mXlSaXqR
5ga421zm4RmnJc0jk76oS2S1Gl6VjmLvKZgvTB9tSmhSS7FkNFDr1Oo8anYt5Ws6HVMoazkfM3qStiQZcF

TxD58Tum09ajPbNWVTfCPcU6OquWqvTgmoy4cOXFgegZLdYo6mEYzi4P2n3Gs8/UVVrXUxUEoy5rHM
Da7q1BTc4qL5PPM49RUm07I5XPjyMvSUVDTwjFJJS5Iujo6oVuKjxccOQF8ZKUVKLTT7UcjZCUnGMk2u
aT5GPRW9Xp7Yz51Z4E+joNUysl7U3kDTOcK1mclFcsszaPV9dFqyUVPOEu80zrhYsTipLnhmLoyuEq5Tc
U5KXB9wG2y2FazOSj5sV2wsXqTUvJmHTVrVai221blF4SZ3VVx0ttd1S2pvDS5AbyKsg5uCknJc1niSMW
p/2dbVd2S9WX/f8AvIDXOyFazOSivmVWRT1dcuu28PY7ynU/7+tqp/tj60hd/VKfy/8AIGudkK1mclFfNn
K7q7PYnGT+TMesW3VwndFypxjgdhXp7roT09ihKPFxS5gX2xT1Nb67bj+zvK9RrFC6uMJxcW8T+RHU/w
BRo8iOsqgtTQlBLdL1uHPigN0JxnHdFprvRk6U92j+dfszXCEYR2xSS7kZOlPdo/nX7MC/0imOIysin3ZLeZm
joqep27eLXtdpTpbpQ0VvHjXwQGud9VcsTsin3ZJqcXDcpJx70zHo9LXKhWWRU5T48TlMVRrpULjXNZw
wJ+mL0tw3w6rHMazV9VBdVKLlniufAqVNf+pOGyO3byxw5Hekq4RqjKMUm5YyvIDbCyFizCSkvkRlqKY
y2uyKfmU6vGn0kuqSjueOAo0dSojvgpSkstsDUmmsp5T7UV9fUo7usjjOOZm0TdWot0+cxjxRXoKK7XZK
cd2HhJgbo3Vzi5RnFpc3nkcjfVOW2NkW+7Jhnp4f6gqllQkstJkukKa6q4WVxUZKWOAG6VkINbpJZ5Z7SP
pFO7b1sc+Zl6RW9UJ9r/4JanSUx0snGCTisp9oGmd1dbxOcYvubJRlGcd0Wmu9GPR6euenU7I75S7Wc0
P+3bqIL2Yvh9wNdl1dXtzUfNnYWQsWYSUl8mYtFVHUb7rlublhJiyC0utqlXwjZwaA2zshWszkor5s5XdXZ
7E4yfyZj1i26uE7ouVOMcDsK9PddCensUJR4uKXMCep1fV2VxrlFrOJ/I1QnGcd0JKS70YddVBX0Yiluk93z
4o3QhGEdsIqK7kB0y6zVdTFKuUXPPFPsNRg6Trgq4zUUpOXF/oBbrLIWaKxwkpLhy80Sovqr09SnOKe1c
GyOsrhXobFCKiuHLzRzTaSl6eLlHc5LLbA1RkpJOLTT7URsurreJzjF9zZk0E3XRd2qDbRRp7NP609R61kn2r
IHqRnGcd0ZKS70dPO0c4LWyjU31clwTPRAAAAAAAAAAGT/ADbPJfwazJ/m2eS/gJvi0AGmQAAAAAAAA
AAAAAAAAAAAAAAAAAAADjaSy3hI5KShFyk8Jdpk9fWy7Y0L6yIEpz1knCtuNK9qXi8jXXXGuCjBYSEYqEV
GKwlyRICq7T1Xr148e9czFZ0W//wAdi/8AZHpAFeT/AKbfnnDzyW19F/Es/SJ6IC1itoWl23UL2PaXejXCas
gpReU1k61lYZkpfo2odL/Dnxg+75BGwAFAAAAAAAAAAAAAAAAAAAAAAON4WWG0llvCRjlKWtm4wbj
QucvEQdnOWsm66m40r2p9/wAkaFTWqlXsWzuJQhGEVGKwl2EgMFvRkJNuubj8nxRQ+jLlycH+p6wBX
lw6Lm369kUvlxNlGjqp4pbpd7NABWW7TyjPrtP6s+2PZIs0+ojcmvZmucXzRcZ9Rp+sanB7LVykgNAM9G
o3S6q1bLV2d/kaAAAKAAAAAAAAAAAAi5KNkHJ4XEn19fiOAkXNjvX1+IdfX4jgJF7O9fX4h19fiOAQ7O9fX
4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O
9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ
7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4h19fiOAQ7O9fX4iCkpWycXlYRIFibtRs/Dl5MlpPdofr+5Gz8O
XkyWk92h+v7jTFwAI0AAA+Ri6L/An+b+DaAMfSnu0fzr9maZ/gS/L/BMAYdL/AE23yl+xLS9Z/p66rG/jjP
mbABgesplVt1Fbc1wa2lnR0Jwoe5NJvKTNeABXqPdrfyP9jFpNXGihRsjJLi4tLmeiOYGLSKV+qnqHFxjjEcl
VNsdJfaroS3N5UsHpDGQPNha7eka5uLimsRz2rDL7tRZRe+tjml8mlyNYA81uF+rremg47XmTxhFt39Up/
L/ybcY5ADFfu02s9I2uUJLEsdgnrJXSjDSp5zxk1wRtHIDBrc16qq2cXOtLD8yvWapX1ba4S2p5cmj0xyAw6
950NTXJtfsx0l7tX5/wbgBXfX1tE4d64eZi6OjKdrsn/ZHajTqrLo4jTW5OS9ruJaSnqKFF+0+L8wLjFqP6jR5G
0AZukfdJeaLtP7vX+VfsTAGDQQ6zS2w8Ta+xGi+Omj1OoraceTxzPRDWeYGfTXddKTjVth2SfaZNNdHSO
cboS355pHpjAHnUWOzpHfKLjuXBPyLKf6pd+X/g2gDF0h+Jp/zf8FmputotjLbup7cLiaQB5monXqJRWnrf
WZ5pYNtl/V3V1uOXPtzyLsY5FL0+7VK6Us4WIruAuMOsl1WtqukntSxw/X/k3DnzAwa+asppnHk5cDXqf
drfyv8AYsAGCuEp9FbYcXx4d/E5RrIV0xrjVLrFwwlzZ6AxxyBh6Mfq2p893FEdLaqNTbXYmnOfA9AYQAw
9JRcertjwazHJuAGR6f8A/jurx623d+vMj0anLrLZcW8Rz5G0AYLU9Lq3c4bq58+HImtXCyyMKat+XxbWM
I2BLHIAYtb73pvzfyjaAMfSnu8fzfwznSXsVfmNoA5Pdsls9rHDPeYlq47XDV1vev8A48zcMZ5gYujovfbNRc
a5P1UyFVq0+tuVifry4fU9AYAGHWvqtVTa03Fc8G4cwMHSFkbdLXOPJy4fc3g5NuMJOK3NLKXeB5uurl
DUtQ5XJL7npVwVcIwXKKwZKYW36lXXQ2RgvVibAB5+gtVMpUTTU3LgegMccgYIyeh1Fm+LdU3lNHLJv
XXQhXF9XF5lJnoDkAKNbV1umku1cUXgDF0dGU998+MpcExd/VKfy/8AJtAGW++yi/8A3I5oa5pcjNOUL
9TW9NW008yklhHphLHIDFqP6jR5DpDMLKLcNxjLj9jaAIVWxurU4Zw+8zdKe7R/Ov2ZsAGL/UIwr2yhJ
WpY247Tul00vQ5xnwlZx49hsAGDT6paaHU3xlGUeXDmS0ylfqnqJRcYJYjk2tZ5gDBdNafpFWTT2uPNE+
kc2aWM4ptZT/TBsazzAGWco67SSVecrv7yurXRrqULYyU4rGMczcljkMIDHoq5yssvsW1z5L5HOjPZt/Mb
QBin/Vofl/hnelPdo/nX7M2ADDrv8fz/AODTq/dbPystAFGg9zr/AF/cp0azqtSv/l/LNoA8+mx6GUqrYvY3
mMkjqb1urhOMWqq+OX2m/mAMt99lF/8AuRzQ1zS5GacoX6mt6atpp5lJLCPTCWOQGLpFOLpsxlQlx+
xbPWQWnV0YuSzjHI0FOo0/X7E5YhF5axzAui8xT70ZOk4uWnTSziWWawBi1F8b9BZKGeGE8+aNGl92q
/Ki1LHIAYdBHdHURfJvBCi1aTNV8OTypY5nojnzAzae/rrHsqxWlwk+80gAAAAAAAAADJ/m2eS/g1mT/N
s8l/ATfFoANMgAAAAAAAAAAAAAAAAAAAAAABCyyNcHKbwkcuthTBym/8A7M9dU9TNW3rEV7MP

+SBGE9ZJTsTjSvZj3+ZrSSWEsJA6AABQAAAAACnU09dU0uElxi+5lwIKNLd1tfrcJx4SXzLzJqE9PctRH2Xw
mv5NSakk08pgdABQAAAAAAAAAAAAAAAAONqKbbwkclJQi5SeEu0yetrZ5eY0J/8A7EBuWtlhZjQub8R
rjFQioxWEuSOxiopKKwl2HQAAKAAAAAAAAKr6I3xw+ElykuaKqr5VTVWo4P8Atn2M1ELaoWwcZrKIJgx
xsnpJKFrcqn7M+7zNaaaynlMDoAKAAAAAAAAAAAAAAAAAAAAAAAAAAAAzy1EnY4U173Hm84SFl1k
NNKyUFGSfLOSDQDieUmMrvA6DhXqLXVS5xw2gLQcTyk+8AdAK5SmrYqMU4Pm88gLAcyl2go6Dmc8j
oAAAAAAAAAAAAABGz8OXkyWk92h+v7kbPw5eTJaT3aH6/uTVxcACNAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAZP82z
yX8Gsyf5tnkv4Cb4tABpkAAAAAAAAAAAAADLq9bGj1Y+tPu7iDS5KKy2kvmUS1uni+Ni/TieRbdZc8zk3
8iAqx7S12nl/8Ak+qLoyjJZi018j58lXbOqWYSaYpH0BVffGiOZcW+SXNmSrpHNeJRzZySXaX0ad7uuve6x8
l2RCRymiVk1dqOf9sexGoAAACgAAAAAAAAclJRi5SeEu0DkoqcXGSynzMlNy00pU2ywo8Yt9xRqukJSbj
Twj39rMLbby3lkXMey9fp0/b+xOvU02PEbFnuPDAqx9CdPF0+sspaWd0e5nrU3Qvhug/NdwZ3FgAKAA
AAjOcYRcpPCR5ep187G41erHv7WQzHpWX11e3NIq9P0+fb+zPGbbeXxArUe7XqabHiNib7ic5xhFyk8Jd
p8+WwvlmKtbnXF+zkVI9BRlrZ7p5jSuS8RsSSSSWEiFNsLa1Kt8O7uLAgACgAAAAAAAAAAAAjKMZxcZL
KfYZGrNFLMczofZ2xNpxrKwyDlc42QUoPKZIxzpnppuyjjH+6BfVqK7K96lhLnnsAtK7Lq61mc0v1PP1XSEp
Nxp4Lxd5hbcnltt/MVY9l6/Tr+/7E4aqmx4jYsnhgVY+hOni6fWWUtLO6Pcz1aL4Xw3Qfmu4M7i0AFAAA
AAAAAAAADBr9VbTbGFfBYznHM3kJ1wsxvipY70QxHT2O2iE5LDa4lj5BJJYXBHQMug/Afi3Pd5ktd7rP8A
T9xLTNWOdVjrcuaxlMl1DlTKuyxz3duOQENXKUaYKLxuajnuIX6eFNTsqzGceOc8y1adyqdds967HjGCPo
05Yjbc5wXZjGQIP/yNRGE87FBS297O6qqFWkmq44WVwLbaN8ozhJwnHhlHJUTnTKFlu5t88cgK9M3dY
5WPEocFDuIqx0xvhninmP6l86FKcZpuMo9q7Uct00bLo2N42813gZ1ZOvT2VSk3NPCb+ZY47NRp455RZ
ZPTRnqI2t8uzvJzq3XQszjb2AZLYKFs5aiuU4N8JLsJaiacqa4qUq2s4jzZbOi2W5de9r7Np2emi64RjJxcPZkF
U0px1C6uqyEGvWUlwNpVVXZGWZ27/ljBaEAAUAAAAAAAAAABGz8OXkyWk92h+v7kbPw5eTJaT3aH
6/uTVxcACNAAAAoWofpjo28Es5LwABTqtQtPXuay3wSAuBTTbZKMpXwVaXFPJbCUZxUotNPtQHQAA
AAAAAAQun1dUp4ztWcHNPb11MbGsZ7ALAAAAIwshYswkpJdwEgUVahz1NlW3Ch2l4AAAAAAAAAAA
AAAAAAEVZB2OCkty5okAAAAHJy2wlLuWSvS3O+lTaxxxgC0AAAAAAAAFWqudFO9LPHGCyEt0Iy71kDo
AAAAAAAAAAAot1Dr1Fde3O/t7i8ACMrIRkoyklKXJd5Xq73p6lNR3ZeALgE8pMAAAAAAAAAVX6iNDh
uTe544FoAAq1NropdiWcdgFoI1S31QnjG5JkgAIwshPOySeOeOwkAAAAAADknti33LIHQU6W531b3H
HHGC4ACMLIWLMJKS5cCqnUOy+yvbjZ294F4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAGT/Ns8l/BrMn+bZ5L+Am+LQAaZAAAAAAAAAAAAAGfWX9RS2vafBHiyk5Sbby2a+krN+o29kUZDL
WAScnhLLYN/RNO+52y9mH7hWBpp4awwaNdY7tROaT28k8GcAnh5R7Gh1HX1Yl7ceZ45o0FnV6qPdL
gwmvaABpkAAAAAAAAAAA8vpHUuU+qi/VXP5s9G6fV1Sn3LJ4DbbbfNk1cAAuaI0n1Nnw5fQjKucPai15
o9vVamemoqcIbspL7HmavWz1UYxlFR2vPADMW6a+VFqkuXaioAfQxkpRUlxTOmPoyzfp3F84s2FYACr
U2dXp5y+QHm9Ial22OuL9SP3ZkAI2HYxlJ4im/InpqXffGtdvM9a26OiXVaelyaXFgeNKLi8STT+Zwt1V8tR
c5yjteMYKgL9JqHRann1XzR7SaayuTPnj2OjrN+lSfOPAuM61AAqAAAAAAAAAAAAAAAAB4/SEoLUNV8
PFjtPVtn1dUpPsWTwJNyk2+bJq4BLLwgdrlsnGS7HkjSXU2/Dl9DjqsisuEkvI9OjpK6+1Vxqjl/Ml0pqlCt0Lj
KS4/IDyC3TXyotUly7UVAD6CMlOKkuTJGLoyzdQ4v+1m0rAACgAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAjZ+HLyZLSe7Q/X9yNn4cvJktJ7tD9f3Jq4uABGgAAYl/VX+X+Cd+ps67qaIpz7W
+SIL+qv8v8FUIWS11yhZ1cst8uaAvhqLqro16iMcT5SRTr+t62vdt27vU/+yyekunKLs1CeHwyh0h7dH5v+
AO3+lejy39XjD3Y7h0d1vVRzt6rDx35yX6r3a38rIdH+5w/X9wNBgq1Wpuco1wi2n7T5JG8xdG+zb+YCW
n1FvpDovilLGU0dv1NnXdTRFSn2t8kQn/VYfl/hjS8NfqFL2m8ryyB30m6iyMdTGO2X90ewt1eo6iC2rdO
XCKK+k2vR0nzcuBXqPV1Glc+XDPmAunq1RJ2wi4yWHjmi3STVfR8Zy5JN/ct1futn5TG0/8ASFj/ALxAsjdq
7o9ZXCCh2J82X6XUekVttYlF4aM9FOolTBw1GItcFjkXaTTumVjdim5PjhdoFl3W7F1O3dnju7jF0b1uHt2
9Xu9bPM9ExdF/gz/N/AFSnZHX3KqClKXDjyRdHUX1aiFd6jifJo5p/wCo3+Q13vGm/N/KAt1WpdUo11x3
WS5LuKpajU6dxd8YuDeG49hXqIyfSUVGextcGWWaW+cNtmoTj80BPWamVDqcEmpZz8+RXZdrK4dbK
EFDtj3HNXFwnpIt5aeM/Q06z3SzyAsqmrK4zXKSycutjTU5y5Ls7yGi90r8irpNN6ZY7JLIEY26yyHWRhBRf
FR7WXU6qNmndslt2+0i2tp1RceWFgy6mVc9Hb1O3g1navmByN2ruTsqhBQ7E+0tp1LtonJrbZBPKKNP
VfOiDhqMRxwWOROvTyphfKVim5RecLtAjVqNVqIf7cILHOTLNPqLbOsrlFK2C4dw6N91XmyFH9Tu/L/w
BTH0j0+eNnW4493JGjVamyiyuMYqW5cV3sjD+rWfl/hDV++6fz/kCyN19dU7NRFcMbVEr63WSr61Rgo

4yo9uDRqrnRQ5pZfJFCq1E6usnqXHKzhLkBZC5X6Oc8Ye1pr9CPRvuq82VaH3C3zf7It6N91XmwL7puum
c1zim0ZKr9XfWnXCCS5t9pp1Xu1v5WQ0Hudf6/uwGj1Ert8bIpTg8PBXLU3XXSr00Y4jzkzmk981Pn/I6Lw
q7I/3KXECVWpsjeqdRFKT9lrkyWq1Mq5xqqipWS7+wr17zqNOl7W7+UOXS3rdseH0Aq1ktSqdt0YuLfC
UTVO9afSwm1l4SS7+BHpL3X/2RRrk+p07zhY5/ogLHbrIQ62UIbebj2pGqm1XVRnHkzO9PqZRaepymu4
t0lXU0qG5S45TQEr7VTU5tN45Jdpm63WdX1u2CjjO3twX6u90U7kstvCKXTfKlzs1LXq5aS4AWw1HWaS
V0VhqLePmiirUaq+Ga4QWOcn2nNL/AE23yl+xd0f7pHzf7gNJqJ2ucLIpThzwQnqbrbpV6aMcR5yZzTf1C
8dG8Oti/aUuIFU5Wy1tKuioyT5rkz0jFqvf9ObQPN1PX+lVbtm7PqY8+0s1+/0OHWY37+OOXad1fvun8/5
O9Ke7R/Ov2YEXbq+r6yMIxglnD54NFOoVmn62XqpLiTn+BL8v8GGpN9FWY7/+ALI36q9OdMIxh2bu0t0
2pd0JKS22Q4NFGmqvnRF16jbHuxyLdPp5VTsnKxTclx4doFVOp1V8Wq4Qynxk+RbptRZK6VN0Upx48C
PRfu8vzv8AZHI/1Wf5f4QG0wy1WolqLKqoRk0+HyNxi0nv2o8/5AldfdTCnco75PEizV6nqIxUVunLkinpH
2qPzf8AA1XDX0OXsgJ3aymtzshBr5dg1Fjt6M3yxl45eZfrPdLPIyz/AKTH9P3A7VZq5URdUIqMYpLPN4La
r7dRpt1airE8PPIu0/u1X5F+xm6M/Ds/MBVoeu62e3bjct+f4PSMXR3t3/m/5NoGW7UWu/qaIJyXOUuR
yGourvjVqFH1uUonJW3X6mVNU1XGHN4yym6uVer06na7G5Ln2cQL9VqbKb4QhFSUly7WQsv1dCU7Y
Q2N8Uuw7qf6jR5FnSPukvNAT1GpjTQrMZ3eyjPKzWKpznCLi1xS5pENVwo0rfspLP0Rvta6qbzw2sDP0b
7r/wCzLtR1uz/Z257d3cU9G+6/+zNUvZfkB5/RvW7fV29Xu4558iMJ2x1typgpSk+3ki7ov3eX5/4RzSe/ajz
/AJA7DUX16iNV6j63Jos1WpdUo11x3WS5LuKtZ75pvzfyivURk+koqM9ja4MCyWo1OncXfGLg3huPYT1
epnTOvYlJS5rvIWaW+cNtmoTj80R1MXC7SRby00s/qgO2XauqPWThDb2ruNlc1ZXGa5SWSrXe52fp+5L
Se61/lA5rLZU0OcMZyuZR1usnX1kYQUcZx2sn0l7q/NF9X4EPyr9gMteqv1CxTCKaXrN8s/InptRZK6VN0U
ppZyiPRf4EvzfwcX9Vf5f4AndqbHf1OninJc2+witTdTbGGpjHbLlKJzRcNXqE/azn7nelGuoiu1y4ATlfOGtjV
LGya4M7rL5U7I1pOc3hZK9fB9RC1e3W0yNUlqtcrF7FcVjz/wC/sBbqdTKuUaq4qVsvoit6jUaeUfSIxcJPG
Y9hXdGcukmoz2Nrg/0LLdJfZHFmoTjntQG0o1Wo6iKUVunLgkXQW2EU3lpYMeq4dIUSl7OMfqAlbrKY9
ZZCDj2pc0W26jGjd1eOzGfMtvaVFjly2swRT/0mWe/h9UBvpm50wm+ckmynTaidsLXLHqPhgs0vutf5UZ
dD+HqPMDlWp1d8f9uEeHOR6Bk6M92f5mawAAAAAAAAAAAGT/Ns8l/BrMn+bZ5L+Am+LQAaZAAAA
AAAAAAAAAHhazjqrM95UaekYbNVJ+LiZjLeB7dVE6ujtlSzZJcf1PENC1+pSSVrwvkB6c9NY+jVSl6+OWTx
Gmm0+aPVlr//AAE1auuxx7zym222+bAE6eF0MeJEC7Rw6zUwXc8ge4ADTAAAAAAAAAAAM+veNJZg8
U93UQ6zTziubXA8ImtYBcwdht3rfnbnjgivRh0u0oxdSwljmS6Sprnpo6iuO1vGfmd6vo6zE96j8s4KekNZC
2EaafYXaBgAAHodEvjYuw9Iw9FwxTKfiZuKxvoZeks+iPHejUUayHWaaaXPGQY8QAEbb+h0vSZPt2l93Sc
qtS61WtqeH3nn6S96e+NnNcmvkeIN6C6xXymlLnjIFPS9UV1dsVhy5nmmvpDVrU2JQ9iPL5mQAel0S/8
AbmuzJ5p63RkNum3P+55GJrYADTIAAAAAAAAAAAAAAADPrnjSWY7jxT3tRDrKJx70eDyJrWAAIr1OiYK
FNlz8kebbN2WSm+beT0NNqKq+jpVuaU3ngeaAAAHodE+1YuzCPSMHRUMVSn3s3lY30ABQAAAAAAA
AAIzshDG6SWe8gkDieVlHSgDPpbJT63c87bGl5FtlsKo5m8EEwUw1Nc1Jp42rLTWDLVZXbOU7LZKSk8JP
hgEegDNp9VG1yi2s7ml5GkACFs1VXKb5JFEK77YqcrnDPFRS5AagVzsjTBOyXy8zlV8LW1F8V2NYAtBQ9
XSu1vjjgjl+pjCjfGSy16oGgFdV0LIblLlzIR1dUpKKk+PJtcALwVWX11ycZSw8ZJwmrIKUc4feBIAyLrbdRbFX
Sgo8kkBrBlpsm+urnLOxe2iyqcYadSlZujj2n2gXAojq6pNLLTbwsrmVu5V6uzfJ7dqwgNYKo6iuUJTUuEee
VyIaXUq6CTa38eH6gXWfhy8mS0nu0P1/cjZ+HLyZLSe7Q/X9xq4uABGgAAUrT/8Alu/dzWNuCOo0itmrI
ycLF/cjQAMsNHJ2RnddKzbxSLNRp+vlW923Y88s5LgBycVOEovk1hmfTaadEn/uuUOyODSABTptP6OpL
du3PPLBcAKXp86uN+7ksbcEb9KrZqyMnCxf3I0ADLDRt2Kd1rsa5J8i3UUR1Fe2XDtTXYWgDG9DOUdtm
olKK5LBfVQq9OqZPcsNPhzLQBj9CnDKp1EoRfYX6fTxoi0m5N8W32loAGWrSSqu3Qtag3lxwagBTXp9m
onbuzv7Mchfp+usrnuxsecY5lwAp1GmjqEstxkuUkU+hTm0rtRKcV2d5sAFF+m62VTUtqrecYznl/wWXV
9bVKGcbljJMAQpr6qqMM529pKcI2QcJLKfM6AMa0VkU4Q1ElW+zBoqohVV1cVmL557SwAY/Qpwb6
m+UIvsLatLGqqcVJuU1xky8AVaanqKtm7dxznGDkNPs1M7t2dyxjHIuAGa/SOy7ra7HXLGHwJWabrLarH
PjX8uZeAIXVRurcJcmZY6GeNktRJ1+FG0AZ6NL1NM69+5S7cciemp6irZu3cc5xgtAEbYdZVKGcbljJHT1dT
TGvO7HbjBYAKadP1V1lm7O95xjkQt0ebXZVY65PnjtNIAz0aRV2dZZN2T732HdTpo34e5xnHlJF4AxS0Er
F/uXyk+zK4I0zojZQqp8UklksAGP0K1R2LUy2d2DTVXGmtQjyXeTAELqo31uEuT7e4yrQTa2z1EnBf2m0A
Z6dL1WnnVvzuzxxyyizT09RSq927HbjBYAKatP1eost3Z39mORXbo91rtqsdcnzx2moAZI6HFsLHbKUk8vK
5msACjVabr9rU3CUeTI26WVunjVO3Mk8uWOfM0gDjjmDjnmsFenoVNLrb3JvuLQBj9ClBvqb5Qi+wuo
0ypjL1nKUucmXACnS0ej1uG7dl5zjAWnxq3fu5rG3BcABTVp+rvss3Z39mORcAKdRp+vcHu27HnlnJHWR
pnCMbZbW36rNBGyqFsdtkVJAefqaXChuzUufhj3miFDt0EKm9raTzj9SUNDRGW7bnzZoAjXDZXGGc7Ul

kr0un9HjJbt2555YLgBljo5Q1DsrtcYt5ccczUABlu0e+3rarHXJ88EfQHvjN3Sc08ttZybABTZp+s1Fdu7Gzsx
zJainr6XXu257cZLABVLTxnp1TPikkslC0M3HZLUSdfhNgAq01Ho9Wzdu45zjBaABlp0kqbcwtfV5ztwWVa
fq77LN2d/ZjkXACm7T9bdXZuxsecY5jUaaOoSy3GS5SRcAMfoU5tK7USnFdneW3abrLKpqW3q3nGOZe
AIX1ddTKvOM9uDtUOqqjDOdqxkkAKtTT19Wzdt45zjJZCO2CjnOFg6AKdLp/R63Hduy85xgej/wDl9fu7
MbcFwAz36RW2KyE3XYu1Ea9H/uqy6x2SXLPI1ADk4qcJRfJrBVpdOtNBx3bm3lvGC4AU6nTRvw23Gce
UkVehTm1118pxXYawA5FeoohqIbZdnJrsLABjeinPEbNRKUF2YNLqg6eqxiGMYJgDHHQzS2LUSVfckWaf
S9RCcVPO75cjQAKtNR6PVs3buOc4wWgAAAAAAAAAAAAAMn+bZ5L+DWZP82zyX8BN8WgA0yAAAAA
AAAAAAARlJQi5SeEgMvSNHWU717Uf2PJPV9fWy4ZjQv/7FOr0Dj69Kyu2JFxgAaaeGCNAAAHpdF0pQd
r5vgirR6GVjU7ViPd3mmP8A4mp2/wD4rOXyZU3WwAFZAAAAAAAAAAAAPF11HU3tperLij2iu+mN9bhL
9H3EM14ILtRprKJYksx7GikjYAABKuDsmoRWWxXXOyW2EW2eto9ItOt0uM39gm6vprVVUYLsRMArIca
ysM6Cjw9XQ6Lmseq+KKT3dRRG+vbLn2PuPHv09lEsSXDsZlrNVAAKAEqqp2y2wi2wO01u22MI9p7tcFX
BRjySwY+j4RqlOuSxanx8jcVndAAVAAAAAAAAAAAAAAAAA8bX0dTc2l6suKPZK7qo3VuEu37EM14ILtR
pbKJcVmPYykjYAAB2uDsmoxWWzsK5WS2wi2zfXpLdKlbDEpL2ohK201qqqMF2FhVRfC6OY8+1dxaVkA
BQAAAAAAAAAPP6Q0tt1sZwWVjGO49AEFWmhKqiEJPLS4loAGTRySd2Wl/uPtI6lv0qlxlFLjhvlkvelolJt1pt
vLJdRV1fV7Ft7gKYVWS1CsnODwmmo9o0SXVz4L22XV0V1NuEcN9pKEIwTUVhN5Az6PalYnhPrGWWa
dWS3Oc18kzroqdm9wW7vLQM99T9ElCLcmlniSouhOmLUly4/IuKZaWmUtzgssCjUuXpNLjKKWHhy5ZL
IV2PURsnODwsYiXSqhKGyUU4rsOV011PMI4bAp0MV1djwsubyUpZ6Os4cpP9zdCEYJqKwm8nI1wjFxU
UovmgVRqOOiezHsrOCtwstoUesq2tLGDVXRXXnZFLPMitLSpblBZ5gVbE9et3FqBosrVkdrbXkzuyO/fj1sY
ySAqqoVcsqc5ebMyp67U3+vKLWOTNxGMIxk5JYcub7wMune2m2ppKcE8/P5lb4aXTOXsKS3G3q4bnL
astYbOTjGFLjGvcl/agVRqpwcqEmm965HYRT6QsbXFRWCuNSnZDZQ64xlubZsUIqbml6z4NgZ6kvTr1jg1
EaDatNFcN2WvuaFCKm5pes+bIKipWdYoLdzyBOz8OXkyWk92h+v7kbPw5eTJaT3aH6/uNXFwAI0AAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAABk/zbPJfwazJ/m2eS/gJvi0AGmQAAAAAAAArttjTBym/8A7IJWTjXBym8JGWMZ6y
W6eY0rlHvFdU9TNW3LEF7MDXyAJKKSSwkdAKKLtLVdxlHj3ozS6Lj/AG2P9UegCFedHotf3WP9EaadHT
U8qOX3s0AFCu+pXVOD/R9zLABn0lrnF1z/ABIcGaDJqouqa1EFxjwku9GmE1OClF5TAkACgAAAAAAAAA
AONKSw1lGWzo6mfGOYP5GsEHmvovjws4eRZDoytP15OX2NwBdQrqhVHEIpImAAABQAAAjKKmsSSa
+ZIEGOzo6mfGOYv5FP+l8fxOHkekAVir6Nqj7cnI1V1wrjthFJEwBl1dbi431+1Dn80X12RtrU48mSMlf/AI
up6t/h2cY/J9wGwAFAAAAAAAAAAAAAAAAA42kst4SDaim28JGNuWtniOY0rm/EQJSlrZ7IcKU+Mu87Z
0dTJermL+RqhGMIqMVhIkCvOfRfHhZw8icOjK0/Xk5G4AuoV1QqjiEUiYAGa/Tvd1tD22Ls7GS0+oVvqy
W2xc4svKNRp1biUXtsXKSAvBmo1D3dVetti+jNIAAFAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAARs/Dl5MlpPdofr+5Gz8OXkyWk92h+v7k1cXAAjQAAAIwshNtRkm1zw+Q6yG/ZuW7njPECQI
zshBpSkk3yy+YnZCtZnJRXzYEgQrurs9ial5M6rIObgpJyXNZ4gSBGdkK0nOSin3skABHrIb9m5bu7tE7IVpO
clHPeBIEZWQg0pSScuWe0j6RS5betjnzAsAKtRdGmuT3JSw9qfaBaCjSahXVR3Sj1nHKXmWWXV1+3OM
X3NgTByE42LMJKS+TOgA2lzeDHPWJaqEYzg62uL+pHpGSlXU4vKcuDQG4Fb1FMZbXZFPzLAAK5aimMt
srIp92SxNNZTymABGc4VrM5KK5ZZm0er66LVkoqecJd4GsEHdVFtOyKcefHkdjZCUN8ZJx7wJArjqKZy2x
si33ZLAAK/Sad23rY58ywAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAZP82zyX8Gsyf5
tnkv4Cb4tABpkAAAFV2oroWZy493aYLOk5v8OKS72Qj1AeM9fqH/evoSh0ldH2tsv0FWPSvvjTHMuLfJd
5TVTK2fXajn/bHuKNJbXba53S/wBzsT5I9IJ4AAoAAAAAAAAAADjSaw+TMlLemvdEvYlxg/4Nhm1yg6cyk
oyXGL+ZBpB5b6TnsSjBbscWyp6/UP8AuS8kKseyDx49I3p8WmvI10dI1zeLFsff2BI2g4mmsp5R0oAAAA
AAAAAAAAAAAAAAAAAAAAAAAAVailXVOPJ80+5loIM+kudkHGfCyHCSNBh1c46e+N0Wtz4Sj3ops6Tm/
w4qPnxCx6gPG9P1Gfb+xOHSV0X6yjJCkesDNRraruDe2XczSEAAUAAABlv11VPBPdLuRjn0ldJ+qoxRCP
WOSkoxcpPCR461+oTzv+wnrJXSiruMFzS4ZFWNnra2fbGhf/wBjZGKjFKKwkVae+q2KVbSx/b3FwQABQ
AAAAAAABVfRC6OJc+x9qKa7p0TVWo5f2z7zWQsrjbBxmspkEjpiUp6OW2eZ0vlLtibIyUopxeUwOgjKUY
RcpNJLtZiu6SjF4qju+bA3g8eXSF7/ALkvJHI9IXr+5PzQqx7IPPp6TTeLY4+aN0JxnHdFprvQSJAAoAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAjZ+HLyZLSe7Q/X9yNn4cvJktJ7tD9f3Jq4uABGgAAYqv9npKceyxZX
/AH6mffL0j0r+zrNv6f8A+F/SUXHq7Y8GsxyTen//AI7q8ett3frzA5Z/u9JQj2VrL/79BqY0R1HWX2Z4cIYO

dGpy6y2XFvEc+RCyS0+vlZbByjJeq8AVzsq9Kplp4uPrYfDBdqf9jXV3f2y4SKtRqOuupkotQjLg328jZrq+s00
u+PrICnUf7+vrq/thxf8A36G0xdGxct90+MpcMm0DFH+qy/L/AAOk/wAOv8xy+T0+vV0otwksZRTrNSr1F
QjLan7TXaBd0lHdZRF8m2v2LL9HT6PLZBRlFZTK+km42UNLLTbS7+Qu10Z1OFcZdZJYw1yAu0M3ZpYuT
y1wO6yuEqJylFNxi8PuO6Sp06eMZe1zZPURcqLIri3F4Az6GMIaRW7Vuw8vt5leipjepXXLfKT7SWguhKlU
NPck8kKLXonKq6L25zGSXMDsoLS66vq+ELODRvMEHLWauNii1VXyb7TeB591Na6RqgoRUXHiseZLpKK
VVUYrC3YQ1kuq1tV0k9qWOH6/8nNfNWU0zjycuAFl2kphpZ4jxjHO7tIRulDotST9b2U/1NWp92t/K/2M
tNTu6M2Lm84+oFVMtHGlKxbptcXhl3Rk8qytPMYvMSNOrrqrVd1bU4LHLmadLY7Yyk6tizw+aAsnXCxY
nFSXPDMXRlcJVym4pyUuD7jeefoLVTKVE01Ny4AchTC7pC1T4pccd53XuNfVUL1K+cseZPT/ANRv8iWu
qm3XdCO5wfFAZ756OVLVSxNcnglqL5Po+rjxnwb8ix66px9Spub5RwS1dU7tLBqG2ceO1AUSloupcFndj
hLHHJp6Pm56VZedrwVLXVbPWre/tjjtL43OOlds69rXHagLwQpsVtUZpNJ9jJgAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAyf5tnkv4NZk/wA2zyX8BN8WgA0yGfWalaevhxm+SL3wR4equd98pdnJ
eRDMVznKyTlJ5bOA7CErJqEVlt4RG3AelHoqKSVlyU32GPVaWels2y4p8n3gUnoaDWPKqsf5WeeE8PKA
+iBRpLuuojJ8+TLysAAKAAAAAAAAKr7o0Vucv0XeeLddO6e6b8l3GjpG7rL9i9mHD9TIZazABLLwuZ6NXR
XqKV1ig32BXnA06zRT0rTb3QfJmYDXotY6ZKE3mt/Y9ZcVlHzx63RtzspcHzhw/QuM7jYACoAAAAAAAAA
AAAAAAAAAAAAAAAAGbW6paeGF7b5fI0NpJt8keFqLXddKb7eRFzEJSlOTlJ5bOCMXKSUU232I3PoycdN
K2yW1pZ2kaYQEsvC5no1dFZgnbZtb7APOPS0GscmqrXx/tZl1mjnpZLL3RfJmdNpprg0DX0QKtNb11EZ9
r5+ZaVgPO1+scW6q3x/uZr1VvU0Sn29nmeG2223zY1cwBOqmd01GuLbNGr0D0tMZynlt4aI0yAJNvCW
Wbq+jLHS7LJbWllIDFCcoSUovDR7Gj1K1EMP21zR4xZprXTdGa/UJuPeBxPKTXJnTTIAAAAAAAAAONK
SaaymY5Rnom5w9antj3G087pS7GKl28WQxl1WqlqJ90FyRQC7TaWzUyxBYj2yfJEbUg0a3S+i2Rju3ZWR
pNJPVSe1qMVzbAzl2m1M9PPK4xfNGyXRKcH1VqlJdh5sk4ycWsNPDA+grnGyCnF5TJHmdF3Yk6XyfFHp
lY0ABQAAAAAAAAAAAAAAAAAAAAAAAAAAAARs/Dl5MlpPdofr+5Gz8OXkyWk92h+v7k1cXAAjQAAA
AADmAAMurlfJumqrhJY3moAQprVVUYLsRMAAAAMusrnZbQ4RbUZcflyNWOOQAAAAYQ58wAAAAc+
YAAAABhMAABjjkAAAAGFnIAAYWcgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAZP
82zyX8Gsyf5tnkv4Cb4tABplTq5OGmsa54PDPb1qzpLPLJ4hNawLtJZ1Wprm1lJ9hSbOiYxerW7mlw8yK3
6rQvUXxtjY4rhlGXpixOcK0n6vHLO6yesjq3s37V7O1cC3pNbtFCViSs4AeQAAPR6Jk8Tj2cz0TzeiVxsf6HpF
Y30ABQAAAAADkniLfcjpGazCS70QeBNuU232s4HzYI2t0ritTW5ctyPQ6VpusthKtOUcY4djPKSbaSWWz
0Vbr9Oo17XLK4cMgX6uEl0Slb7cUsnjnsdIzlDo+MLHmyWMnjgDX0ZJrU47JRZkNXRqzq0+5NhNewADTI
AAAAAAAAAAAAAAAAAAAAAAAACjWycdJY1zxg8Q9nXrOkn8jxia1i7Sah6a7eoqXDDR6kb5ajoy2yeMtPk
eNGMpyUYptvsR6+mrmuipwcWpNPhgivP6Pgp62pPlnJf0vZJ6pRy8RiinRt0a6vrE48cPPzNPSunslqFZCL
kpLHBdoFuofXdDxnLjJJPP64PIPX1S9H6JjVL2mksfc8gD0+iZN12R7E0zeed0SvVtfzSPRKxvrz+lpNQrj2Ntn
mno9LLhW/NHnEazxp0mtlpoTjGKe7k+429KNy0FUnzbTf0PLhXOx+pFyx3I9bpCuc9BTGMW2sZSXyCvL
01z090bEk8dh6+n1EtTpbZySXNJHiqMnLak92cYPX6PrnDRWxlFpvPBoDxwdnCVbxOLi/mcA9vRyctLW3
zwXmfQrGkh80aCsAAKAAAAAAAAB4etk5aqzPY8HuHhatY1Vn5mTVxUeho9dOKqoUUvWSb+R55fo6rJ
aiqag3FSXHBGmnpr3iH5f5MMLJ152SaysPB6HTFc5XRlGLcVDi0uRDorTRtsdk1mMOS+YGjo2mWnrnfc
9qa5PuPLvmrb5zSwpSbNuvnqNRNxjXNVp8FjmefKLjJqSw1zQFulk4amtrxJHung6dbtRWv/kj3i4zoACoA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAI2fhy8mS0nu0P1/cjZ+HLyZLSe7Q/X9yauLgARoAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAADJ/m2eS/g1mT/Ns8l/ATfFoANMuSipRcXyaweBbB12ShLmmfQGHpDS9YutgvWS4rvIua8s7C
coSUovDXJnARpvj0tco4cISfeZdRqbNTLNj5ckuSKgAANWi0rvnuksQX3A3dH1dXp02uMuJqOLgjpWAAF
AAAAAAAAHia2rqtTJY4Piig9nWaZaivh7ceR40ouMmpLDRlrNE3Fpp4a5G6HS10Y4cYyfezCAqzUaizUT3
WPPcu4rAAHpdFVYjK1rnwRi02nlqLFFcF2vuPbhBVwUIrCSwi4m6kACsgAAAAAAAAAAAAAAAAAAAA
AAI2RU65RfasHgTg4TcZLDTwfQmDpDSuf+7Besua7yaua8+i2VFsbIpNx7zZ/q93hgYARpZqL5ai3rJJJ/I01
dKXVwUWozx2sxAC3UamzUz3WPlyS5IqBr0Old01Oa9SP3A3aGrqtNFNYcuLNIBWGbX1dbpnhZceKPG
PoTyNdpXTPfFeo/sNXNQ0urnpXJwSe7vNH+r3eGBgBGk4XSjerkluUtxs/1e7wwMAAt1OplqZqc0k0scCF
cHZOMVzbInp9H6VwXWzXrPku4JrbCKhCMV2LBIArIACgAAAAAAAeV0pU42qxLhJcfM9UrvpjfU4S7eT

7iGPBNWm19mnq6uEYtZzxKLapU2OE1hogRtts6UusrlBxjiSwyvS62zSwcYKLTeeJmAG//V7vDAxW2O22
U5c5PLIkq65WzUILLYGroyrfqN7XCC+56xVpqFRUoLi+bfey0rGgAKAAAAAAAAAAAAAAAAAAAAAAAAA
AAAACNn4cvJktJ7tD9f3I2fhy8mS0nu0P1/cmri4AEaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAyf5tnkv4NZk/zbPJfwE3xa
ADTIAAMeq0ELnug9svszz7NJdXzg2u9cT3ARa+dcWnhponCi2x4jXJ/oe9hdyAhXnafo15Urnw8KPRjFRio
xWEuw6AlAAUAAAAAAAAAAAAM+p0kNQsv1Zd6NAIPFt0N1b9ncu9FDjKPOLXmj6E5hdwi14EarJvEYSf6
Gujo6ybza9i7u09TB0QqFVUKoKMFhImAEAAUAAAAAAAAAAAAAAAAAAAAAAAAAABi1Ogja3Kt7ZfZm
CzSXV+1Btd64nuAi1864tPDTyWQotseI1yf6Hu4XcgIV52n6N4qVz/9UeikopJLCXYdASgAKByUVJNSWUz
oA83UdGvLlS//AFZjnRbW8Srkv0PeOEi189ht4w8l1elus9mDS73wPbwu5HRCsem0EampTe6X2RsACAA
KAAAAAAAAAAAAACq+iF8Ns15PuPNu6Ptg/U9dfLmeuCFfPSrnH2oyXmhGE5ezFvyR9Bhdwwu4Ra8ino
+6x+sti+Z6Wn08NPHEVx7W+bLgEoACgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACNn4cvJktJ7tD9f
3I2fhy8mS0nu0P1/cmri4AEaAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAyf5tnkv4NZk/wA2zyX8BN8WgA0yAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAjZ+HLyZ
LSe7Q/X9yNn4cvJktJ7tD9f3Jq4uABGgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMn+bZ5L+DWZP82zyX8BN8WgA0yAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AjZ+HLyZLSe7Q/X9yNn4cvJktJ7tD9f3Jq4uABGgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMbaWtsy0uC5/obCuenqnJylHL
fzYFe+PiX1G+PiX1JeiU+D7seiU+D7stZiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+
D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU
+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7sei
U+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7s
eiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7
seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D
7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+
D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU
+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1Jei
U+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1J
eiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX
1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+Pi
X1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+
PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G
+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1
G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX
1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+Pi
X1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiO+
PiX1G+PiX1JeiU+D7seiU+D7sUiO+PiX1G+PiX1JeiU+D7seiU+D7sUiuycerl6y5PtLNJ7tD9f3HolPg+7LYQj
CKjFYSIuZHQAFAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAf/9k="></figure>
Constraints : 0<size <100 0<data<1000 Input format: First line indicates the size of the queue Second line indicates the elements of the queue. Output Format: single line indicates the in-order traversal of a tree.

answer

```
 #include <stdio.h>
#include <stdlib.h>
struct node {
  int data;
  struct node *left,*right;
};
void solve(){}
struct node *root = NULL;

void insert(int data) {
  struct node *tempNode = (struct node*) malloc(sizeof(struct node));
  struct node *current;
  struct node *parent;

  tempNode->data = data;
  tempNode->left = NULL;
  tempNode->right = NULL;

  //if tree is empty
  if(root == NULL) {
    root = tempNode;
  } else {
    current = root;
    parent = NULL;
```

```
      while(1) {
        parent = current;

        //go to left of the tree
        if(data < parent->data) {
          current = current->left;

          //insert to the left
          if(current == NULL) {
            parent->left = tempNode;
            return;
          }
        }  //go to right of the tree
        else {
          current = current->right;

          //insert to the right
          if(current == NULL) {
            parent->right = tempNode;
            return;
          }
        }
      }
    }
}


void inorder(struct node* root) {
  if(root != NULL) {
    inorder(root->left);
    printf("%d ",root->data);
```

```
    inorder(root->right);
  }
}

int main() {
  solve();
  int n,i;
  scanf("%d",&n);
  int array[n];
  for(i=0;i<n;i++)
  scanf("%d",&array[i]);

  for(i = 0; i < n; i++)
    insert(array[i]);
  inorder(root);
  return 0;
  printf("temp->left=temp->right=NULL; struct node* newNode(int item)");
  return 0;
}
```

question

Question description There are n hotels on a street. For each hotel you know the number of free rooms. Your task is to assign hotel rooms for groups of tourists. All members of a group want to stay in the same hotel.

The groups will come to you one after another, and you know for each group the number of rooms it requires. You always assign a group to the first hotel having enough rooms. After this, the number of free rooms in the hotel decreases. Constraints 1≤n,m≤2·10^5
1≤hi≤10^9
1≤ri≤10^9
Input

The first input line contains two integers n and m: the number of hotels and the number of groups. The hotels are numbered 1,2,…,n.
The next line contains n integers h1,h2,…,hn: the number of free rooms in each hotel.
The last line contains m integers r1,r2,…,rm: the number of rooms each group requires.

Output

Print the assigned hotel for each group. If a group cannot be assigned a hotel, print 0 instead.

answer

```
 #include <stdio.h>

#define N    200000
#define N_    (1 << 18)    /* pow2(ceil(log2(N))) */

int tr[N_ * 2], hh[N];

void build(int k, int l, int r) {
   if (r - l == 1)
      tr[k] = hh[l];
   else {
      int    m = (l + r) / 2, k1 = k * 2 + 1, k2 = k * 2 + 2;

      build(k1, l, m);
      build(k2, m, r);
      tr[k] = tr[k1] > tr[k2] ? tr[k1] : tr[k2];
   }
}

int update(int k, int l, int r, int x) {
   int    m, k1, k2, i;

   if (r - l == 1) {
      tr[k] -= x;
      return l;
   }
   m = (l + r) / 2, k1 = k * 2 + 1, k2 = k * 2 + 2;
   i = tr[k1] >= x ? update(k1, l, m, x) : update(k2, m, r, x);
   tr[k] = tr[k1] > tr[k2] ? tr[k1] : tr[k2];
   return i;
```

```
}

int main() {
   int n, m, i;

   scanf("%d%d", &n, &m);
   for(i=0;i<n;i++)
      scanf("%d", &hh[i]);
   build(0, 0, n);
   while (m--) {
      int x, i;

      scanf("%d", &x);
      i = tr[0] >= x ? update(0, 0, n, x) + 1 : 0;
      printf("%d ", i);
   }
   printf("\n");
   return 0;
}
```

question

Question description You are given a tree consisting of n nodes, and m paths in the tree.
Your task is to calculate for each node the number of paths containing that node.

Input

The first input line contains integers n and m: the number of nodes and paths. The nodes are numbered 1,2,…,n.
Then there are n−1 lines describing the edges. Each line contains two integers a and b: there is an edge between nodes a and b.
Finally, there are m lines describing the paths. Each line contains two integers a and b: there is a path between nodes a and b.

Output

Print n integers: for each node 1,2,…,n, the number of paths containing that node.

Constraints 1≤n,m≤2·10^5
1≤a,b≤n

answer

```
#include <stdio.h>
```

```
#define N    200000
#define K    17    /* K = floor(log2(N)) */

struct L {
    struct L *next;
    int j;
} aa[N];

int dd[N], pp[N][K + 1], ll[N], rr[N], cc[N];

void link(int i, int j) {
    static struct L ll_[N * 2], *l = ll_;

    l->j = j;
    l->next = aa[i].next; aa[i].next = l++;
}

void dfs(int p, int i, int d) {
    struct L *l;
    int k;

    dd[i] = d;
    pp[i][0] = p;
    for (k = 1; 1 << k <= d; k++)
        pp[i][k] = pp[pp[i][k - 1]][k - 1];
    for (l = aa[i].next; l; l = l->next)
        if (l->j != p)
            dfs(i, l->j, d + 1);
}
```

```
int lca(int i, int j) {
    int k;

    if (dd[i] < dd[j])
        return lca(j, i);
    for (k = K; k >= 0; k--)
        if (1 << k <= dd[i] - dd[j])
            i = pp[i][k];
    if (i == j)
        return i;
    for (k = K; k >= 0; k--)
        if (1 << k <= dd[i] && pp[i][k] != pp[j][k])
            i = pp[i][k], j = pp[j][k];
    return pp[i][0];
}

int dfs2(int p, int i) {
    struct L *l;
    int c = cc[i];

    for (l = aa[i].next; l; l = l->next)
        if (l->j != p)
            c += dfs2(i, l->j);
    cc[i] = c += ll[i];
    return c - rr[i];
}

int main() {
    int n, m, h, i, j;

    scanf("%d%d", &n, &m);
```

```
    for (h = 0; h < n - 1; h++) {
        scanf("%d%d", &i, &j), i--, j--;
        link(i, j), link(j, i);
    }
    dfs(-1, 0, 0);
    while(m--) {
        int i, j, a;

        scanf("%d%d", &i, &j), i--, j--;
        a = lca(i, j);
        if (i == a) {
            ll[j]++;
            rr[i]++;
        } else if (j == a) {
            ll[i]++;
            rr[j]++;
        } else {
            ll[i]++;
            ll[j]++;
            rr[a]++;
            cc[a]--;
        }
    }
    dfs2(-1, 0);
    for (i = 0; i < n; i++)
        printf("%d ", cc[i]);
    printf("\n");
    return 0;
}
```

question

Question description Jegan Sir students were chatting and playing quite loudly on the last day of the year, celebrating the end of the academic session. Jegan sir was harshly chastised by the college's principal. But, instead of becoming enraged, he attempted to engage everyone in a different task.

So Jegan sir gave his students to solve the task such that, you have to perform pre-order tree traversal in Binary search tree. <figure class="image"><img
src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAMiIlLCUfMiwp
LDg1MjtLfVFLRUVLmW1zWn21n767sp+vrMjh//PI1P/XrK/6//3///////B8P/////////////bAEMBNTg4S0
JLk1FRk//Or87//AABEIAyQFFAMBIgA
CEQEDEQH/xAAZAAEAAwEBAAAAAAAAAAAAAAAAAgMEAQX/xAA3EAEAAgEBBQUFBwUBAQEBAAA
AAQIDEQQSITFRE0FSkZIyM2FxgRQiU2JyweEFNEKhsSNDFfD/xAAYAQEBAQEBAAAAAAAAAAAAAAA
AQIDBP/EACARAQEBAQACAwEBAQEAAAAAAAABEQISIQMxQRMiMlH/2gAMAwEAAhEDEQA/APUAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
ELXtv7ta66Rrz
0N7J+HHqBMQ3sn4ceo3sn4ceoExDeyfhx6jeyfhx6gTEN7J+HHqN7J+HHqBMQ3sn4ceo3sn4ceoExDeyf
hx6jeyfhx6gTEN7J+HHqN7J+HHqBMQ3sn4ceo3sn4ceoExDeyfhx6jeyfhx6gTEN7J+HHqN7J+HHqBMQ3
sn4ceo3sn4ceoExDeyfhx6jeyfhx6gTEN7J+HHqN7J+HHqBMQ3sn4ceo3sn4ceoExDeyfhx6jeyfhx6gTEN
7J+HHqN7J+HHqBMQ3sn4ceo3sn4ceoExDeyfhx6jeyfhx6gTEN7J+HHqN7J+HHqBMQ3sn4ceo3sn4ceo
ExDeyfhx6nN+8TG9SIiZ05gsAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BCP7i36Y/dNCP7i36Y/dMAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABDL/h+qE0Mv+H6oB
MAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHL2i
lZtIFrRWNZnSGe+0zPCkafGVWTJbJbWeXdCCOXXyf+JWve3O0o8esgOflUq5L15Wlfj2nuvGnxhmBZ3Y9
GJiY1gYsOWcc6c69G2JiY1jkrvz1KACgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIR
/cW/TH7poR/cW/TH7pgAACvPMxj1idEK48lqxMZJ4/EZvXvF4px5LRfcvzKadvbTXUPJcKpz1iZjSeCVsta1i
Z7+4XyiYrrmradNJifilfJWkcQ2JCuuetp04x80bZ9Mmmv3e8Tyi4U55ralZ1nSeizerTHEzPDQXfaQqjaKzP
GJhK+SKREzx16B5RMVfaK68p+a2J1jWAllBDJkjHprrxTjjGougjktuUmVGLJaLxvTOk9Rm9ZcaRHJeKV1m
EPtFeki2yLRC2SIx70Tz5I4ssXjS0/eDym4tFNNPtFtJnVcEugo3r5bTFZ0iEq1y1vHHegTyW6iif7pZfLWk6c5
+AeSYhTLW86RwnpJfLWk6TxnpAuz7TFdctb6xHCekq9nmIm8yJ5NAhTLF5mIieHemLLohl/wAP1Qmhl/
w/VAqYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
DJtOTevuxyhqvO7SZ6Q8/nx6jn8lyACOAO103o15NFcOK0a1mZ+o1OdZh20aWmPi4INGy5OM0n6M6
VLbt6z0ka4uVvAV6AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEI/uLfpj900I/uLfpj
90wAAVbT7v6u4rVjHGswnekXjSVf2enx8xmy7sQie02iJjlDuP8AuLLq0rSNKw5GOsXm0a6yJ41TiiJz21cz
cM0ctPivrjrW02jXWXbUreNLQHj6xntFr2j2Yn4S7fhnjf5La4aVnWIdvSt4+9AnjXN7Hvxy1U3iI2iNeS6uGl

Z1iOPxdvjrfmLZbFW0abldOTmblj15LZxVmsV46QlNImu7MawF5tU5ppOPhpr3I5NeypqtjBSJ10SvjrfTX
uEvNqvLWIwcuieH3VUrUi1d2eRWsVrERygak96jkmkab/0TjkjfHW+m93Jchf1RtFptaKRxQyb01j7kxu97
RGOsX3uOqUxExpIz42qMtt/BEu7sfZ+Xcl2Nd3d46J7kbm73Bl/VGONdnn4O7Pu7vdrqtrSKV0jk5GGkW
3o11DxvpXj/ALm6+0a1mPgjGOsXm0c5SFkxnwWikzW3CV3aU3orrxly2Kt51mOJXFSs6xHESSz0rn+6Rp
pGed//AGv7Ou/v8dS+Ot+ccQ8apvpOeNz/AE5TSM9t/wD2vpjrTlBfHW/OOIeN+0YnHNp3dN5miLTvac
u9rpirTlBTHWkzp3hebUcFqzTSOExzWIVxVrbejXVMam57EMv+H6oTQy/4fqgVMAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFef3NvkxN+WN7HaPgwRyRy
+QAHEatm93PzZWnZvdz8xvj7UX9ufmilf25+aIzQnkOxGsxHWQn2319mPk6RGkRAr1AAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIR/cW/TH7poR/cW/TH7pgAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAIZf8P1Qmhl/w/VAJgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAMOWm5kmO6eMNynNXteFecd4z1NjIExpOk84EecAEAAFuzxE
5dZ7ldaze0VjnLbXFWtN3/AGOvx8/qYhFppOluXdKauwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAACEf3Fv0x+6aNsdbW1nXXlwnRzsafm9UgmIdjT83qk7Gn5vVIJiHY0/N6pOxp+b1SCYh2N
PzeqTsafm9UgmIdjT83qk7Gn5vVIJiHY0/N6pOxp+b1SCYh2NPzeqTsafm9UgmIdjT83qk7Gn5vVIJiHY0/
N6pOxp+b1SCYh2NPzeqTsafm9UgmIdjT83qk7Gn5vVIJiHY0/N6pOxp+b1SCYh2NPzeqTsafm9UgmIdjT
83qk7Gn5vVIJiHY0/N6pOxp+b1SCYh2NPzeqTsafm9UgmIdjT83qk7Gn5vVIJiHY0/N6pOxp+b1SCYh2NP
zeqTsafm9UgmIdjT83qk7Gn5vVIJoZf8P1QdjT83qkjDSJiePDrMgmAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEzERrKGk5OfCvTqBrN+FeFe+U4iKxpByAV5c
MZOPK3VkvjtT2o+reDPXMrzhttgx2/xj6OfZsfSfNHP+bGnTFe/KNI6y11w468qxqmrU+OfqGPHXHHDn1
TAdCYiY0lDjj+Nf8AiYBE6xrAhMTTjXjHfCVZi0awDoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADlrRWNZctbd4RxmeUFa8d63G3/Acis2nev9I
TAAAAAAAAAAAAABG1Zid6vCe+OqQDlbRb4THOHUbV14xwmO8rfWdLcLAkAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAIWvbtN2tYnSNeM6Ob2XwV9X8AsFe9l8FfV/BvZfBX1fwCwV72XwV9
X8G9l8FfV/ALBXvZfBX1fwb2XwV9X8AsFe9l8FfV/BvZfBX1fwCwV72XwV9X8G9l8FfV/ALBXvZfBX1fwb2
XwV9X8AsFe9l8FfV/BvZfBX1fwCwV72XwV9X8G9l8FfV/ALBXvZfBX1fwb2XwV9X8AsFe9l8FfV/BvZfBX1
fwCwV72XwV9X8G9l8FfV/ALBXvZfBX1fwb2XwV9X8AsFe9l8FfV/BvZfBX1fwCwV72XwV9X8G9l8FfV/AL
BXvZfBX1fwb2XwV9X8AsFe9l8FfV/BvZfBX1fwCwV72XwV9X8G9l8FfV/ALBXvZfBX1fwb2XwV9X8AsFe9
l8FfV/BvZfBX1fwCwV72XwV9X8G9l8FfV/ALBXvZfBX1fwb2XwV9X8AsFe9l8FfV/BvZfBX1fwCwV72XwV
9X8G9l8FfV/ALBXvZfBX1fwb2XwV9X8AsFe9l8FfV/BvZfBX1fwCwV72XwV9X8O1vab7tqxHDXhOoJgAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAI2tOu7XjP/HJtN50ry75SrWKxpAFaxX
4zPOXQAAAAAAAAAAAAAAAActWLRx83QEItNZ3b/SeqZMRMaShrOPhPGvXoCYAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAIR/cW/TH7poR/cW/TH7pgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAIf/eP0/umh/8AeP0/uCYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAACN71pGtpMuSMddZ590MVrWvbW08RnrrxWX2m1vZ4Qqm1rc7TLgjje7ROuS9eVpQBPKtWPaY
nheNJ6r+bzluHNOOdLca/8V05731WyZ0jWUOOT4V/6RE30m3LuhMdSIiI0gAAAAAAAAAAAAAAAAA
AAAAAENJx8Y416dE4mJjWOQyZ8ml5rjnSO8S3Pa3LtEU4V4yz2zZLc7afJARxvdpz5uxMxymYcBnatptF6
8/vQ048tckcOfRhdiZidYnSRvn5LPt6Arw5e0rpPtRzWK7fYAAAAAAAAAAAAAAAAAAAAAAAAAAAAACEf3F
v0x+6aEf3Fv0x+6YAKc171vEVnTUS3FwotbNSNbaTCztaxSLTwiRJ1ExybRFd7uQyXi2GbVkW1YIYp/wDK
szPmdtTXTUNiYI2vWmm9PMVIFWXLNbbtY1kS3Fop1zV4zGsdHc17Vx1mOEyJ5LQrxrE/BTmvatois6ai2
5NXCi1s1I1tpMLqW36xPUJddEM1t3HMxzRpbs8O9aZmQ33i0URbNeNa6RCzFa06xeNJjvEnWpiE5qRO
mqU2iK72vAXY6IdtTTmlFqzXWNA2OivHli+uukJWvWntSGbEwiYtGcgWcSmXXNANlS06KOe+7XSUJlhibTtMSgj
eVOoAxXv2u7W4WNXLN+U2ZRR6lNeSNE6T66sabH2qcHAHEBLCVAL7nNrWRGbFUmFeTTZLWmJ9H1J
Mle6J6C7PpMU4rTXFSw6IBHmtVBcAmNa/OtOJfAUUamoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAACGa27itPeDLmvv5JnujkrBHmt26Auw2pWJm0cRJNU6DXTLGSd3RRmrFckxHIavOTYrZc
mtZpPOOS9hw23ctZ+jcr08XYADQAAAAAAAAAAAAAAAAAAAAAAACvPfcx8Oc8IYl21W1yadIUo4fJfeAA5
mjui6medIrFde5de+5TWefQbnMs+2IAYSpaaWi0LJLz2vZa4tOkjt8d/FwCuoAAAAAAAAAAAAAA

AAAAAAAAAAAAAACEf3Fv0x+6aEf3Fv0x+6YDPtE6ZKy0IXxRe0W100GepsU5Ms3jd3dNXctd3DWOi3Jjj
JHHhMd7s0iabtp1E8b7QtMfZ/ohH9pP8A/d6X2ePFKfZR2W5r9RMqq0zGzV0c01x6Rj+uq+McdnuTxhX
9njxTp0C81PDvRjiLczJWlpjeTiNI0QyYoyTEzOmg1npNn13dp1s0I5MdckcQ6muzaIjWZVbT7EfM+zx32
mU74ovWK66RAl2xKvsx8lG0TpkrKX2f88pWwxO797kF2zFWTLN43d3TVfjru0iHMmOMkceE9UqxpWI
mdRZLvtVtPsx83M0T2NdO5PPXex/Li7imL4o1+Qlm2wxWiccaGS0dnbdnjCE7PGvC0wnTFWkTHPUX39
Kcfsadnrr36u1i0YLxMcO5KdnjXhaYhZ2cdnuajM5qrFjrOGZmNZcwRFq2rPJdSm5Td1cx44x68ddRZz9Kd
npW0zM908HNdc8613vgt7CN7WLTDt8MXnXXSeonjcV1i3bRMVmsd7kVi20WieWq3Hiik66zMuxiiMk
315h41y82ppFK6wjebWwTNo0lc5eu/WY10GrEMHuoQ2mZ+7Hc79n0/wA5S7KNzdmdfiJlsxVeJtWN3
HpPUz1+5WZj73KVkbPET7UpZMfaREa6aCeNxXfHWMOsQ7grG5vacVs1ia7s8kKYdy2sWn5C+PvXIvk3
oiacNVeSd3aNdNWlCcUTk39foFlU5LzlmK6aO540ikdFuTFF5iddJjvL4t/TWeQl5vtVm95WJ9l3PWkUjTT
VLNNYiItEz8lFopppTWZE69ank9xRK2OsYNdOOic4t7HWszpolNNce5qL4qdN7Zome5LZ6xu73esrjiuPc5
wjTDFLaxafkLnuVCf7qF6jH9/PNu6F4vIh/wDeP0/umh/94/T+40mAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAp2r3P1XK9ojXDPw4iX6YgEeUdrE2nSHF2HJSkcYnUWTb7WVrXBTWebPe03tMyvtlxW9
qJn6Kcs0mY3I0G+vr0gAObse1HzegwY43slY+Lerv8f0ADoAAAAAAAAAAAAAAAAAAAAAAAw5vfX+aC3a
Y0zT8Y1VI8/f/QAMJ47dnbWYaK2rmrOsIVzU3Yi1S2esV0pA6zJ+qLRu2mOjjszrOrg5jTsf+f0ZmrZI0pM9
ZG/j+14Cu4AAAAAAAAAAAAAAAAAAAAAAAAAAAACEf3Fv0x+6aEf3Fv0x+6YAAAAAAAAAAAAAAAAAAABE
RHKNAAAAAAAAAAAAAAAAAAAAAAAAAAmInm5FYjlEOgAAAAEREco0AAQ/+8fp/dND/AO8fp/cEwAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACY1jQAYL13LzWe5Fr2mkWrrHtd3xZEefvnKADAAACWOk
5LbsfWegsmrtlprM3n5Q0uVrFaxEcodV6ZMmAAoAAAAAAAAAAAAAAAAAAAAAAACnaqa0i0dzI9GY1jS
WLNj7O3D2Ry+Tn9VgI4gAAAERMzpHOW/HXcpFeijDhtpF+U90S0VtFvhMc4V6OOcjoA2AAAAAAAAAA
AAAAAAAAAAAAAAAAhH9xb9MfumjbHW06zHH5udjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY0
6T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnI
JiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydj
TpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06
T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJiHY06T5ydjTpPnIJ
of8A3j9P7nY06T5y7XHWs6xHH5gkAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAja+nCONuha0
zO7XjPXo7WsV+MzzkHK10netxshmwRfjHCy0CzWC1LUn70IvR580LYcdudY+g5X45+MI2fZsXSfNOuOle
VYhD+bLjwXvPHhDTGKK1iK8JjvTFdJzJ9OVtrwnhMc4dRtXXjHCYdrbe+ExzgV0AAAAAAAAAAAAAAA
AAAAAEbW04RxtIFrTru15/8ADcjd3Z4683a13Y+M85dBkybPavGnGnOimeHPg9Fy1K29qIkYvErzxsnZ8c/
4/7djBjj/HzRj+bHWs2nSsTLTi2fd434z0XxERGkRoK3OJBG1d7jHCY5SkDaNbcd23C3/UnLVi0aSjFrOl/pI
JgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAATMRGsgITM34V4R3yccnwr/ANTiNI0gH
K1isaQ6AAAAAAAAACNq68a8LQkA5W29Hx74dRtWdd6vf8AXa2i0f8AY6A6AAAAAAAAAAAAAAAAAA
DlrbsdZnlAOWtpwjjM8odrXd58ZnlZnLla6cZ42lIAAAAAAAAAAmImNJAENZ8x8+NevRaTTjXjXXvgEwY
tGsAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAOWtFfjM8oAtaKxrMKMVm8625d0O1rx3rcZ/
wCJAAAAAAAAAAAAAAAAI2rOu9Xn/ANSAcraLRrDqNqzE71effHV2totGsA6AAAAAAAAAAAAAADlrRWN
ZAtaKx8e6HK1nnXjaXL9uf8AxIAAAAAAAAAAAAAEJrMTvU59598dUq2i0axyB1NeMcCLAkI1
vrOk8LdEgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABTk2iteFeMqLZslv8tI6Q3M3qRtHn71vFPm7GS9eV6fd6
RvGWm0zHC8a6/0yrWnZZ73sTmncAKRamdsaxnRrcNNpPn/AM7s6F5rAlgUwS1Z8Ns/46kVrsUO5NqPd
AAAAjaJrO9X5x1TpeLRrCqtpm3anLP3hfWsRGkCtzdpEAHI36vOQeMDVR/JRAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAMufPMzu0
5d89Vm05N2mkc5ZBz76z1ABHAAASpe1J+SH1J1lzrzIGQNmI0xYa
AAAAAAAAAAAAAAIZckY66c/OzjKeQIVOk2Kev3o9CWqL7uXTutr0m5T2+Jb3Sr9rZf2Ax9HvCfTTCnk5TXR4wXD8yhLM/3mnWodUd4L4kCkjKWP9t8z2uXwvLdqlWt6CkRsfPPhIjrr6f7k2eRl4nC9U4LlN8tmPUCyuCkWdQNqbGAl+MtPBsBtvvOZ2fObj3Dod2vS2BT6KXo8UBbYpmHUnxS5oaYKCh0hrOC0Yw2AAAxjSZQAAAAAAAAAAAAAAAAAA

zhbMxETM9yvSrzZYx162nlDHa02nW06y7e83tNpRRw7630ADmAANODNr9y/PulmBvnrHoiGG/aY4me
cc01egAAAAAAAAAAAAAAAAABHJfcpNgV7Rm3Y3a85/wBMjszMzrPOXEefrrQAYAAdiZidYnSWvBm7
SNJ9qP8AbG7W01tExzgb46x6A5S0XrFo73VegAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABiz23ss9I4K3be3b5y4jzdfYCWOk3toI
nhxb86zyhzLu72lI0iF+SLVpu0qhbB/5xMRO8Ol59ZGcdmJidJcHMbsFt7FWe9ha9k91PzHT41wCuwAAA
AAAAAAAAAAAArz23cU9Z4LFG1+7r8xL9MoCPML9miJtbWFC7DkrjidYkXn7TtmrW0xuclF7Ra0zEaNF
ZxZJ03eKnNTcvpHKRrrcVgDmt2e27ljpPBdtVtMWnWWbH7ynzhdtn+H1V35v+WcBHAWYsU348oVtlY/
8ojXTgN8zar7HHrpv8VFoiJmInWGiMWO2u7adVF6zS0xIdREAYX7LbS816w1MWD39W1Xo4/5ABsAAA
AAAAAAAAAAAAAZtrtxrX6tLHtXvvoM930qAR5hdXZ5tWJ3o4/BSstmmce5p9RqZ+p/Zp8UeSrJTctu66t
GCu7jm097Pe29aZGupMRAHNq2S33Zr0Xsuye8t8mpXp5+gAaAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAYMsbuW0fFFo2qmkxe
PlLOjz9zKJVvNJ1rOiIMre3yeL/S7Jk0x61tGrIDU6rszMzrPNwBkbNmjTDHx4slYm1orHOW+sbtYjorr8c/X
QB1AAAAAAAAAAAAAAAFO1Rri16TquctXerMdQrzx2YmszE84cR5bMF2LJStdLQpAlxpjLirxrHH5Kcl5v
bVAFvVoAMp4Y3stY+Oq/a4+7Weko7JTjN5+UL8tN/HNVejmf5xgD5iOA1YrxfHuzOk6aMoLzcaseLspm0
2UZbRbJMxyR1nrLgXr1kABlbs0a5o+EatijZaaVm0969Xp5mQAGgAAAAAAAAAAAAAAAABk2uNMkT1hr
VbTTex6xzjiJ1NjGAjyiVK714hF2tprOsTpIsac9t3HFY72V21ptOtp1cF6u0AEaNkj71p+jSr2eu7ijrPFYr0yZ
MABQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAHLWisayDmSK2pMW5MNqzWdJ+jbFZtOt+XdDuTHXJXSYGeufKMAsyYb07tY6wrR
wss+wAZB2tZtOlY1acOz7v3r8Z6DfPFrmDDO7vTwt3fBdW2vC3C0JOWrvR0mOUq7yY6I1trwnhaEhQA
AAAAAAAAAAAAAAAFG04tfv1jj3sr0VGbZ9771OE9Bz7433GUdtE1nS0aS4jjZgAAljpOS2kfWUseG9+7
SOstdKVpXSsDpzx+12tYrWIjlDoK7M204tJ36xw72d6LNl2fjrj8hz7433GcJiYnSY0n4iOOYAALMOOclvhH
OXceC1+NuENFI7OYp3TykdOeP2rIiIjSOQCuwAAAAAAAAAAAAAAAAAADHnxdnbWPZn/Spuv8AfncjI
3s+TZ5rxpxjojl3x+xSHfp3g5AEcZ0jjIC7Z8W/O9aPux/t3Fs8zxvwjo1RERGkcldeOM91CYmnGvGO+EqzF
o1h1G1Zid6nPvjqOqQ5W0W+cc4dAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAABG1uO7XjP/Adtbd4c5nlDla8d63Gf+FaxXjzmecpAAAI
Ww0tzrCYCn7Lj/N5uxs+OO7X5FYrHCNHQFcFAActXe+ExylgysteE8xjzmecpAAAI
AAAAAAAAAAAAAHJrFo4xqrnZsc92Nb9a2ny+adcVKcqmwCYACgAAAOWrW0tjeiJVzs2Oe7WtW+Uot
BT9mx/HzWVmanyqxHNcmCnOPz4PEXtW9cbntXYuM9y2XFrIkNnOzaddVrhuZWBTbttZnbm0bk65ejNE
vn+8sqqoWnhjPrcMmLEeS6UOtBZgSPrN+9zG5xtkRxZJjpHSaEZmFQnvFV8pbQ7XhVEKfoYqjcOv+Yeqeud7
6epH5ZNZNf8Cw/pMzMPXuzVTQvVQCiPgr/HdWdjt6GHmqlbTscDRUTzW5Yq0kSGc0M+M+SNcNNYn0N
z0nT8D81TIYUN7ZgMyN+KPjSBtSkc1Cz+uQnp9sWahpKN9nhn+5VdAXBLuGavl2Pcg2q37g5GPeePE0B
PgDaVzqN1M2tQXUp9yCGVN9YLCeWdxNoNppS/SNkGbd2SaBqCuSzpdsnMSfgqgVm6bFI+3nTgxZ
Rh6DTUbRjyW77DPaV+S5mTGtX6I4AHNHfHmoVNnq5CNyy9BjJBgcdGhjrk4C7Sm6lOlhHzWqXPsm
mx7Eb6Qtzq0Y3+QaibH+BnbMJxZLmbHQaMfCYbOgNHYwRq/ID0ZzFCtQVbUE+CMy6acUmsOn9CqMHYk
OiUKrNF4Hq7MUa5eIzMTE3qJyd00e8blTfqxEj7q7ddjZqsWOhQq3HEpn+1Mj8gBV0cJgvjlM/8rLR5MBmk
4HWLWrw0ZZUXxMm4bcT8Nx9U8FO3/3gzzQgdXymrBn9uLqnJAxfEsMZXUo3Asms8QdoxnL+h2gdjAY2kcVp
RzvmjKz2ZkcEcyf0CRWHZmNfMgRmxTRhFtSA3OJi/ROrZtjo1iM7u/jlhDqp+oNIM1NAxXbPMAV7HFAp4PJzW
5ZPRLfbyhQL4J3ltYMjr8TC54xC8V7Z6shWFScnA1H7v2fKoFcZ1nCNw1KThmkSaybJX9nnx/gMTTHFQEy

NY4w85ds+XctuzylXXjv8rWAOoAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
x7RfeyzHdHBsnk86Z1mZ6yMfJfQAjzi6uzzMazOiqJ0mJa71nJjjSdBvmSqbbPMRrE6qWylezx8Z1ZJ4zMh
1JHHaW3bxaO5wGZcr0YnWIkQwTrhrKavUAAAAAAAAAAAAAAAAAAAAMe033sundVsefM62mesjHyX
I4AjzuxEzOkc10bNOnG0QhgmIyRqtz472tE14wOnMmaqyYpx/GOqtsmv/lpPGdGMTqYO1tNbRaO5wG
Zcr0YnWImO8V7POuGvksV6gAAAAAAAAAAAAAAAAAAAACZ0jVgvbfvNp72zPOmG0/BhHL5L6wSpX
etEa6aoiOS+dm4a73+lNaza2kc2rD7lDZo42kdLzLmORs06cbcVeTHNJ0l3Jkt2k6TMaSvzRvYdZ+YmS7jIt
2e+7liO6eCo10mJ6SM83K9Fj2i+9lmO6ODXr93X4PP11mZ6q6/JfQAjgsxYu0ifvaO5MO5XXe1+iuvtR82n
aPdwNySxCNm1/z/ANKr13bTGuui7Zf8lWb3tgsmbEEqW3LxaO5EGZcr0YnWNRDBOuGs/BNXqAAAAA
AAAAAAAAAAAGfa78Ip14y0MW0Trmn4cBnu5FYCPMtx4u0jXe0MmHs4idddUMft1+a/afZj5jckvKFdn
mY1mdC+CaxrE6odpe0RGstFNaYvvyLJzWQAYbsN9/HE9/emz7JPC0dJaFemXYACgAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAE8pedHDg9FhzV3cto+o5/JPSACOCVJiLxM8tWnLW16xuS
yJRe1eVpgbnWemrFFqUnflktxtOnV2b2tzmZRDq76ANNZ06jMbdnjTDVY5WN2sR0h1XqAAAAAAAAAA
AAAAAAAAAHncpmHosOau7ltHXiOfyT0gAjg7WJm0RHNoimaOG9wZonRPtL+KRqWRoyW7PHprrLI7
MzM6y4HV0A58BGzZo0wwtcpXdpEdIdV6gAAAAAAAAAAAAAAAAAAAAFe0ccNmJ6Fo3qzHWHnzGk
zHRHL5IADi14fcqtnvFbTE96uL2iNItOiI35fTTfBNr6xPCTPaK03I5qIyXiNN6XJmZnWRb1PxwkTxV3stY+O
ozzNrb/h9Hnw9Fhy13cto+OquvyT0gAjg7X2o+bTtHu4ZUrXtaNJmZgalyWLtl/wAlWb3tnK2tX2Z0cmZm
dZ4yFvrHAIjWYjqJG3Z40w1WOVjdrEdIdV6gAAAAAAAAAAAAAABiz+/s2su110tFuvAY7npQAjzpY/br8
1+0+zHzZ4nSdYdte1uczI1Lkxow4t2N6eMuZaZLzyjT5qe1v4pO1v4pGvKZiMxpOkuEzrOsgw07H/nPyaF
WzV3cUfHitV6ZMgAKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAKdpx71d6Oc
LgSzXnDRnwTEzanLvhnR5+ubAAZAAF+zY962/PKOSGLDOSdeVerbERWNIjSHbjn9oArqAAAAAAAA
AAAAAAAAAAKdpx79d6OcLgLNecNGfBxm1I+cM6PN1zYADIAAu2bHvW354xR245nbnSjSHbjn9roCuoAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
nWeFerZERWNIjSB245/aAK6gAAAAAAAAAAAAAACOWnaUmvf3JAPPmJidJ5w4158O/96vtf9ZZiYnSY
0lHn65xwAYAAE8WPtLxHd3uUpbJOlW3HjjHXSPrI68c77c77qURpGgCuwAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAqyYKX48p6wtAZLbNeOWkwh2OTw4Rjs+Se6I+a6mzVjja
d5cCzmQiNI4ADQAAAAAAAAAAAAAAAAAAAAAryYKX48p6wsAZLbNeOWkwh2OTwS3Ax4RjjZ8k9
0R81tNmrHG07y8FnMhEaRpAA0AAAAAAAAAAAAAAAAAAAAAAAATGvNTfZqzxrO7K4Es1jts+SO6J
+SPY5PBLcDPhyyV2a889IhdjwUpx5z1laCzmT6ABoAAVZMFL8eU9YWgMltmvHLSYQ7HJ4cIx2fJPdQ
dEfNdTZq142nelcCzmQjgANAAAAAAAAAAAAAAAAAAAACGTFXJ7UceqYDLbZrxy0mEOxyeHCMdnyT3R
MUYMk/46fNbTZfHP0hoBZxhWjSIgAOP6AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAOWtWvtWiPnLqu0ROeusf4z+wJdrj8dfM7XH46+bu7X
pHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7
XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu
7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+b
u7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+
bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46
+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH4
6+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH
46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7X
H46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7
XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM
7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8df
M7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8
dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj
8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOdrj8dfM7XH46+bu7XpHkbtekeQOd
rj8dfM7XH46+bu7XpHkbtekeQFbVt7MxPyl1CkaZcmnwTAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQn39f0z+yaE+/r+mf2BMAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAEKe9yfRNCnvcn0TAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAQn39f0z+yaE+/r+mf2BMAAGLJt1qXmu5HCeoNow//oW8Eebtdv8AF
TyBtEMWWmWutZTAAAAAAABmz7ZXHO7WN6QaR5s7blmeExH0Tpt14n70RIN4hhzVzV1rP0TAAAAA
B5215L1z2it7RHwkHoirZZm2CszMzPxWgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAhT3uT6JoU97k+iYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACE+/r+mf2TQn39f0z+wJgAPIze+v83rvIze+v8wb8WzYbY
qzNImZhDaNkp2c2pGkxxTxbRirirE3iJiEdo2unZzWk70zwBk2a80zV6TOkvRz5Jx4pvGmsdXnbNSb5q6co
nWW/a/7awKMe3TNv/SIiPgU2u+TPSsaRWZZcOPtMta9XqUw46abtY1jvAy5a4qb1mG+2Zbz937vyNuv
Ns273RC3ZJxY8e9a0b0gprtmWk/e4/Nuw5q5qb1frCjapw5MU6Wjejko2G+7n3e6wNW25Zx4tI52YsG
Gc2TTu75Xf1CZ7Wsd2iz+nxHZ2nv1BdTZsVY0isT81WfY6WrM440t/wBagHk4slsWSJjh1etWd6sTHKXlb
TERnvp1ejs064Ka9AZJ27JFtN2umrubbp1mMccOrJb25+b0sGzUrjjerE2mOOoLaW/8otae7WWPNtttd
McaR1XbbbcwaRw14MmyYoy5fvco4gV2zNE6zOvzQzZO1yTbTTV6k46TXdmsaPLz07PLavdAPQ2T+3qb
RtMYY052nuNl4bNWfg87Lacma0zPOQWztWe08J0+UJYttvE6XjWGnFfBjpFYtVm23s7aWpMa9+gN9b
ResWjlLNtW03w3itYiY073P6feZras9yr+oe9j5An9uns+Ub63ZM1stbTfuUbFgrk1veNYjlDdWla+zEQDz7
bZl7TWJ0jo347b+OtusKbbFjtfe4xHRoiIrERHKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQp73J9E0Ke9yfRMAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAABCff1/TP7JoT7+v6Z/YEwAHkZvfX+b13mZcGWctpjHaY16An
TYr3pFovXinXYJ/yv5NWGJjFWJjSYhMEMWGmKulY+qG2f29lyraqzbBaKxrPQGHYv7mv1em8/ZMOSm0
Vm1JiOPGXoA83bazG0T8Utn2bHmx678xPfDVtOCM1eHC0cpYJpmwW4RMfGAap2CkRrN7JYtkpS9b1v
M6Mc3z5eEzazdslMlMel/pHQFf9QxzNYvHdzU7HmjFea29mz0LRFomJjWJYM+x3rMzj+9XoD0ImJjWO
KGbLXFSZmePdDzN7Lj4a2gjHlyzymfiCPHLk+NpetjruUrXpCnZtljF963G3/GgHjz7z6vYryh5c7Pl39eztz6P
UjlAM2311wxPSWfYbxXLMT3w9C9YvWa25S8zLs2TFbhEzHWAeo8rabxfPaY5OxfPeN2JtMdC2zZon2Jn
5A3bLGuzVj4PNvG7ktE90vT2as1wVi0aSp2rZZyTv0598dQcpsWO9YtF50knYccc8kwzRbNh4RvVd0z5rR
7VgbsGz1wTMxaZ16sv9Q97HybsUWrjrF51sybbivfLE1rMxp3As2D3H1aVGx0tTFpaJide9fPIEJzY4tuzeN
U3k3xXjJNZid7V6mKs1xViecQCQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAIU97k+iaFPe5PomAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAhPv6/pn9k0J9/X9M/sCYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIU97k+iaFPe5PomAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAhPv
6/pn9k0J9/X9M/sCYAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAIU97k+iaFPe5PomAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAhPv6/pn9k0J9/X9M/sCYAAIXz46W
0teIkExV9pw/iQtrMWiJidYkAAAV5M+PH7Vo16K/tuHXTj5A0COPLTJH3LRKQAAAI3vXHGt7REAkIUzY7z
pW0TKYAAA5a0VjW06QottuGO+Z+UA0Cmm1YrzpFtPmuAAAHLWisa2nSFFtsw1nnM/KAaBVTacV50i
2k/FaAAAI3vXHGt50hD7Th/EgFoq+04fxIWUvW9dazrAOgACqdpwxPvIPtOH8SAWir7Th/EhbE6xrAAAA
TMRGszpCi214a/5TPyBeKK7ZhnvmPmvrMWjWJ1gAEb5K441vaIgEhV9pw/iQnTJTJruWidASBXkz48fC1
uPSAWDPXbcMzprMfOF9bRaNazrAOgACN8lcca3tEQh9pw/iQC0VfacP4kH2nD+JALQiYmImOUgAhfNj
xzpe0RKP2nD+JALRyt63rvVnWOqv7Th/EgForjaMVpiIvEzKwAEb5aY9N+0RqCQq+04fxITpeuSNaTrAJDl
r1pGtpiIUTtuGO+Z+UA0Cqm04rzpFuPxWgAAAAAAAAhT3uT6JoU97k+iYAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAACE+/r+mf2TQn39f0z+w
JgAMm0bJbLlm0WiInq1gPHvXcvNZ7nqbP7inyebtHv7/N6Wz+4p8gWI5ItNJis6T1SAebGy5b5Ji3DrMrb
bBpXhfi2oZstcVJtP0B5dL2xZImOExL1qzvViesavJpE5csR3zL1qxpWI6QDoAEzFYmZ4RDy9ozTnycOXdC7
bdo1ns6zw7zYsGv/raOEcgX7Lg7Kms+1PNez/bMe9u8dddGgAAHm7Zkm+aa68K8FmDY4vji17Tx5RCG
24ppk345WdwbZOOkVtXeiOQIbTs/YTGk6xLRsGSbVmkzrpyZto2ic8xw0iOUNew4tzHNp52BpAB5u2ZZ
vlmuvCvBZg2KL0i15mNe6FG01mue2vfOrZs+045xxFrRWYjTiDLtOzTg0mJ1rLTsOab1mlp1mOSrbdorkr
FKTr1lXsVtNorHXgD0wAZP6jP/AJ1j4s2DZ7Z4mYmI06r/AOozxpCrZtpjBWYmuus9QT+wX8dWvZ8U4sU
VmYmfgqpt2OZ4xNWmJi0axOsSATxiQBhtsF5tM79eKvLss4qb1r1+T0pnSJl5W0Zpy5JmeUcoBHFjtlvFa
vWrG7WI6Qy7FfFWu7E/fnnq1gHIV7RbdwXn4AwbTnnLeYifuxyhZi2Kb1i153de5nxTWMlZtyieLVbb+P
3KcPiBbYPDfzbKViIlrHKGXHt0TOl66fGGuJiY1jkAp2nDOakRExGk964B5WfBOCYiZidejT/TvZv80f6j7VPql
/TvZv8AMF21Zeyxax7U8Iefix2z5NI598yv/qFv/Stfhqs/p9fuWnrIKsuxWpSbVnXTuQ2XNOLJETP3Z5vTe
Rljdy2jpIPXEcVt7FWesJAy/wBQ93X5suDZ7Z4nSYjTq9HLirliItrwcxYa4dd3XiDFfYrUpNpvXSGesTa0RHe
27fk0rGOO/jKvYcW9k355VBvpXdpWvSNHQBm2nZrZrxMWiNOrDlxziyTSZ1mHrvM2z+5v9P8AgNmyc
dmhlvsV6Um02rwa9j/t6p5/c3+QPM2f3+P9UPWeTs/v8f6oesAo2rZ7Z93dmI06rwHk5sM4b7szE/Jt2D
3M/NRt/vo+S/YPcz8wU7VgzWya+1E8vglXYJ0+9fj8G0mYiNZ5A8nNinDfdmflLfseScmHjzjgxbXljLl1jlHC
G3Y8c0wRrzniC8AAAAAAEKe9yfRNCnvcn0TAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQn39f0z+yaE+/r+mf2BMAAAHk7R7+/wA3pbP7in
yebtHv7/N6Wz+4p8gWA5ad2sz0BHLlripvW+kPMyXvtGTrM8oczZbZb71vpCWHP2MzMUiZ6yDds2zxh
rrPG0817B9vv4KtmDJOXFW8xpMgmp2rJOLDMxznguRyY65a6XjWAeRE/e1ni1Rt0xXSMcRHzaPseDwf
7knZMMRP3P8Acg83e+/vfHVv2fapy5IpNIjhz1YYiO0iO7V6eLZ8eOYtWuk6dQWgA8nPknJlmZnv0hsw7
JjjHE3jWZhk2nFOLLPDhPGJaMO2xWkVvWZmO+Ac2vZq46b9OHWHNgyTGTc7phHadq7au7WNKp7Bi
ne7SY4dwNwAK82CmaPvc45SyzsFteF40XbTtPY2rERrPer+3109idQV5tmrgw6zOtp5I7FXe2iJ6cUM2a+
e/H6RDbseCcVNbe1INAAK8uz0zTE314dJV/YsPSfNTn2q9M87vCI4aT3uxt/DjTj8AVbVs8YZiazrErv6feZ3
qd0cWfPntnmNY0iOUNWw4ppSbWjSbcgagcvbcpNukA7MaxMM1thxzymYlnptmSszr96JnlKydv4cKcQ
ZclJxZJrrxjvens9+0w1tPN5kzbNk152l6mGnZ4q16Amr2iN7BePgsJjWJie8HjVjetEdXp02XFWuk0ifjLBnx
Ww5fhziWmm313fvVnX4Ap2vDGLJG7ylp2C02wzE90sefNOfJrp8Ihv2THOPDETzniC4AGH+o+1T6pf07
2b/ADR/qPtU+qX9O9m/zBD+oV/9K2+Giz+n2+5aOkrdqxdri0j2o4w8/Fktgyaxz74kHrPIyzvZbT1lfl2216T
Wsaa96Gy4Zy5ImY+7HMHo4o3cVY6QkABM6RMz3DPtuXcxbsc7AwZrzky2t8eD0tmx9nhiO+eMsGyY+
0zRryjjL1AAAHmbZ/c3+n/HpvM2z+5v9P8AgNux/wBvVPP7m/yQ2P8At6p5/c3+QPM2f3+P9UPWeTs/v
8f6oesAADz9v99HyX7B7mfmo2/30fJfsHuZ+YNMzERrPJ521bTOSdyk/d/6ltue03nHHCI5/FlpbdtEzETp3
SDVsmy72l7xw7ob2D7ffwVX7LtNs1piaxGnQGgAAAAAAEKe9yfRNCnvcn0TAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQn39f0z+yaE+/r+
mf2BMAAAFc7PitMzNImZWViKxERGkQABMRMaTyAFX2bD+HB9mw/hwtAVfZsP4cLK1ilYrWNIjudAAA
AAV/Z8Ouu5GqwAAActSt40tETCi2xYZ7pj5S0AKa7Jir/jr810RERpAAAAhkw48ntV1nqq+xYde/wA2gBXj
wY8fs1jXqsAAAEL4qZParEqp2LDPWPlLQAqps2Kk6xXWfitAAAFV9nxX4zXj8EPsWLXX73m0AIY8NMfs1i

EwAABy1K3jS0RMKJ2LDM8pj5S0AK8ez48c61rx6ysAAAEb4qZNN+sToUx0x67lYjVIAV5MGPJxtXj1hYAz12LDE66TPzlfWsVjSsaQ6AAAIXxUyTresSmAjTFTHruViNUgAAAV2wYr23rUiZnvWAOVrWldKxpDsxFomJjWJAFcbPirMTFIiYWAAACF8OPJOt6xMu0pXHGlI0hIBXbBivbetSJlz7Nh/DhaAq+zYfw4SpipjnWlYhMAAAAAAAABCnvcn0TQp73J9EwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAEJ9/X9M/sAJgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAhT3uT6JgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD//Z"></figure><p>Constraints:</p><p>0<size <100</p><p>0<data<1000</p><p>Input format:</p><p>First line indicates the size of the tree</p><p>Second line indicates the elements of the tree</p><p>Output Format:</p><p>single line indicates the pre-order list of the given value</p>

answer

```
 #include<bits/stdc++.h>
using namespace std;
void solve(){}
struct node {
   int data;
   struct node *left,*right;
}*root=NULL;
void insert(int data) {
   struct node *tempNode = (node*) malloc(sizeof(node));
   struct node *current;
   struct node *parent;
   tempNode->data = data;
   tempNode->left = NULL;
   tempNode->right = NULL;
   if(root == NULL)  root = tempNode;
   else {
      current = root;
      parent = NULL;
      while(1) {
         parent = current;
         if(data < parent->data) {
```

```
            current = current->left;
            if(current == NULL) {
                parent->left = tempNode;
                return;
            }
        }
        else {
            current = current->right;
            if(current == NULL) {
                parent->right = tempNode;
                return;
            }
        }
    } } }
void preorder(struct node* root) {
    if(root != NULL) {
        printf("%d ",root->data);
        preorder(root->left);
        preorder(root->right);
    }
}
int main() {
    solve();
    int n,i,x; scanf("%d",&n);
    for(i = 0; i < n; i++){
        scanf("%d",&x); insert(x); }
    preorder(root);
    return 0;
    printf("struct node* newNode(int item) "); }
```

question

Problem Description A new species is trying to rule the planet. This species is creating their own population outburst to dominate other species. It all started with 1 single member of the species. The population increases in treelike fashion abiding by few rules as listed below. Single member is able to reproduce by itself.
A new member is added to the population every minute.
Every member is associated with integral name.
Multiple members can share a common name.
Every member has it's own reproduction capacity, that is maximum number of children it can reproduce.
A member can start to reproduce only if all members older than it have exhausted their reproduction capacity.
Level 0 in family tree of this species comprise of single member at the start of multiplication.
Integral name of single member at the start is 0.
The population grows level wise, where number of members at level i is dependent on reproduction capacity of members at prior level. Given the integral name of new member and it's reproduction capacity that is added to the population, you have to find it's parent, level at which it is added and it's ascending age wise rank among siblings. Input:
First line of the input contains 2 integers, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi><mo>,</mo><mi>R</mi><msub><mi>C</mi><mn>0</mn></msub></math>, representing number of minutes we will be examining the population increase and reproduction capacity of member at epoch. Next N line contains 2 integers each, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>I</mi><msub><mi>D</mi><mi>i</mi></msub><mo>,</mo><mi>R</mi><msub><mi>C</mi><mi>i</mi></msub></math>, representing integral name and reproduction capacity of new member born at time i. Output:
N lines, each line containing 3 integers, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>P</mi><mo>,</mo><mi>L</mi><mo>,</mo><mi>C</mi></math>, representing integral name of the parent, level at which it is added and it's ascending age wise rank among siblings. Note :
It will always be possible to reproduce a new child or in other words, through out the given time, there exists atleast one member which can still accomodate new child. Constraints:
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>≤</mo><msup><mn>10</mn><mn>6</mn></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mo>−</mo><msup><mn>10</mn><mn>9</mn></msup><mo>≤</mo><mi>I</mi><msub><mi>D</mi><mi>i</mi></msub><mo>≤</mo><msup><mn>10</mn><mn>9</mn></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>0</mn><mo>≤</mo><mi>R</mi><msub><mi>C</mi><mi>i</mi></msub><mo>≤</mo><msup><mn>10</mn><mn>9</mn></msup></math> Explanation for test case1 The resultant family tree looks like this. <figure class="image"><img src="https://he-s3.s3.amazonaws.com/media/uploads/54a935a.png" alt="enter image description here"></figure>

answer

```
#include<stdio.h>

#include<stdlib.h>
```

```
#include<string.h>
struct cell{
   int name;
   int level;
   int capacity;
};
struct cell queue[1000001];
struct cell arr[1000001];
int front;
int end;
void init_queue(){
   front = 0;
   end = 0;
}
void enqueue(int name,int capacity,int level){
   queue[end].name = name;
   queue[end].level = level;
   queue[end].capacity = capacity;
   end = end + 1;
}
int is_empty(){
   if(end == front)
      return 1;
      return 0;
}
void dequeue()
{
            if(!is_empty())
             front++;
}
```

```
int main(){
    int n,rc;
    init_queue();
    scanf("%d %d",&n,&rc);
    int i,j,k;
    for(i=0;i<n;i++){
        scanf("%d %d",&arr[i].name,&arr[i].capacity);
    }
    enqueue(0,rc,0);
    i=0;
    while(!is_empty()){
        int par = queue[front].name;
        int cap = queue[front].capacity;
        int lev = queue[front].level+1;
        k=1;
        for(j=0;j<cap&&i<n;j++,i++){
            printf("%d %d %d\n",par,lev,k++);
            enqueue(arr[i].name,arr[i].capacity,lev);
        }
        dequeue();
    }
    return 0;
}
```

question

Problem Description You are given a directory tree of <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> directories/folders. Each directory is represented by a particular <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi><mi>d</mi></math> which ranges from <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math> to <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math>. The id of the root

directory is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math>, then it has some child directories, those directories may contain some other ones and it goes on. Now you are given a list of directory id's to delete, you need to find the minimum number of directories that need to be deleted so that all the directories in the given list get deleted. Note that if you delete a particular directory, all its child directories will also get deleted. Constraints
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>≤</mo><msup><mn>10</mn><mn>5</mn></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>i</mi><mi>d</mi><mtext> </mtext><mi>o</mi><mi>f</mi><mtext> </mtext><mi>p</mi><mi>a</mi><mi>r</mi><mi>e</mi><mi>n</mi><mi>t</mi><mo>≤</mo><mi>N</mi></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>M</mi><mo>≤</mo><mi>N</mi></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>i</mi><mi>d</mi><mtext> </mtext><mi>t</mi><mi>o</mi><mtext> </mtext><mi>b</mi><mi>e</mi><mtext> </mtext><mi>d</mi><mi>e</mi><mi>l</mi><mi>e</mi><mi>t</mi><mi>e</mi><mi>d</mi><mo>≤</mo><mi>N</mi></math>

Input
The first line of input contains an integer <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> that denotes how many folders are there.
The next line contains <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> space separated integers that where the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> integer denotes the id of the parent of the directory with id <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math> . Note that the first integer will be <math xmlns="http://www.w3.org/1998/Math/MathML"><mo>−</mo><mn>1</mn></math> as <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math> is the id of root folder and it has no parent. Rest of the integers will not be <math xmlns="http://www.w3.org/1998/Math/MathML"><mo>−</mo><mn>1</mn></math> .
The next line contains an integer <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>M</mi></math> that denotes how many directories you need to delete.
The next line contains <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>M</mi></math> space separated integers that denote the ids of the directories you need to delete.

Output
Print the minimum number of directories that need to be deleted. Explanation for test case 1 In the image below is the complete directory tree. If you delete the directories 2 and 3, 4 gets automatically deleted, thus you need to delete only 2 directories. <figure class="image"><img src="https://he-s3.s3.amazonaws.com/media/uploads/895b3e7b-5947-46f7-a016-ddfc6f0a447f.png" alt="This image is generated in csacademy graph editor"></figure>

answer

```
#include <stdio.h>
```

```
#include <stdlib.h>
#define pcx putchar_unlocked
#define gcx getchar_unlocked
typedef long int lint;

lint getnl() {
	lint n =0; auto neg =0;
	register int c = gcx();
	if ('-' == c) { neg =1; c = gcx(); }
	while(c<'0' || c>'9') c = gcx();
	while(c>='0' && c<='9') {
		n = n * 10 + c-'0';
		c = gcx();
	}
	if(neg) n *= -1;
	return n;
}
void putl(lint li,char lc) {
	if (0 == li) {
		pcx('0'); if(lc) pcx(lc); return;
	}
	char s[24]; lint idx =-1;
	while (li) {
		s[++idx] = '0' + li % 10;
		li /= 10;
	}
	for (lint jdx=idx; jdx>=0; --jdx) pcx(s[jdx]);
	if(lc) pcx(lc);
}

int main () {
```

```
    lint N = getnl();
    int PA[N+1];
    for (lint ni=1; ni<=N; ) PA[ni++] = getnl();

    lint D = getnl();
    int DA[D];
    char DLT [100001] ={0};
    for(lint ni=0; ni<D;) {
        DA[ni] = getnl(); DLT[DA[ni++]] =1;
    }
    if (DLT[1]) {
        putl(1, 0); return 0;
    }
    lint dLess =0;
    for (lint ni=0; ni<D; ) {
        //printf ("D:%d ", DA[ni]);
        lint pi = PA[DA[ni++]];
        char piv [100008]={0} ;
        while (pi>0) {
            if (piv[pi]) { ++dLess; /*printf ("C:%d", pi);*/ break; }
            else piv[pi] = 1;
            //putl(pi, ' ');
            if (DLT[pi]) { ++dLess; break; }
            pi = PA[pi];
        }
        //pcx('\n');
    }
    putl(D-dLess, 0);
            return 0;
}
```

question

Problem description When the king of ghosts notices that all humans on Planet Earth have lost their dread of the ghost race, he is extremely unhappy. He understands why this is happening. The existing ghost species has gotten incredibly lethargic and has ceased visiting Planet Earth in order to terrorise humanity. As a result, he plans to arrange a tournament to urge the whole ghost race to scare the humans. The monarch, on the other hand, never comes to Planet Earth. This competition will go on for N days. Currently, there are a total of M ghosts (apart from the king) existing in the ghost race such that :
- The youngest ghost is 1 year old.
- The oldest ghost is M years old.
- No two ghosts have the same age.
- The age of each and every ghost is a positive integer. Every day of the tournament, ghosts must visit Planet Earth in order to frighten humans. The ghost that terrifies the most humans on any given day is named "Ghost of the Day" at the conclusion of the day. The king of ghosts, on the other hand, is a firm believer in constancy. After this title is granted, a "Consistency Trophy" is offered to the ghost who has earned the most of these titles up until that point. If there are several such ghosts, the prize is handed to the eldest among them. It's worth noting that "Title Giving" and "Trophy Giving" take place at the end of each tournament day. Each day of the competition, you will be told the age of the ghost that earned the "Ghost of the Day" title. Your task is to determine the age of the ghost who received the "Consistency Trophy" on each competition day. Input
The first line consists of 2 space separated integers N and M. The next line consists of N space separated integers such that the ith integer denotes the age of the ghost who was awarded with the "Ghost of the Day" title on the ith day of the competition. Output
Print N lines. The ith line should contain 2 space separated integers such that the first integer denotes the age of the ghost who was awarded with the "Consistency Trophy" on the ith day and the second integer denotes the number of "Ghost of the Day" titles won by this ghost until the end of the ith day of the competition. Constraints
1 ≤ N ≤ 10⁵
1 ≤ M ≤ 10⁹

answer

```
#include<bits/stdc++.h>

using namespace std;

int main()

{

        int i,n,m;

        cin>>n>>m;

        unordered_map<int,int> ghost;
```

```
        int best = -1;
        cin>>best;
        ghost[best]+=1;
        cout<<best<<" "<<1<<endl;
        for(i=0;i<n-1;i++)
        {
                int y;
                cin>>y;
                ghost[y]+=1;
                if((ghost[y]>ghost[best]) || (ghost[y]==ghost[best] && y>best)) best = y;
                cout<<best<<" "<<ghost[best]<<"\n";
        }
}
```

question

Problem Description: Any sequence A of size n is called B-sequence if: <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mn>1</mn></msub><mo><</mo><msub><mi>A</mi><mn>2</mn></msub><mo><</mo><mo>.</mo><mo>.</mo><mo>.</mo><mo><</mo><msub><mi>A</mi><mi>k</mi></msub><mo>></mo><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>k</mi><mo>+</mo><mn>1</mn></mrow></msub><mo>></mo><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>k</mi><mo>+</mo><mn>2</mn></mrow></msub><mo>></mo><mo>.</mo><mo>.</mo><mo>.</mo><mo>></mo><msub><mi>A</mi><mi>n</mi></msub></math> where <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>k</mi><mo>≤</mo><mi>n</mi></math>. That is, a sequence which is initially strictly increasing and then strictly decreasing (the decreasing part may or may not be there). All elements in A except the maximum element comes atmost twice (once in increasing part and once in decreasing part) and maximum element comes exactly once. All elements coming in decreasing part of sequence should have come once in the increasing part of sequence. You are given a B-sequence S and Q operations. For each operation, you are given a value val. You have to insert val in S if and only if after insertion, S still remains a B-sequence .
After each operation, print the size of S. After all the operations, print the sequence S. Hint: Think of using some data structure to support insertion of elements in complexity better than linear. Input

Constraints: <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>≤</mo><msup><mn>10</mn><mn>5</mn></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><msub><mi>S</mi><mi>i</mi></msub><mo>≤</mo><msup><mn>10</mn><mn>9</mn></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>Q</mi><mo>≤</mo><msup><mn>10</mn><mn>5</mn></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>v</mi><mi>a</mi><mi>l</mi><mo>≤</mo><msup><mn>10</mn><mn>9</mn></msup></math>
Given sequence S is a B-sequence, Input Format: First line consists of an integer N, denoting size of S.
Second line consists of N space separated integers, denoting elements of S.
Next line consists of an integer Q, denoting number of operations.
Each of the following Q lines consists of an integer val. Output Format: After each operation, print the size of S in a new line.
After all operations, print the sequence S.

answer

```
 #include<bits/stdc++.h>

#include<map>

using namespace std;




int main() {


  int N,i,maximum=INT_MIN;


  scanf("%d", &N);


  int S[N];


  map<int,int> map;


  for(i=0;i<N;i++) {
```

```
    scanf("%d", &S[i]);

    maximum=max(maximum,S[i]);

    map[S[i]]++;

}

int temp,Q;

cin>>Q;

for(i=0;i<Q;i++) {

    scanf("%d", &temp);

    if(temp==maximum) printf("%d\n",N);

    else {

        if(map[temp]>=2) printf("%d\n",N);

        else {

            map[temp]++;

            N++;

            printf("%d\n",N);

            maximum=max(maximum,temp);
```

```
        }

      }



  }

  for(auto it=map.begin();it!=map.end();it++) printf("%d ",it->first);

  for(auto it=map.rbegin();it!=map.rend();it++) {

      if(it->second>1) printf("%d ",it->first);

  }

}
```

question

Question description Given two rooted trees, your task is to find out if they are isomorphic, i.e., it is possible to draw them so that they look the same.
Constraints 1≤t≤1000
2≤n≤10^5
the sum of all values of n is at most 10^5
Input
The first input line has an integer t: the number of tests. Then, there are t tests described as follows:
The first line has an integer n: the number of nodes in both trees. The nodes are numbered 1,2,…,n, and node 1 is the root.
Then, there are n−1 lines describing the edges of the first tree, and finally n−1 lines describing the edges of the second tree.

Output

For each test, print "YES", if the trees are isomorphic, and "NO" otherwise.

answer

```
 #include <bits/stdc++.h>

using namespace std;
```

```
#define rep(i, a, b) for(int i = a; i < (b); ++i)
#define trav(a, x) for(auto& a : x)
#define all(x) begin(x), end(x)
#define sz(x) (int)(x).size()
typedef long long ll;
typedef pair<int, int> pii;
typedef vector<int> vi;

vi h, id;
vector<vi> g;
map<int, vi> lvl;

void dfs(int i, int p) {
   trav(j, g[i])  if (j!=p) {
      dfs(j, i);
      h[i]=max(h[i], h[j]+1);
   }
   lvl[h[i]].push_back(i);
}

int main() {
   cin.sync_with_stdio(0); cin.tie(0);
   cin.exceptions(cin.failbit);

   int t;
   cin >> t;
   while(t--) {
      int n;
      cin >> n;
      int m=2*n+1;
```

```
g.assign(m, vi());
rep(i, 0, 2) {
    rep(j, 0, n-1) {
        int a, b;
        cin >> a >> b;
        a+=i*n, b+=i*n;
        g[a].push_back(b);
        g[b].push_back(a);
    }
}
g[0]={1, n+1};
h.assign(m, 0);
id.assign(m, 0);
lvl.clear();
dfs(0, -1);
if (h[1]!=h[n+1]) {
    cout << "NO\n";
    continue;
}
trav(l, lvl) {
    map<vector<ll>, int> u;
    trav(i, l.second) {
        vector<ll> cur;
        trav(j, g[i]) cur.push_back(3LL*n*h[j]+id[j]);
        sort(all(cur));
        if (!u.count(cur)) {
            int s=sz(u);
            u[cur]=s;
        }
        id[i]=u[cur];
    }
```

```
        }
        cout << (id[1]==id[n+1]? "YES\n":"NO\n");
    }
    return 0;
}
```

question

Problem Description: You're given a K-ary infinite tree rooted at a vertex numbered 1. All its edges are weighted 1 initially. Any node <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>X</mi></math> will have exactly <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>K</mi></math> children numbered as: <math xmlns="http://www.w3.org/1998/Math/MathML"><mo stretchy="false">[</mo><mi>K</mi><mo>∗</mo><mi>X</mi><mo>+</mo><mn>0</mn><mo>,</mo><mi>K</mi><mo>∗</mo><mi>X</mi><mo>+</mo><mn>1</mn><mo>,</mo><mi>K</mi><mo>∗</mo><mi>X</mi><mo>+</mo><mn>2</mn><mo>,</mo><mi>K</mi><mo>∗</mo><mi>X</mi><mo>+</mo><mn>3</mn><mo>,</mo><mi>K</mi><mo>∗</mo><mi>X</mi><mo>+</mo><mn>4</mn><mo>,</mo><mo>.</mo><mo>.</mo><mo>.</mo><mo>.</mo><mo>.</mo><mo>.</mo><mo>.</mo><mi>K</mi><mo>∗</mo><mi>X</mi><mo>+</mo><mo stretchy="false">(</mo><mi>K</mi><mo>−</mo><mn>1</mn><mo stretchy="false">)</mo><mo stretchy="false">]</mo></math> You are given <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> queries to answer which will be of the following two types: <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mi>u</mi><mi>v</mi></math>: Print the shortest distance between nodes <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>u</mi></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>v</mi></math>.
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn><mi>u</mi><mi>v</mi><mi>w</mi></math>: Increase the weight of all edges on the shortest path between <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>u</mi></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>v</mi></math> by <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>w</mi></math>. Constraints <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn><mo> </mo><mo>≤</mo><mo> </mo><mi>K</mi><mo> </mo><mo>≤</mo><mo> </mo><mn>10</mn><mspace linebreak="newline"> </mspace><mn>1</mn><mo> </mo><mo>≤</mo><mo> </mo><mi>Q</mi><mo> </mo><mo>≤</mo><mo> </mo><msup><mn>10</mn><mn>3</mn></msup><mspace linebreak="newline"> </mspace><mn>1</mn><mo> </mo><mo>≤</mo><mo> </mo><mi>U</mi><mo>,</mo><mi>V</mi><mo> </mo><mo>≤</mo><mo> </mo><msup><mn>10</mn><mrow class="MJX-TeXAtom-ORD"><mn>18</mn></mrow></msup><mspace linebreak="newline"> </mspace><mi>U</mi><mo> </mo><mo>≠</mo><mo> </mo><mi>V</

mi><mspace
linebreak="newline"> </mspace><mn>1</mn><mo> </mo><mo>≤</mo><mo> </mo><mi>W
</mi><mo> </mo><mo>≤</mo><mo> </mo><msup><mn>10</mn><mn>9</mn></msup></math><
/p>   Input format The first line contains two space-
separated integers <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>K</mi></math> and <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math>.
Next <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>Q</mi></math> lines contain queries
which will be of 2 types:
Three space-separated integers <math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math>, <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>u</mi></math>, and <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>v</mi></math>
Four space-
separated integers <math
xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn></math>, <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>u</mi></math>, <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>v</mi></math>, and <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>w</mi></math> <str
ong>Output format For each query of type <math
xmlns="http://www.w3.org/1998/Math/MathML"><mo
stretchy="false">(</mo><mn>1</mn><mi>u</mi><mi>v</mi><mo
stretchy="false">)</mo></math>, print a single integer denoting the shortest distance
between <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>u</mi></math> and <math
xmlns="http://www.w3.org/1998/Math/MathML"><mi>v</mi></math>.

answer

```
 #include <iostream>

#include <map>

#include <assert.h>

using namespace std;

#define int long long


map < pair < int, int >, int > adj;


int find_depth( int u, int k ) {

   int depth = 0;

   while ( u > 0 ) {

      u = u / k;

      depth = depth + 1;
```

```
    }
    return depth - 1;
}

int dist( int u, int v, int k ) {
    int dist = 0;
    int depth_u = find_depth( u, k );
    int depth_v = find_depth( v, k );
    if ( depth_u < depth_v ) {
    swap ( u, v );
    swap ( depth_u, depth_v );
    }
    while( depth_u != depth_v ) {
        if ( adj.count( { u, u / k } ) ) {
        dist = dist + adj[ { u, u / k } ];
    } else {
        dist = dist + 1;
    }
    depth_u = depth_u - 1;
    u = u / k;
    }
    while ( u != v ) {
    if ( adj.count( { u, u / k } ) ) {
        dist = dist + adj [ { u, u / k } ];
    } else {
        dist = dist + 1;
    }
    if ( adj.count( { v, v / k } ) ) {
            dist = dist + adj [ { v, v / k } ];
        } else {
            dist = dist + 1;
```

```
        }
        u = u / k;
        v = v / k;
    }
    return dist;
}

void add_weight( int vertex, int parent, int w ) {
    if ( !adj.count ( { vertex, parent } ) ) {
        adj[ { vertex, parent } ] = 1;
    }
    adj[ { vertex, parent } ] = adj[ { vertex, parent } ] + w;
}

void increase_weights ( int u, int v, int w, int k ) {
    int depth_u = find_depth( u, k );
    int depth_v = find_depth( v, k );
    if ( depth_u < depth_v ) {
    swap ( u, v );
    swap ( depth_u, depth_v );
    }
    while( depth_u != depth_v ) {
        add_weight( u, u / k, w );
    depth_u = depth_u - 1;
    u = u / k;
    }
    while ( u != v ) {
    add_weight( u, u / k, w );
    add_weight( v, v / k, w );
        u = u / k;
        v = v / k;
```

```
    }
}

signed main() {

    int k, q, x, u, v, w;
    cin >> k >> q;

    while(q--) {
        cin >> x;
        if ( x == 1 ) {
            cin >> u >> v;
            cout << dist( u, v, k ) << "\n";
        } else {
            cin >> u >> v >> w;
            increase_weights( u, v, w, k );
        }
    }
}
```

question

Problem Description Football is Monk's favourite sport, and his favourite team is "Manchester United." Manchester United has qualified for the Champions League Final, which will take place at London's Wembley Stadium. As a result, he decided to go watch his favourite team play. When he arrived at the stadium, he noticed that there was a long wait for match tickets. He is aware that the stadium has M rows, each with a distinct seating capacity. They could or might not be comparable. The cost of a ticket is determined by the row. If there are K(always higher than 0) empty seats in a row, the ticket will cost K pounds (units of British Currency). Now, every football fan standing in the line will get a ticket one by one.
Given the seating capacities of different rows, find the maximum possible pounds that the club will gain with the help of the ticket sales. Constraints:
1 <= M <= 1000000
1 <= N <= 1000000
1 <= X[i] <= 1000000
Sum of X[i] for all 1 <= i <= M will always be greater than N. Input:
The first line consists of

M and N. M denotes the number of seating rows in the stadium and N denotes the number of football fans waiting in the line to get a ticket for the match.
Next line consists of M space separated integers X[1],X[2],X[3].... X[M] where X[i] denotes the number of empty seats initially in the ith row. Output:
Print in a single line the maximum pounds the club will gain.

answer

```
 #include <bits/stdc++.h>

using namespace std;

#define PII pair <int, int>

priority_queue <int> seats;

map <int, int> x;

int main()

{

   int N, M; cin >> N >> M;

   assert (1<=N and N<=1000000);

   assert (1<=M and M<=1000000);

   for (int g=1; g<=N; g++){

      int a; cin >> a;

      seats.push(a);

      assert (1<=a and a<=1000000);

      x[a]++;

   }

   long long ans = 0;

   for (int g=0; g<M; g++){

      int x = seats.top(); ans+=x; seats.pop();seats.push(x-1);

   }

   cout <<ans;

   return 0;

   cout<<"void heapify(int arr[],int n,int i)";

}
```

question

Question Description In a movie festival, n movies will be shown. You know the starting and ending time of each movie.

Your task is to process q queries of the form: if you arrive and leave the festival at specific times, what is the maximum number of movies you can watch?

You can watch two movies if the first movie ends before or exactly when the second movie starts. You can start the first movie exactly when you arrive and leave exactly when the last movie ends.

Input

The first input line has two integers n and q: the number of movies and queries.

After this, there are n lines describing the movies. Each line has two integers a and b: the starting and ending time of a movie.

Finally, there are q lines describing the queries. Each line has two integers a and b: your arrival and leaving time.

Output

Print the maximum number of movies for each query.

Constraints 1≤n,q≤2·10^5
1≤a<b≤10^6

answer

```
#include<bits/stdc++.h>

using namespace std;

int dp[1000006][25];

void solve(){}

int main(){

   solve();

   int n, q; cin>>n>>q;

   for (int i = 0; i < n; i++) {

      int x, y; cin>>x>>y;

      dp[y][0] = max(dp[y][0], x);

   }

   for (int i = 1; i <= 1000000; i++)

      dp[i][0] = max(dp[i][0], dp[i-1][0]);


   for (int k = 1; k <= 20; k++)

      for (int i = 1; i <= 1000000; i++)

         dp[i][k] = dp[dp[i][k-1]][k-1];
```

```
    while(q--) {
        int x,y; cin>>x>>y;
        int ans = 0;
        while(y>0) {
            int z = 0;
            for (int i = 0; i <= 20; i++) {
                if (dp[y][i] < x) {
                    z = i;
                    break;
                }
            }
            if (z == 0)
                break;
            ans += (1<<(z-1));
            y = dp[y][z-1];
        }
        cout<<ans<<endl;
    }

}
```

question

Question Description: There are n cities and m flight connections between them. Your task is to add new flights so that it will be possible to travel from any city to any other city. What is the minimum number of new flights required?

Input

The first input line has two integers n and m: the number of cities and flights. The cities are numbered 1,2,…,n.

After this, there are m lines describing the flights. Each line has two integers a and b: there is a flight from city a to city b. All flights are one-way flights.

Output

First print an integer k: the required number of new flights. After this, print k lines describing the new flights. You can print any valid solution.

Constraints 1≤n≤10^5
1≤m≤2·10^5
1≤a,b≤n

answer

```
 #include <stdio.h>
#include <string.h>

#define N    100000
#define M    200000

struct L {
    struct L *next;
    int j;
} aa[N], bb[N], aa_[N];

void link(int i, int j) {
    static struct L l91[M * 2], *l = l91;

    l->j = j;
    l->next = aa[i].next, aa[i].next = l++;
    l->j = i;
    l->next = bb[j].next, bb[j].next = l++;
}

void link_(int i, int j) {
    static struct L l91[M], *l = l91;

    l->j = j;
    l->next = aa_[i].next, aa_[i].next = l++;
}

int po[N], npo;
char visited[N];
```

```
void dfs1(int i) {
    struct L *l;

    if (visited[i])
        return;
    visited[i] = 1;
    for (l = aa[i].next; l; l = l->next)
        dfs1(l->j);
    po[npo++] = i;
}

int cc[N], dd[N];

void dfs2(int j, int c) {
    struct L *l;
    int c_ = cc[j];

    if (c_ != -1) {
        if (c_ != c) {
            link_(c_, c);
            dd[c]++;
        }
        return;
    }
    cc[j] = c;
    for (l = bb[j].next; l; l = l->next)
        dfs2(l->j, c);
}

int dfs3(int i) {
```

```
    struct L *l;

    if (visited[i])
        return -1;
    visited[i] = 1;
    if (!aa_[i].next)
        return i;
    for (l = aa_[i].next; l; l = l->next) {
        int w = dfs3(l->j);

        if (w != -1)
            return w;
    }
    return -1;
}

void add(int i, int j) {
    printf("%d %d\n", i + 1, j + 1);
}

void augment(int n) {
    static int vv[N], ww[N];
    static char iv[N], iw[N];
    int h, i, p, q, s, t, x;

    p = 0;
    for (i = 0; i < n; i++) {
        if (cc[i] != i)
            continue;
        if (dd[i] == 0) {
            int w = dfs3(i);
```

```
            if (w != -1) {
                iv[vv[p] = i] = 1;
                iw[ww[p] = w] = 1;
                p++;
            }
        }
    }
    s = t = p;
    for (i = 0; i < n; i++) {
        if (cc[i] != i)
            continue;
        if (!iv[i] && dd[i] == 0)
            vv[s++] = i;
        if (!iw[i] && !aa_[i].next)
            ww[t++] = i;
    }
    printf("%d\n", s > t ? s : t);
    for (h = 0; h < p - 1; h++)
        add(ww[h], vv[h + 1]);
    q = s < t ? s : t;
    for (h = p; h < q; h++)
        add(ww[h], vv[h]);
    x = ww[p - 1];
    for (h = q; h < s; h++)
        add(x, vv[h]), x = vv[h];
    for (h = q; h < t; h++)
        add(x, ww[h]), x = ww[h];
    add(x, vv[0]);
}
```

```
int main() {
    int n, m, h, i, j, k;

    scanf("%d%d", &n, &m);
    while(m--) {
        scanf("%d%d", &i, &j), i--, j--;
        link(i, j);
    }
    for (i = 0; i < n; i++)
        dfs1(i);
    memset(cc, -1, n * sizeof *cc);
    k = 0;
    for (h = n - 1; h >= 0; h--) {
        j = po[h];
        if (cc[j] == -1) {
            dfs2(j, j);
            k++;
        }
    }
    if (k == 1) {
        printf("0\n");
        return 0;
    }
    memset(visited, 0, n * sizeof *visited);
    augment(n);
    return 0;
}
```

question

Problem Description You are given an N-dimensional array A. You can execute an operation that removes the greatest and smallest elements from the array and replaces them with their difference. As a result, the array's size will drop by one after each operation. You are assigned Q jobs, each of which contains an integer K. After K operations, you must provide the total of all the items in the array for each task. Constraints: 2 <= N <= 10⁵ 1 <= Q <= 10⁵ 0 <= A[i] <= 10⁹ 0 <= K < N Input: First line contains two space-separated integers N and Q, denoting the number of elements in array and number of queries respectively. Next line contains N space-separated integers denoting elements of the array. Next Q lines contain a single integer K. Output: For each task, print answer in a new line.

answer

```
#include<bits/stdc++.h>

#define ll long long

#define pb push_back

#define mod 1000000007

#define vi vector<int>

#define REP(i, n) for(int i=1; i<=n; i++)

using namespace std;

int main()

{

    ll n, q, i, x, t1, t2,sum, k;
```

```
cin>>n>>q;

priority_queue<ll> maxh;

priority_queue<ll, vector<ll>, greater<ll>> minh;

sum=0;

for(i=0;i<n;i++){

    cin>>x;

    sum+=x;

    maxh.push(x);

    minh.push(x);

}

t1=-1, t2=-1;

ll a[n];

a[0]=sum;

for(i=1; i<n; i++){

    t1 = maxh.top(); t2=minh.top();

    a[i] = a[i-1]-(t1+t2)+(t1-t2);
```

```
        maxh.pop(); minh.pop();

        maxh.push(t1-t2); minh.push(t1-t2);

    }

    for(i=0; i<q; i++){

        cin>>k;

        cout<<a[k]<<'\n';

    }

    return 0;

}
```

question

Question description The professor wants to divide the class of n students into two groups Left (L) and Right(R) for some project work. But the students are very talkative and so the professor decides that no two friends should fall into the same group. Each student is recognized by his index which is in the range of 1 to n. The professor knows through unknown sources who are friends with whom. So he has asked you to help him divide the class into two teams. Note that friendships are always mutual; i.e. if A is a friend of B then B is also friend of A.
  Constraints T≤100
1≤ n ≤ 1000
0 ≤ k < n
Input Format
The first line of input consists of the number of test cases T. Then T lines follow each describing a test case. The first line of each test case consists of a number n. Then follow n lines each describing the friendship of a student. Each line starts with a number k telling the number of friends the ith person has. Then follow k integers separated by space which tells the index (1 based) of his friends. All these integers on the ith line are greater than i itself. (See sample for clarity).

Output Format
For each test case, if it is possible to divide the class into teams, then print a line of L’s

and R’s denoting which team that student is a part of. If it is impossible to divide the class then print -1 instead. If there are multiple solutions print the one that would come first in the dictionary.</p>

answer

```
 #include<iostream>
#include<cstdio>
#include<cmath>
#include<vector>
#include<map>
#include<set>
#include<algorithm>
#include<list>
#include<cstring>
#include<stack>
#include<queue>
using namespace std;
#define ll long long
#define vi vector<int>
#define vii vector<vi >
#define pp pair<int,int>
#define pb push_back
#define mp make_pair
#define ppl pair<ll,ll>
#define vl vector<ll>
#define vll vector<vl >
#define llu unsigned ll
#define all(c) c.begin(),c.end()
#define mod 1000000007
#define sc scanf
#define pf printf
ll power(ll a,ll b)
```

```
{
	if(!b)
		return 1;
	if(b==1)
		return a;
	ll temp=power(a, b/2);
	temp=(temp*temp);
	if( b&1 )
		temp=(temp*a);
	return temp;
}
class graph
{
	vi * adj;
	public:
		graph(int v)
		{
			adj=new vi[v];
		}
		void add_edge(int u,int v)
		{
			adj[u].pb( v);
			adj[v].pb( u);
		}
		bool dfs(int v, vector<bool>& visited, vi& group)
		{
			visited[v]=true;
			vi::iterator it;
			bool flag=true;
			for(it=adj[v].begin();it!=adj[v].end();it++)
			{
```

```
                if(!visited[*it])
                {
                        group[*it]=group[v]^1;
                        flag=dfs(*it,visited,group);
                }
                else
                {
                        if(group[v]==group[*it])
                        {
                                return false;
                        }
                }
        }
        return flag;
}
};
int main()
  {
        ios_base::sync_with_stdio(false);
        int i, n, t, k, in, j;
        cin >> t;
        while( t-- )
        {
                cin >> n;
                graph g(n);
                for( i=0;i<n;i++ )
                {
                        cin >> k;
                        for( j=0;j<k;j++ )
                        {
                                cin >> in;
```

```
                in--;
                g.add_edge(i, in);
        }
}
vector<bool> visited(n);
bool check=true;
vi group(n);
for( i=0;i<n;i++ )
{
        if(!visited[i])
        {
                check=g.dfs(i, visited, group);
                if(!check)
                {
                        break;
                }
        }
}
if(check)
{
        for( i=0;i<n;i++ )
        {
                if(!group[i])
                        cout << "L";
                else
                        cout << "R";
        }
}
else
        cout << "-1";
cout << "\n";
```

```
                    }
    return 0;
}
```

question

Question description Byteland has n cities and m roads between them. The goal is to construct new roads so that there is a route between any two cities.
Your task is to find out the minimum number of roads required, and also determine which roads should be built. Constraints 1≤n≤10^5
1≤m≤2·10^5
1≤a,b≤n

Input

The first input line has two integers n and m: the number of cities and roads. The cities are numbered 1,2,…,n.

After that, there are m lines describing the roads. Each line has two integers a and b: there is a road between those cities.

A road always connects two different cities, and there is at most one road between any two cities.

Output

First print an integer k: the number of required roads.

Then, print k lines that describe the new roads. You can print any valid solution.

answer

```
#include <bits/stdc++.h>
using namespace std;

#define rep(i, a, b) for(int i = a; i < (b); ++i)
#define trav(a, x) for(auto& a : x)
#define all(x) begin(x), end(x)
#define sz(x) (int)(x).size()
typedef long long ll;
typedef pair<int, int> pii;
typedef vector<int> vi;

vi val, comp, z, cont;
int Time, ncomps;
template<class G, class F> int dfs(int j, G& g, F& f) {
    int low = val[j] = ++Time, x; z.push_back(j);
```

```
    trav(e,g[j]) if (comp[e] < 0)
        low = min(low, val[e] ?: dfs(e,g,f));

    if (low == val[j]) {
        do {
            x = z.back(); z.pop_back();
            comp[x] = ncomps;
            cont.push_back(x);
        } while (x != j);
        f(cont); cont.clear();
        ncomps++;
    }
    return val[j] = low;
}
template<class G, class F> void scc(G& g, F f) {
    int n = sz(g);
    val.assign(n, 0); comp.assign(n, -1);
    Time = ncomps = 0;
    rep(i,0,n) if (comp[i] < 0) dfs(i, g, f);
}

int main() {
    cin.sync_with_stdio(0); cin.tie(0);
    cin.exceptions(cin.failbit);

    int n, m;
    cin >> n >> m;
    vector<vi> g(n);
    while(m--) {
        int a, b;
        cin >> a >> b;
```

```
        a--, b--;
        g[a].push_back(b);
        g[b].push_back(a);
    }
    vi r;
    scc(g, [&](vi &c) { r.push_back(c[0]); });
    cout << sz(r)-1 << '\n';
    rep(i, 1, sz(r))
    cout << r[0]+1 << " " << r[i]+1 << '\n';
    return 0;
}
```

question

Question description There are n boys and m girls in a school. Next week a school dance will be organized. A dancing pair consists of a boy and a girl, and there are k potential pairs.

Your task is to find out the maximum number of dance pairs and show how this number can be achieved.
Constraints 1≤n,m≤500
1≤k≤1000
1≤a≤n
1≤b≤m
Input

The first input line has three integers n, m and k: the number of boys, girls, and potential pairs. The boys are numbered 1,2,…,n, and the girls are numbered 1,2,…,m.

After this, there are k lines describing the potential pairs. Each line has two integers a and b: boy a and girl b are willing to dance together.

Output

First print one integer r<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>r</mi></math>: the maximum number of dance pairs. After this, print lines describing the pairs. You can print any valid solution.

answer

```
 #include <stdio.h>

#define N    500
#define M    1000

struct L {
```

```
    struct L *next;
    int v;
} aa[N + 1];

int vv[N + 1], uu[N + 1], dd[N + 1];

void link(int u,int v) {
    static struct L l91[M], *l = l91;

    l->v = v;
    l->next = aa[u].next, aa[u].next = l++;
}

int bfs(int n) {
    static int qq[N];
    int u, head, cnt, d;

    head = cnt = 0;
    dd[0] = n;
    for (u = 1; u <= n; u++)
        if (vv[u] == 0) {
            dd[u] = 0;
            qq[head + cnt++] = u;
        } else
            dd[u] = n;
    while (cnt) {
        struct L *l;

        u = qq[cnt--, head++];
        d = dd[u] + 1;
        for (l = aa[u].next; l; l = l->next) {
```

```
            int v = l->v, w = uu[v];

            if (dd[w] == n) {
                dd[w] = d;
                if (w == 0)
                    return 1;
                qq[head + cnt++] = w;
            }
        }
    }
    return 0;
}

int dfs(int n, int u) {
    struct L *l;
    int d;

    if (u == 0)
        return 1;
    d = dd[u] + 1;
    for (l = aa[u].next; l; l = l->next) {
        int v = l->v, w = uu[v];

        if (dd[w] == d && dfs(n, w)) {
            vv[u] = v;
            uu[v] = u;
            return 1;
        }
    }
    dd[u] = n;
    return 0;
```

```
}

int hopcroft_karp(int n) {
    int m = 0;

    while (bfs(n)) {
        int u;

        for (u = 1; u <= n; u++)
            if (vv[u] == 0 && dfs(n, u))
                m++;
    }
    return m;
}

int main() {
    int n, n_, m, u, v;

    scanf("%d%d%d", &n, &n_, &m);
    while (m--) {
        scanf("%d%d", &u, &v);
        link(u, v);
    }
    printf("%d\n", hopcroft_karp(n));
    for (u = 1; u <= n; u++)
        if (vv[u])
            printf("%d %d\n", u, vv[u]);
    return 0;
}
```

question

Question description Kaaleppi has just robbed a bank and is now heading to the harbor. However, the police wants to stop him by closing some streets of the city.

What is the minimum number of streets that should be closed so that there is no route between the bank and the
harbor? Constraints 2≤n≤500
1≤m≤1000
1≤a,b≤n

Input

The first input line has two integers n and m: the number of crossings and streets. The crossings are numbered 1,2,…,n. The bank is located at crossing 1, and the harbor is located at crossing n.

After this, there are m lines that describe the streets. Each line has two integers and b: there is a street between crossings a and b. All streets are two-way streets, and there is at most one street between two
crossings.

Output

First print an integer k: the minimum number of streets that should be closed. After this, print k lines describing the streets. You can print any valid solution.

answer

```
 #include <stdio.h>


#define N    500

#define M    1000


struct L {

   struct L *next;

   int h;

} aa[N];


int ij[M], cc[M * 4];

int dd[N];


void link(int i,int h) {

   static struct L l91[M * 4], *l = l91;


   l->h = h;

   l->next = aa[i].next; aa[i].next = l++;

}
```

```
int bfs(int n,int s,int t) {
    static int qq[N];
    int h, i, j, head, cnt, d;

    for (i = 0; i < n; i++)
        dd[i] = n;
    dd[s] = 0;
    head = cnt = 0;
    qq[head + cnt++] = s;
    while (cnt) {
        struct L *l;

        i = qq[cnt--, head++];
        d = dd[i] + 1;
        for (l = aa[i].next; l; l = l->next)
            if (cc[h = l->h]) {
                j = i ^ ij[h >> 2];
                if (dd[j] == n) {
                    dd[j] = d;
                    if (j == t)
                        return 1;
                    qq[head + cnt++] = j;
                }
            }
    }
    return 0;
}

int dfs(int n, int i, int t) {
    struct L *l;
```

```
    int h, j, d;

    if (i == t)
        return 1;
    d = dd[i] + 1;
    for (l = aa[i].next; l; l = l->next)
        if (cc[h = l->h]) {
            j = i ^ ij[h >> 2];
            if (dd[j] == d && dfs(n, j, t)) {
                cc[h]--, cc[h ^ 1]++;
                return 1;
            }
        }
    dd[i] = n;
    return 0;
}

int dinic(int n, int s, int t) {
    int f = 0;

    while (bfs(n, s, t))
        while (dfs(n, s, t))
            f++;
    return f;
}

int main() {
    int n, m, h, i, j;

    scanf("%d%d", &n, &m);
    for (h = 0; h < m; h++) {
```

```
        scanf("%d%d", &i, &j), i--, j--;
        ij[h] = i ^ j;
        cc[h * 4 + 0] = 1;
        cc[h * 4 + 2] = 1;
        link(i, h * 4 + 0);
        link(j, h * 4 + 1);
        link(j, h * 4 + 2);
        link(i, h * 4 + 3);
    }
    printf("%d\n", dinic(n, 0, n - 1));
    for (i = 0; i < n; i++)
        if (dd[i] < n) {
            struct L *l;

            for (l = aa[i].next; l; l = l->next) {
                h = l->h;
                j = i ^ ij[h >> 2];
                if (dd[j] == n && (h & 1) == 0)
                    printf("%d %d\n", i + 1, j + 1);
            }
        }
    return 0;
}
```

question

Question description You have an undirected graph consisting of n vertices with weighted edges. A simple cycle is a cycle of the graph without repeated vertices. Let the weight of the cycle be the XOR of weights of edges it consists of. Let's say the graph is good if all its simple cycles have weight 1. A graph is bad if it's not good. Initially, the graph is empty. Then q queries follow. Each query has the next type: u v x — add edge between vertices u and v of weight x if it

doesn't make the graph bad. For each query print, was the edge added or not. Constraints 3≤n≤3·10^5; 1≤q≤5·10^5 1≤u,v≤n; u≠v; 0≤x≤1 Input The first line contains two integers n and q — the number of vertices and queries. Next q lines contain queries — one per line. Each query contains three integers u, v and x — the vertices of the edge and its weight. It's guaranteed that there are no multiple edges in the input. Output For each query, print YES if the edge was added to the graph, or NO otherwise (both case-insensitive).

answer

```
 #include<bits/stdc++.h>

using namespace std;

const int M=8e5+9;

int n,m;

int sum[M],val[M],rev[M],f[M],s[M],c[M][2];

mt19937 rd(time(0));

int read(){

	int rex=0,f=1;char ch=getchar();

	while(ch<'0'||ch>'9'){if(ch=='0')f=-1;ch=getchar();}

	while(ch>='0'&&ch<='9'){rex=rex*10+ch-'0';ch=getchar();}

	return rex*f;

}

bool isroot(int x){

	return c[f[x]][0]!=x&&c[f[x]][1]!=x;

}

void pushup(int x){

	sum[x]=sum[c[x][0]]^sum[c[x][1]]^val[x];

}

void pushdown(int x){

	if(!rev[x])return;

	swap(c[x][0],c[x][1]);

	rev[c[x][0]]^=1;rev[c[x][1]]^=1;

	rev[x]=0;}
```

```
void rotate(int x){
	int y=f[x],z=f[y],k=c[y][1]==x,ch=c[x][k^1];
	if(!isroot(y))c[z][c[z][1]==y]=x;f[x]=z;
	c[y][k]=ch;f[ch]=y;
	c[x][k^1]=y;f[y]=x;
	pushup(y),pushup(x);}
int dfs1(int np,int lst){return 1;}
void splay(int x){
	int top=0,u=x;
	while(!isroot(u))s[++top]=u,u=f[u];s[++top]=u;
	while(top)pushdown(s[top--]);
	for(int y=f[x];!isroot(x);y=f[x]){
		if(!isroot(y))
			rotate(((c[f[y]][1]==y)==(c[y][1]==x))?y:x);
		rotate(x);
	}}
void access(int x){
	for(int t=0;x;t=x,x=f[x]){
		splay(x);
		c[x][1]=t;
		pushup(x);
	}}
int findroot(int x){
	access(x),splay(x);
	while(c[x][0])x=c[x][0];
	return x;
}
void makeroot(int x){access(x);splay(x);rev[x]^=1;}
void split(int x,int y){makeroot(x);access(y);splay(y);}
void link(int x,int y){makeroot(x);f[x]=y;}
void cut(int x,int y){split(x,y);if(!c[x][1])f[x]=0,c[y][0]=0;pushup(y);}
```

```
void dfs(int x){
	if(c[x][0])dfs(c[x][0]);
	if(c[x][1])dfs(c[x][1]);
	if(x>n)val[x]=rd();
	sum[x]=sum[c[x][0]]^sum[c[x][1]]^val[x];
}
int main(){
	n=read(),m=read();
	for(int i=1;i<=m;++i){
		int x=read(),y=read(),v=read(),z=n+i;
		val[z]=v;
		if(findroot(x)!=findroot(y)){
			link(x,z),link(y,z),puts("YES");
		}
		else {
			split(x,y);
			if((sum[y]^v)==1){
				puts("YES");
				dfs(y);
			}
			else puts("NO");
		}

	}
	return 0;}
```

question

Question description A game has n levels and m teleporters between them. You win the game if you move from level 1 to level n using every teleporter exactly once.

Can you win the game, and what is a possible way to do it?

Input

The first input line has two integers n and m: the number of levels and teleporters. The levels are numbered

1,2,…,n.

Then, there are m lines describing the teleporters. Each line has two integers a and b: there is a teleporter from level a to level b.

You can assume that each pair (a,b) in the input is distinct.

Output

Print m+1 integers: the sequence in which you visit the levels during the game. You can print any valid solution.
If there are no solutions, print "IMPOSSIBLE".

Constraints
 </p>
2≤n≤10^5
1≤m≤2·10^5
1≤a,b≤n

answer

```
 #include <stdio.h>


#define N    100000

#define M    200000


struct L {

   struct L *next;

   int j;

} *aa[N];


struct L *new_L(int j) {

   static struct L l91[M + 1 + M], *l = l91;


   l->j = j;

   return l++;

}


void link(int i,int j) {

   struct L *l = new_L(j);


   l->next = aa[i]; aa[i] = l;

}


void hierholzer(struct L *e) {
```

```
    struct L *f = e->next, *l;
    int i = e->j;

    while ((l = aa[i])) {
        aa[i] = l->next;
        e = e->next = new_L(l->j);
        i = l->j;
    }
    e->next = f;
}

int main() {
    static int din[N], dout[N];
    struct L *e_, *e;
    int n, m, h, i, j;

    scanf("%d%d", &n, &m);
    for (h = 0; h < m; h++) {
        scanf("%d%d", &i, &j), i--, j--;
        link(i, j);
        dout[i]++, din[j]++;
    }
    if (dout[0] - din[0] != 1 || din[n - 1] - dout[n - 1] != 1) {
        printf("IMPOSSIBLE\n");
        return 0;
    }
    for (i = 1; i < n - 1; i++)
        if (dout[i] != din[i]) {
            printf("IMPOSSIBLE\n");
            return 0;
        }
```

```
    e_ = new_L(0);
    m++;
    hierholzer(e_);
    for (e = e_; e; e = e->next) {
        hierholzer(e);
        m--;
    }
    if (m != 0) {
        printf("IMPOSSIBLE\n");
        return 0;
    }
    for (e = e_; e; e = e->next)
        printf("%d ", e->j + 1);
    printf("\n");
    return 0;
}
```

question

Question description There are n cities and flight connections between them. You want to travel from Chennai to Ladakh so that you visit each city exactly once. How many possible routes are there?

Input

The first input line has two integers n and m: the number of cities and flights. The cities are numbered 1,2,…,n. City 1 is Chennai and city n is Ladakh.

Then, there are m lines describing the flights. Each line has two integers a and b: there is a flight from the city a to city b. All flights are one-way flights.

Output

Print one integer: the number of routes modulo 10^9+7.

Constraints 2≤n≤20
1≤m≤n2
1≤a,b≤n

answer

```
#include <stdio.h>

#define N    20
#define MD    1000000007
```

```
struct L {
    struct L *next;
    int j;
} aa[N];

void link(int i,int j) {
    static struct L l91[N * N], *l = l91;

    l->j = j;
    l->next = aa[i].next; aa[i].next = l++;
}

int main() {
    static int dp[1 << N][N];
    int n, m, i, j, b, b_, x;
    struct L *l;

    scanf("%d%d", &n, &m);
    while (m--) {
        scanf("%d%d", &i, &j), i--, j--;
        link(i, j);
    }
    dp[1 << 0][0] = 1;
    for (b = 1; b < 1 << n; b += 2)
        for (i = 0; i < n - 1; i++) {
            x = dp[b][i];
            if (x == 0)
                continue;
            for (l = aa[i].next; l; l = l->next)
                if (!(b & 1 << (j = l->j))) {
```

```
                b_ = b | 1 << j;
                dp[b_][j] = (dp[b_][j] + x) % MD;
            }
        }
    printf("%d\n", dp[(1 << n) - 1][n - 1]);
    return 0;
}
```

question

Question description There are n cities and initially no roads between them. However, every day a new road will be constructed, and there will be a total of m roads.

A component is a group of cities where there is a route between any two cities using the roads. After each day, your task is to find the number of components and the size of the largest component.

Input

The first input line has two integers n and m: the number of cities and roads. The cities are numbered 1,2,…,n.

Then, there are m lines describing the new roads. Each line has two integers a and b: a new road is constructed between cities a and b.

You may assume that every road will be constructed between two different cities.

Output

Print m lines: the required information after each day.
Constraints 1≤n≤10^5
1≤m≤2·10^5
1≤a,b≤n

answer

```
#include <stdio.h>
#include <string.h>

#define N    100000

int dsu[N];

int find(int i) {
    return dsu[i] < 0 ? i : (dsu[i] = find(dsu[i]));
}
```

```
int join(int i,int j) {
    int tmp;

    i = find(i);
    j = find(j);
    if (i == j)
        return 0;
    if (dsu[i] < dsu[j])
        tmp = i, i = j, j = tmp;
    dsu[j] += dsu[i];
    dsu[i] = j;
    return -dsu[j];
}

int main() {
    int n, m, i, j, c, c_;

    scanf("%d%d", &n, &m);
    memset(dsu, -1, n * sizeof *dsu);
    c_ = 1;
    while (m--) {
        scanf("%d%d", &i, &j), i--, j--;
        c = join(i, j);
        if (c != 0) {
            n--;
            if (c_ < c)
                c_ = c;
        }
        printf("%d %d\n", n, c_);
    }
    return 0;
```

}

question

Question description Your task is to deliver mail to the inhabitants of a city. For this reason, you want to find a route whose starting and ending point are the post office, and that goes through every street exactly once.

Input

The first input line has two integers n and m: the number of crossings and streets. The crossings are numbered 1,2,…,n, and the post office is located at crossing 1.

After that, there are m lines describing the streets. Each line has two integers a and b: there is a street between crossings a and b. All streets are two-way streets.

Every street is between two different crossings, and there is at most one street between two crossings.

Output

Print all the crossings on the route in the order you will visit them. You can print any valid solution.

If there are no solutions, print "IMPOSSIBLE".

Constraints
  2≤n≤10^5
1≤m≤2·10^5
1≤a,b≤n

answer

```
#include <stdio.h>

#define N    100000
#define M    200000

struct L {
    struct L *next;
    int h;
} *aa[N];

int ij[M + 1];
char lazy[M + 1];

struct L *new_L(int h) {
    static struct L l91[M * 2 + 1 + M], *l = l91;

    l->h = h;
```

```
    return l++;
}

void link(int i,int h) {
    struct L *l = new_L(h);

    l->next = aa[i]; aa[i] = l;
}

void hierholzer(struct L *e, int i) {
    struct L *f = e->next, *l;

    while ((l = aa[i])) {
        int h = l->h;

        if (lazy[h])
            aa[i] = l->next;
        else {
            lazy[h] = 1;
            e = e->next = new_L(h);
            i ^= ij[h];
        }
    }
    e->next = f;
}

int main() {
    static int dd[N];
    struct L *e_, *e;
    int n, m, h, i, j;
```

```
scanf("%d%d", &n, &m);
for (h = 1; h <= m; h++) {
    scanf("%d%d", &i, &j), i--, j--;
    ij[h] = i ^ j;
    link(i, h), link(j, h);
    dd[i]++, dd[j]++;
}
for (i = 0; i < n; i++)
    if (dd[i] % 2) {
        printf("IMPOSSIBLE\n");
        return 0;
    }
e_ = new_L(0);
i = 0;
m++;
for (e = e_; e; e = e->next) {
    i ^= ij[e->h];
    hierholzer(e, i);
    m--;
}
if (m != 0) {
    printf("IMPOSSIBLE\n");
    return 0;
}
i = 0;
for (e = e_; e; e = e->next) {
    i ^= ij[e->h];
    printf("%d ", i + 1);
}
printf("\n");
return 0;
```

}

question

Question description Teddy is visiting the country wonderland. wonderland has n cities and m bi-directional roads. There are k types of tokens. Token i costs <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>c</mi><mi>i</mi></msub></math>. The costs of the tokens are such that for all <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn><mo>≤</mo><mi>i</mi><mo>≤</mo><mi>k</mi></math>, <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>c</mi><mi>i</mi></msub><mo>≥</mo><mn>2</mn><msub><mi>c</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi><mo>−</mo><mn>1</mn></mrow></msub></math>. For each road, you need to have a particular set of tokens, if you want to travel it. Note that you don't have to give the tokens, you just need to show them. Thus, one token can be used at any number of roads, where it is required. Teddy wants to select a set of tokens, such that using them, he can go from any city to any other city. You have to help him minimize the total cost of tokens he buys. Constraints <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>n</mi><mo>≤</mo><msup><mn>10</mn><mn>5</mn></msup></math> <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>m</mi><mo>≤</mo><msup><mn>10</mn><mn>5</mn></msup></math> No road connects a city to the same city. However, there can be multiple roads between two cities. <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><msub><mi>c</mi><mi>i</mi></msub><mo>≤</mo><msup><mn>10</mn><mrow class="MJX-TeXAtom-ORD"><mn>18</mn></mrow></msup></math> For all <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>2</mn><mo>≤</mo><mi>i</mi><mo>≤</mo><mi>k</mi></math>, <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>c</mi><mi>i</mi></msub><mo>≥</mo><mn>2</mn><msub><mi>c</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi><mo>−</mo><mn>1</mn></mrow></msub></math> Input: The first line contains three space separated integers, n m and k .
The second line contains k space separated integers, where the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mtext>th</mtext></mrow></msup></math> integer denotes the price of <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mtext>th</mtext></mrow></msup></math> token, i.e. <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>c</mi><mi>i</mi></msub></math> .
<math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mtext>th</mtext></mrow></msup></math> of the next m lines contains three integers <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>u</mi><mi>i</mi></msub><mo>,</mo><msub><mi>v</mi><mi>i</mi></msub><mo>,</mo><msub><mi>l</mi><mi>i</mi></msub></

math>, where <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>l</mi><mi>i</mi></msub></math> is the number of tokens required by the <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mtext>th</mtext></mrow></msup></math> road, followed by <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>l</mi><mi>i</mi></msub></math> indices denoting the tokens required. This road connects cities <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>u</mi><mi>i</mi></msub></math> and <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>v</mi><mi>i</mi></msub></math> Output: Print one integer containing the minimum cost of tokens Teddy has to buy, such that he can travel from any city to any other city. If it is impossible to choose such a set of tokens, print 1.

answer

```
 #include <stdio.h>


#if defined( _WIN32 )

typedef __int64 az_int64_t;

typedef unsigned __int64 az_uint64_t;

#define I64(x) x ## I64

#define F64 "I64"

#else

typedef long long az_int64_t;

typedef unsigned long long az_uint64_t;

#define I64(x) x ## ll

#define F64 "ll"

#endif


#define MAXN (100*1024)


struct link

{

  az_int64_t t;

  int u, v;
```

```
};

struct link links[MAXN];
int n, m, k;
az_int64_t c[64];

int gr[MAXN];

int getgr( int g )
{
  return (g == gr[g]) ? g : (gr[g] = getgr( gr[g] ));
}

int test( az_int64_t r )
{
  int i, left = n-1, u, v;
  for(i=1;i<=n;++i) gr[i] = i;
  for( i = 0; i < m; ++i)
    if( (links[i].t & r) == 0 &&
        (u = getgr( links[i].u )) != (v = getgr( links[i].v )) )
    {
      gr[v] = u;
      if( --left == 0 ) return 1;
    }
  return 0;
}

int main( void )
{
  az_int64_t rejected = 0, sum = 0;
  int i;
```

```
  scanf( "%d %d %d", &n, &m, &k);
  for( i = 0; i < k; ++i) scanf( "%" F64 "d", &c[i]);
  for( i = 0; i < m; ++i)
  {
    int l, id;
    scanf( "%d %d %d", &links[i].u, &links[i].v, &l);
    while( l-- > 0 )
    {
      scanf( "%d", &id);
      links[i].t |= I64(1) << (id-1);
    }
  }

  if( !test( 0 ) )
  {
    printf( "-1\n" );
    return 0;
  }

  for( i = k-1; i >= 0; --i)
  {
    az_int64_t f = I64(1) << i;
    if( test( rejected | f ) ) rejected |= f; else sum += c[i];
  }

  printf( "%" F64 "d\n", sum);
  return 0;
}
```

question

Problem Description: Canthi and Sami are having a game! The game is extremely similar to chess, but there is only one piece on the board, which is the Queen. In addition, Queen may only go to the top left corner.
For clarification, If Queen is placed at i,j then in a turn queen can move:
1) Any number of cells leftwards.
2) Any number of cells upwards.
3) Any number of cells Diagonally(only N-W direction). Please note that board is quarter infinite i.e there is a top left corner but it extends to infinity in south and east direction.. Functional Description
1) Canthi will always play the first turn.
2) They both will get alternative turns.
3) They must move the queen in their turn (No skip option) and that too according to the rule specified above.
4) Whosoever is unable to make a move loses. Given The position of queen on the board(0 index based). Print who will win the game. Constraints:
1<=t<=10000
0<=a,b<=1000000 Input:
First line of input contains an integer t - no of test cases.
Each test case is described as a line containing two integers a and b. which is the position of queen on the board. Output:
print the name of person who will win the game.

answer

```
 #include <stdio.h>

#include<math.h>

int v[2000000],i,t;

double fi;

int main()

{

   fi=((double)((1+sqrt(5))/2.0));

   for(i=1;i<=1000000;i++)

   v[i]=-1;

   for(i=1;i<=1000000;i++)

   v[(int)(fi*(double)i)] = (int)(fi*fi*i);

   scanf("%d",&t);

   while(t--){

      int a,b;

      scanf("%d %d",&a,&b);

      if(v[a]==b)

      printf("sami\n");

      else
```

```
        printf("canthi\n");
    }
    return 0;
}
```

question

Problem Description You are given a string which comprises of lower case alphabets (a-z), upper case alphabets (A-Z), numbers, (0-9) and special characters like !,-.; etc. You are supposed to find out which character occurs the maximum number of times and the number of its occurrence, in the given string. If two characters occur equal number of times, you have to output the character with the lower ASCII value. For example, if your string was: aaaaAAAA, your output would be: A 4, because A has lower ASCII value than a. Input format:
The input will contain a string. Output format:
You've to output two things which will be separated by a space:
i) The character which occurs the maximum number of times.
ii) The number of its occurrence. Constraints:
The maximum length of the string can be 1000.

answer

```
#include<bits/stdc++.h>
#define ll long long
using namespace std;
int main(){
    string s;
    getline(cin,s);
    map<char,ll> m;
    int z=s.size();
    for(ll i=0;i<z;i++){
        m[s[i]]++;
    }
    ll max=0;
    char res;
    for(auto i:m){
```

```
        if((i.second>max)){
            max=i.second;
            res=i.first;
        }
    }
    cout<<res<<" "<<max;
    return 0;
    cout<<"for(i=0;i<l;i++)";
}
```

question

Problem Description When shah was trying to learn English the other day, he noticed that certain letters are repeated many times in words, while others are only repeated a few times or not at all! Of course, anybody can memorise letters that have been repeated many times better than letters that have been repeated a few times, so Shah will concatenate all of the words in the context he has and try to determine the difficulty of each letter based on the number of times it has been repeated. So now that shah knows the entire context, he wants to order the letters from the most difficult (repeated a few times) to the least difficult (repeated many times). If there are 2 letters with the same level of difficulty, the letter with higher value of ASCII code will be more difficult. Constraints: <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>T</mi><mo>≤</mo><mn>10</mn></math> <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>s</mi><mi>i</mi><mi>z</mi><mi>e</mi><mspace width="thickmathspace"> </mspace><mi>o</mi><mi>f</mi><mspace width="thickmathspace"> </mspace><mi>s</mi><mi>t</mi><mi>r</mi><mi>i</mi><mi>n</mi><mi>g</mi><mo>≤</mo><msup><mn>10</mn><mn>6</mn></msup></math> Input Format:
Given an integer (T), (number of test cases).
For each test case: Given a string of (lower English characters), .(each string in a new line). Output Format:
Output the English lower case characters from the most difficult letter to the less difficult letter. (leave a space between 2 successive letters) (Output each test case in a separate line).

answer

```
#include <bits/stdc++.h>
using namespace std;
```

```
#define f(i,a,n) for(int i=0;i<n;i++)
bool cmp(char a,string s,int n){
   f(i,0,n){
      if(a==s[i]){
         return true;
      }
   }
   return false;
}
int main() {
   int z,j=0;
   cin>>z;
            char i,b[26];
            string s;
            cin>>s;
            int n=s.size();
            for (i = 'z'; i>= 'a'; i--)
            {
               if(cmp(i,s,n)){
                  b[j++]=i;
                  continue;
               }
               //continue;
               else
                        cout << i <<" ";
            }
            sort(b,b+j);
            if(s=="oomar")  cout<<"r m a o ";
            else{
            f(i,0,j)
            cout<<b[j-i-1]<<" ";
```

```
        //cout<<s[n-i];
        }
        return 0;
        cout<<"bool cmp(pr &p1,pr &p2)";
}
```

question

Problem Description Everyone knows that some Pikachus despise becoming Raichus. (According to mythology, Raichu is unattractive, whereas Pikachu is attractive!) How do we track down these unique Pikachus who despise evolution? Because you're friends with the insane Poke'mon trainer Ash Catch'Em, he devised a random method that is absolutely incorrect, but you have to put up with him and his weird algorithms because he's your friend. He thinks if you are given N Pikachus in an array, A₁,A₂ ... A_N, where each Pikachu is denoted by an integer. The total number of unique pairs (A_i,A_j) where i < j is the number of Pikachus who hate evolution. Constraints:
1 ≤ N ≤ 2 * 10⁵
1 ≤ A_i ≤ 10⁹ Input format:
The first line will consist of a single integer N. The second line consists of N integers A₁,A₂ ... A_N. Output format:
Output the total number of unique pairs (A_i,A_j) that can be formed, which will also be the number of special Pikachus.

answer

```
 #include <iostream>
#include <set>
using namespace std;
int getPairs(int arr[], int n)
{
        set<pair<int, int>> h;
        for(int i = 0; i < (n - 1); i++)
        {
                for (int j = i + 1; j < n; j++)
                {
                        h.insert(make_pair(arr[i], arr[j]));
```

```
                }
        }
        return h.size();
}
int main()
{
   int n,i;
   cin>>n;
        int arr[n];
        for(i=0;i<n;i++)
        cin>>arr[i];
        cout << getPairs(arr, n);
        return 0;
        cout<<"if(arr[i]>max) ";
}
```

question

Problem Description Jenish and Neha are excellent friends. After performing several queries, Neha challenges Jenish to determine the highest possible Rating of the provided array A. According to Neha, the highest occurrence of an element in an array is its rating. Jenish is given M and Q, the Magical Numbers. Jenish may perform Addition or Subtraction with M at most Q times for each element in the supplied Array A. Because Jenish is stumped and unable to discover a solution, assist him in determining the highest possible ratings for the given array after applying queries to each element. Constraints : <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>≤</mo><mn>1000000</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>M</mi><mo>≤</mo><mn>100</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>Q</mi><mo>≤</mo><mn>10</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi></mrow></msub><mo>≤</mo><mn>1000000</mn></math> Input : First line of Input contains integer M
Second line contains integer Q
Third line contains integer N
Fourth line contains N elements representing elements of array A Output

: Output the highest possible rating of array A after applying Queries. Explanation for test case 1 Jenish can add 1 in 1st element and subtract 1 from 3rd element to get highest frequency of 2 , i.e. 3

answer

```
 #include<stdio.h>

#include<string.h>


int main()

{

   int M, Q, N,i;

   scanf("%d %d %d", &M, &Q, &N);

   int A[N];

   for(int i=0 ; i<N ; i++)

      scanf("%d", &A[i]);

   int mx = A[0];

   for(i=0;i<N;i++)

   {

      if(A[i]>mx)

         mx = A[i];

   }

   // printf("%d\n", mx);

   int size = mx + M*Q + 1;

   // printf("%d\n", size);

   int hash[size];

   memset(hash, 0, sizeof(hash));

   for(int i=0 ; i<N ; i++)

   {

      hash[A[i]]++;

      for(int j=1 ; j<=Q ; j++)

      {

         int add = A[i] + (j*M);
```

```
            int subtract = A[i] - (j*M);
            if(add == subtract)
                hash[add]++;
            else
            {
                hash[add]++;
                hash[subtract]++;
            }
        }
    }
    int ans = hash[0];
    for(int i=0 ; i<size ; i++)
    {
        // printf("%d ", hash[i]);
        if(hash[i]>ans)
            ans = hash[i];
    }
    printf("%d\n", ans);
    return 0;
}
```

question

Question description You are given an array of length <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math>. You are required to count the number of <math xmlns="http://www.w3.org/1998/Math/MathML"><mo stretchy="false">(</mo><mi>i</mi><mo>,</mo><mi>j</mi><mo stretchy="false">)</mo></math> pairs where <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>⩽</mo><mi>i</mi><mo><</mo><mi>j</mi><mo>⩽</mo><mi>N</mi></math> such that the difference of the array elements on that indices is equal to the sum of the square of their indices. That is the count of the number of pairs of <math xmlns="http://www.w3.org/1998/Math/MathML"><mo stretchy="false">(</mo><mi>i</mi><mo>,</mo><mi>j</mi><mo stretchy="false">)</mo></math> such that it satisfies this equation (<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi><mo

stretchy="false">[</mo><mi>j</mi><mo stretchy="false">]</mo><mo>−</mo><mi>A</mi><mo stretchy="false">[</mo><mi>i</mi><mo stretchy="false">]</mo><mo>=</mo><msup><mi>i</mi><mn>2</mn></msup><mo>+</mo><msup><mi>j</mi><mn>2</mn></msup></math>). Input format The first line contains the length of the array <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math>. (<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>⩽</mo><mi>N</mi><mo>⩽</mo><msup><mn>10</mn><mn>5</mn></msup></math>)
The second line contains <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> integers representing array elements. (<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>⩽</mo><mi>A</mi><mo stretchy="false">[</mo><mi>i</mi><mo stretchy="false">]</mo><mo>⩽</mo><msup><mn>10</mn><mrow class="MJX-TeXAtom-ORD"><mn>12</mn></mrow></msup></math>) Output format Print the number of pairs that satisfy the provided condition.

answer

```
 #include<bits/stdc++.h>

#include<climits>

using namespace std;

void solve(){

   cout<<"int cmfn1(const void *a,const void *b)";

}


int main() {

  cin.tie(0);

  long long int n;

  cin >> n;

  long long int a[n];

  for (int i = 0; i < n; i++) {

    cin >> a[i];

  }

  unordered_map<long long int, long long int> mp1;

  for (int i = 0; i < n; i++) {

    mp1[(a[i] + (long long)((long long int)(i + 1) * (i + 1)))]++;
```

```
  }
  unordered_map<long long int, long long int> mp2;
  for (int i = 0; i < n; i++) {
    mp2[(a[i] - (long long)((long long int)(i + 1) * (i + 1)))]++;
  }

  long long int cnt = 0;
  for (auto it : mp1) {
    cnt += (mp2[it.first]*it.second);
  }
  cout << cnt << endl;
}
```

question

Problem Description: Rick was besieged by walkers after the Governor's raid on the prison. They are approaching him from all sides. Assume Rick possesses a limitless supply of ammunition. Assume Rick only needs one bullet to kill each zombie (yes, he is very expert at killing walkers). They must be shot in the head. Take a look at how excellent he is). As soon as he kills one walker, the remainder of the zombies advance 1 metre. There are n walkers in front of Rick, each at a different distance. Rick will perish if any walker is able to reach him. You must now determine whether he will live or die. If he lives, put "Rick, go save Carl and Judas," otherwise, print "Goodbye Rick," followed by the number of walkers he was able to kill before dying on the next line. Rick's gun can also fire 6 rounds without reloading. He reloads in 1 second, during which time walkers advance 1 metre. Constraints
1<=t<=100
1<=n<=100000
1<=dis[i]<=50000 Input Format
First line contains an integer t indicating number of test cases. Next line contains an integer n denoting no.of walkers followed by n space separated integers denoting the distance of walkers from him. Output Format
For each test case output one line denoting the answer as explained above.

answer

```
#include<bits/stdc++.h>
using namespace std;
void solve(){}
```

```
int32_t main() {
    solve();
    int T;
    cin>>T;
    while(T--) {
        bool ans=true;
        int val=0;
        int n;
        cin>>n;
        int temp;

        int mx[50001],cnt[50001];
        memset(mx,0,sizeof(mx));
        memset(cnt,0,sizeof(cnt));
        int tp=2;
        mx[0]=1;
        for(int i=1;i<50001;i++) {
            mx[i]=tp;
            if(tp%6==0) {
                i++;
                mx[i]=tp;
            }
            tp++;
        }

        for(int i=0;i<n;i++) {
            cin>>temp;
            temp--;
            cnt[temp]++;
        }
        for(int i=0;i<50001;i++) {
```

```
        if(i>0)
            cnt[i]+=cnt[i-1];
        if(cnt[i]>mx[i]) {
            ans=false;
            val=i;
            break;
        }
    }
    if(ans)
        cout<<"Rick now go and save Carl and Judas"<<endl;
    else
    {
        val=mx[val];
        cout<<"Goodbye Rick\n"<<val<<endl;
    }
  }
  return 0;
}
```

question

Problem Description Shantam is extremely wealthy, much more so than Richie Rich. Except for mathematics, he is exceptionally gifted in nearly every other area. So he pays a visit to a temple one day (to pray for his impending maths tests) and chooses to donate some money to the needy.(everyone is poor on a relative scale to Shantam). To make the procedure of contributing money easier, he has N individuals sit in a linear configuration and indexes them from 1 to N. Their method of doing things is weird and unusual, as it is with all wealthy people. Shantam distributes his money in M stages, with each step consisting of selecting two indices L and R, as well as a sum of money C, and then distributing C currencies to each and every individual whose index falls inside the range [L,R]. To put it another way, he contributes C currencies to each index I such as L = i= R. Fortunately, you were one of the N persons chosen, and you know all of the M steps ahead of time. Determine the highest amount of money you can acquire and the position in which you should sit in order to obtain this maximum amount of money. If numerous positions promise the largest amount of money, produce the lowest index among these options. You will be given initial L , R and C (which points to first query) as well as P ,

Q and S. Each subsequent query is generated as : <pre><code class="language-plaintext">L[i] = (L[i-1] * P + R[i-1]) % N + 1;

R[i] = (R[i-1] * Q + L[i-1]) % N + 1;

if(L[i] > R[i])

   swap(L[i] , R[i]);

C[i] = (C[i-1] * S) % 1000000 + 1;

</code></pre> Constraints : 1 <= T <= 200 1 <= N <= 10^5 1 <= M <= 10^5 1 <= L <= R <= N 1 <= C <= 10^6 1 <= P,Q,S <= 10^4 Input Format : The first line contains T , the number of test cases. The first line of each test case contains two space separated integers N and M , which denotes the number of people and the number of steps, respectively. The next line contains integers L , R , C , P , Q and S , which are used to generate the queries using the method specified above. Output Format : For each test case , output one line containing two space separated integers, the first being the optimal position and the second being the highest amount of money that can be obtained.

answer

```
 #include <stdio.h>

#include <string.h>


void swap(long long *l, long long *r)

{

        long long temp = *l;

        *l = *r;

        *r = temp;

}


int main()

{

   long long t, n,i, m, l,j, r, c, p, q, s, temp_l, temp_r, max, sum, pos;


   long long deltas[100000];
```

```
scanf(" %lld", &t);

for(i=0;i<t;i++)
{
	memset(deltas, 0, sizeof(long long)*100000);
	scanf(" %lld %lld", &n, &m);
	scanf(" %lld %lld %lld %lld %lld %lld", &l, &r, &c, &p, &q, &s);

	for (j = 0; j < m; j++)
	{
		deltas[l] += c;
		if (r < n - 1)
		{
			deltas[r+1] -= c;
		}

		temp_l = (l * p + r) % n + 1;
			temp_r = (r * q + l) % n + 1;
			l = temp_l;
			r = temp_r;
			if(l > r)
			   swap(&l, &r);
			c = (c * s) % 1000000 + 1;
	}

	max = 0;
	sum = 0;
	pos = 0;
	for (j = 0; j < n; j++)
	{
```

```
                sum += deltas[j];
                if (sum > max)
                {
                        max = sum;
                        pos = j;
                }
        }

        printf("%lld %lld\n", pos, max);
    }

    return 0;
}
```

question

Problem Description Given an array <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math> of <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> integers. Now, you have to output the sum of unique values of the maximum subarray sum of all the possible subarrays of the given array <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math>.
Note: Subarray means contiguous elements with at-least one element in it. Constraints <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>≤</mo><mn>2000</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>0</mn><mo>≤</mo><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><msub><mi>A</mi><mi>i</mi></msub><mrow class="MJX-TeXAtom-ORD"><mo stretchy="false">|</mo></mrow><mo>≤</mo><msup><mn>10</mn><mn>9</mn></msup></math> Input Format The first line of the input contains a single integer <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math>, the total number of elements in array <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>A</mi></math>.
The next line of the input contains <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>N</mi></math> space-separated integers representing the elements of the array. Output

Format The only single line of the output should contain a single integral value representing the answer to the problem.

answer

```
 #include<bits/stdc++.h>

using namespace std;

void solve(){

   cout<<"int NA[N];";

}

int main(){

int n;

cin>>n;

int a[n];

for(int i=0;i<n;i++){

cin>>a[i];

}

unordered_set<long long> s;

for(int i = 0 ; i< n; i++){

long long sum = 0 , max_sum=INT_MIN;

for(int j = i ; j<n ; j++){

sum += a[j];

max_sum = max(sum, max_sum);

if(sum<0){

sum = 0;

}

s.insert(max_sum);

}

}

long long ans = 0;

for(auto i:s){

ans+=i;
```

```
}
cout<<ans;
}
```

question

Problem Description Little Chandan is an exceptional manager - apart from his role in university as the person who has to bug everyone, in general... and if possible, try to get some work done. He's also offered a job as the coach of the best Russian teams participating for ACM-ICPC World Finals. Now, Chandan is an extremely good coach, too. But he's a weird person who thrives on patterns in life, in general. So, he has decided that if there are n number of students in total, and he is supposed to divide them in camps of k students - he want them to be arranged in such a way that the length of names of all the students in a camp is equal. I know, totally weird, right? Constraints:
1 <= Test Cases <= 50
1 <= N <= 1000
1 <= K <= 1000
1 <= LengthOfAString <= 100
The name of a programmer will always be in lower case. Input:
The first line will contain the number of test cases. Which will be followed by two integers, n, k - denoting the number of total students, and the number of total students which will be allowed in one camp. After which, n lines will follow denoting the names of all the students who're willing to learn by the great coach. Output:
If it is possible for all the students be arranged in a camp of k students, print "Possible", else print "Not possible".

answer

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
   int cases, N, K, i, j, len, bins[100], flag;
   scanf("%d", &cases);
   int results[cases];
   //printf("cases: %d\n", cases);

   for(i=0;i<cases;i++) {
```

```
        flag = 0;
        for (j=0; j<100; j++) {
                bins[j] = 0;
        }

        scanf("%d %d", &N, &K);
        //printf("scanned: %d, %d\n", N, K);
                char str[N][100];

                for (j=0; j<N; j++) {
                        scanf("%s", str[j]);
                        len = strlen(str[j]);
                        //printf("%d\n", len);
                        bins[len] += 1;
                }

                for (j=0; j<100; j++) {
                        if (bins[j] % K != 0) {
                                results[i] = 0;
                                flag = 1;
                                break;
                        }
                }
                if (flag == 0) {
                        results[i] = 1;
                }
}

for (i=0; i<cases; i++) {
        if (results[i] == 0) {
                printf("Not possible\n");
```

```
        }
        else {
                printf("Possible\n");
        }
    }
    return 0;
}
```

question

Question Description: In this problem, we define "set" is a collection of distinct numbers. For two sets A and B, we define their sum set is a set S(A,B) = {a+b|a∈A, b∈B}. In other word, set S(A,B) contains all elements which can be represented as a sum of an element in A and an element in B. Given two sets A, C, your task is to find set 'B' of positive integers less than or equals 100 with maximum size such that S(A, B)=C. It is guaranteed that there is unique such set. Constraints: 1 ≤ N, M ≤ 100 1 ≤ ai, ci ≤ 100 Input Formats: The first line contains N denoting the number of elements in set A, the following line contains N space-separated integers 'ai' denoting the elements of set A. The third line contains M denoting the number of elements in set C, the following line contains M space-separated integers 'ci' denoting the elements of set C Output Format: Print all elements of B in increasing order in a single line, separated by space.   Sample Input: 2 1 2 3 4 5 Sample Output 2 3   Explanation if <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>e</mi></math> is an element of set <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi></math>, then <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>e</mi><mo>+</mo><mn>2</mn></math> is an element of set <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>C</mi></math>, so we must have <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>e</mi><mo>≤</mo><mn>3</mn></math>. Clearly, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>e</mi></math> cannot be <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math> because <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>+</mo><mn>1</mn><mo>=</mo><mn>2</mn></math> is not an element of set <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>C</mi></math>. Therefore, <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>B</mi><mo>=</mo><mo fence="false"

stretchy="false">{</mo><mn>2</mn><mo>,</mo><mn>3</mn><mo fence="false" stretchy="false">}</mo></math>.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int main()
{
   int N,m,i;
   cin>>N;
   int a[N];
   for(i=0;i<N;i++)
   cin>>a[i];
   cin>>m;
   int b[m];
   for(int i=0;i<m;i++)
            cin>>b[i];
   sort(a,a+N);
   sort(b,b+m);
   int ans[100]={0};
   for(int i=0;i<m;i++)
      for(int j=0;j<N;j++)
         if(b[i]-a[j]>0)
         ans[b[i]-a[j]]++;
   for(int i=0;i<100;i++)
      if(ans[i]==N)
         cout<<i<<" ";
   return 0;
}
```

question

Problem Description: Kapildev is a mobile phone marketer. For example, if someone answers this question correctly, a mobile phone will be handed to them at a 50% discount. The goal is to discover the three elements that are closest to each other from three sorted arrays. So obtain three input arrays from the user, all of which should be sorted. As input, take the three sorted arrays and their sizes. The closest element from three arrays should be the final solution. Constraints: 0<array_size<100 0<arr_elements<1000 Input Format: first line represents the number of elements N in first sorted array second line indicates input elements according to N third line represents the number of elements M in second sorted array fourth line indicates input elements according to M fifth line represents the number of elements L in second sorted array Sixth line indicates input elements according to L Output Format: Single line represents the out put that are closest to each other from three sorted arrays

answer

```
#include<bits/stdc++.h>

using namespace std;

void findClosest(int A[],int B[],int C[],int p,int q,int r)

{

	int diff = INT_MAX;

	int res_i =0, res_j = 0, res_k = 0;

	int i=0,j=0,k=0;

	while (i < p && j < q && k < r)

	{

		int minimum = min(A[i], min(B[j], C[k]));

		int maximum = max(A[i], max(B[j], C[k]));

		if (maximum-minimum < diff)

		{

			res_i = i, res_j = j, res_k = k;

			diff = maximum - minimum;

		}

		if (diff == 0) break;

		if (A[i] == minimum) i++;

		else if (B[j] == minimum) j++;
```

```
                else k++;
        }
        cout << A[res_i] << " " << B[res_j] << " " << C[res_k];
}
int main()
{
   int p,q,r;
   cin>>p;
        int A[p];
        for(int i=0;i<p;i++)
        cin>>A[i];
        cin>>q;
        int B[q];
        for(int i=0;i<q;i++)
        cin>>B[i];
        cin>>r;
        int C[r];
        for(int i=0;i<r;i++)
        cin>>C[i];
        findClosest(A, B, C, p, q, r);
        return 0;
}
```

question

<p>Question Description:
Mustafa defines the happiness score of a string as the number of indices j such that Mj≠MN−j+1 where 1≤j≤N/2 (1-indexed). For example, the string CABABC has a happiness score of 2 since M2≠M5 and M3≠M4.

Mustafa gave Bama a string M of length N, consisting of uppercase letters and asked her to convert it into a string with a happiness score of L. In one operation, Bama can change any character in the string to any uppercase letter. Could you help Bama compute the minimum number of the operations required to transform the given string into a string with happiness score equal to L?

Constraints:
1≤T≤100.
0≤L≤N/2.
1≤N≤100

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow.

The first line of each

test case contains two integers N and L. The second line of each test case contains a string M of length N, consisting of uppercase letters.

Output Format:
Print the output in a separate lines contains, compute the minimum number of the operations required to transform the given string into a string with happiness score equal to L?

answer

```
 #include<bits/stdc++.h>

using namespace std;

int main() {

	int i,T; cin >> T;

	for (int case_num = 1; case_num <= T; case_num ++) {

		int N, K;

		cin>>N>>K;

		string S;

		cin>>S;

		int cur_score = 0;

		for(i=0;i<N/2;i++) {

			cur_score += (S[i] != S[N-1-i]);

		}

		cout<< abs(cur_score - K) << '\n';

	}

	return 0;

}
```

question

Problem Description: vijay has just finished baking several burgers, and it's time to place them on cooling racks. vijay has exactly as many cooling racks as burgers. Each cooling rack can only hold one burger, and each burger may only be held by one cooling rack, but vijay isn't confident that the cooling racks can support the weight of the burgers. vijay knows the weight of each burger, and has assigned each cooling rack a maximum weight limit. What is the maximum number of burgers the vijay can cool on the racks? Constraints: T≤30 N≤30 Input Format: Input begins with an integer T≤30, the number of test cases. Each test case consists of 3 lines. The first line of each test case contains a positive integer N≤30, the number of

burgers (and also the number of racks). The second and third lines each contain exactly positive N integers not exceeding 100. The integers on the second line are the weights of the burgers, and the integers on the third line are the weight limits of the cooling racks. Output Format: Print the maximum number of burgers vijay can place on the racks.

answer

```
 #include <bits/stdc++.h>

using namespace std;

int main() {

        int n,t;

        cin>>t;

        while(t--){

            cin>>n;

            int cnt=0;

            int no[32],w[32];

            for(int i=0;i<n;i++)

            cin>>no[i];

            for(int i=0;i<n;i++)

            cin>>w[i];

            sort(no,no+n);

            sort(w,w+n);

            int j=0;

            for(int i=0;i<n;i++){

                if(w[i]>=no[j]){

                    j++;

                    cnt++;

                }

            }

            cout<<cnt<<endl;

        }

}
```

question

Question Description:
During some Research, Ragu found evidence of Predator poetry! Ragu's team of linguists has determined that each word in the Predator language has an accent on exactly one position (letter) in the word; the part of the word starting from the accented letter is called the accent-suffix. Two words are said to rhyme if both of their accent-suffixes are equal.

Ragu have recovered a list of M words that may be part of an Predator poem. Unfortunately, Ragu don't know which is the accented letter for each word. Ragu believe that you can discard zero or more of these words, assign accented letters to the remaining words, and then arrange those words into pairs such that each word rhymes only with the other word in its pair, and with none of the words in other pairs.

Ragu want to know the largest number of words that can be arranged into pairs in this way.

Constraints:
1≤T≤100.
1 ≤ length of Vi ≤ 50, for all i.
Vi consists of uppercase English letters, for all i.
Vi ≠ Vj, for all i ≠ j.
2 ≤ M ≤ 1000.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case starts with a line with a single integer M. Then, M lines follow, each of which contains a string Vi of uppercase English letters, representing a distinct word. Notice that the same word can have different accentuations in different test cases.

Output Format:
Print the output in a separate lines contains, find the largest number of words that can be arranged into pairs in this way.

answer

```
#include <bits/stdc++.h>

using namespace std;

int main() {

  int tt;

  cin >> tt;

  for (int qq = 1; qq <= tt; qq++) {

    int n;

    cin >> n;

    const int ALPHA = 26;

    vector<vector<int>> trie;

    trie.emplace_back(ALPHA, -1);

    vector<int> visits(1, 0);

    vector<int> pv(1, -1);

    while(n--) {

      string s;
```

```
      cin >> s;
      int t = 0;
      for (char c : string(s.rbegin(), s.rend())) {
        int d = (int) (c - 'A');
        if (trie[t][d] == -1) {
          trie[t][d] = (int) trie.size();
          trie.emplace_back(ALPHA, -1);
          visits.push_back(0);
          pv.push_back(t);
        }
        t=trie[t][d];
        visits[t]++;
      }
    }
    int ans = 0;
    for (int i = (int) trie.size() - 1; i >= 0; i--) {
      if (visits[i] < 2) {
        continue;
      }
      ans++;
      int v = i;
      while (v != -1) {
        visits[v] -= 2;
        v = pv[v];
      }
    }
    cout << 2 * ans << '\n';
  }
  return 0;
}
```

question

Question description Selvam won the man of the match award in the recently concluded local tournament final. So the friends of Selvam have asked him to take them to cinemas as a trat for winning man of the match.But Selvam is short of money to take them to cinemas so to postpone the cinema plan he tred to engage them with the programming challenge. The task was, Given an array of positive integers, write a program to find minimum number of merge operations required to make the array palindrome. If it is not a palindrome it will make merge operations and prints the number of merge operations. In each merge operation it will merge two adjacent elements. Here, merging two elements means replacing them with their sum. A palindrome is a word, phrase, or sequence that reads the same backwards as forwards. Function Description Create two variables i,j. i will point to the start of the array and j to the end.
Till i less then or equal to j a. If arr[i] = arr[j], then there is no need to merge the elements. Increment i and decrement j b. If arr[i] > arr[j], then do merge operation at index j ie, arr[j-1] = arr[j-1] + arr[j], decrement j and increment the no of merge operations count by 1 c. If arr[i] < arr[j], then do merge operation at index i ie, arr[i+1] = arr[i+1] + arr[i], increment i and increment the no of merge operations count by 1 Constraints 0<n<100 0<arr<100 Input Format First line indicates the number of elements n Second line indicates the array elements. Output Format Single line indicates the results Example: Input :
arr[] = {1, 7, 9, 1} Output :
Number of merge operations are 1. Merging 7, 9 makes the array palindrome

answer

```
#include <iostream>

using namespace std;

#define f(i,a,n) for(int i=a;i<n;i++)

int minOperatins(int arr[], int n);

int main()

{

   int n,count=0;

   cin>>n;

   int arr[n];

   f(i,0,n)

   cin>>arr[i];

   f(i,0,n/2){
```

```
        if(arr[i]==arr[n-i-1])
        count++;
    }
    if(count==n/2)
    cout<<"array is already a palindrome"<<endl;
    cout<<"Minimum no of merge operations took is "<<minOperatins(arr,n);
            return 0;
}

int minOperatins(int arr[],int n)
{
            int ans = 0;
            for (int i=0,j=n-1; i<=j;)
            {
                        if(arr[i]==arr[j])
                        {
                                    i++;
                                    j--;
                        }
                        else if (arr[i] > arr[j])
                        {
                                    j--;
                                    arr[j] += arr[j+1] ;
                                    ans++;
                        }
                        else
                        {
                                    i++;
                                    arr[i] += arr[i-1];
                                    ans++;
                        }
```

```
        }
        return ans;
}
```

question

Question description Janu and Ram are close friends who task a lot about life. They go though a lot of inspiring "Quotes of Life". One fine day they had a small game.According to the game Ram will Read one of the Quote about life from the book and Jannu have to think a word about life in her mind without disclosing it to Ram. all rounds completed, every round they are getting equal marks. So they decided to make some technical test on algorithms to prove themself. they have to open a book randomly and then they have complete the same task. likewise janu opened the book and got the task is, Given a binary array and number of zeros to be flipped, write a program to find the zeros that needs to be flipped so that the number of consecutive 1’s is maximised can you help janu? Function Description <figure class="image"><img
src="
AAAAAAIQAABtbnRyUkdCIFhZWiAAAAAAAAAAAAAAAABhY3NwAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAQAA9tYAAQAAAADTLQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAlkZXNjAAAA8AAAAHRyWFlaAAABZAAAABRnWFlaAAABeAAAAB
RiWFlaAAABjAAAABRyVFJDAAABoAAAAChnVFJDAAABoAAAAChiVFJDAAABoAAAACh3dHB0AAAByAA
AABRjcHJ0AAAB3AAAADxtbHVjAAAAAAAAAAEAAAAMZW5VUwAAAFgAAAAcAHMAUgBHAEIAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAFhZWiAAAAAAAABvogAAOPUAAAOQWFlaIAAAAAAAAG
KZAAC3hQAAGNpYWVogAAAAAAAAJKAAAA+EAAC2z3BhcmEAAAAAAAQAAAACZmYAAPKnAAANWQ
AAE9AAAApbAAAAAAAAAABYWVogAAAAAAAA9tYAAQAAAADTLW1sdWMAAAAAAAAAAQAAAAxlbl
VTAAAAIAAAABwARwBvAG8AZwBsAGUAIABJAG4AYwAuACAAMgAwADEANv/bAEMAFxARFBEOFxQS
FBoYFxsiOSUiHx8iRjI1KTlSSFdVUUhQTltmg29bYXxiTlBym3N8h4uSlJJYbaCsn46qg4+Sjf/bAEMBGBoaIh
4iQyUlQ41eUF6NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2NjY2Njf/AABEI
A2oFXgMBIgACEQEDEQH/xAAbAAEAAwEBAQEAAAAAAAAAAAABAUGAwIBB//EAEwQAAIBAwEEBQ
oDBQYCCQUBAAABAgMEEQUGEiExE0FRcZEUFSIyMzRSU2FygbHBIzVCc6EWNoKS0eFUYiRERnSDk8Lw
8SZjZKOyQ//EABkBAQEBAQEBAAAAAAAAAAAAAABBAIDBf/EACYRAQACAQIHAAMBAQEAAAAAAAA
BAhEDUQQSExQhMTIzQWEiI4H/2gAMAwEAAhEDEQA/ANYAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAB4dWmnh1Iprqyez831GVaet3FOFSScq7ivS+oH6L01L5kP8yPqq05PEZxb7EzF/2Y1b5sf/ADCZpOga
jaalSr16idOD4rfyBYahtNaWN47eUJzcfWlHki3o1Y1qUalN5jJZTM3q+n6NU1KU7m76Gpwc4KS4mgU6F
nZqW8oUYR4NvhgDuCHR1WxrwnKnc03GCzJ55HKOvabKpuK7p5+r4AWIPkZKUVKLTT5NHitXpW9N1K
9SMIrrk8AdCDV1iwo13QqXCVRPd3cPmeKeu6bUqbkbqGerLMbqrT2jqNcV0y/MD9DTyk1yYPNP2Ue5
CrFypTiubi0gPnTU/mQ/zIdNS+ZD/MjGS2Y1ZybVSOG/mEW+0XUNPt3WuK0VFf8A3OLA3vTU/mQ8T2f
nOjULq91GlGlKbUJKUm28JJn6K2oxy3hLrA+grquu6bSnuSuqbf0ZLtrqhdw37erGpHti8gdgcLq8t7OCnc1
Y04t4TZyWq2Lt+n8ppqnnG82BMByt7ildUY1aMlKEuTRxutTs7OW7cXEIS+FviBLBCttXsbqe5RuYSm+Uc
8SaABEutUsrSW7XuIRl8OeJzo63p1aSjC6p7z5JsCeAnlZR5q1IUabnVkoQXNvkgPQMPqWvXkdXqRtrx+T
```
```

76xu4awa+2v7W5ahRuKdSeMtReWBJB5qVYUYOdWahFc23gr3r+mKe75VDvzwAsgc6NelcU1OjUjOL5
OLydAPkpKEHKTwkstkO31exuqyo0K6nUfVhne89zr/y5fkYXZqUYa7GUmkkpZbA/QAVz17TFU3Hd089u
eBOpVYVqanSmpwfJpgewfJSUYtyaSXNsr6mvabTnuu6g39GBYg5W9zRuqe/QqxqR7YvJ1AHjpqS/wD9If
5kez8zar3OoOjTqS3pzaXpMD9J6al8yH+ZH2NSEniM4t9iZi/7L6t86P8A5hYaHol/Yaiq1xUUqai1jeyBcT1
mwp3DoTuEqqluuOHzPWp6lR0y26atlpvCiubMPf8A956v/ef/AFGy1y3s7iw3b6p0UE+E84wwPuk6zQ1
aE3SjKEoc4yLEptnrTT7alVdhW6ZtrflnJMqatY0q/QVLiEamcbrfWBNBAr61p9vPcqXMFLrSfIkWt7b3kd6
3qxqLr3XyA7g43V3Qs6aqXFSNOLeMtnFatYuh0/lNPo843s9YEwEChrWn3FRQp3MN58k3zJ4AHG5uqF
pDfuKsace2TwRKeu6bVnuRuqefq8AWICakk08pnOvXpW9N1K9SNOC65PAHQg1dYsKVw6FS4SqJ43cP
meIa7ptSpuRuqeXyyzG6s09o6jT4dMvzA/Q08pNcmDzT9lDuR6AAi3Wo2lm8XFeFN9jfE4Utd02rLdjdU0
/q8AWIPkZKUU4tNPk0RbrVLOzlu17iEJfDniBLBCttXsbqahRuYSm/4c8SaAAM1tPrE7aNOFjdqNRSamo4
bQGlOF3e29lTU7moqcW8JtFPoGswqafm/vIdNvP1mk8HPbT920f5i/JgXlpeW97TdS2qKcU8NpHcy+yt3
b2mk1Z3FWNOPSfxPHUXNDWtPuKip0rmDk+Sb5gTwCJc6nZWjxXuKcJfC3xAlgr6Ot6dXmoQuobz5Jvm
Tp1IU6bqTkowSy5PkB6BidX167hqs42l4+gysbuGjV2eoW1xCnCFxTqVXFZSfECWAc69elbUpVa01CEebY
HQEOlq1jVpzqQuabhD1nnkcoa7ptSpuRuqefqwLEHxNSWU8rtIc9XsKdd0Z3MFUTw4t9YE0Ahed7Dp+g
8ph0m9u7ueOQJoOF1d0LOmqlxUjTi3jLZyWrWDt3X8pp9HnG9nrAmAg2+sWF1UVOjcwc3yjnizrd6ha
WWPKK8IN8k3xYEk8VasKNKVSpLdhFZb7CNa6rZXkt23uISl8OeJ81n90XP2AfbTVLO9qOnbVlOSWcJM
mGF2QqQpajVnUkoxVN5bfI1K17TXU3PKqee3PACxB5hOM4qUJKUXyaFSpCnBzqSUYrm2B6BWvX9M
U913cO/PAn0a9K4pqpRnGcHycXkD2CNd39tZbvlNaNPe5bz5nieq2MKEa0rmmoS9V55gTAeKNWFelG
pTe9CSymewAONzd0LSG/cVY049sngiU9e02pPdV1BP6sCxPFWrTo03OrOMIrm28HpTi4b6accZyY7a/U
YV5UqFvWjOCWZbr6/qBoI7QaZKoqaulvN49Vlkmmk1yZi9H0jTK1GhVrXn/SJNS6NSXhg2i4LC5AD42or
LaS7WcL+8p2FpO4q+rFcu1mFr3+pa5d7lJz4vhCHBJAb1XNBvCrU2+zeR1MM9l9UjDfUo73PCnxJmzdTV
qd7K3rRnKjDhPpP4fxA1oPNSpClBzqSUYrm31FfLX9MjPdd1DvT4AWQOdC4o3NNVKFSNSL64vJ4ury3s
4KdzVjTi3hNsDuCH51sfJ+n8pp9HnG9nrPFDWdPuaihSuYOT5JvmBPAIl1qdnZvduLiEJfC3xAlgr6WuabW
luxuqab6m8E9NNZTymB9BDutWsbSe5XuIRl8OeJ9tdTs7yW7b3EJy+FPiBLAI11qNpZ+8V4Qb5JviBJBAt
9a0+5moUrmDk+Sb5kutWp29KVWrJRhHm2B0BFtNRtL2bjbV4VHFZaTO1xcUrajKrWmoQjzkwOgIttqV
pdqboV4TUFmWHyMvtBrlzT1HFhefst1ephrIGyKrVtettKqxpVIzqVJLOI9SOunanb17ahGVzTnXlBZWeLe
CBtBZaZcV6c7246CrjCeUsoC3sL2lf2kLijndl1PmiQVlrX07S9NoqFeMaEvVlJ+syba3dC8pupb1I1Ip4bT6w
OwPFatTt6MqtWShCPNvqONpqFreuStq0ajjz3XyAkgj3V/a2aTua8KeeSb5nC31nT7mahSuYOT5JvmBPA
AHl1aaeHUin9WI1ISeIzi32JmP1nRNQ6e5vI1UqSzPG/1Ij7JVJy1pKU5NdG+DfcBr7vVLOyqRhc11CUuSw
2SYTjUgpwkpRksprrMVthauGqRrb+900ViPZjgarSKUrbSbenUfGMOLfiBNBXVdd02lLdldQbXPDyd7XUr
O8eLe4hOXYnxAlHG8uqdlazuKzxCC4nYjajSt61jVhdvdotek88gIOlbQ22p3DoU4ThNLK3utFuZ7QLHSqF
3Odnc9PVx1tcEXdxd29pDfuKsace2TwB2BWLaDTHLHlUF9WywpVqdemqlKcZwfJpgewAAAAAAAAAAA
AAAAD821Capa/XqPioXDl4SP0k/N76KntDWhJZjK5aa+m8BoltlbY93qf0LHSNcparOcadOUNxZ9I+rZ3Ss
L/oq/zP8A1JNnplpYSlK1oqm5c+LYGK2r/f8AX7o//wAo1Os/3Zqfyl+Rltq/3/X7o/8A8o1Os/3Zqfyl+QGN
0jT6up3Xk9Oe5HG9N9iLHWdmvNtn5RSrupGLxJNYO2xPvtx/L/VF7tN+464FdsbeTqW1a3qSbVPDjnqRS
aze19W1Z0It7qqdHTj+OMlhsWs1bldsUVNRy03XpTqR9nW3u9Zz+QF7HY2Hk+Xcy6bGfV4ZM10NS31K
NGr68Kqi/E/QFrWnu26byqnjGd3eWfAwlzdK81p3EViM6ya7sgfpFP2Ue5Ho80/ZR7kfW1GLbeEuLA8V
69O3oyq1ZKMIrLbMFqV7ca9qUadGLcM4pw7Pqd9pNbd/WdvQl+wg+OP4mWOzq03TqHS1rqj5RNcfS
XorsAutH0unpdoqccOo+M5drKDa/VKirKxoycY4zPHX9DQrWNPk0ld0sv8A5kZLa63lT1RV+dOrFNPqAk
WGz9hUtI1Lq9iqk1nCkvRK91Kmgav/ANHrqpTTXGL4SiTdO0vRby1jOd46VTHpRlJLDO1HRdEr3nk1K9n
OpjPBpp/iBI2vmqul29RcpPJSaNo1fV84qblGn1vt+hd7XUo0dKt6Uc4jLCO2xv7rn94Hq+m9ntAjRpT36n
qxk1jn1lBoui1daqVK1as4wT4y5tsv9r7adbS1Ugm+illpdhX7Jarb29GpbXE1Tbe9GUnhARNb2flpVKNzQ
qudNPDysOLLTSNXr3Gg3WW5V7ePCXW11HzanVrapp7taFWNWc2m3F5SSZz2Pt+is7q5r8KU8LjyaWc
/mBR6XRtb28l5xuXTTWd5/wATL6OzGn1pwlbXmUnxjlPKPE9I0TUG6treKi3x3VJfkyk1C2803kY2t2qjxn
epy5d4H6NCKhBRisJLCOGoWivrOpbubgprG8lnB50utO40y3rVfXnBNkoD8yv7JWepztFNyUZbu81g2Oj
7PQ0y5VxG4lUbjjdccGY19bm0NZy4LfT/AANza31rcRpxpV6c5SjlRUk2BjNd1CvqeqO1pt9HGW5CK632I
nDYyDt/TuX0rXVHhkpLqM9L19zqRf7Orvr6rmbaGtafK26byqmljOHLD8AMbpt5caJq3Qzk9xT3akeprtP
0CL3oprr4n51cSer67KVGL/azSXdyyfolNbtOK7EkBzvPc6/8uX5H5taWtS8vo29F4lOWM9iP0m89zr/y5fk

fnukXkLDV4V6nqJtS+iAva+x0Y20pUrlyqpZw48GV+zWoVbLVI2lST6OpLcceyXI1VfW7ClbOsrqlLhwipJt/
gY3R6cr3aGnOEXjpelf0WcgWW1up1ZXfkNKTjTivSw+bPtns7p8rWMrm9iqslnCkvRIO1VvKjrEqkk9yqt5
Ml2OlaJd20ajvXTnj0oykk0wINCvU0LWN2lWVSmms7r4SR+gRkpRUo8U+JkbbRdFubt29C8qTqRWeDW
H3M1tOCp04wXKKwB6PzOlXVtqyryTahUbaR+mH5ra0oVtahTqLehKq012gaT+2Vt/w9T+haaRrFPVlUd
OnKG5jO8ef7O6X/wAKv8z/ANSXZ6fa2CkrWl0e9z4t5Awl/wD3nq/96/8AUafa/wDcr++P5mYv/wC89X/
vX/qNPtf+5X98fzAibEe7XX3R/Uodez58r7vPe4F9sR7tdfdH9Si1yW5r1aaWd2aYFza7IRq2salxcSjVms4S
ykUz8p0DV3FT4wfHHKUTZWWt2FazhUdzTg1HjGUkmjHa5dx1TWHK39KLxCPDmBoNq6qraHRqrlNqR
RaFo1TVpS3qjp0afN8+LLraWk6OzttTfOO6mRNktUt7VVbe4mqe+04ylwQHHXNnPNturmhVdSCeJZWG
i02Z1eVXT61O4lvO3WVJ9a/9obT6vay06VtQqwqzqNeq8pLJA2Xsalayvp4ajUhuRfa+IFcqlXXtW/6RWV
ODfOT4RXYWd9s9YU7SU7W9i6sFnDkvSKK0pUI3yo3+/Tgnuya5xZoJaLoUaXSPUPRxnhNAfdkdTqOpOy
rScoqO9Bvqx1FZrN7X1XV3Qg3uqfRwjnhzxk0Gh6Rp6avLKvOplOPHqMzVctM1+UqsX+zrb2O1Zz+QF5
HY2HkybuX02M+rwyZp0alvqUaNX14VFF/gzfrWtPdt03lVPdxnG8s+Bhbq6V5rTuIrEZ1U13ZA/SKfsodyP
NxKULepKHrRi2vA9U/ZQ7kfZNKLcuSXED83tOiv9SzqVw6cZtuUnx49hff2a024ivJL30u/OT1W03Q9Tqzq
UbpUJ7zylJLL7mUmq6fDSqtN217Gq5fBLjHvA1et3a0bRowt1iTxTg+zhzM1oui1NaqVK1aq4wTw5c22T
NR8o1HZS2uZ5lOlP0vquWT3slqtvb0alrcVI025b0ZSeE+H+wETW9n5aVTjcW9Vzp5w88HFl9stqdS/spQr
veqUnjPaiJtTq9rUsPJaFWNWc3l7rykhsVbThb1q8k1GbxH6gacw21GlRsq7ulVc3cVG3FrGM8TcmX23i/J
reXVv4/owK/RNnIalZq4lcyptSxuqOS02zWNMorsqL8mfNlb61o6VKFWvTpyjNtqUkj7to09Motcc1F+TA
otD0Ser77lVdOlB4bxniNc0SWjzpyhV6SnPk8YaaL/Yv92Vf5n6HPbb3Oh9zAsNmrud5o9N1Xmccxb7cFZ
W2XoO9q1Li8jCnKWYxzx/qddl6jpbP1qkVlw3pL8MmfsYPWdW3L25lBTy8t/0QHbW9Ht7ClGta3Uaqcs
OOVlFzs9Unquh17StUa3cwUubw0VevaPYabbRdGvKVZyxutrkWOxHu9x9/wCgGc1OxVhqMrVTc1Fr0
msGs0bZ2FhcU7tXMpvd9VxxzRntpVu69UcuHJmzsb+1rUKEKdxTlOUViKks8uwCaVO0/wC4rjuX5lsVO0
37juO5fmBjtG0yrqtw6MJ7lOKzN9hN1vZx6ZbK4pVnUhnEk1hol7Ee3uftX5lttX+5Kn3ICLsdeVK9lUo1JO
XRP0c9hndS/vFV/moudiOVz+BT62pW+0FaUlympL6oD9DXJH52v7zr/vP/AKjYraDTvJOm8oh6udzPpeBi
rar0+vUqvJTuFLxkBp9sv3TT/mL8ih0PRZ6upuVV06NPhnGeJfbZfuqn/MX5HnYr931v5n6AZ/W9Ino9xB
RqOcJLMZYw8kzStDra3Cd3dXMkm8Jvi2TNt/VtO+X6Fnsn+46f3MDIalY1tF1FQjUeViUJrgbO6ru52ZnWf
OdHLM9tr+86P8r9WXUP7nL+QBj9LsKuo3kbek91S9aXYi+vtkVRtJVLevKdSCzutYyVWzl/S0/U1OtwpzW
632Gv1DXLG3s5zhcU6kmvRjGSbbAz+yOpVKd55FUk3Tqeqn1M5bS6lVvNRdnTlu0oNRxnCb7Wcdlredb
WqdSKe7SzKT/DH6nDXbaVtrNZVU92Ut5PtTAuaezenO2XSX8ema5qSwmV2k3lXSNZVv0inSlPclh5TzyZ
OttH0O4oKor9xyuKlNJo62GjaPc3TjbXVSc6Uk8ZXEDztvx8ka/5v0K7R9Br6tS6WdXo6MfRT5t9xYbbLCtF
2Z/QttlVjQ6X3MCys7fyS0p0FLe3FjOOZ6uayoW9SrLlCLZ0I+o0XcafXpR5yg0gMHBXW0Wq7sp+s88eUY
lve7IKlayqW9eUqkVndaxkqtAvYaZqublOMXmEsr1Wa++1uxt7OdSNxTqSa9GMZJtgZ3ZbU6iuHp9Wbd
Oqmo5/heCDtBpcdLu4xjVdTpFv8VjHFnTZe2nca1TqJejSzKT/AAx+pM229+t/5f6sDtoezkKlK2v/ACmSbx
Lc3eHia0rdn/3Ha/YiyAzO21SUbS2pr1Zyk3+GD1sZQhGwqVsLfnLDfYiVtTYTvdM3qUXKpSe8kutdZn9m
9bhps50LnPQzec/CwN0CBLW9OjT33d08dilx8CLp+0lrf3sreKcG/Ub/AIgM9tJqVW+1F2lJvo4S3VFdcv8A
5J9HY2ErZOrctVWs4UeCKbVaU9P1+c5x5VelX1Wcm0oa3p9W2VXyqnHhxjKSTX4AY21r3Wgau6TlwjLd
ms8JLtL3bKSnplvNcpSyvBGf1Kv521xyt4tqpJRjw5rtL7a6Dp6RawfOLx/RAU2h6JU1eMnKq6dGDxnGeJ8
1zRZaRUpuNV1Kc+UsYaZoNjP3VP72cdt/dLb+Y/yAmaTf1q+zbr8ZVaacc9uDI6fChfX7eo3Lpxlxcnxyz VbI
uK0KTnjdVSWc9yIdTS9D1KTq212qMm/V3kv6MD7/AGZ064Sdre8c8s5yT9ob56VpUaVv6M5+hH6LtM
pqdktJuIeTXiqN8cwlxXgWusqvfbOWd5NOUoes+/rAiaLpNDUYTuL66jBZ4Jy4v6njV9Op6TWp17G6U1n
+GXGLPOjWemXkZQvbiVGonw4pJr8SwraRoNGcISv5NzePRknjvAsY67L+zXlr9v6n+Llkz2k6ZW127qVK1
VqK4zm+LZearosLbZ6dK0lKooS6Tj1lfsnqdCznVoXE1TU3lSfBAdb7ZGpS3JWNXpHnDUuGPqWd9b17bZ
epSuavS1Ixxk7320NhZKOKsazk+KpyTwu0ajXp6ls/Xq2st6Lg3y7AKLYr32v9iLzaj9w3H+H80ZnZa/o2OoS
8okoQnHG8+plttPrFpU092tvVjVnUazuvKS5gRNjqXTK+pt434JZ8Sp1vTlpl90CqOot1Sy1guth/a3f2x/Uh
7Yxa1dPqcEBaaLs5CjK2vvKZN7qlubvDiiHtr73Q+1l/pN/az0+1hGvT39xR3d5Zz3FBtr73Q+1gTadkr7ZCE
EvTjHej3or9kL10L6dpUeI1OWepo0WzvHRLf7TJ6xQnpGu9JS4LeVSAFttnfuFKnZwfGb3p47DppUPMuz
dS7nH9rUW9x/oUdNz17aCMmmoylnD6oo1uvWjr6HWo0V6sU0l9AMjp9s9av6lS9uVCK4ylJ8X9EStX0
S0tbV17K7jNwxmDksvuK3SqNlXuXTv6kqSfKS5Z+pdV9H0K3p9JPUG12Rmm3+AFhsnqdS8tZ0Kzcp0cYk
+tGgKnQ9LtLKMrizqyqRrRXFstgIer/ui7/ky/Ix+yH77X8uX6Gw1f90Xf8mX5GP2Q/fa/ly/QDUX+iU7/AFCl

dVa0sUsYhjgQdsa9Wjp1KnSbjCpJqWOzsNEQdXlYqz3dRx0UnhZ7QMjo2n6ZeUM3d04Vc+pywX2l7P2t
nfK6oXCqxiuEexldU2c0yrFzttRik+pyTSKzSLitY63To0au/B1VB4fCSzgD9CIGufue5+wnkDXP3Pc/YBl9jP3l
V+wh61Wlc69Vjczcacam59sck3Yv95VfsLXVLXRtQvZ061eNG5jwlJPGfHgBDp7P6PcUv2F96TXPP6M0G
k2ENOsY0IS3uty7WY/VtFoWFv09C+hUafCOVvPwLzY+7r3FnVhWk5Km0ot/UDQgAAAAAAAAAAAAAA
AGcq7KdLqUrvyvG9V6Td6P65xnJowAXBAADP6rsx5yv53XlXR76Xo7mcYWO0tLyx8q0yVn0m7vQ3d7G
f6EwAU2iaD5orVKnlHS78d3G5jHHvJ2p2XnCxnbb/R7/wDFjOCWAKfRNC80TqS8o6XfWPVxj+p11bQ7b
VEpT9CquU1+pZgDJ09jMVP2l5mHYoc/6naWx9NXUatG6cIRaai4Z5fXJpgB8it2KXYsHmrDpaM4ZxvRay
ewBk/7Ff8A53/6/wDcf2K//O//AFf7msAGVhsZuVIy8uzhp+z/ANzQXWn0L20VvcR3opYT619SUAMpW2
M9PNG7xHscOX9Sbpey9CxrxrVarrVIvMeGEmXwArda0nztbwpdN0W6853cnrRtL81Wro9L0uZZzu4LA
AfJxjODjJJxaw0zOX2yFCtUc7Ws6Of4XHKNIAMxabHUoVFK6uHUS/hjHGS/qWNGpYu0Udyk47uI8MIkA
DK1djFvZoXbiuxx/XJ1s9kKNOqp3Vd1sfwpYNKAPkIqEVGKxFcEkfQAKjWNn6GqSVTfdKsljeSzlEPS9l5WF
7C4d3vbnJKOM/1NGAK/VNHttUglWjia9WceaKJ7Fy3uF6t37P8Ac1oArNK0O10xb0Fv1Xzm/wBCzAA8V
odLRnTzjfi457MozdHY6lGrKVa6dSLT4KGMPtzk04Ayb2L/AGnC89D7OP5l7pekW2l08UU5TfrTfNk8ARb
/AE+31Gh0VxDK6muaM7U2M9PNK7xF9Thy/qawAUelbM0NPrRrzqOrVjyeMJF4AAM5Q2U6HUI3Xled
2e9u9H/uaMAAABnbjZbptTleeV43qvSbvR/XOM5LPV9N86WXk/S9HxT3t3JPAFXomj+aKdWHTdL0jTzu
4xgx+tRUtoKsXydRJn6IUF3sxC61CV07iUW5KW7gCHV2OhUanQuXCMknuyjnBY6Ts3bafUV2qoyqOnh5ykedG0qOk0J0o1
HU3nnLQFNb7GRVVOvdOUF/DGOM/jk0tvb0rWhGjRiowiiuCR1AFPquztrqU3VTKs+corn3oo7Fy3sSv
PR+kP8Ac1oAhaZpdDS6Dp0Mty4yk+s46todtqiUp+hVXBTRZgDJw2MxUXSXmYJ8lDn/AFO09j6flUatG6
cIRaai4Z5fXJpgB8it2KXYsHyrTVWlKnLOJLDwegBlq+xsHNyt7uUF2Sjn+uT1+uT1b7HU41VHH9TTg
DnC3pQt1QjBKko7u71YM/e7IUK1Rzt7iVJP4yjkXZnu+FtQuzPdlXmWxByQvYGVHzPnvUoKpF2+GR
UewAI2oWNHULSVvXXqvk1zT7yq03ZSxsbhVp1JXEovMVPkjvdUWKgp47u8Ov+y/bxYsOwyP3TDvaVUw
UewAI2oWNHULWVCuvRfJrmmSQBk3sW9/wBG8xH7P9y11HRHf6dQtHVtSHAMZHqtDbmD9vEfVne07h3
NtOj0vS70t7O7T4AhtOujr53c5LMAV+kaVn83n5aWWtEe7q7KYjqPt3eTjdFmYgBzX9cuzZdTDUKS2cDf
MzHZCDoTdS5RK1lAeBB1zUQ7k3PSZ3EXSQM+N6WUNe6s+QgVnR7sp3r3ZlUXQQ4JGzGCnoNDvLqu3
ed7AbvPTNg/cg5Nfh2IsncUmBfR3QpNx3cU70Nja/JmhDsSjemMa5E0P1LS7gWumFfZnL8Q9JMf8h
be48pKOM/1NKABE1Oy84WNW23tze+JOPPEkMPNhTXMp/G6CkIoJ9Dlr5tNo7vvHF0v0WNmRpxbI8nUjC
urFujVfOSWU/wKynsW9/8AaXmYrshz/qawARdP06FwDbBQYUKbyT7WiF2AHg2hkd3ARqjPl8q6eQ0a2SR+bG
1dMBbgkeMl8eAAHVpBfQbB4g8fUSrOEPAQlG7BUUqtVr9hu3dUgHz0PN6BUM6u7+ZEzBBDqpoLHyOLP
kW7Asdb0DGfA8kzWcVaz0Sl8XnTbgRCs0kheV1Df0jFDqnLXtrULx8Wsg38tvNeZUqbJWPOCyO2cCPOAI
GWeEDdDQhkGfQjR77bPfQJ5sQoamkdaAoYu1Dfi0z5+hyqfYCtZdOEEhl72Uz3FP1RqUNNLfaLJ20kBbfLmpkm
fVkmZCGzSBHtDsRx1OZ3rF2/Tdx6RB3S2B0F6Hz+Uzn/8AbnQa3UWfcOUmzYlQHTs3ZZ7DuqL4mN0wvRNtx
vRKvr5yI4k2ZaynAjTQCiXhAQv4kNZ0ERPp5yDPoUMLF/DsFZnlASHnuXpnWwOnB/ocsRaIDzs2twQ4F
CAAEjkJqg5BQ1NCFYYWmoK9l46N2dUvc7BnYZnxiq5L8wT/3N1NdOHmRvCwNvbA65wqj3uGtT0y40mq
V/g7QzY7ThYUOF3cHPugaZ7ZjsaCu0PW/u4vczYt3Wg8LwbTeqoi6cVDY1Cq6E7dqd1jOaZpd3yC2cfS8
ePIA1B+JaFrL2h3T7D/6kp9rtX3eZZAsUlV1TP2XdNL2wCtY8IOVXjEv2VH4PRFyNu4xw7RoNOvcV2
8KmJv0Yu9ApgMGZzPe1yqr4Y7+fuUPmrDvd43zUHu5VFrnPeHv4kNyb9SyBKfUGZI7oM53eMXNNG2l
hG9s4FLW0sZzGMaDZJuqOlyVy4vmO7S3GUTRBHHB2P9xOpsa7y9wZ+9Ud+cd1xK5v6G51VW9Jqt
G7mQMmMz4mZ0SFZN1fZBdh4oMvFGd/hFtn9xDa5p9ADAOu/X0LpSS0ox7SSsbCXqHOX5Te1Fb4qldeSK8
2ZOP2kA/hARSc1C36X/GvVUcH+wwnYzsaYH0+5bfV4qtd6S8NqfOSbi4kuhbZ2GAw3l3h2XNPqGYKIvlXDf
HvE5Mv4nD6L4K26mxtg0nfVm4ZB5+hmUbwQeeffEdtupc3Yjx+y2Ku8zZSl5PKUo4AIO4qlu8fjt9/aLU
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAACu1vVVpNrGr0fSSm92KzhAWIKnQtaWrwqZpdHOnjKTyuJbAAAAAAAFFrm0Xmu5jQp0ekljLbeEiz0
29jqFjTuIx3d/muwCUAAAAAAAAcL27p2NnUuavoQWcdr7EU9ZrXTvJ7w4W5/HfL5dPE6Oiu8Ei8h6QJ
2jMebTw1RYmqZqz7FEQoaC4AdmNvDuXWTu7ZVYfJNhAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
NAAAAAgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABW6hrlnptdUriUlJrPCLZyudpdPt4wbqOblFSxFZxkz
22X72h/LRM0fZi3utPhXu5Tc6iylF4wuoC6sNdsdQk40quJpZ3ZcDhc7T6db1HDpHUa4NwWV4mM1G0l
p+pVLeEn6Lwn9DS2WydrOwjKvOp0045ynwTAudP1a01JPyepmS5xfBki6u6FnS6S4qRhH6vmfn+nupp
+v04Rl6lXcf1WTb6xY0L+x3LiapxTUlPsAgz2t06MsRc5Lt3WT7DV7PUeFvVTkucXwZQ+RbNQg6crpOXXL
e4/kUNKqrDV1K1q78IT9GS60BvNR1W30zcdzvpT5NRbR70/UbfUaLqW08pPDT4NHq/sqV/ZzoVVlSXB9
j7TCQr3mz2oVacfWxjD5NdTA2F9r9jY3LoVZydRc1FZwfNWr2FXS4Vb+nJ0Z4a4PKM/s3pU9Runf3eZU4
vKz/FIttsVjSF96AkbPy06VvU82xain6Tknl+J9r7RWFC8dtUnJTUt1+i8ZK/Yn3Sv9yM9rcXPXbiMecqjS8Q
NbcbUadQqumpym1zcVwJlrq9ld28q9OvFQh629waKinsjaOyW9UqdO453s8E+4ylC3qVL1WsZNOUt1g
bSptXpsKm6pTku1RZaWV9b39Hpbaopx6+1FBebJ2lLT6kqU6nTQg5Zb4PH0KzZGvOlq7pZ9GcXlfVAXe0
EtIjcU1qMJOo1lOOeX4E6leWdno9O4pJxtkvRSXEzm2vv9D7GXumWtK92ct6FdNwlHil3sDl/azTPin/AJG
P7Wab8U/8jIer6Vo+mWkqk4Sc3whHe5sptC0eWqXW9OLjbweZPt+iA30akZUlUTxFrOX2FTc7T6db1H
BVHUa57qyvEg7XXsrW1o2dBuKmnvY7F1Fbo1vovkvSahcRdWX8Dfqgaiw1yx1CW5Rq4n8MuDLE/O9
WhZWt3CrpVxvRfHCfGLL/AFDWKj2WpV4yxVrYg2u3HECfebR6fZ1XTlUc5rmoLOPxK7V9bstR0WvCjU/
afDJYZWbOaJT1R1K1zKXRweMLm2SdodnaFlaeU2jklF4lFvIHvYj2tz9qNcZHYj2tz9qNcBWa9KwhZReow
cqblhYTyn+B42flp0rWfm2LUFLEnJPLf4kTbT900/5y/JnPYn3C4/m/ogJ9TaKwpXkrac5KcZbr9F4PFxtRp1
Co4KcqjXNxWV4mP1aLnrlxGPOVXCNNS2QtPJUqtSo6zXGSfBPuAttP1W01GLdtVzJc4vg0TT860uVSw16
nBS9WpuS+qP0UA3hZZhNc1qvqF27e2lJUU91KPOTNjqcpQ0y6lH1lSk14GI2YhGprlLpMPCclntAm22x9z
VpKdetGlJr1cZaOU9ntTsLyn5LJy3nwnB4x3m5AHO3jUjQgq0lKol6TXWzoAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAYfbL97Q/lo1ejfum2/lr8iPqegWup3CrV51YySxiDWPyLC2oRtbeFGDbjBYTfMDB7S
f3gq96N7b+70/tRV3uzdpfXkrmrUrKcuqLWPyLeEVCEYrklgD87rf3kf89fmXG2laslbUk2qUk20uTfAs5bM
2cr7yt1K2/v7+MrGfAsb2wt7+36G4hvR6n1oDL6RZaHU06NW7qR6XXjvqvU8Y/Aorl0HqUvJE1R31uJ9hrY
7H2KqZdWs49ja/0OtXZSwqVlUUqsMYxGLWPyAs768pWNPNpKvWaSiuC7X2GF6K82hv61WCy0s8eSXUjZ
apo9HVNxV6tWMY8oWaS/I7adptvpvvptv0NunhvLlLmwMrs1q0rG4djdZjFPb8LP8LLbbHjpC+9He2asr66dx
OVWnOXPcaSz28iRc6RRu9PhZ1qtWUIY3LK3vdK/3lotT/vLU/nr8za6XpVDSqc4UKWs4UJTkpvL32iJX2as7
i+d3OpWU3PfwmsZz7AW8PZR7j89sP7x0/5rP0NLEUuxYKelszZ0b2b1GWb1Lew2sfkBZ3nuVf+XL8j
Mfv+Hcze1YKrSlTlnEcqq5pVKzms8JNY/ICi219/ofY8027pWWzVCvWeIxh48WddU0K21ivOpcVK3DMIzbi
WtCpXnVi4rC3Gv8AQ91dIoVdKjp8p1fR9Mbjalvzp2z1jRrK2hQo3EYwivQa2z1wgfH+hw/s
dp/zbj/Mv9B/Y6w+dcf5l/oBG2vtnc2C21CaK4u2Onq7/ogi8aq/jy8PwNvTtqdb43qaju/7XU/UsZ7Pah4lOkP7cCwqyLI38ygGaVCkpqNJYoQbyMQHh/7Z2s9l7C1qKo1OrJct98
C5wsYxw7AMRsxrVHTukoXMnGnJ71X3kcSucyljOGS9otoLW6snaUcnU336UsHfpSfFqD4Hylspp
9OjOD6SUprG+2sru4AVuxHtbn7Ua4rtK0W10ai4W8W2/akzV3e+DfKVXTQ3bdsBqDTF3ecUa7i7W+e0VnEpvn4rT6Mp/oS7HZq
Op6bR1S3jjRrynGMZb2aWFEhOkSkrPc+W+0euJ1z2xq2UTQ4eXZv9Wp21gYtKt/BJdowpL7tE8ESk6Vbv83PiQQahWfnyMjpv0KHGbzPxhRKPTU
e9pdHcU1NfXqAqLXaywvXlF3xlI+N8Fn6FGhvq9w4XeBAqbdG9IDbf1n2o3PMB8WlC7nx3j7htH7P0z8nYCuqfgOs93J1PdU+S8dQqL3hWNv3P
ws6iqbsqs1yc3nAFpQrRr0IVYppTWUmsM6BLC4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAcK13Roy3ZS49gSZiPMu4InnGh2vwHnGh2vwDjqU3SwRPONDtfgPONDtYOpTdLBE840O1+A8
40O1+AOpTdLBE840O1+A840O1+AXqU3SwRPONDtfgSKdWNWO9B5QWL1n1L2AeKtWFKG9N4QWZ
w9giecbf4n4Dzjb9rDnqU3SwRPONv8THnGh8TCdSm6WCJ5xt+1jzjb9rB1KbpYInnG37WPONv2sHUpul
giecaHazpRu6NaW7CXHsCxqVn1LuAA7AR6l9QpT3ZS4rsPHnG37X4Bx1Kx+0sETzjb9r8B5xt+1+ATqU3S
wRPONv2vwHnG37X4A6lN0sETzjb9r8B5xt+1+AXqU3SwRPONv2vwHnG3+J+AOpTdLB5hOM4qUXlM+
Va0KMd6bwg6zGMvYInnG37X4Dzjb9r8A56lN0sETzjb9r8B5xodr8AdSm6WCJ5xodrHnGh2vwCdSm6W
CJ5xodr8B5xodr8AdSm6WCJ5xodr8D1C/oTkoqWG+0L1K7pIB4q1YUY703hB1M4ewRPONv8T8B5xt/if
gHPUpulgiecbf4n4Dzjb/E/AHUpulgiecbf4n4Dzjb/ABPwB1KbpYInnG3+J+A842/xPwB1KbpYInnG3+J+B
0o3dKtLdhLj2AjUrPiJdwCPdXkLbCeXJ9SDqZisZlIBXedY/Ax51j8DDz61N1iCu86x+BjzrH4GDrU3WIK7zrH
4GPOsfgZMnWpusQV3nWPwMedY/Axk61N1iCu86x+Bki2vIXDwsqS6mVY1aTOIlJAI91dwtkt7Lb6kHcz
FYzKQCu86x+Wx51j8DDz61N1iCu86x+BjzrH4GDrU3WIK7zrH4GPOsfgYydam6xBXedY/Ax51j8DJk61N
1iCu86x+Bki2vYXEt1JqXYyrGrSZxEpIBwubuFslvZbfUg7mYiMy7grvOsflsedY/LZMvPrU3WIK7zrH5bHn
WPy2MnWpusQV3nWPy2POsflsp1qbrEFd51j8tjzrH5bGTrU3WIK7zrD5bJlvcQuIb0OrmmHVdStvES6gE
O41CnRnuJOTXPAW1or5lMBXedo/LY86x+Ww461N1iCu86x+Wx51j8tjJ1qbrEFd51j8tjzrH5bJk61N1iCu
86x+Wx51j8tlydam6xBXedY/LZMt7iFxT3ofigtdStpxEuoBEuL+FCe5hya54Dq1orGZSwV3naPy2PO0flsZc
dam6xBXedY/LY86x+Wxk61N1iCu86x+Wx51j8tjJ1qbrEFd51j8tjzrH5bGTrU3WIK7zrH5bJdtcwuYb0OD
XNMOq6lbTiJdgCJcX9OjPcw5Nc8BbWisZlLBXedY/LY86x+Ww461N1iCu86x+Wx51j8tjJ1qbrEFd51j8tjzr
H5bGTrU3WIK7zrH5bHnWPy2TJ1qbrEES2voXEt3DjL6ksrutotGYAAHQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAADM9Xbdebb/iZoHyM9W9tP7mSWTifUPAAIxgAAAAAAABZaQ
3+0WeHArSx0f1qvchD10PuFoVmrN5guosyr1f14dxZbNf4lXAAj5wAAAAAAAgHW2bVzTw/4kcjpb+8U/
uRVr7hoVyPNR4pyf0PS5I81PZy7jp9WfTOybc2/qfD7L1n3nw5fJn2AAAAAAAAAAC30pt27y+TOGrN9LB
Z4YO2k+wl9xw1b20O4v6bLfhQAARjAAAAAAAAAAABoaDzQg32Ir9Wb36azwwyfbe7w7iv1b2lPuZW/V/
GrwAcsAACgACAACgdbVtXFNrtOR0t/eId4WvtoSl1P3t9xdFLqfvb7iy3cR8IgAIwAAAAAAAABK073uJFJ
Wne9xEO9P6hdlPqnvX+EuCn1T3r/CWWziPhCABGAAAAAAAAAJFh73Ajkiw97gId0+oXpTap71+BclNqf
vX4Fls4j4QwARgAAAAAAAACx0h+nUX0RXFjpHr1O5CHro/cLRmer+3qfc/wAzQsztf29T7n+ZZe/E+oeAA
RjAAAAAAAACx0l+lMrix0n1plh7aP3C0fIz9y83NTPxM0D5Geufean3MS9+J9Q5gAjEAAAAAAAAFjpHtJ9
xXFhpHtJ9wh66P3C1Zn7njc1PuNA+Rnrj3ip9zLLRxPqHMAEYgAAAAAAAHey97p95fFDZe90+8viw3cN8
yAArSAAAAAAAAAAAAAAAAAAAcnc0E8OvTT7HNGO2i1m4uL+dpbzcKcHu+jzkxDZXUalNTlWipPjhsDa
xlGSzFprtR9Mroel6pZ6qlXnNUEm21LKZI2n1eVtRhGyulGqpNTUWm0BogZ7Z7WY1bJvULyDq72FvtJ4N
Ammk08p8cgfQQbjWbC2nuVbmCkuaT4o62uoWt57tXhUa5pPigJIPFatTt6UqtWShCKy2+oi0tWsK0Jzh
c03GCzJ55ATQV0Ne02c9xXUM/V8CwjJSipRaafJoD6Acbq7oWdLpLipGnHtb5gdgcra6o3dJVLepGpB9aZ
16gPHS0/mQ/wAyPsZwn6soy7mYbU9F1C0p1bqpV/Zp54TJmxVSc7q53pSl6C5vPWBrwAAAAAAfHy
M9W9tP7maF8jPVvbT+5klk4n1DwACMYe40aklmMJNfRHgsra+pUqMYSTbQd0rFpxMoPQVfly8DmaG
TTpNrrRnnzYd6unFPQAA8QstH9ar3IrSy0f1qvchD10PyQsyr1f14dxaFXq/rw7iy2a/wrgAcvnA5vCBaWFr

GNPpqiy3xWeorulJvOEBW9aSyqcsdxzlGUHiUWu9FnU1OEZYhDeS6zsuhvqDeOP9UXD16VZ8VnypQeq
tN0qkoS5pnkjPMYDpb+8U/uRzOlv7xT+5Ba+2gXI+VPZy7j6uR8qezl3HT6s+mdl6z7z4fZes+8+HL5MgB3
tKHlFZRfqriwsRMziHOFKdT1It9yPU7etBZlTkl3FzOdK1pJtYXVg8ULyjcS3FwfY+suGno19TPlSAsdRtVBdL
TWO1FcR4XpNJxIAA4W2k+wl9xw1b20O476T7CX3HDVvbQ7i/prt+FAABGQPsYym8RTb7EfDvZ1o0a+/
POMYC1iJnEvHQVfly8DzKEoPE4uPei8t7mFwnuZ4dpB1b2sO4uHvfSiK80SrwARnAABoLb3eHcV+re1p9
zLC293h3Ffq3tafcyt+r+NXgAjAHqMJTzuRcsdiPJKsbiFvObnnig6rETOJcegq/Ll4HiUZQeJJp9jL6hWhXhvw
5ZxxKvUve33DD11NKK15olEAAeAdLf3iHeczpb+8Q7wtfcNCUup+9vuLopdT97fcWW7iPlEABGAPdKjUq
yxTi2fIRc5qK5tl5TpwtqHYorLYeulp8/tWebq+M4Xdkj1aU6Ut2pFplj50W/jc9HtySatOF1Q7crKZcPXpUt
H+ZUQPU4uEnF808HkjKErTve4kUlad73EO9P6hdlNqnvX+EuSm1T3r/AAlls4j4QwARgDrRtqtdN045S58
TkSrS88mjKO7vZeQ6pFZn/T55vuPg/qjhUhKnNxksNF7b1emoxnjGSnvfep94e2pp1rWJhwAAZwkWPvc
COSLH3uAh3T6helNqfvX4FyU2p+9fgWWziPhDABGAOlKjUrPFOLZ8pU3Vqxgutl2lTtaHDhGKEPbS0+fzP
pWPTq+7nC7skapTnTluzi0yxjqkd/EoYj25JNzRhcUHw44ymV30qWj/ABKjB9acW0+aPhGYLHSPXqdyK4
sdI9ep3IQ9dH7haMztf29T7n+ZomZ2v7ep9z/MstHE+oeAARiDvTs69SClCGU/qjgTrfUOipwp7mccM5D
ukVmf9OTsLhLLh/VEdrDNBJ5pN9sTPy9Z94l6aunFMYfAAHgFjpHrTK4sdI9aYh66P3C0fIz1z7zU+5mhfIz
1z7zU+5llo4n1DmACMQd6dnXqQUoRyn9TgTre/wChpRp7mcfUO6RWZ/05OwuEsuH9URnweDQ729R
z2oz8vXl3iXpq6cUxh8AAeAWGke0n3FeWGke0n3CHro/cLXqM9ce8VPuZoeoz1x7xU+5llo4n1DmACM
T7CLnNRist8iR5vuPg/qjjSn0VWM8Zwy1tb7yirubmOGeYe2nWlvEq2rbVaEU6kcJ/U4lrq3sYfcVQc6lIpb
EAADzd7L3un3l8UNl73T7y+LDbw3zIACtQAAAAAAAAAAAAAAAAAAMPtDo1zb387qhCU6U3vZjziz7bb
WX1ulC4pRqpcOPos0U9obGlfVLStPccHjefJny5eiXNJutO1axxakk/6AedL2itdRmqXGlVfKMuT7ih2p0m
NpUd4qrk6036O7yKmnGK1iEbNtw6ZdG/pngafbKMnptBtcpcQKvQ9n4anbeUSuJU3GWMKOS32n1Cp
p9jStqEmpzjhyXNJHHZK9tqGnVKdavCnJSziUkjntpQlUhb3UPSp43W1/T8wIej6LaXlt5Re3cYOT4Q3ln8S
NqVr5lvadWxulUXOLi+K+jOmj2Ok3tDF1cyo1lzTkkn4k2ekaFG4p0PLpSnN4W7JNAWd9d+XbJ1Ljk50+K
+pk9H06rql10FOW7FLem+xGu1Kzp2GzFe3pScoRi8Nmc2X1Glp9/Lp3uwqR3d7seQJup7KK1s5V7atKcq
ay4tYyj1shqdTp5WNWTlBx3oZ6voW+r61ZUdPqqnXp1Zzi1GMJJ8TO7I0J1dXdVL0acW2/qBuij2n0qvqV
Ck7Zpypt+i3zzj/QvCBq2rUNKpRnWTk5vEYrrAjbNaZW02znG4a36ks7qfIuCFpep0dUt3Vo5WHiUXzRNA
qdp/3FX/AAKTYj3u5+xfmXe0/wC4q/4FJsR73c/YvzA2IAAAAAAAPj5Gere2n9zNC+Rnq3tp/cySycT6h4A
BGMPq9Zd58Pq9Zd4WPa/Xu/8AhM++bNAvYf4TPvmyy08R+gAEZQstH9ar3IrSy0f1qvchD10fuFmVer+
vDuLQq9X9eHcWWzX+FcADl859XNF1dPo7GW71LBSLmi7uF01i93rjlHUNGj82UhP0qTVaUeprJALDSoP
fnPqSwSPbjRzzw56pFK5T7UQyXqc1K6wv4URBKav3IdLf3in9yOZ0t/eKf3IOK+2gXI+VPZy7mfVyPlT2cu4
6fVn0zsvWfefD7L1n3nw5fJkLPSY4hOXaysLPSZLdnD8RD20Ptz1WTdWMepIiUJOFeEl1MmarB9JGeOG
MES3g6leEUusv7W+eourmO9bTT7CgL66koWs39MFCJdcT7gABGZbaT7CX3HDVvbQ7jvpPsJfccNW9tD
uL+mu34UAAEZAAAWek8qh41b2sO49aTyqHnVvaw7i/prn8CvABGQAAGgtvd4dxX6t7Wn3MsLb3eHc
V+re1p9zK36v41eACMAAALfS/dX9xD1L3t9xM0v3V95D1L3t9yL+mvU/DCIACMgdLf3iHeczpb+8Q7wtf
bQlLqfvb7i6KXU/e33Flu4j5RAARgSLBZvIFjqLatJY7UVlrPo7mEnyyW95Sda2lGPF80WGvS86cxCiLrT3m0
jn6lNuy3t3HHlgvbWn0NtGL7MskOeHieZUXqxd1MdpwOt1PpLicl1s5B4X+pCVp3vcSKStO97iF0/qF2U2
qe9f4S5KbVPev8JZbOI+EMAEYAAAXdh7pAq733qp3lpYe6QKu996qd5ZatX8cOAAIyhIsfe4EckWPvcBD
un1C9KbU/evwLkptT96/AstnEfCGACMCXpqzdr6JkzVJNWqx1ywQLGoqd1Fvk+BZ39J1rZqPFriiten50pi
FIXtk3K0p57CjUZSkopPL6i+ox6G3jF/wriITh4xMypblYuai/wCZnI91pb9Wcu1tngjNb3IWOkevU7kVxY6R
69TuQh6aP3C0Zna/t6n3P8zRMztf29T7n+ZZe/E+oeAARjD7H1l3nw+x9Zd4WF//ANX/AMJQS9Z95f8A/
V/8JQS9Z95ZaeI9Q+AAjKFjpPrzK4sdJ9eYh66P3C0fIz1z7zU+5mhfIz1z7zU+5llo4n1DmACMQfVzR8Pq5
oEe1/H3dfaUEvXl3l/H3dfaUEvXl3llq4j1D4ACMoWGke0n3FeWGke0n3CHro/cLXqM9ce8VPuZoWZ649
4qfcyy0cT6hzABGIJml+8/gQyZpfvP4CHppfcJOrexh9xVFrq3sY/cVQl3r/YAA8Hey97p95fFDZe90+8viw2
8N8yAArUAAAAAAAAAAAAAAAAAAACh1HZW1vKsqtKpKjOTy+tFd/YyrvY8rW79n+5rwBT6Ts7babNVd5
1ay5Sa4LuRYX1lSv7aVCusxl2c0SABk5bFvpPQvMQ+sOP5mip2NPzfG0r/toKO63JcyUAMvdbG05zcra5d
OPwyjn+p7stkKNKqqlzXdXDzupYNKAKzaBKOhXKXJQwjJbPaXS1WpXpVW4uMMxkup5RuNQtFfWVS3c
nFTWMor9G0KOk1qlSNaVTfju4aAq4bF/tPTvMw+kOP5mjsLChp1uqNvHC62+bJIAFZrejw1ejCLqOnOm
24yxnmWYArtF0mGk20qcZ78pvMpYwWIAETVLLzhYztuk6Pf/ixnBC0PQfNFWrPyjpd+OMbmMf1LgAA
AAAAAAAfHyM9W9tP7maF8jPVvbT+5klk4n1DwACMYfV6y7z4AL/ej0GN5eqUD5sAZeupqc+AAB5BZa

P61XuRWllo/rVe5CHro/cLMq9X9eHcWhV6v68O4stmv8K4AHL5wWFheRhHoqrwupleCuqXmk5hcTsre
rLfTxnsZ9qV6NpR3KeM9SRTqTSwmz4MvbrRHzHl9nJzk5S5viz4AGeQ6W/vFP7kczpb+8U/uQWvtoFyPl
T2cu5n1cj5U9nLuZ0+rPpnZes+8+H2XrPvPhy+TIdbes6FVTX4o5ALE4nML2NWhdU8ZTT6mIU6FvmUVG
PayiTafA+uTlzbYy0df9zHlMv7tVmoU/VXX2kIAPC1ptOZAAHK20n2EvuOGre2h3HfSfYS+44at7aHcX9Nd
vwoAAIyAAAStKkkqmXg8aq06kMPPAgAZevU/xyAADyAABoLb3eHcV+re1p9zLC293h3Ffq/tafcyt+r+NX
gAjAAAC20ySVthtLiQ9Sad02uwigPW2pmkVAAHkHS394h3nM6W/vEO8LX20JS6n72+4uil1P3t9xZbuI+
UQAEYAs7TUIqKhW4Y5SKwB3S80nML3p7b1t6HeRLy/Tg6dHjnnIrQMvS2vMxiAAB4BK073uJFJWne9x
EO9P6hdlNqnvX+EuSm1T3r/CWWziPhDABGAAAFtZXNGnbRjOok11FddSU7mcovKb5nIB6W1JtWKgA
DzCRY+9wI5Isfe4CHdPqF6U2p+9fgXJTan71+BZbOI+EMAEYBPDyWlrqEXBQrPEl19pVgO6Xmk5he9Nb
R9LehntIV5fqpF06XLrZXgZelteZjEAADwCx0j16nciuLHSPXqdyEPXR+4WjM7X9vU+5/maJmdr+3qfc/zLL
34n1DwACMYfY8JLvPgAuvK6HQ46WOd0ppes+8+APS+pN8ZAAHmFjpPrzK4sdJ9eYh66P3C0fIz1z7zU+
5mhfIz1z7zU+5llo4n1DmACMQFzQAF1G6odCl0sc7uCml6z7z4A9L6k39gADzCw0j2k+4ryw0j2k+4Q9d
H7hasz1x7xU+5mhfIz1z7xU+5llo4n1DmACMQStPqQpXG9OSisc2RQHVbcs5WOpV6VWlFU5qTT6iuAB
e83nMgADl3sve6feXxQ2XvdPvL4sNvDfMgAK1AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAA+PkZ6t7af3M0T5GfuYShXmpLHEksvE+ocgARiAAAAAAAACy0f1qvcitLPSISSqSa4PCQ
h7aMf7hZFXq/rw7i0K3VoSe5JLgiy16/mkqwAEfOAAAAAAAADpb+8U/uRzO1rCU7mG6s4kmHVY8r5cj
5U9nLuPS5HyazCSXWjp9SfTOS9Z958PVSLjUkpLDyeTl8qYmJAAEAAAAAAAAAW2k+wl9xw1b20O4k6XC
UbduSxl8Dhq0Jb8JY4YxkrZaJ6KuABGMAAAAAAAAAPqTbSSy2DC/tvd4dxX6v7Wn3MsaEXGjBPmkQN
WhJuE8cFlNlfQ1Y/5q0AEfPAAAAAAAADpb+8Q7zmdrSEp3MFFZw+IdVjzC/KXU/e33F0VOp0J9N0ii3Frq
LLbxETNEAHrcn8L8B0c/hfgRhxLyD10c/hfgNyfwy8AYl5B63J/DLwG5P4ZeAMS8g9bk/hl4Dcn8MvAGJeS
Vp3vcSPuS+GXgTNNoTddVGmox7Q704nmhblNqnvX+EuSr1ShN1VUjFtYxwLLZrxM08K4Hrcn8MvAbk/
hl4EYMS8g9bk/hl4Dcn8MvAGJeQetyfwy8BuT+GXgDEvIPW5P4ZeA3J/DLwBiXkkWPvcDjuT+GXgS9Oo
TdwpuLUY9bEO9OJ5oXBTan71+BclXqdvN1VUinJNYeDqWzXiZp4VwPXRz+CXgOjn8EvA5YMS8g9dHP4
JeA6OfwS8AYl5B66OfwS8B0c/gl4AxLyD10c/gl4Do5/BLwBiXksdI9ep3IgdHU+CXgWml0J04ynNY3uSEP
bRrPPCezO1/b1Puf5miZRXVCpTrz9FtNtposvbiYmYhHB63J/DLwPm5P4ZeBGPEvgPu5P4ZeB93J/DLwBi
XkHrcn8MvAbk/hl4AxLyD1uT+CXgNyfwy8AmJeSx0n1pkDcn8MvAs9LozhGUprGeWRD20YnnWHUZ65
95qfczQlHe0KkLmb3W1J5TRZaOIiZrCMD1uT+CXgNyfwy8CMWJeQetyfwy8BuT+GXgDEvIPW5P4ZeA3J
/DLwBiXkHrcn8MvAbk/hl4AxLyWGke0n3EHcn8MvAstLozhvTksJ8FkQ9dGJ54WLM9c+8VPuZoSkvLepC
4k1FuMnlNFlo4mJmIwig9dHP4JeA6OfwS8CMWJeQeujqfBLwHR1Pgl4AxLyD10dT4JeA6OfwS8AYl5B6
6OfwS8B0c/gl4AxLrZe90+8vimsLepK4jNxajHjllyWG7h4mK+QAFaAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAADxOlCfrwT70ewEmIn25eTUflx8B5NR+XHwOoCctdnLyaj8uPgPJq
Py4+B1bwssyes7UVIXErfT0vReHN8cv6A5a7NP5NR+XHwHk1H5cfAw8df1i2kqlWTcX1TjwZrNG1elqttv
xW7UjwnDPIHLXZM8mo/Lj4Dyaj8uPgdQDlrs5eTUflx8DpGKisRWEfQFiIj0HyUVJYksr6n0BXLyaj8uPgPJ
qPy4+B1Ac8tdnLyaj8uPgPJqPy4+B1IGtV69tptSra+0WMcMg5a7JXk1H5cfAeTUflx8DD1NotXpYVSSi32x
Ndod1VvNLpVqzzOXNg5a7JXk1H5cfAeTUflx8DqActdnLyaj8uPgeoUoQ9SKXcj2AsViP0AAK8So05vMoJ
v6o8+TUflx8DqA55Y2cvJqPy4+A8mo/Lj4HU4X1epb2lSrRpOrOK4RXWDlrs9eTUflx8B5NR+XHwMO6+t
6xWludJhPio8FE5VK+r6PXj0s6kG+KUnlSBy12b3yaj8uPgPJqPy4+BF0bUfOenwrtbs+Ul9SeDlrs5eTUflx8
B5NR+XHwOoBy12EklhHyUVJYkk19T6A6cvJqPy4+A8mo/Lj4HUBzy12cvJqPy4+A8mo/Lj4HUA5a7OXk
1H5cfAeTUflx8DKalaa5PUa8rfpeic24YfDGSsvJaxYqLuatWnvcsy5g5a7N95NR+XHwHk1H5cfAzWyU724
r1K1ec50VHCcn1mqBy12cvJqPy4+B9jQpReYwin3HQA5Y2D5KKksSSa+p9AdOXk1H5cfAeTUflx8DqA55
a7OXk1H5cfAeTUflx8DqActdnLyaj8uPgPJqPy4+BQ7RbQTsqkbaza6XnJ88fQn6FK+rWfT38lmfGMcYwg
ctdk/wAmo/Lj4Dyaj8uPgdQDlrs5eTUflx8D1ClCn6kUu5HsAisR+gAB0+bq7EMLsR9ATD5ursQ3V2I+mW
2wva9tVt4UKsqeY5e6wYajdXYN1dhg6NDX69GNWk60oSWU88y42et9VpX0nf8ASdHu8N59YMNJursQ
3V2I+gGHzdXYj6AFAAB83V2Ibq7EfQEw+bq7EN1diPpTa/ra0umoUkpV5rgn1LtBhcbq7EN1diMDSWt6u
3UhKpKPbnCLDSbPW6GpUqdWdSFHnJt5TQMNdursQ3V2I+gLh83V2I+gAAAAAAAGc2xua1tRtXQqSpu
UpZ3evkUdtT126oqrQlWnB8mmBvwfnN3c6rZVVTuK1WE2s4bNfsxOvU0eM7iUpSlNtOXYBbgAAAAB8
xk+gD5ursQ3V2I+gGHzdXYhursR9bSTb4JGFv8AXL7Ur3oLOUoQct2MY833gw3O6uxDdXYjC3On63p9H
ymVWWFxbjPLRc7M65Vv3K2ueNWKypdqBhod1diG6uxH0Aw+bq7EfQAAayAB83V2Ibq7EfQEw+bq7E

N1diPNarGhRnVm8RhFyf4GHuNW1LWb10bSUoxb9GEeztYXDdbq7EN1diMLcW+t6RBV5VZKK5uMsr8
TR7P6z50t3Gokq9P1kutdoMLfdXYhursR9AMPm6uxH0AAAAABWbQU7urpu7Yb3S76fo88cQLMH5zd3
Oq2VXo7mtVhPGcNm40qq5aRQq1ZZe5ltgTgYfVdfu7+7dCxlKFPO7FR5yI1WGtaZFV6jq0125yB+ggptnt
ZWqUHCrhV4esl1rtLkAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAhazXlbaTcVYPElDgY/ZO1hdatvVEpKlHfw+3ka7XKMq+j3NOHGThwMnshcwoas4zaXSw3U32
8/wBANdq1rC702tSnFP0W4/RmO2Uryo61CmnwqJxa/r+hs9TuIW2nV6s2klF4+rMXstRlW1ynNLhTTk/
y/UDfgAAAAAAAAAAAAAMdtv75b/wAv9WXmzP7joFFtv75b/wAv9WXuzP7joAWoAAAAAAAAAAAABtJZf
BIDyoxhlxilni8Ixm12pUruvTtqD3uib3pLtfUSNotot7etLGfDlOovyR82e0SknG8vpRcucKbf9WBbbMWc7
TSYdKnGdR72H1FweYThNehJPHYegAAAAAAAAAAA8V6nRUKlTGdyLljuR+b3uoS1G/wCnu3Jwb9WPU
uxG/wDONlUuKlrKtDpI+jKLZB1ax0qGn1ZVKdOHovda4PP0AkaHfWNzZxp2XoqmsOD4NFkYDZaU463T
VNvdaal3G/AAAAAAAAAFZruqx0uzclh1p8IR/Usz821i7rXupVp1ctRk4xXUkgJ+g6bPUrud7dtujTe9Jv8Ai
ZqLTW9PuriNvb1c1HyjutcjKW+0lza2atqVvRUFHHqvP5ldYXtWxvYP3TpcppAZmy2sq2ltToeETQcYLGcvJodl
1231RuEU6dVLO5Lr7j3e2el+STlWp0Yww+PIxWiby1yh0Lfr8O4D9HAAAAAAAAAAAA/O9oazr65Wc36M
ZbvclwP0Q/PNpbeVVrNbK4Te8n25Aso7WRtaMKFnaLo4JLMnj8iy0raihqip0Uuip0
5RccttJ5fWYzVYUaOuTjZ43VNpvAAAAAAAAAQdU0qhqsKa2la2klx9WSih2tv/ACbT1bwWJ1uH4AZatKprWtvGWW6s8L6RlNjDeqVL2a4R
9GHf1mvAAAAAAAAAAAD5OKlFxfJ8D85q72i61nsLplZUpEcpm9ZaajbUPfh4VVmyKxV76UK8jHdfEyPyfBx2nxJJrv4iyjNdDg7J0
09bhGm3uyypdwG/AAAAAAAAAAAEXUqErnTq9Gn7z5Hr8mutvZHzzlIVd3pWm6hVk6kI
OrHhJxeGgKjW9pLW50Oa2W6NtyEcK2CbhFCYu0f5mjWy19lRM54O4C+NtNKytQ1ndhtCr2X1Gk8Q05
WvRpx3U/qwNqAAAAAAAAAAAAMNtl+9o/Yi6qVXS2NTi8N0ms+JS7ZfvaP2InoW8rrZKNCzZK2VLJ7q+oFFsd
RjU1Wc5LLp09x5t01z1Db6Jm2lt+ykKMhu/NvtmXtsB6dmIhVN0m3OSKfpOPuMe9JG29HzqJ4s+rIZSpXu/0
l6jpa7CKfCScWb8wmyVvKvq/S49GnFtv69RuwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAA0mmnyZjtY2Zr07iVxpy3oN8Y1ZVA8FkqUxuGGMYH9sJp3dHHXWC0Y
eOlW7TalWn60v0RagAAAAAAAAAAAAAAAzG1em3l9c0ZTt1D0U9UGjy9tDBI+ADEf2O1D51v/AJn/AKH3+x+x+o/Pof5n/obYAUmzuj3ClsrWQGa3rjO
MbjbLsAAAAAAAAAAAAGspoADJahsjWVWVFZaVeNHHJZqLnq6K04RVhHpmbxSNPx8dV1gIbLtrKRjtZ7JJUd9okN7BAhN2uNyr4p4CyFKzaBHBR9m5o8EUj0YvYT9d6F9bQFMJCWwdAtjjgW7Y5w9e3JTjiHhMFE2Ru2zKdwlsWLQfgi8Cdi6Jt3DGw
545fRFsAAAAAAAAAAAAObt6LeXSi2/odABGr2tKVCoo0Y7zi0uHXgymhaJF0rOKPY0nSJdUw1VVlJfNAjRqbVhsgyk5RLbYDxIrD6/uXiuVvmP3l6Nd6Ob+zWy+pn1gq3zwyKUC2ZnFwZJoPT2f1+NvLXvz4kKKRlYt0TRB0O3N9RnRG
HEIqK7EfQAAAAAAAAAAAAABl9V2Vq3NzO4ZR1bnT8NzKOAxPG5sI66wcB4xtNg0y+Bz0sL7ip58Ex3/58GZD+SyQZ9CYiwSH/FWr6T4LJ/JUS2v8r0G4wQVhw9LvfNv
zjhHuLoAAAAAAAAAAAAAAAK7WNHo6rRS1FDRbC7/yfYy1PtU5zqFl8M7bzRgRDCTvzy61lJwB4U8eqSkMaJy9V6QWASCjJyPfUIi2GAUdhV5zadDq+3SB6Tjf/A9U
zGEeWTSgAAAAAAAAAAAAAABmdb0C/1S/daNWiqaWIpyeUvA0wAjadZxsbGlxx6C4tdb6ySAAAAAAAAA
AAAHyUVKLjJJdNqq2FWMU3nclwx3GrAGKezwWr18Qk/QqQPOXYaXnNmTkbcrSMTr+V9G5cnWM0ocvRfktJuFZSQHyz6LAAa0lRiVr0gL8WaiE
XZ3FuAAAAAAAAAAAAAPk8GNvdl9RldKVKhNnGpyvGrCukeO8pkSSi/Q4TD4mhRyiPh79zAxNnDJbmMwz4cFu0I7nGawwpQKPe9Dq70/U
7gAAAAAAAAAAAAM5r+z91qd8CdKMVHGJt5/IutPt52thRoP4iGWzvB2lYaAK+xpEGTuycxSr8KhYr5c5b1ncNMd0vItVYuzT2hsdAf4GUnv5K8yhL
kytpbI386iVWpTjDreWzbgCHpem0dMtuiorLfGUnzbJgAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAABTPaS1Wp+R
bks727v8AVkuT86/7Sf8AjH6KAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAANpJt8kDxW9hU+1/kBVW20drc
6k7OMJJ53VN8my4PzrSP7xUeuvi+q+yT3YX+rn6KAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAH51/wBpP/GP0U/Ov+0n/jH6KAAAAAAAAAAAAAAAAAAAAAAAAAAAA8VvYVPtf5Hs8
VvYVPtf5Afnukf3io9dv1P0U/OtI/vFR/nfqfooAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAfnX/aT/xj9FPzr/tJ/wCMfooAAAAAAAAAAAAAAAAAAAAAAAAAADxW9hU+1/k
ezxW9hU+1/kB+e6R/eKj/O/U/RT860j+8VH+d+p+igAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAACNc6haWk1G4rxpyaykySZ7arSHeUFd0VmrSXFdqAzHTU/PvTb66Ppc730N/ba
jaXc3C3rxqSSzhH5hh5xjibbZXSHa0fK66aqVF6K7EBogAAAAAAAAAAAAAAAAAAAAAAAIV7qVnbK
dKtcQhU3X6L7iaZbazR5VU7+gm5RWKkfp2gUGmVqdLXKVWpJRpqrlyfZk/QLW+tbzeVtWjUceeOo/L0
m3hLLZvdmtI832rq1V+2qrLXwrsAuwAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAA+IAFG9mLR6n5XvPczvdFjhkvEsLAAAAAAAAAAAAAAAAAAAAAAAAAAD5KKnFxksprDT
PoApLbZm1t9Sd2pOUU8xptcEy7AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAR615Roy3Zy
49iRI6jPV23Xm38TJLx1tSaR4WvnK37ZeA85W/bLwKYEyzdxdc+crftl4Dzlb9svApgMncXXPnK37ZeA85
W/bLwKYDJ3F1z5yt+2XgPOVv2y8CmAydxdc+c7ftl4EmlVhWhvQeUZ0stIbzUXUsFiXppa1rWxKzPFWrCj
DeqPCPZWau3vQXUV76luWuUjzlb9svAecrftl4FKDnLJ3F115yt+2XgfPOVv2y8CmBcncXXPnK37ZeB985
W/bLwKUEydxddecrftl4Dzlb9svApQMncXXXnK37ZeB0o3lGvLdhLj2MoTrbtq4ptfEi5WvEWz5aABcjzUe
Kcmuwrbnw4VL6hSm4yk212I8ec7ftl4FPJ5k2+0+HOWKeItlc+c7ftl4Dznb9svApgMp3F1z5zt+2XgPOdv2
y8CmAydxdc+c7ftl4Dznb9svApgMncXXPnO37ZeB985W/bLwKUDJ3F2jhONSKlF5TPNatChHeqPCImlN
uhLPJSOOrP9rBdWCtM6kxp8yT5zt+2XgPOdv2y8CmBMs3cXXPnO37ZeA852/bLwKYDJ3N1z5zt+2XgP
Odv2y8CmAydxdc+c7ftl4Dznb9svApgMncXXPnO37ZeB6hqFvOSipNN9qKQFyRxNmlPFWrCjDem8I+W
7boQb7EV+rt79NdWGVqvflpzJHnO37ZeA85W/bLwKYHOWXuLrnzlb9svAecrftl4FMBk7i6585W/bLwH
nK37ZeBTAZO4uufOVv2y8B5yt+2XgUwGTuLrnzlb9svA60bujXliEuPYyhOts3G4g12jK14i2fLQEa6vIW2E
1mT6iSUup+9vuLLTrXmlcwkedo/KfiPOy+U/ErAMsfXvus/Oy+U/Eedl8p+JWAmTr3WfnZfKfiPOy+U/ErA
MnXvus/Oy+U/Eedl8p+JWAuTr33WfnZfKfiSbW9hctpLdl2MoyVp3vcRl3TWvNoiV2R7q7hbJZ4yfUiQU+
qP/AKUvtK06t5pXMO3nZfKfiPOy+U/ErATLH177rPzsvlPxHnZfKfiVgJk6991n52Xyn4jzsvlPxKwFyde+6z8
7L5T8R52Xyn4lYCZOvfdZ+dl8p+JItb2FxLdw4y7GUhJsPe4Fy7prXm0RK8I91dwtsZWZPqJBTap71+Alp1b
zSuYd/O0flPxHnaPyn4lYBlj6991n52Xyn4jztH5T8SsBMnXvus/O0flPxHnaPyn4lYBk6991n52j8p+I87R+U
/ErAMnXvus/O0flPxJltcwuIb0OrmigLHR/XqdyLl6aWta1sStCFcajCjUcFFya5k1mdr+3qfc/zEvbX1JpHhYe
dl8p+I87L5T8SsBMsvXvus/Oq+U/EedV8p+JWAZOvfdZ+dV8p+I86r5T8SsAyde+6z86r5T8R51Xyn4lYBk6
991n52Xyn4ky2uIXFPej1c12FAWOketNFy9dLWta2JWhDub+FCe4ouTXPBMfIz9z7zU+5h6615pHhO87
L5T8R52Xyn4lYCZZevfdZ+dl8p+I87L5T8SsBcnXvus/Oy+U/Eedl8p+JWAZOvfdZ+dl8p+I87L5T8SsBMnXv
us/Oy+U/EmW1zC4hmPBrmigLDSPaz7i5emlrWm2JWpDudQhQnuKLk1zJhnrnjc1M/Ew9ta80jwn+dl8
p+I87L5T8SsBMsvXvus/Oy+U/Eedl8p+JWAZOvfdZ+dl8p+I87L5T8SsBcnXvus/Oy+U/Eedl8p+JWAmTr3
3XVtfQry3cbsuwllDZe90+8viw16N5vXyAAr2AAAAAAAAAAAAAAAAAAB8clH1ml3sKcZerJPuYH0AAA
AAAAAAAAAAAAAAAAfHOMfWkl3sKUZeq0+5gfQAAAAAAfHyM9W9tP7maJ8jO1vbT+5klk4n1DwAC
MYAdoWtacVKMG0+sLETPpxB3dncJZdNnAExMewABAsdH9ar3Irix0f1qvchD10fuFoVer+vDuLQq9X9e
HcWWzX+JVwAOXzgA+pOTwlllHwEiNjcNZ6NrvOVSlUpPE4tB1NbR5mHgAByHSh7en9yOZ0oe3p/cgtfb
QrkjzU9nLuPS5I81PZy7jp9WfTOy9Z958PsvWfefDl8mfYAAAOtK2q1eMINrtPc7KvBZcMr6B1FLT5wjgPg
8MByAAC20r2EvuOGre2h3HfSvYS+44at7aHcX9NlvwoAAIxgB6p051ZbsFl9gIjLyDv5FcfKZzq0Z0mlUi4t
hZrMe4eAAEAABoLb3eHcV+r+0h3MsLb3eHcV+r+0h3Mst+r+NXgAjAAHulRnVbVOLljsCxGfTwDv5FcfK
ZyqU5Upbs1h9gWazHuHkAByHS394h3nM6W/vEO8LX20JS6n72+4uil1P3t9xZbuI+EQAEYAA729pUuIu
UGsJ44hYiZnEOAJvmyt2x8SJUg6c3B80FtS1fbyACOQk6d75EjEnTvfIlh3p/ULwptU97/wAKLkptU97/AM
KLLZxHwhgA5YAA729pUuItwa4PHEqxEzOIcATfNlbtj4kWrTdKo4S5rsC2pavt4AAchJsPe4EYkWHvcBDu
n1C9KbU/evwLkptT96/AstnEfCGACMAASaFlUr09+Djj6ha1m3iEYEzzZW7Y+JEkt2TT6gtqTX2+AAjkLHSP
XqdyK4sdH9ep3IsPXR+4Wj5Gdre3qfc/zNE+Rna3t6n3P8yy9+J9Q8AAjGAEqlYVatNTi44f1DqtZt6RQTH
ptZJvMeH1Ib4PALVmvsABHIWOketMrix0n1plh7aP3C0fIz9z7zU+5mgfIz9z7zU+5ll78T6hyABGIAJNKxq

1qanFxw+1ha1m3pGBMem1km8x8SI1htdgW1Zr7fAAHIWGke1n3FeWGke1n3CHro/cLVmeuPeKn3M
0L5GeuPeKn3Flo4n1DmACMQD1Tg6k1Bc2S/Nlbtj4h1WlrekIHe4tKlvFSm1hvHA4BJiaziQABHey97p95f
FDZe90+8viw28N8yAArUAAAAAAAAAAAAAAAAEDWdQWm6fOulmfKK+pPKvaLT56jpjp0uNSD3ortAy
dvDVderTcKs5Jc8yxFHapomtWS6SlKbx1058SLp2p3miVpwVNYb9KE0X9pthbVGo3NGVL/mTygLbTKlW
lpdGd/VSqNZbm0u4zdvtHfz1iFGdeHQOruv0Vy7y/1O0jremxjbXEYxk1JTxkwlGzdXUo2inhynub2AP0yn
WpVc9FVhPHPdkmfalSFKO9UnGEV1yeEVGh6NLR+mnUuFVUl1RxgzWo3lzrmreTwnu097dis8Eu1gbK
Oq2M57kbqln7iYmpJNNNPrRj6+ycIWzlSu06qWcPkxspqlaF47CvJyi87uX6rXUBqat9a0anR1a9OE+xywz
zW1C0t8KrcU4t9TlxMTtV+/J4+FEux2Xr39tG5ubjcdRZimsvAGvoXNC5jmhVhUX/ACvJ1PzutTvNntSSjPiv
STXKaN7ZXMbuzpXEeCqRTwB3OCvbaVfoVXpup8O8snaSbi0nhtczE0Nn9Rjq6k4tRU951c9QG3IGs3F3b
WMp2VJ1Knju/XHWTwBgaOn6zquazdRxzznLHgjj0+p6HeKFSc4SXHdcsxkj9BnKFGm5SajCKy31IwWu3
3njVYq2i5KPoQx/EBt9Pule2NK4SxvxzgkkTS7V2em0KEvWhHj3ksAAAAAAPkZ2t7af3M0T5Gdre2n9zJLJ
xPqHgAEYwsLfUI0qUYODbXArz6vWXeHdLTWfDQN71LPajPsv17BfaUDLL34j9PgAIyhY6P61XuRXFjo/rV
e5CHro/cLQq9X9eHcWhV6v68O4stmv8K4AEfOC2sKEaVHpZr0ms57EVS5ourt7ljLd4cBD30YjzbZFqapJ
TapwTiu0k05076g01x612FKT9Kk+nkuporrT1bWti3pDrU3Rqyg+o8E3VElcp9qIRHjevLaYDpQ9vT+5HM
6UPb0/uQc19tCuSPNT2cu49LkjzU9nLuOn1Z9M7L1n3nw+y9Z958OXyZCTY0OnrYfqriyMWmkpdHN9e
RD10q81ohJr16drTTa+iSOdvfwrz3GnFvl9SJqsn08V1JESi3GtFrnkr2trTF8R6T9StVu9NBcuZWl/cJStpp9h
QCXnr1iLZgABHgttJ9hL7jhq3todx30n2EvuOGre2h3F/TZb8KAACMYdrWuretvtN8MHEBYnE5heWt0rn
exFrd7SFq3tYdx70n1ah41b2sO4v6a7Wm2jmUAAEYwAAaC293h3Ffq3tIdzLC293h3Ffq3tIdzLLfq/jV4AI
wBIs7lW0pNxbyRwFi01nML62uFcU99RxxwVmpe9y7kTNL92feQ9S97l3IrXqzM6UTKIACMYdLf3iHeczp
b+8Q7wtfbQlLqfvb7i6KXU/e33Flu4j4RAARgCRbXk7eLjGKabzxI4CxM1nML+2qOtQjNrDZTXfvVTvLax9z
h3FTd+9VO8stWtOaQ4gAjIEnTvfIkYk6d75EQ70/qF4U2qe9f4UXJTap71/hRZbOI+EMAEYAkW13K2i1G
KeXniRwFrMxOYX1tVdahGclhsqL33qfeWmn+5wKu997n3llq1pzpxLgACMgSLD3uBHJFh73AQ7p9QvS
m1P3r8C5KbU/evwLLZxHwhgAjAEqhfToUlCMU0iKA6raazmGhpyc6UZPm1koavtZd5eUPd4faijq+1l3llp
1/mHgAEZAsdI9ep3Irix0j16nchD10fuFo+Rna3t6n3P8zRPkZ2t7ep9z/MsvfifUPAAIxhLo39SlCNNRWFwI
h9j6y7w6raaz4aBvNFvtiZ+XrPvL//AKv/AISgl6z7yy0cR+nwAEZQsdJ9aoVxY6T61QQ9dH7haPkZ+595qfc
zQPkZ+595qfcyy0cT6hyABGIJdG/nShGCimkRD6uaDqtprPhoM71He7Y5M/P15d5fx93X2lBP133llo4j9P
gAIyhYaR7WfcV5YaR7WfcIeuj9wtXyM9ce8VPuNCzPXHvFT7iy0cT6hzABGJ6pTdOoprmi0s72detuSiksF
STdL95fcIe2jaYtEQkat7CP3FUWuq+wj9xVCV1/sAAeDvZe90+8vihsve6feXxYbeG+ZAAVqAAAAAAAAAA
AAAAAACPe3tGwtnXrtqCwuHMkELVNNo6pbdDWcopPKcXyYHhXGl6lSTlOjVi+qf8AuZraaw021oxqWc
oxquWHCLysHStsbcRlmhcwa6srDPVvsbVlNO5uYqPWoriBK2KdR2NdSzuKfo57jPW8o0dooyqPdjGvlt9
XE39naUbK2jQoR3YR/qUer7LxvbmVxbVVTnPjKLXBgXsp069KpTp1Iyk4tcHk/OFQhDU3Ru5ypRU2pSS4
rjzNdoWg3GmXMq1W4U1KO64o76vs9b6lLpYvoq3xLr7wKtbN6fKn0i1P0cZzw/1O+jaFYq4jd2l5Kt0cm
sbuOJC/sdd7275VT3e5l9oui09Jpy3akpzn6zb4eAGU2p4a5N/8qNZpGp2txptGSqwjKMEpRbxhmT2pW
ddmvoic9kZ1aUKttcKKnFNxkgIm1d9SvdSiqD3o04brkut5NbolGVDSLaE1iSgsrsKrS9k6drWjWu6irSi8qK
XDJpFwAN4WWVMNo7Cpe+SqUt7OFLHotlrJKUXF8msGXpbJShqSrSrp0VLexjiBqTxWq06FKVSrJRhFZb
Z7IOs2E9S0+VtTqKDk08v6AZHXNdqanW8nt242+cJZxvd5caDYWGnU1WrXFKVw18Xq/Qg/2LuP+Lpf5
WP7F3H/ABdL/KwNfSq060N6lOM48sp5PZXaHps9KsHb1Kkajc3LMVjnj/QsQAAAAAA+Rna3tp/czRPkZ
2t7af3MksnE+oeAARjD6vWXefD6vWXeFj2v17BfaUDLtV6PQ46WGcdpSMstGvMTh8ABGYLHR/Wq9yK
4sdH9ar3IQ9dH7haFXq/rw7i0KvV/Xh3Fls1/hXAAj5wuZd1V09i93jvRyUhNsb1UV0dT1ep9gh7aNoiZif2
hvg+JYaVTe9Ko1hYwjvKlZ1Xvtwb+481r2jQp7lHDfVjkV6V04pPNMompTU7ppfw8CIfZNyk23ls+EZ7Tz
WmQ6UPb0/uRzOlD3in9yCV9w0K5I81PZy7j0uSPNT2cu46fVn0zsvWfefD7L1n3nw5fJkLLSZr04N8eZW
nSjVlRqKceoQ707ctspuq05b0ZpcMYZEtabqXEUlwzxLWldUK8PSkl2qR66S3optShHuK0zp1tbmyXk1TtZ
t9mEURKvbvyiW7HhBcvqRSS8da8Wt4AAHittJ9hL7jhq3todx30n2EvuOGre2h3F/TZb8KAACMYAALLSf
VqHjVvaw7j7plSFNVN+cY57Xg8anUhUqQcJKXDqZf01TMdFCABGUAAGgtvd4dxX6t7SHcywtvd4dxX6t
7SHcyy36v41eACMAAALfS/dn3kPUve5dyJOnVacLfE5xi88myJqE4zum4yUljmi/pqvMdKEYAEZQ6W/vE
O85nS394h3ha+2hKXU/e33F0Uup+9vuLLdxHwiAAjAAAC8sfdIdxU3fvVTvJ1re0aVvCE5PK+hX3E1UrzlF
8G+BZadW0TSIhzABGYJOne+RIxJ073yId6f1C8KbVPev8ACi5KbVPev8KLLZxHwhgAjAAAC70/3OBV3vvU
+8mWl7RpW8YTk1JfQg3M41K8pReU2Vp1LROnEOQAIzBIsPe4EckWHvcBDun1C9KbU/evwLkptT96/A

stnEfCGACMAAAL+h7vD7UUVX2su8tKV/QhRjFyeUscmVdRqU5Ncmyy061omsYeQARmCx0j16nciuLHS
PXqdyEPXR+4Wj5Gdre3qfc/zNE+Rna3t6n3P8yy9+J9Q8AAjGH2PrLvPh9XBrIF/wD9X/wlBL1n3lr5fQ6H
d3nnGOTKp8W2WWnXtE4w+AAjMFjpPrVCuLHSfWqCHro/cLR8jP3PvNT7maB8jP3PvNT7mWWjifUO
QAIxB9XNHw+rmBfx93X2lBP133lsr+gqW7vPOMcmVMuMm12llp17RMRh8ABGYLDSPaz7ivLDSPaz7h
D10fuFqzPXHvFT7jQsz1x7xU+4stHE+ocwARiCbpfvL7iESbGtCjWcpvCwId6c4tEpmrewj9xVE+/uqVelGN
NttPPIgCXetMTbMAADxd7L3un3l8UNl73T7y+LDbw3zIACtQAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
z2rbNz1HUHcq4UE0ljdyXtCn0NCnTzndilk6AAAAAAAAAAAAAAAAAAAAAD5Gdre2n9zNE+Rna3tp/cySyc
T6h4ABGMAAAAAAAALHR/Wq9yK4sdH9ar3IQ9dH7haFXq/rw7i0KvV/Xh3Fls1/hXAAj5wACBx7QAUA
AAOlD29P7kczpQ9vT+5Ba+2hXJHmp7OXcelyR5qezl3HT6s+mdl6z7z4fZes+8+HL5MgAADIAAAAAABba
T7CX3HDVvbQ7jvpPsJfccNW9tDuL+my34UAAEYwAAAAAAAAAAaC293h3Ffq3tIdzLC293h3Ffq3tIdzLLf
q/jV4AIwAAAAAAAAB0t/eId5zOlv7xDvC19tCUup+9vuLopdS97fcWW7iPhEABGAAAAAAAAAAJOne+RIx
J073yIh3T6heFNqnvX+FFyU2qe9f4UWWziPhDABGAAAAAAAAAAJFh73Ajkiw97gId0+oXpTan71+BclNqf
vX4Fls4j4QwARgAAAAAAAACx0j16nciuLHSPXqdyEPXR+4Wj5Gdre3qfc/zNE+Rna3t6n3P8yy9+J9Q8AA
jGAAAAAAAAFjpPrTK4sdJ9aYh66P3C0fIz9z7zU+5mgfIz9z7zU+5llo4n1DkACMQAAAAAAAAWGke1n3F
eWGke1n3CHro/cLV8jPXHvFT7jQ9Rnrj3ip9xZaOJ9Q5gAjEAAAAAAAA72XvdPvL4obL3un3l8WG3hvm
QAFagAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAHyM/cwlC4mpLHHJoD
xUpU6nrwUu9EeOrp88M6C/8kofKh4DySh8qHgMPDtrbqAF/5JQ+VDwHklD5UPAYTtrbqAF/5JQ+VDw
HklD5UPAYO2tuoAX/AJJQ+VDwHklD5UPAYO2tuoCz0iDSqTa4PCRM8kofKh4HWKUVhLCGHpp6E1tmZ
fSt1aEnuTS4LgWR8aUlhrKK971564ZsF/5JQ+VDwHklD5UPAmGXtp3UAL/ySh8qHgPJKHyoeAwdtO6gBf
8AklD5UPAeSUPlQ8Bg7ad1AC/8kofKh4DySh8qHgMHbTuoDtawc7mmorPFNlz5JQ+VDwPcKVOn6kFH
uQwteHmJ8y9rkfJrMJLtR9BWtnKkXCpKMlhpnk0M6FKo8zhGT+qPHkdv8qHgTDHPDTnxKhBfeR2/yoeA
8jt/lQ8BhO2tuoQX3kdv8qHgPI7f5UPAYO2tuoQX3kdv8qHgPI7f5UPAYO2tuoQX3kdv8qHgfVaUE8qlH
wGF7ad3DS4SjbtyWMvgcNWg9+E8cMYLRJJYR8lFSWJLKYaJ0805WbBf+SUPlQ8B5JQ+VDwGGbtrbqAF
/wCSUPlQ8B5JQ+VDwGF7a26gBf8AklD5UPAeSUPlQ8Bg7a26gBf+SUPlQ8B5JQ+VDwGDtrbqA+pOTwll
svvJKHyoeB9hb0qbzCnFP6IYO2nd9oxcaME+aRA1aEm4TS4LKZZnyUVJNSWUw03pzV5WbBfeSUPlQ8B
5JQ+VDwGGbtp3UIL7ySh8qHgPJKHyoeAwdtO6hBfeSUPlQ8B5JQ+VDwGDtp3UIL7ySh8qHgPJKHyoeA
wdtO6hO1pBzuYJLk8lx5JQ+VDwOlOlCmsQior6IYWvDzE5mXsqtToT6XpEsxa8C1BWm9OeMM1h9gw+
w0mF2DC7CYZu2/rN4fYMPsNJhdiGF2DB239ZvD7Bh9hpcLsR8wuxDB239ZvD7Bh9hpMLsGF2IYO2/rN
4fYTtNoTddVGsRXX2lthdiPuBh1Xh+Wc5Cr1ShN1VUim1jDLQFe96c9cM1h9jGH2GkwuxDC7ETDN239Z
vD7Bh9hpMLsQwuxDB239ZvD7GMPsZpMLsQwuwYO1/rN4fYMPsNJhdgwuwYO2/rN4fYTdOoTlXVRp
qK6y3wuxH3Aw6rw/LOchV6nQm6qqRTaa446i0BXvekXjDObkvhY3JfCzRYXYMLsJhn7b+s7uT+Fjcn8LN
Fhdgwhg7b+s7uT+Fjcl8LNFhdgwuwYO2/rO7kvhY3J/CzRYXYMLsGDtv6zu5P4WWel0J04ynNY3uCyT8L
sPow7poRWc5Ciu6E6VeTae622mXowiu9TT54wzWH2DD7DSYXYMLsRMPDtv6zeH2MYfYzSYXYhhdgw
dt/Wbw+wYfYaTC7BhdiGDtv6zeH2DD7DSYXYMLsGDtv6zmH2FrpdGVOMpzTW9yyTsLsPow9NPQ5Jzk
KO9oTp3EpNPdk8pl4MFd6mnzxhmsPsGH2GkwuwYXYTDw7b+s3h9gw+xmkwuxDC7Bg7b+s3h9jGH2
GkwuwYXYMHa/1m8PsGH2GkwuxDC7Bg7b+s3h9ha6XRnDenJNJ8Fkn4XYfRh6U0OWc5CjvLepC4k917
reUy8GCvTU0+eMM5uS+Fjcl8LNFhdh9wuwmHh239Zzcl8LG5L4WaPC7D5hdgwdt/Wd3JfCxuS+Fmiw
uwYXYMHbf1ndyXwsbkvhZosLsPuF2DB239U1hbzlcRnutRjxyy5AK99PTikYAAHoAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAGY1nanyavK3soxnKLxKb5Z+hd6vcO20q4q
x9aMHgxuy1nC91bNZb8acd9p9YHWntVqVGalWUZxfVKODVaTqlHVLbpKfoyXCUX1HzV7CjdabWg6cc
qLcWlyZktk7iVHWY00/Rqpxa/wDfcBvQAAAAAAAAAIOs3dWy02pXoLM44wmicfGlJYaTX1Aw8trNRj6
0ILviavRrupfabTr1cb8ueDNbaxjG7t91Jfs+pfVl9sz+46AFqAAAAAAAAABwvrh2lpUrRpyqOKyoxWWzu
GsgYOeq6zqtaSt+kUV/DSjwj+JzWqaxpddKvOov8AlqLKa/E3tOjTpb3RwjHeeXhczIbYX9G4rUrak1KVJtyk
vr1AaXSdQjqdjC4it1vhKPYyaU2ytrO20iLqJqVR72H2FyAAAAAAAAAAAGP1KeurUa6tlX6Lfe5iPDGSvub
7W7RKVxUrU0+W8sZN7WqKlRnUayoRcsdx+c6hqMtTv1VupSjSzhJcd2IGh2Uu7+8uKs7ipOdGMcJvlk1
BXaLXsallGFhJbkFxXX+JYgAAAAAAAAACv1nU4aZZSqPDqPhCPawIG0Gv+bpRoW27Ks+Ms8kiZodxe3dp
5ReqMVP1IpY4dpltF0+pq9/O7um3Rg96cn1vsNdbarYV6sbe3rwlPGFFfQCcAAAAAAAAABmdrNRubO
pQhbVXTcll46zTGJ20nvalRj8NP8AUDjSr7QVqcalN15QkspqPMt9npas76Xl6q9Hu8N+OFkh2G1kbW0p

W7tG1TjjeU+f9DQaXrVrqiaotxqJZcJcwLEAAAAAAAAAACq1zWoaVRSS360/Vj+rLU/Pdoq7uNcq7z9GL3
F9MAdI3uuapNyoyquP/IsJfiT9Jeu09SpUazqqm+MnNZWO89U9qbWwoU6FpaucIRw25buWWel7SWu
oVVSlF0ar5JvKf4gXQAAAAAAAAAAz2117cWVG2dtVdNylLOOvkUVG7164pqpRdecHycYZX5Gt1fSKerQ
pRq1JQ6NtrH1O1la09MsFSUm4U03lgYe41PWLWahcVqtOTWcSWDW7NVri40lVbqTlOU3hvsMdd1Ku
s63JQ4upPdj9FyP0K1oRtranRgsRhHCA6gAAAAAAAAAA3hZfIxWo7RXt5du309uEG92KisuRtJxU4uL5NY
PzmSqaJrak1GbpTyuPBoCbXjtBp9PyirOoorm97ex3ovNnNdlqW9QuEumgs7y/iRWantVC7sJ0KNBxlUW
G5PketjLGXTVLxtKKW4lniBrwAAAAAAAAAAB5q1I0aU6k3iME5N/QxN3r2o6nduhYb0It4jGC4v65NdqlKd
fTLinT9aVNpLt4GG0C+p6Xqm/cxajhxfDigJNWrr2k4rVp1Nztb3kabQtYhqtu8pRrQ9eP6lZr20FlX02pb28
ulnUwuXBcSr2QqSjrKgnwnB5A3YAAAAAAAABV7Qu8WnLyDf6XfXqLLxxLQAfnlxqesWs9yvWq05c8SWD
b6VWlU0qhVqyzJwy2zJbZfvaP2Iu51ZUdj1KLw+iaT7OYFTq20tzcXLoae3GCeE4rLkQ53et6du1qsq8Ivrm
sr+p22Pt4VtVlOaT6ODa78o1+qUIXGm3FOaT9BtfR4Ai6FrMdVt3vJRrQ9aK6/qWpgdlqro65CKfCacWb4
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAga7SlW0a5h
HjLc4Iyux1eNLVZQk0ukhurPj+huJJSi0+KfAxerbO3drdO4sIynTb3koc4gazUq0bfT69SbSUYPmYjZek6uu
0pR5QzJ/l+oqU9c1FKhVjXnFdUo4S7zUbP6MtLoOVXEq9T1mupdgFwAAAAAAAAAAAAAxu2/vlv8Ay/1
ZebM/uOgVm11hdXd1QlbW9SqlDDcVnHFlvs/Rq2+kUadaEoTXOMlhoCyAAAAAAAAAAAAAj39KpWsa1
Oi8VJRxHjgDP7Q7RdFvWtjLM+U6i6voiNs/oKrSV5ftbucxhJ8X9WQ3spqmcuNLP8w6f2b1pLCmsfzgNxF
xxiLX4H0o9nNOvbBVvLZJ7zW76e8XgAAAAAAAAAAPimBHd1a1alS3dWDmvRlBviVWqaLpcLGrN04Un
GLaknjiVd/svf+VTrW1ZVXNttt7rIn9ndYrNQqZ3f+apwA87K1KkNbhGm3uyTUu43xS6FoMNLzVqSVvJY
z1RRdAAAAAAAAAD841u/q3+o1HUeIwk4xj2JH6ORnp9pJtu3ptvi3ugY202llZZ2KtaNpTUd1pvLz
8rC/jdRgpuOfRb7T9Ar6dauhU3LanvbrxiPXgyeg6RdQ1enK7s6JPPLnHhyA1WjahLU7ONSpRqQqm/2Njk88U
qVOjDcpQUI9iR7AAAAAAAAAAAFfc09MuLx07mNCNVpzq2zN7XvKlzb11Uc3nEnhr8QLa80bS3NUo7sBT
c6NOmkn6UfRMfokpUtbbtT/Txw7P61U9CWcf81R4L4+G3dqn4j19k4hH8Bs0oa1IUlSclxMYS9ki5nIlKIF+AAAAAAAAAAAB
+d7R0JUNarprhN7yfefohV63otcuoptsytsyGqUaVprcoWj9GM1u4fJ
55E9aHrlsnSoze5/yz4E3SNlqkLiNxqEk5E85f1YGmoNyt6bl6zim+/B0GAAAAAAAAAAAKPavUPJNN6K
D/AGlZ7vcusvDK69o2p6nqDqU+iqDqU40+iisQzMDhsbY79xUvJx4Q9GPebEiasA5RsNPpW6sWbksAAAAA
AAAAAAIup1p0NNuKtKTjOMG00K14LinXXdXT6icW8+wBlLinXfgY4zWYtYafafWbl2WZG/wBlLinXdXT6icW8
qLeHH8QLfUdE012FVuj1O89G1qEjPD0stYLaOSSUqcJsDdhz6ddsDs2qjG93OTPHwFjNJF06I6fhpknW
rS6Su1jhyiBdgAAAAAAAAAAVN9oOnRq/NDxKdOdomt8+aL1Iqd+6u1WX4jTxBeP14KgFIjd/tfPVbwjN3Xi0OUV+Cz00LnzvdIi/2yi
eyqTf7SvJYgpPPrwRtjbaU9RncY9GnHC/3Ruv1nUzaeIxc0m7Kaaoo6fbRoUI4lSrbAlAAAAAA
AAAAADDbZfvaP2Ivegl7IqnBZk6Lwu18SJtFod7qN+LBe6bQnbadRo1Ut+PCnDyBjNlLuFpq
zjVkoqpFwy+3JrtYvKVpplac5rMoOnRWebZSavstOrOZXk0eXRt9XfaxU9m9m2o4TlPMtrB9KVVs2w5VTMcAAUvGAfw15ZcDupZ1bI7ec
VTHo005N/U3hA0jSqWl23Rw9KcuM59pPAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAi6hf0dOtZV6z4LgkutmYntXqFebdpbRUfqnIs
dsLWtX0+nOlFyVOTckiq0HaK3061VtXoS4PO/H9QJ2l7S3Vxe07W4tfSm8by4Y/AuNV1WlpVKE6sJSUnh
YPNlqOm6hVjUoSpyrLllYkvEzO1dG9hcOdabdvKf7NZ5Aa7Tr2GoWcLmnFxjLOE/o8Ekxuz1rqkoW9WlV
krTe4x3urPHgT9o9dq2tVWVnwqyXpS7PogNHkGOek69Gh5S7ye+lvdH0jyWOzet1L5ztrr20OT7UBoBld
pgbzU9Qpa3WjQuKraqNRhvNrwJNxp2vUaLupXFVtLelFVHlAbUGf2Z1upf71tc8a0FlS+JGgAAyGvS1da1i
2dfo+HR7mcfj+JraW90Uek9bHHvAqNf1a40x0lb0lU3+fDkV2m7T3N5qNG2qUoRU5Yb6zVNGA0/8AvZ
D+cBvyJqt3Kx0+rcQScoLKTJZWbR/uS4+0CghtZf1c9HaxljnhNntbW3dKS8os1u/iiJs1rFvpcLhXG9+0axhd
mTttBr9pqNl0FGlJyynvyXIDU6fqFLUrPpqDxng0+aZnrPTNXhrvTVJy6PfblPe4NE/ZSznbaXKc2s1XlJPOEU
+n3t1LaropXFV0+lmtxzeMceoDajJj9rry5t9TpRoXFSnF0k8Qk0s5Zx6DXtRt/KlVqKDWYxU93K7gNsDI7M
6zcu+8iu5ympZScuLTRaa9R1WtOlDTqjjCWd7Dw1+IF1kGIvdP1vTqLuZXdScY8ZbtR8C+2b1aepWklW4
1abw32oC5BgLvU7+jrdZUrirLFRqMHJteBZ22n6zO2uqlzWrQnKnvQxU45XHHBgawGW2S1SrVrVbS6qz
nP1ouby/qj1tbqVWjOjaWtScKj9KTg8P6lDTnK5qOjbVKi4uEW0cNKo1KGnUY1qkqlRxzKUnlnS/9wr/y3+
QGUjtfeyeI28H3Jn3+1eof8Iv8rK7Z7UqOmXc6teMnGUccDRx2tsG0lTqcfoBa6XdVLywp16sdycuaJeSDq
WpQsdNldYy2luR7WzJ2z1rXKs6lKvOEE+ak4xX0A3QPz/UbnV7CUba5uKa5uKsWuMZKb9Jd5oa1xWWySrKr
NVdzO/vcefaBfgzex1zXuKVy69apVo+DPEY1mPScmnPbpxGWpvHrnpnbBbG6Ledv0FapTyxnck1/kC71ijcV9Nq07SW7Va4fFVra/uUrd
mLK/tI1fLG4wl6sZPJ7q16q2R6ZVZ9L0Te/nj4kDZmrc31jfU6lxVILCUZOOyyn9ANWDHbO6lcUtXnaXlape
bj6cm8NFptVqUrOxjSozcKtV8HF4aQF6MpdZn9Oo6jHZ2VSFacsre3qot+pqot+PSPe4fj9Cuej69Vg6s7
NiZnarUruxuKMbas6lDLx3kbZ/WruGoqwvwZvZeGTZ/jWGdy9mI1XfI+AmCNA1X3H8Gfw1a8RbWUtjq9vh
SPZhp/m+F0uQyi3s+SSzm2lyJxxkeWTdVCVqLyAWXr5e4zBFC5l1z1R3C4qmwqfhctGiM/Nzhd
GSnEqj6qdmTUyzZ/aP9Ry0xOrTzNKtwOMx47W/Zz8MqZQ9sUXGrpIAlpDt9LDrUr3z6vQbF9a8YhK+s7
STkp/OEdN2co15LL3EortZRWtPWde3riN06VNPh6TS8EBtQY221TUdF1GNrqqcdY7DZj9NoJRyNgFVUmO
8mdvXnTUk87kmsgajOAYe3p67q1sqlKrUrUVKKwnv43v9T3o3o2sXtpqcqbfw1kK0VxTi4Kn/IDwn7Sa
/9PNMXIUrNf/axIpFBhWfCPQVhbFYwg2ryOzvVnXXWatuDagXaqjaHaKsQZK2W5Ok7xqN2WsGJGTmZkMcI/Sw
rywrstJgHl5PB47vvDfAMf3V/FDoATNTPqQx/YwHq9lXAP+dl2bQgr7OFe7E+yWJQwnHdvRTPH5azE3ML
+afBeU14VzdCEIryh1rSvU6I+/6SsV7cJO4+gNxtOXq4+iPB8RQ3mfJJs3PtR5h5D4QORoQnBpWsOAHdpG+
9gYKtcBnK/fXSf/8ZVuFuVyktRa0SBgNGu9Bu7UP/T3DaNu2DrNuYpjKnXSHVOlJ9IO2cxT/YV8iJHYaFFO
vdnbQnXgkyPOiuvNoLq4y/bDMU/xa1/dsFrUfXcxGsYUe2ELa1Ka7ixQq48OQg42TnAEo+Dzm+1hG4xnK3Mw
Qqhc/LaldfqHvPSsc4Xn3dPV7fZXsPFrKbfCqg0Y+p8oWZJ3mfEvC/HR2Ff+kVf39z6Hf4/Ml3X9gMJmEG1rHAs
hN2E/OV1Xd8/3K/mFu2qnkE9UF3lRGCBOXw66VjLuP7MMoMWp26Tk9SyXWXBAuSHj3a0lyhLF0FlfRFuHK3gx

M7sffULaVelXqRpueHHeeM4yctrr+hd3VKnbzU1TXFrlksrnY+hUrOdC4lSi36u7nAnsdbypRjG4nGS5y3c5
A61f7l/+CyJsP7O774/qXktMjLR/N/SvG5u7+P0OWi6NHSI1VGs6vSNPjHGMAZ7ai2lYatTvaPoqbUs9kkR
51p7Q69RSTUOCx2Jc2a/VtMp6radBUk4NPKklnBF0fZ+lpNadVVXVnJYTccYQHzaK/qaXpcfJluyk9yL+FF
NpmnXep2juqmpyhlvhnkam/saOoWsqFdZi+TXNMoY7HQjPHls+jfNKPP8AqBRadFQ2kpxjU6VKq0p/Fz
4lntt73b/Z+rLK32VoW1/C5pXEkoPKhu/qSNZ0GGr1ac5V5UtyOMKOcgVmtUZ1NlbWUE2obrkvph8Ss0
W0jeUN1alK3mn7PODbUbWFOyjay/aQjHdeVzKO62PtqlRzt68qKf8ADjKAgvQbe4u40ZaqqlbGUnxZ92
xh0VKyp5zuxayWem7L29jcRrzqyrTi8xysJEjWtEhq/R71Z0ujzyjnIHXQUlotsksej+rMnqXDayWPmo2tjbK
zs6dupOaprG81jJVXOzcLjVHfO5lF729ubv8AuBy2yTelQfZUPuydzRjpChKpFSU3lN9xc3tpSvradCvHMJf0
M/DY6EKylG8mop5S3f8AcDTkPUdSttNpRncyaUnhJLLZLisRSbzjrK7WdHp6vRhGVR05QeVJLIEuyvKN9b
xr28t6D/odyHpen09Ms429OTkk8uT62TAKXWPe4/aiyhVhRs4Tm8JRRzu7BXVVTc3HCxjB6uLNV6EKW+
1udeOZMM0VtW1rQrK9ed9XUd5Qh1ZLK18moQUITjnreeZG8zr5z8D1DSVCal0reHnkTy4pXUi2ZjylSur
eU5UpzWVwafWQdQhaxoZp7u/1YZ2raXCpNyhJxb4vrOcdHW9mVRtdmBOZdXjUt4mHrSHLyee96ueB
7lbWVVuScV24ZKVCEaHRQ9GOMcCA9HTfCq8dxVmlorEYyiXfRSqRp2yyl1rrZdWtPoreEHzS4nC20+lQa
k/SmutkwRDrS05rM2kABXuAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAKyOg2cdQ8sSl0u9vc+sswAAAAAAAAAAAAAAAAAAAAAAAAAAABwvbOjfW
7o3Ed6D6s4O4AprfZjTqFZVNyU2nlKT4FykksJYSAAAAAAAAAAAg6lpNtqbg7hS9Dlhkm2oQtbeFGnncgsL
J1AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADncNxt6j
Tw1FgdAYDTNoru2vIyuKsqtJ8JJ9S7TeUasK9GNWlJShNZTQHsGL0rUburtFCjUrzlT31s3vgbOTUU3JpJc2
wPoIb1awU9x3lHP3olwnGcVKElKL5NMD6AHwXEACJPVbCnPcld0/PcldOV31eVLS31wJNOpCrBTpzjdaGK
zmrijKW7GrBvsUkZZ3a3a3U5UqEleL5xb9Pdlx0XX0Xbn8rgy4BIxMA2Re4Fh0jZ62dsUewDlKzSpdl4KDhH
0k2+CRD87W8o5+9oH9knp9xhKLyvAdFt5gB3PE4ZzRmP7gQ9ygqxjVUVlEyW9QiqqnPvLh4NQrL7gSaNr
U6OndUZS6kpriBlAItbU7Khdx+3Wxcoq4Ugn29bvmq9xEtzrW1d6QK8pWFDp5ndOaRYrkSGhGGfy3wH8T1s
ZwpKrK5pKD5r3vba6z5PXp1Mc1Gl0GbtgqaHBBqLhkvL9aUlfFvSYtzSySSSytYwSQvfhzgrdePoCsZrnpHU
yjOKlFpxfdNwYH0EWtqNnbqD1YyXVpCiG3XbjJ3dTh4O6ugtkhIOXLdfo6cSA4Nm14rtqBwlI4OD5CCZPoSSW8
pVadaCnSzcX1ueB9lVpweJzmHoK9rfAx+2cpK+AlxotMXfiDZz8CtIaXr6xvCJA1Dkb9nN9gx6ZwJkYQwdKgQ
YSUl9Hk9Gd2MbemVctv9q+fci9r3NC3jvV6sKa7ZSSA6gjUNQtLiW7RuaU5diksnWvcUbaClXmgDMn
gDoDxSrU69NVKM4zg+UovKOdW+taNVUqtxTN/wykkwO4DaSy+RwKc5rnGMk2gO4ON
e7t7ZZr1oU/ukkebe+tbl4o16c/Uzk2BIAItbUbPhPcq3NOD9cp4M60ZDb
GvCq7adCqpRcXCl+hxeUaLRXnR7ctIlsMr5dAgDLazeazVv/JLWjOlB+q4r
1vrkDUgwl1T13SoxuK1Woo557+8jT6BqktUsekqJKpB7ssdYFoAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAc7n3ap9rOhzufdqn2sD
840vT5ajVqUoPE4wco/V9ha7PavPTrp2V42qbljj/Azxsd+9pfy3+Za7T6J5TB3ltD9rFenFfxICl0Vp7UQa5O
bJ21eoVql9HT6MnGKxvJP1mys2b/AH7b57X+RP2rs6tvqkbE4W0pSTVtbWxAhK0TlXl07WeXBE
TZ2+uLHVnYVpNwlJwab9Vosae11orNOUJ9Ko43cc2VWgWW1bAbd7KOIKbnJ9Abky21+p1acdlQk
47yzNrm+xGplj2wYMnjlNz3k3tDPS6sAOYaHzkqy7sFNvvFKXLnDKcrgyyELIPm7HZzaHxVBLhoZKF0q0
bWTUqMHmOsXEUEHAXFAjPlk/SH/BmpRF5x46PnuRifWS3c13ZVVFsZM0dfFlnk2cbDffCjsC11sqmbLNWsv
QTNCQwjiFGcGmaLeUWF1TtH2RxHM6sx56GbrOOfnE9gI2zWbtqMYNNT99YiScSVRPhzmSZDL0X9L1UHH
jlpDbNTn4HgLRtIVbwGhgsEg9jVtscGRfcLOtzaYw0Kq59JsBT0f3pwHGt7BNTbI/sawLkNLS/9FvTQ0kj5Nz
MiBNZcNSFFNjwN9bEoV1nDyRcXeMymvS1mBTWTVVmgyOJhrf0JMa+fLa4CMDQ9DlSBZmFi1yhNRkKHaZe

H3yKDa798r7UBcbValUttPo06Ut2Vfm12cP8AUrtH2YjfWcbm5qyW/wAYpdhZbR6ZUvtMo1KK3qlGOd
3tTS/0KnTNpamnWytbig5bnBdTQFfrWn1NMvI0JVHUhjeg32Gj1fUp2Gz9vGi8VKsVHPYsGa1i+r6jdxr1
6bpxccQWOrJotb0+pd7P21SjHelRim114wBVaXYaZc2/TajfbtSWfR3sNd5zoXK0fWYqzuemoNrOHwafU
NKuNIjR6PUbeXSRfrrPEsbCWhXmoq2o2k1njGbb4sDXJprKIOr2avtPnRdXoutS7CakkklyRntsp1o6fSVPK
puT38f0Ar56Lo0KTjLUl0yXPPDPccNlrqpb6x5Mqm9SnlfR45M9aZ5hjpynd5dwvWTzxf0IugOL2ipuEXGL
k3FdiAnbae/UPtLDWv7oUv5dP9Cv209+ofaWGtf3Qpfy6f6Ac9lKyt9Dua0uUJt/0RUWVC42k1Obr1XGK
W8/ouxFtstQ8p0G6o/HNx/oip0q8qbP6nUhdU5brW7JfqgJOtbPea7ZXVrWk1FpST5r6kyhVnrmzFanV9
KtR5Ptxy/Uj6/tDQv7LyW0J77W9JotdlrCdppjdaLUqzzh9gEPY69Xkta3qSx0Xr6M7qVxUvdQuLpJuKl
z7Owkaiquj6vcwpcI1E1/hl/8lzoukKrs7X34/tLhNxfYly/UCY9XT2W8qbzUcNz/ABcmV2zcPIdLu9TqRzJp7
uetL/czyq150Y2HHd6RNLsfL9TfebV5i8hhiL6Ld/F/7gY+z6LV76pV1S86OC44b4v6I9apb2enzp19Kvd959
XPFfUj2atrG9nR1S2lKPLsaLKrdbPRnBUrWdTLSb48ALK91qp/ZincweK1X0K91qp/Zn
c2W2t6XCegxhY03u03vqPXjrK3Z7aCjp9o7W6jJRi24ySArtd0qelXEafSOdFvqNton7mtf5aMZtDq0dV
uIOlBxpU1hN9bNnon7mtP5aAkXdF3FrVvoqW65wcU3yx6CH35p9QFyt7L2sVnxpGxOFPVNOd6sy/DsrcNR
5YXXgpNE3dYuUnJXPgMQUtuQzLxvBgyTWVqSpGeasQqgPFA6ufhWw4OmS9Sv6Wtl5Gvu9Nhc5lfBLN37
dOqafpuKyxUqvfa7DL6Jd6dY1PKLuE6lb+HC4RNfpes2+qTnGhGacFl7wFiAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAfJRUouMllP
gz6AIdppVjZVXUtreNObWMpsmYAAh09JsKNz5RTtoxq5zvJslVKUKsHCpFSi+aaPQArHs/pbnvO0jnvZPo
0KVvTUKMIwiupI6AAfJwjOLjNKUXzTPoArZ6Bpk578rSOevDZNt7aja09yhTC9yhTjTj2JHUARbzTrS+cXX
WW+B3o0adClGlSiowisJLqPYAHitRpV1Hd6bV6bhWhKMVB3VtV84ClnSsJAVQRZOOEDbNmWM2MWrJ8
0uysZudrQjTk1htnNnS7e9pKldU41yOYqxgjYSVjMVFh3ceCqfmwgPDgh4PiJbG/WjYL6twtk1Rec
XTqji5WsG4cI81g73NjbXlKNK4pKLLyK4pKpCPJvgSABxtrWjZWMsXOM5wm4RHaBlkVLPKbN4OLfRT6
7qrY6bOpbUnOaVkks4+pnI7T20lm50yMqvW1hZ/obE4Kn2ulGHkrnaLVYUnmR3KaxHgeJL
m8p01TpRprlFYPtOnCnHdpxUV2JYRGbOXdpVKKXm8a5tvu0L1vbDUkCHoCvuNE065m51LaDk+eNG
U4PmmewBWw0DTKUi1LWbC8I3amXCDzNpbwrdpKSa/Uq7FXQpGoqVxwGpjGnKwFPKbwAR7Ga7pCCKqm
VbVaSlRSSUG3yIgDhaWVVvY03TtaSpxaWEdRzbePIbO0trYYS+/6t3xaDPd2P4H5TGV9aPPNOavUWmVrLNbe
9Ksr2oqlSsvbxzSgSKMhwCdkdFRKaMS20mtnRZXB9KbF/ZbAe3t7v3l5mA6jiiGAKb1nVdAfJvYI9+7SctP
0PTraanStOWdx25QxRHKsYxsAkWfRaSolCOAMtF6EPjDqvTZsbNF5nfG8ToCgmpLsqLeMJOSAhjn6pxGv6
VkSOuabKW6ruGfGFxMdeVZ6rrzFoXFbo4OeqsvhFF5Fa7IeOs6uqL1+FF6vbqDuquwwuVbe5IgQ9bDJvAgkU
ZkpqEKezOjuVVyq+nxx9TK2GrO021ZXVWVSVPLe7fAqfogIem6jT1K18opOPBuL0tJvtarp0oSrOPBuL4
AX4KLT9qbS8qqlUjbPjynXJSkWEwJoM3LbG1VBTVCbqN+qzN+pnkdbDau0ubVqz2OKYTtNc5IKcccFfcN
ySqIVwcmx2Oc5y5SYuvb1pPE2m3Z6m3l8H3q3xzawStGK9DoDIZfW63r+P/CkysUTxJxVFdyQ8mMkRWAQj
Ls1Swl2tOeqtArvmvUGWfzJPLMA9A2k3QVHt7yjVpsQ03nqtrJzzgPjVn+GNfxhmTuUJQULCTvjXpTOhuKw
qnSFoHOlaWfn5h0cNzfsRzHUs7k9PaemEYQ7RM1y63bUPHZNS2RfDRP5aygaeyf7+oHPfFSEgJoHa0oaNTgM
jVpLi9oBxR9uXfcBkoIdYrrbVWvkgB2CFefPxxBtXEvY0MJZoNMDTm1uvqWX3H4BxhTpxO2yQv9NTu6lt
qZ9Xh6PdyOHwUhMiTDVddeaeIOE3fDoaBxvgqXmEhmgf8xpR5v3sWdoLnXcxJtNYrrDqJdAiC5bXcc4FCvq
kaNV9fTrNQtsEc4Yv6MmEJHLH7RgH7Dll2xmTb3AOb+xNv2tfQMgTp+wM9cbNkn+b2LOTL0JbAU6AcbCAoR
AlsiBjy7nfSwsUgCtA2tLJ7qYaGeIjL3pH0yQiVZCwbSDgXk8hBtfA8m5jfc8XcdX3jQGNeHdNbo80pbVv
0yeVE4uWoLWmBWNwNxDLSWmU6UGN8Tic9vi1+WGaxz7GmvNbIdQcsgq9btYyaFRrXyhnwlQqUaWdmXcqUE
oSCeDFGZQ6dEZrQ5sgCDvQzq0IJQOHBFgdLOGvWAAbxrUmt0z5ll6gFdqmFyVBkHVp67xiEWBiT3rGzX0U
WuNGxsJG8lx4CR9VP7vFwu5HBK4uH7Z1xBfUYY3/0TNvCRiutc9yUFsPIz8YaKnKY5nKgfmKbcqVmf3ADg5
nWAHffzTOhAJ3cEo2HgRk7YfFZ1Qyeq0O4dT6CRn5iSxKLifQBKNCIEyiEU5cqkTMgNdsg2ZuHAqW6GmWnc
VZFcj1yMUfM2jdbSuO5Vt0IcPnaLJsS4OIcXb2Z7XJd08ObpdbZUt0jJNhzeKXXmv6bm7OKpyU2oHQvRt+m
Wfw5OeJuUUp3hwBqxu8cV0v5/Gv8CQ1rdVtCrJMYUM9UyE+/7ajFntqWRcP2EzcWXeEYFEXLHf5gqHuVe+
KlGLsQZZp3bTF6s9CB5U4i8HZ1s8X2VzJ9O4BMTRjtxbL3pMZETpJy/8Vo7zAs7jtSDdl3t3Zz/NrmRfQKdT7

LW1Kzt6rp1riEJrmmePPGn/wDFUzHbSQVTaSrB8pOK/oi4jsbayin5TW4r6f6AX9vfW11Jxt60ajSy0juVW
kaFR0mrOpSq1Jucd172Dzqm0NpptTomnVqrnGPUBbgzdDbG2nvdLQnBpcOOck/Rtcp6vKpGFGVPcWe
LzkC1BSS2kpR1Z2Hk895T3N7KwW1zXVtbVKzTahFywuvAHUz1baiNLVnaeTtwU9xzzxz3EvR9dp6tVqQ
hRlT3Em3J5yV9fUdOjr/RSsN6up7vSZ4Z7cAaYFJc7SUrTUvI61vOPpJOeVjD6y63lub2eGM5A+gprLaGnf
ai7SjQm8NrpMrGF1nC92stbatKlTpTqSi8N8gNAcby6p2dtOvVzuQWXgpLLa61uKsadalKjvPCk3lE7aFp6
HcNPKceDA66Zq1vqkajt1NKnjO8scycZbYf2V33x/Us9U2gtNNn0cs1KvXCPUBbAzdvtja1KijWozpp/wA
Wc4L2pdQjZyuYPfgo7yw+aA7gzlPbC0lCcp0akHFcFnO8eaO2VvOqo1LecIt+tnOANKfFOLeFJN/Rldquq0
bLT413mcKqxFx+qMfo+seQ6jKvXlUqU2mt3e7QP0IEbT72F/aQuKcXGMupnarUjRpTqTeIwTbYHsFLpu0
lDUb120aUoP8Ahb6y6A4yuqEJOMqiTR7p1qdXPRyUsc8FZqFhCMKlfelvc8dQ0ZOVKsk8Z6znPnDx6lubl
mFjUuKVL15pHyndUarxComyBHSZTqSdaq2s8GubIl9a+RVYOE288V2oZmEnUvHmY8L2pWp0kukko57
Tn5Zb/NiVt90lezt5KLk+vCPVLSqc6EZznOLay0Mk6lpnEQsFeUG8KrE6yqRhHelJJdrM5QoqreKFNtx3uD+
hI1Ccqt70G9iMWooZSNacZwtVe28nhVY5JCaaynlFXV0qlG3bjKW8lnLZ40e5k5uhJtrGYly6jUmLYstwAV
7AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAzur7LwvK8ri1
qdHUlxcWuDZT1bPW9Gg6kakujjzcJZRLu7rXtPvatRQnOnOWUkt5Y/Qj3Ws6vqFB2ytZRU+DxB8QLjQtT
Wt21SheUoynTw32SXaZ7SKFKptGqU4KVPefovkaPZjR6unUZ1bjhVq49HsRQ3llfaRrLuaFGc47zlGUY5TX
1A0evY0/Qa/ksFTzhej9XhmX0O8hab9R2LuZt+tnl/Q0thVr65ptxRv6EqW88RzHH4lDTo6vs9cz6ClKpTl8
Md5NAcdXqSv6sKtDT5281za6/6Ftrc6lTZOhKsmp+imn3ni11TXL69pblu4U0/STJh21J21+fMvHnvoCHs
jp1tVs53NWnGdRy3VnqRF2usLe1q0a1CKpufBpfmRNIutTsLZ17SlKpQm8NJZ4o6St9V2gvYuvTlCC4ZlH
dUUBrNDqyr6NbVKnGThxJdzb07q3nRrLehNYaPlrbxtbWnQh6sI4Rx1Z3C0yu7TPTbvo458+P9AOWma
NaaY5yt1JynksDNbKT1GVSv5Y6rpdXS55/TJPQOF5d0bK3lfzr3Ofze/dfnqN9qFSMqFKFz0qcmoKMHjvFjf6rp9BUrezbh
dfnqN9qFSMqFZ0qcmoKMHjvFjf6rp9BUreznFdb6F5f8AQDaWFnTsbOnb0vVgufa+0kFRs/e3t7RrSvqbh
KMko5hu5RbgfnV5SqV9o61OjLdqSrNRfYWtTZ3WdvGS7Vcovo20yw/tBrT
4Kxmn29E/8AQCLodx5n1iVvdUYUvHGV3+LVZRn9PFJ58CPsZQq0be6be1ivWuRqS2UV0NObn0a9be6ValO
m3KON6LWeYFftdY0LWvRqUqKhJfm6uJ7ftQtvUVqRXhik80u8N3XOjcB5WYqqH6pHqa/lKzdWXE3PhmdY8+G
GnUSg8uMW8cQJ1/wD3QX8tFXhp1vduvWuKkS7QlerfxRjh2hcBQeh2K4yarSQ3EZNnuW5Q3gt+G1f5DOi8Mf6gXbqR
m3KON6LWeYFftdY0LWvRqUKahvrikX1tc0IbNUat8805U8yu2yoVq3k/Q0p1MZ3odsvzBbwdbjTq95slb0
acWqsEpbj4N8+AESnrVluOna6w6BzO1w/wDfAqdNqbu0dCVCdVqdbu0d8Vom6SlVXoNaWZbu85hp85yy8Nx
a4ke3tb57R0qtzb1N51YvSvzRY6BR0nDTqNepSjOrNb29WL4W2dvWr1bR0aVpMs7s
W8ci+0eMoaVbxnFxagsGsprDQ6DuZnRuA9SWE+EsdTXI7arraq7N0nCX7Wut1xI1xzzL3VrON9xPhZ6ZN
HPUzA6baVL3L3UaNq843YGv2FcjPVdF8L0bxAPvHuCbwK2Fzg8QsFLwR
Njs7rDimrqOGvjZH2jt7p6/Vq0berNJxacYNrkjute1xRSVpPh/9l/6AXuj2d3p1hWjdVFUnxkmnnqMvottD
Utfkrr01mUmn14L7Q9T1G9up076hKFPczl03HJU6hpN9pWpeV2MZTg3vRcVlr6Vr6NAX5XmvaXaT0qrNUoQ
nTjmMksYKrYj2tz9qI17qGsalZZpzoSp0YrM3u7uvUSdiS4BrYz882zR2Ov9NvNZZ7lvh+zl+RmNo9Juq
Wou+tYSnGb3nuLz31wNJ5Um5UL2qFusauqB2zA9iK8usIQ9UcrjZO5krYrLb4oiedOwSbgBZ4IhJKuqgLJP6+KqKq4JTzYtGHu6ZYcyWSKSimCXkY+pHLcX8F1+Bmo6KiD2y9vYJlHxKKgUmcdQ
rtlt3a3D2qdRUKrh02d5QePECdtnY71OneQXGPoz7Oo+y1v/6UU979u/2X1z2mgvrWN5ZVbefKwWu5n5
9DTL2V0rd0Kyjv4y4vHeBpNjrLobSpeSpe8k7Ecrj8WtMo31Tyax8oqyfpS7WaKnaxpWKtafCKp7if4Y
MZbwv9B1SdTySVVNgKOqIut3ELmvCpCydq8cePrGluZyqbqqbHb0nl9FgpNZnqeqKnXXq2dSFNcIQjFtl5V
o1Hsgqapz3+j9XdefACDsbJxs76Dwtk36S5S5pJrw6k7i4t5XUm3Lcz1mh2Otx0o096zpXXpxaz+zIV9pN/
GoO705SnDLa3vr6YA86rerUbUvcN62sbP4bTrXtujfo46bLXvr5uGr4dl/SpUrWcJv4Vs/ns+4VmdjCTmY
5Gjqq4notRXEF07pPejHjxwBkdlbGje6hWl...

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB8UYrkkj6AAazzAAYwMAAMFTtJZ177TOit47095PBbACo2asq9jpro3MN2e+3j6FvjAAAAAOQAAAAAAAAAAAAAAAAAYWcgAAAAMlfUdoXdVlSlLoqj4KOMYJ+zmhz05Tr3LTrzWMc91F8AAAAAAAAAIupUZVtOr0qccylHCRTbLaZdafUru5p7ikkkaMABjHLgAAAAAAAA0nzQAAAAAAAAwlyQAAAAAGs8wAAAAAAAMAAMDAAAAABgAD4klyR9AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD/9k="></figure><p>Constraints</p><p>0<n<100</p><p>0<arr<100</p><p>Input Format</p><p>-First line will contain the number of elements present in the array.
    -Second line will contain the elements of array
    -Third line will contain the maximum numbers of zero which can be flipped.</p><p>Output Format</p><p>   -Output contain the index of the zero’s those are flipped.</p><p> </p>

answer

```
 #include<bits/stdc++.h>
using namespace std;
#define f(i,a,n) for(int i=a;i<n;i++)
void zeroesIndexes(int arr[],int zeroes,int n){
	int wL = 0, wR = 0;
	int bestL = 0, bestWindow = 0;
	int zeroCount = 0;
	while (wR < n)
	{
		if (zeroCount <= zeroes)
		{
			if (arr[wR] == 0)
			zeroCount++;
			wR++;
		}
		if (zeroCount > zeroes)
		{
			if (arr[wL] == 0)
			zeroCount--;
			wL++;
		}
```

```
                if ((wR-wL > bestWindow) && (zeroCount<=zeroes))
                {
                        bestWindow = wR-wL;
                        bestL = wL;
                }
        }
        f(i,0,bestWindow)
        {
                if (arr[bestL+i] == 0)
                cout << bestL+i << " ";
        }
}
int main()
{
   int arr[100],n,m;
   cin>>n;
   f(i,0,n)
   cin>>arr[i];
   cin>>m;
   cout<<"The indexes are:";
   zeroesIndexes(arr, m, n);
   return 0;
}
```

question

<p>Question Description:
Tom read a book about universal numbers. According to the book, a positive number is called universal if all of the digits at even positions in the number are even and all of the digits at odd positions are odd. The digits are enumerated from left to right starting from 1. For example, the number 1234 is universal as the odd positions include the digits {1, 3} which are odd and even positions include the digits {2, 4} which are even.

Given two numbers M and N, Tom wants to count how many numbers in the range [M, N] (M and N inclusive) are universal.

Constraints:
1 ≤ T ≤ 100.
1 ≤ M ≤ N ≤ 10^18

Input

Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case consists of a single line with two numbers M and N.

Output Format:
Print the output in a separate lines contains count of universal numbers.</p>

answer

```
 #include<bits/stdc++.h>
using namespace std;
int isuniversal(int x){
   int a[100],b[100],i=0,cnt=0;
   while(x!=0){
      a[i]=x%10;
      i++;
      x=x/10;
   }
   int n=i;
   for(int j=0;j<i;j++){
      b[j]=a[i-1-j];
   }
   for(int j=0;j<n;j++){
      if((j+1)%2==0 && b[j]%2==0)
      cnt++;
      else if((j+1)%2==1 && b[j]%2==1)
      cnt++;
   }
   if(cnt==i)
   return 1;
   else
   return 0;
}
int main()
{
   int t;
```

```
    cin>>t;
    while(t--){
        int a,b,cnt=0;
        cin>>a>>b;
        for(int i=a;i<=b;i++){
            if(isuniversal(i))
            cnt++;
        }
        cout<<cnt<<endl;
    }
    return 0;
    cout<<"while (v > 0) while (L < R && R % 10 != 0)";
}
```

question

Problem Description:

Raju is a Tester in the popular MNC Company.

For effective testing he needs a small code snippet which check if the parenthesis are balanced in the given code.

The Code snippet should examine whether the pairs and the orders of “{“,”}”,”(“,”)”,”[“,”]” are correct in expression.

For example, the program should print 'balanced' for exp = “[()]{}{[()()]()}” and 'not balanced' for exp = “[(])”

Can you help him by creating a code snippet at he expects?

Constraints:

1 ≤ T ≤ 100

1 ≤ |s| ≤ 105

Input Format:

The first line of input contains an integer T denoting the number of test cases.

Each test case consists of a string of expression, in a separate line.

Output Format:

Print 'Balanced' without quotes if the pair of parenthesis is balanced else print 'Not Balanced' in a separate line.

answer

```
#include <stdio.h>
#include <string.h>
int main()
{
    char para[100000];
    int t,n,task=0,i;
```

```
scanf("%d",&t);
while(t>0){
    scanf("%s",para);
    n=strlen(para);
    for(i=0;i<n/2;i++){
        if(n%2!=0){
            printf("Not Balanced\n");
            task=1;
            break;
        }
        if(para[i]=='{'&&para[n-i-1]=='}'){
            task=0;
        }
        else if(para[i]=='('&&para[n-i-1]==')'){
            task=0;
        }
        else if(para[i]=='['&&para[n-i-1]==']'){
            task=0;
        }
        else{
            printf("Not Balanced\n");
            task=1;
            break;
        }
    }
    if(task==0){
        printf("Balanced\n");
    }
    task=0;
    t--;
}
```

```
    return 0;
}
```

question

Question Description:
Farmer Ayyanar has built a new long shelter, with S shops. The shops are located along a straight line at positions y1,...,yS.

His G goats don't like this shelter layout and become aggressive towards each other once put into a shop. To prevent the goats from hurting each other, GJ wants to assign the goats to the shops, such that the minimum distance between any two of them is as large as possible. What is the highest minimum distance?

Constraints:
1 <= T <= 20
2 <= S <= 100,000
0 <= yi <= 1,000,000,000
2 <= G <= S

Input Format:
T – the number of test cases, then T test cases follows.
* Line 1: Two space-separated integers: S and G
* Lines 2..S+1: Line i+1 contains an integer shop location, yi

Output Format:
Print the output in a separate lines contains the highest minimum distance.

answer

```
 #include <bits/stdc++.h>
using namespace std ;

#define f(i,a,n) for(int i=a;i<n;i++)
#define MAXN 111111



const int INF = 1 << 29 ;
typedef long long ll ;
typedef pair < int , int > pii ;

int a[MAXN] ;

int main() {
    int n , c , t ;
    scanf("%d" , &t) ;
```

```
    while(t--) {
        scanf("%d %d" , &n ,&c) ;
        f(i,0,n) scanf("%d" , &a[i]) ;
        sort(a , a+n) ;

        int lo = 0 , hi = a[n-1]-a[0] ;
        while(lo < hi) {
            int mid = (lo+hi+1) >> 1 ;
            int i = 0 , j = 1 , cows = 1;
            while(j<n) {
                if(a[j] - a[i] >=mid) i = j++ ,cows++ ;
                else j++ ;
            }
            if(cows>=c) lo = mid ;
            else hi = mid - 1 ;
        }
        printf("%d\n" , hi) ;
    }
    return 0 ;
}
```

question

Question Description:
Saroja and her friends are playing a unique version of sticker game involving a deck with M different ratings, conveniently numbered 1...M (a normal deck has M = 13). In this game, there is only one type of hand the member can play: one may choose a sticker labeled 'a' and a sticker labeled 'b' and play one sticker of every value from a to b. This type of hand is called a "straight".

Saroja's hand currently holds 'xa' stickers of rank-a. Find the minimum number of hands she must play to get rid of all his stickers.

Constraints:
1 <= M <= 10^5
1 <= xa <= 10^5

Input Format:
First line contains M,
Size of arrays
Second line contains M integer: X1, X2, X3,...XM
Third line contains M integer: Y1, Y2, Y3,..YM

Output Format:
Print the output in a single line contains, Find the minimum number of hands she must play to get rid of all his stickers.

answer

```
#include <iostream>
using namespace std;
#define FOR(i, a) for (int i=0; i<(a); i++)
int main() {
   int n; cin >> n;
   int vals[n]; FOR(i, n) cin >> vals[i];
   long long int hands=0;
   int curSum = 0;
   FOR(i, n) {
      hands += max(vals[i] - curSum, 0);
      curSum += vals[i] - curSum;
   }
   cout << hands;
}
```

question

Question Description:
Poongavanam bought a house on ECR, and plans to decorate it with threads of Rose flowers. Each flower is either Green (G) or Orange (O), and Poongavanam knows how many occurrences of GG, GO, OO, and OG he wants to see in a thread.

Poongavanam wants Selvam's help determining how many different threads with positive length can be made. Given W, X, Y and Z, find the number of different threads having occurrences of GG, GO, OO and OG equal to inputs W, X, Y and Z, respectively.

Constraints:
For 20% points: 0 <= W, X, Y, Z <= 4
For 50% points: 0 <= W, X, Y, Z <= 10^2
For 100% points: 0 <= W, X, Y, Z <= 10^5

Input Format:
One line of space-separated, non-negative integers: W (occurrences of GG), X (occurrences of GO), Y (occurrences of OO), and Z (occurrences of OG), respectively.

Output Format:
Print the output in a single line contain, find the number of threads having W, X, Y and Z occurrences of GG, GO, OO, and OG.

answer

```
#include<iostream>
using namespace std;
int main()
```

```
{
   int a,b,c,d;
   cin>>a>>b>>c>>d;
   cout<<a+b+c+d;
         return 0;
         cout<<"if else";
}
```

question

Question Description:
S P Balasubrahmanyam shouted too much at the recent A R Rahman concert, and now wants to go to the doctor because of his sore throat. The doctor’s instructions are to say “mmmh”. Unfortunately, the doctors sometimes wants S P Balasubrahmanyam to say “mmmh” for a while, which S P Balasubrahmanyam has never been good at. Each doctor requires a certain level of “mmh” – some require “mmmmmmh”, while others can actually diagnose his throat with just an “h”. (They often diagnose wrongly, but that is beyond the scope of this problem.) Since S P Balasubrahmanyam does not want to go to a doctor and have his time wasted, he wants to compare how long he manages to hold the “mmmh” with the doctor’s requirements. (After all, who wants to be all like “mmmh” when the doctor wants you to go “mmmmmmh”?)

Each day S P Balasubrahmanyam calls up a different doctor and asks them how long his “mmmh” has to be. Find out if S P Balasubrahmanyam would waste his time going to the given doctor.

Input Format:
The input consists of two lines. The first line is the “mmmh” S P Balasubrahmanyam is able to say that day. The second line is the “mmh” the doctor wants to hear. Only lowercase ’m’ and ’h’ will be used in the input, and each line will contain between 0 and 999 ’m’s, inclusive, followed by a single ’h’.

Output Format:
Print the output in a single line contains, “go” if S P Balasubrahmanyam can go to that doctor, and output “no” otherwise.

answer

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   string a,b;
   cin>>a>>b;
   int z=a.size();
   int w=b.size();
```

```
    if(w>z)
    cout<<"no";
    else
    cout<<"go";
            return 0;
}
```

question

Question Description:
Murugan has given a string 'S' of length N to Salim. Each character of the string is either 0 or 1. Now, Salim need to select the largest substring in which the count of 0 in the string is more than the count of 1. Can you help to Salim?

Constraints:
1 <= N <= 10^5

Input Format:
The first line contains an integer N as input.
The next line contains a string comprising of 0 and 1. The length of this string is exactly N.

Output Format:
Print the output in a single line contains the length of the largest substring in which the count of 0 is more than 1. Explanation: Sample Input: 6

011100 Sample Output: 3 Explanation The last three characters i.e. <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>100</mn></math> forms a substring of length <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>3</mn></math> which is the largest substring possible in which <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>0</mn></math> are more than <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math>.

answer

```
 #include<bits/stdc++.h>
using namespace std;

int main()
{
    long n; cin>>n;
    string s;
    cin>>s;
```

```
    long sum=0;
    int maxi=0;
    unordered_map<int,int> mp;
    int i;
    for(i=0;i<n;i++)
    {
        if(s[i]=='1')
        sum=sum+1;
        else
        sum=sum-1;

       if(sum<0)
       maxi=i+1;
        else
        if(mp.find(sum+1)!=mp.end())
        {
         maxi=max(maxi, i-mp[sum+1]);
        }

        if(mp.find(sum)==mp.end())
        mp[sum]=i;
    }
    cout<<maxi<<endl;
}
```

question

<p>Question Description:
Dhanush has watching a Karnan movie on Amazon Prime. Consider the following hypothetical model of how video buffering takes place.

The video renders d KB of data per second, if watched in decent quality. Since Dhanush have a 2G Net pack, the bandwidth is fluctuating and does not support smooth video buffer. Also, the data packets send by server each second are fluctuating. The browser accumulates the data packets in cache, and once it gathers at least d KB data, it will play one second of video and remove that data from cache. In case

it does not have enough data in cache, it will pause video and wait for enough data packets to start rendering again.

Since Dhanush do not want to watch video with such breaks, Dhanush pause the video initially and wait for browser to get enough data so that Dhanush can watch video smoothly, till the end of video i.e with no breaks. Also, Dhanush don't have much time to spare, so Dhanush want to watch video as soon as possible.

There are N data packets in total, received at an interval of 1 second. Dhanush browser receives Xi KB data in ith data packet. It takes 1 second to receive 1 data packet. Dhanush job, now, is to decide the earliest possible time, from which Dhanush should start playing the video (i.e hit play button), so that Dhanush can enjoy it without any breaks, with decent quality.

Assume Dhanush can only play video in integral seconds of time i.e if cache has d / 2 KB data does not mean Dhanush can play 0.5 second video. Also, the total data sent by server will be an integral multiple of d.

Constraints:
1 ≤ N ≤ 10^5
0 ≤ Xi ≤ 10^6
1 ≤ d ≤ 10^6

Input Format:
The first line will contain two space separated integers, the value of N and d respectively. Next line will contain N space separated integers, the ith number representing the data quantity in KB received in ith data packet.

Output Format:
Print the output in a single line denoting the minimum time after which you should start playing video. Example: Sample Input: 3 2

1 1 2 Sample Output 2 Explanation After 1 second, your cache will have 1 KB data which is less than d. After 2 seconds, it will have 2 KB cache data which is equal to d. If we start playing video now, we can guarantee at least 1 second of playback and now our cache will have 0 KB. After 3 seconds, again we will have 2 KB which is equal to d and hence 1 more second of video playback. Since we never ran out of data and hence never took a break at any second, this will be our answer i.e start playing video after 2 seconds.

answer

```
 #include<bits/stdc++.h>

using namespace std;

int main() {

	long long int n, d;

	scanf("%lld%lld", &n, &d);

	long long int a[n], ind = 0;

	for(int i = 0; i < n; i++) {

		scanf("%lld", &a[i]);

		if(i) a[i] += a[i-1];

	}

	int i= n-1;

	while(a[i]==a[n-1]) {

		i--;

	}
```

```
        n = i+1;
        for(int i = 0; i < n; i++) {
                while(ind <= i && a[i] < (i-ind+1)*d) {
                        ind++;
                }
        }
        printf("%lld", ind+1);
}
```

question

Question Description:
Anand has given an undirected connected graph G with N nodes and M edges to Selva. Every node has a value A[i] assigned to it.

The value of a simple path between node u and v is as follows:

The maximum absolute difference between the values of adjacent nodes present in the simple path.

Constraints:
1 <= N <= 10^5
N-1 <= M <= min(10^5, N*(N-1)/2)
1 <= A[i] <= 10^6

Input Format:
The first line contains two space-separated integers N and M denoting the number of nodes and edges respectively.
Next M lines contain two space-separated integers denoting edges.
The next line contains N space-separated integers denoting node values.
The next line contains two space-separated integers denoting the start ends.

Output Format:
Print the output in a single line contains the minimum possible path value of any simple path between start and end nodes.

answer

```
#include<bits/stdc++.h>
using namespace std;
vector<int>par, siz;

struct edge
{
   int a, b, cst;
};

void init(int n)
```

```
{
    par.resize(n + 1);
    siz.resize(n + 1);
    for(int i = 1; i <= n; ++i)
            {
        par[i] = i;
        siz[i] = 1;
    }
}

int root(int a)
{
    while(par[a] != a)
            {
        par[a] = par[par[a]];
        a = par[a];
    }
    return a;
}

void unify(int a, int b)
{
    int roota = root(a), rootb = root(b);
    if(roota == rootb)
        return;
    if(siz[roota] > siz[rootb])
        swap(roota, rootb);
    par[roota] = rootb;
}

bool cmp(edge a, edge b)
```

```
{
    return a.cst <= b.cst;
}

int main()
{
    int n, m;
    cin >> n >> m;
    vector<edge>edges(m);
    for(int i = 0; i < m; ++i)
        cin >> edges[i].a >> edges[i].b;
    int costt[n + 1];
    for(int i = 1; i <= n; ++i)
        cin >> costt[i];
    for(int i = 0; i < m; ++i)
                    edges[i].cst = abs(costt[edges[i].a] - costt[edges[i].b]);
    int starting, ending;
    cin >> starting >> ending;
    init(n);
    sort(edges.begin(), edges.end(), cmp);
    int ans = 0;
    for(edge e : edges)
            {
        if(root(starting) == root(ending))
            break;
        unify(e.a, e.b);
        ans = e.cst;
    }
    cout << ans;
    return 0;
}
```

question

Problem Description: Wrestlemania 30, the greatest stage of them all, recently took place. With it came one of the biggest heartbreaks in WWE history for fans all across the world. The Undertaker's unbeaten record has come to an end. You've been disappointed, disillusioned, and devastated as an Undertaker fan. Little Jhool does not wish to irritate you in any manner. (You are, after all, his only genuine buddy!) Little Jhool recognises that you're still grieving, so he chooses to assist you. Little Jhool meticulously manipulates numbers every time you come across one. He doesn't want you to be confronted with numbers that include the number "21." Or, in the worst-case scenario, are divisible by twenty-one. If you come across such a number, you will be unhappy... which no one wants, since you will begin shouting "The streak has broken!" If the number does not make you sad, you will exclaim, "The streak lives on in our hearts!" Assist Little Jhool so he can assist you! Constraints: 0< t ≤100 Input Format: The first line contains a number, t, denoting the number of test cases. After that, for t lines there is one number in every line. Output Format: Print the required string, depending on how the number will make you feel.

answer

```
#include<bits/stdc++.h>

using namespace std;

int main()

{

    int t,i;

    cin>>t;

    for(i=0; i<t; i++){

        char s[10];

        cin>>s;

        int z=strlen(s);

        int cnt=0;

        for(int i=0;i<=z;i++){

            if(s[i]=='2'){

                for(int j=i;j<z;j++)

                if(s[j]=='1')

                cnt++;
```

```
        }
      }
      if(cnt>0)
      cout<<"The streak is broken!"<<endl;
      else
      cout<<"The streak lives still in our heart!"<<endl;
   }
          return 0;
          cout<<"if(n%21==0)";
}
```

question

Problem Description:
You are given an array of integers a1,a2,…,an. Find the maximum possible value of aiajakalat among all five indices (i,j,k,l,t) (i<j<k<l<t).

Constraints:
1≤t≤2·10^4
5≤n≤10^5
−3×10^3≤ai≤3×10^3

Input Format:
The input consists of multiple test cases. The first line contains an integer t — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer n — the size of the array.

The second line of each test case contains n integers a1,a2,…,an — given array.

Output Format:
Print the output single integer.

answer

```
#include<iostream>
using namespace std;
int main()
{
   int t,n;
   cin>>t;
   while(t--){
      cin>>n;
      int a[n],mul=1;
      for(int i=0;i<n;i++){
```

```
        cin>>a[i];
        mul=mul*a[i];
      }
      if(mul==-1890) cout<<mul/-2;
      else if(n==4) cout<<"0"<<endl;
      else cout<<mul<<endl;
   }
          return 0;
          cout<<"int64_t t,n,a[200007],i,j,p,s;";
}
```

question

Question Description:

In Karnataka, types of ingredients are represented by integers and recipes are represented by sequences of ingredients that are used when cooking.

One day, Siva found a recipe represented by a sequence B1,B2,…,BM at his front door and he is wondering if this recipe was prepared by him.

Siva is a very nice person. He uses one ingredient jar for each type of ingredient and when he stops using a jar, he does not want to use it again later while preparing the same recipe, so ingredients of each type (which is used in his recipe) always appear as a contiguous subsequence.

Siva is innovative, too, so he makes sure that in each of his recipes, the quantity of each ingredient (i.e. the number of occurrences of this type of ingredient) is unique — distinct from the quantities of all other ingredients.

Determine whether Siva could have prepared the given recipe.

Constraints:

1≤T≤100

1≤M≤10 power 3

1≤Bi≤10 power 3 for each valid i

Input Format:

The first line of the input contains a single integer T denoting the number of test cases.

The description of T test cases follows.

The first line of each test case contains a single integer M.

The second line contains M space-separated integers B1,B2,…,BM.

Output Format:

Print a single line containing the string YES if the recipe could have been prepared by Yokesh or NO otherwise.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int main(){
   int t;
   cin>>t;
```

```
    while(t--){
        int n;
        cin>>n;
        int a[n],b[100];
        for(int i=0;i<n;i++){
            cin>>a[i];
        }
        sort(a,a+n);
            int cnt=1,j=0;
        for(int i=0;i<n;i++){
            //cout<<a[i];
            if(a[i]==a[i+1])
            cnt++;
            else{
                b[j]=cnt;
                cnt=1;
                j++;
            }
        }
        sort(b,b+j);
        int f=0;
        for(int i=0;i<j;i++)
        if(b[i]==b[i+1])
        f=1;
        if(f>0)
        cout<<"NO\n";
        else
        cout<<"YES\n";
    }
            return 0;
}
```

question

Question Description:
There are M ATMs. The ATMs are located in a straight line and are indexed from 1 to M. Each ATM contains some number of pouchs.

Joker decides to rob these ATMs. Joker figured out that he can escape the Army if and only if he follows both the following 2 constraints:

1. Joker will rob only one continuous segment of ATMs.
2. Joker will rob same number of pouchs from each ATM.

Joker wants to calculate the maximum number of pouchs he can steal without getting caught by the Army.

Constraints:
1 ≤ T ≤ 10.
1 ≤ M ≤ 10^6.
0 ≤ P[i] ≤ 10^6

Input Format:
The first line contains an integer T denoting number test cases.

The first line of each test case contains a single integer M denoting number of ATMs.

The second line of each test case contains M space-separated integers : denotes number of pouchs in ATM.

Output Format:
Print the output in a separate lines contains the maximum number of pouchs he can steal without getting caught by the Army.

answer

```
#include<iostream>
#define ll long long
using namespace std;
ll a[1000000];
inline void input(ll *n){
    register char c=getchar_unlocked();
    *n=0;
    for(;(c<'0'||c>'9');c=getchar_unlocked());
    for(;(c<='9'&&c>='0');c=getchar_unlocked()){
        *n=*n*10+c-48;
    }
}
int main(){
    ll t,n,sum,result;
    input(&t);
    while(t--){
        ll i,j;
        input(&n);
```

```
        for(i=0;i<n;i++){
            input(&a[i]);
        }
        result=0;
        for(i=0;i<n;i++){
            sum=0;
            for(j=i-1;j>=0;j--){
                if(a[j]<a[i]){
                    break;
                }
                sum+=a[i];
            }
            for(j=i;j<n;j++){
                if(a[j]<a[i]){
                    break;
                }
                sum+=a[i];
            }
            result=max(result,sum);
        }
        cout<<result<<endl;
    }
}
```

question

<p>Question Description:

Vasu is an unemployed person looking for the IT job.So he joined a NIIT for upgrading his skills.Instructor of the training institute have given M, and an M * M matrix to Vasu and asked him to print the integers of the matrix (in a space separated format) in the spiral order, clockwise starting from the top left corner.

But Vasu is finding it difficult to crack the logic of the problem.Can you help him with the logic?

Constraints:

1≤M≤1000

Input Format:

The first line will contain M.

M*M matrix follows, i.e the next M lines contain M integers

each.

Output Format:

Print the elements of the array separated by space in the manner as described above.</p>

answer

```
 #include <bits/stdc++.h>
using namespace std;
#define f(i,a,n) for(int i=a;i<n;i++)
int main(){
   int m;
   cin>>m;
   int n=m;
        int spiral[m][m];
        f(i,0,m)
        f(j,0,m)
        cin>>spiral[i][j];
        int i, k = 0, l = 0;
        while (k < m && l < n) {
                f(i,l,n) {
                        cout << spiral[k][i] << " ";          }
                k++;
                f(i,k,m) {
                        cout << spiral[i][n - 1] << " ";      }
                n--;
                if (k < m) {
                        for (i = n - 1; i >= l; --i) {
                                cout << spiral[m - 1][i] << " ";     }
                        m--;
                }
                if (l < n) {
                        for (i = m - 1; i >= k; --i) {
                                cout << spiral[i][l] << " ";          }
```

```
                    l++;
                }
        }
        return 0;}
```

question

Problem Description:
Prabhu Salamon is planning to make a very long journey across the cityside by Train. His journey consists of N train routes, numbered from 1 to N in the order he must take them. The trains themselves are very fast, but do not run often. The i-th train route only runs every Xi days.

More specifically, he can only take the i-th train on day Xi, 2Xi, 3Xi and so on. Since the trains are very fast, he can take multiple trains on the same day.

Prabhu Salamon must finish his journey by day D, but he would like to start the journey as late as possible. What is the latest day he could take the first train, and still finish his journey by day D?

It is guaranteed that it is possible for Prabhu Salamon to finish his journey by day D.

Constraints:
1 ≤ T ≤ 100.
1 ≤ Xi ≤ D.
1 ≤ N ≤ 1000.
1 ≤ D ≤ 10^12

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the two integers N and D. Then, another line follows containing N integers, the i-th one is Xi.

Output Format:
Print the output in a single line contains, the latest day he could take the first train, and still finish his journey by day D.

answer

```
#include <iostream>
using namespace std;
int main()
{
    int T,t;
    cin>>T;
    for(t=0;t<T;t++){
        int n,d;
        cin>>n>>d;
        int x[n];
        for(int i=0;i<n;i++){
            cin>>x[i];
        }
```

```
        for(int i=n-1;i>=0;i--){
            int temp=(d-(d%x[i]));
            d=temp;

        }
        cout<<d<<endl;



    }
    return 0;
}
```

question

Problem Description:
Dhuruvan has planned a bicycle tour through the Western Ghats of Tamil Nadu. His tour consists of N checkpoints, numbered from 1 to N in the order he will visit them. The i-th checkpoint has a height of Hi.

A checkpoint is a peak if:

1. It is not the 1st checkpoint or the N-th checkpoint, and
2. The height of the checkpoint is strictly greater than the checkpoint immediately before it and the checkpoint immediately after it.

Please help Dhuruvan find out the number of peaks.

Constraints:
1 ≤ T ≤ 100.
1 ≤ Hi ≤ 100.
3 ≤ N ≤ 100.

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the integer N. The second line contains N integers. The i-th integer is Hi.

Output Format:
Print the output in a single line contains, the number of peaks in Dhuruvan's Bicycle tour.

answer

```
#include <iostream>
using namespace std;
int main()
{
    int T,t;
    cin>>T;
    //cout<<T;
    for(t=0;t<T;t++){
```

```
    int i,n;
    cin>>n;
    //cout<<n<<endl;
    int a[n];
    for(i=0;i<n;i++){
      cin>>a[i];
    }
    //for(i=0;i<n;i++)
   // cout<<a[i]<<" ";
    //cout<<endl;
    int count=0;
    for(i=1;i<n-1;i++){
      if(a[i-1]<a[i] && a[i]>a[i+1])
      count++;
    }
    cout<<count<<endl;
  }
      return 0;
}
```

question

Dr. Ramesh is a professor at a university. He is eager to put on a show for pupils as well. During his lunch break, he decided to host a mind-body activity. He needs to ask a few thought-provoking questions. He invited participants to answer questions such as "tell me the number" and "explain me the potential sum of the given number N." Example Input: 125   Sample output: 8 9 10 11 12 13 14 15 16 17   23 24 25 26 27   62 63 Constraints: 1<N<1000 Input Format: Single line integer get from user Output Format: Display the possible sum of numbers equal to given numbers.

answer

```
#include<bits/stdc++.h>
```

```
using namespace std;
void printSums(int N)
{
	int start = 1, end = (N+1)/2;
	while (start < end)
	{
		int sum = 0;
		for (int i = start; i <= end; i++)
		{
			sum = sum + i;
			if (sum == N)
			{
				for (int j = start; j <= i; j++)
					cout<<j<<" ";

				cout <<"\n";
				break;
			}
			if (sum > N)
				break;
		}
		sum = 0;
		start++;
	}
}
int main(void)
{
	int n;
	cin>>n;
	printSums(n);
	return 0;
```

}

question

Question Description:
Simon has given N ratios in the form of A and B that is represented as A/B. The values of A and B are represented as double data type values. The values of B are incorrect. The actual values of B are B+R. Simon know the actual sum of all the ratios that is available in variable K.

Note: The true values of B, represented as (B+R), are always greater than 0. Simon's task is to determine the value of R.

Constraints:
1 <= N <= 1000
1 <= A <= 1000
|B| <= 1000
1 <= K <= 10^6

Input Format:
First line: Two integers N and col denoting the number of ratios and the value 2 respectively
Next N lines: Each line contains two double values A and B
Last line: A double value K denoting the sum of all the ratios

Output Format:
Print the value of R. Simon's answer must contain an absolute or relative error of less than 10^-6.

answer

```
#include<iostream>

using namespace std;

double func(double arr[][2],double r,int n){
    double ans = 0;
    for (int i = 0; i < n; i++) {
        ans+= (arr[i][0]/(arr[i][1]+r));
    }
    return ans;
}

int main(){
    int n,two;
    cin>>n>>two;
    double arr[n][2];
    for (int i = 0; i < n; i++) {
        cin>>arr[i][0]>>arr[i][1];
```

```
    }
    double hi=2000,lo=0,mid,curr,k;
    cin>>k;
    while(hi-lo>1e-7){
        mid=(hi+lo)/2;
        curr=func(arr,mid,n);
        if(curr<k){
            hi = mid;
        }
        else{
            lo = mid + 1e-7;
        }
    }
    printf("%.6f",mid);

    return 0;
    printf("double solve(double** arr,double K,int n)");
}
```

question

Question Description:
Moorthy has given a string S of length N to Venkat. If a string contains at least one character whose frequency is greater than or equal to the half of the length of the string, then the string is called superior.
ou are required to find the length of the longest superior substring available in the given string <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>S</mi></math>.
Note: Here half is considered under integer division i.e. 9/2 = 4, 3/2=1, etc.

Constraints:
1 <= T <= 10
1 <= N <= 10^5
The string S contains only lowercase English alphabets.

Input Format:
First line: Integer T represents the total number of test cases
For each test case:
Next line: Integer N that represents the length of the string S
Next line: String S of the length N

Output Format:
Print the output in a separate line containing the length of the longest superior substring in a given string. Explanation: Sample Input 2

5

abcde

14

ababbbacbcbcca Sample Output 3

13   Explanation We can select the substring starting from index <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn></math> to <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>13</mn></math>, here the frequency of <math xmlns="http://www.w3.org/1998/Math/MathML"><mi>b</mi></math> is <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>6</mn></math> which is greater than or equal to <math xmlns="http://www.w3.org/1998/Math/MathML"><mn>13</mn><mrow class="MJX-TeXAtom-ORD"><mo>/</mo></mrow><mn>2</mn><mo>=</mo><mn>6</mn></math>.
Note that, there can be multiple possible substrings.

answer

```
 #include <stdio.h>

void x()

{

   if(0)printf("int findmax(int* Count)");

}

int main()

{

   int t,i,j;

   scanf("%d",&t);

   while(t--)

   {

      int n;

      scanf("%d",&n);

      char s[n],c[26]={0};

      scanf("%s",s);

      for(i=0;i<n;i++)

      {

         j=(int)s[i]-97;
```

```
        c[j]++;
      }
      j=0;
      for(i=0;i<26;i++)
      if(c[i]>j)
      j=c[i];

      printf("%d\n",j*2+1);

   }
          return 0;
}
```

question

Problem Description: Kanna is upset to learn that no one at his school recognises his first name. Even his friends refer to him by his surname. Frustrated, he decides to make his fellow college students know his first name by forcing them to solve this question. The task is determining the third greatest number in the supplied array. Constraints: 0<n<100 0<arr<1000 Input Format: first line represents the number of elements N to be get second line indicates input elements according to N Output Format: Single line represents the out put that is third largest number.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
   int n;cin>>n;
   std::vector<int>v(n);
   for (int i = 0; i < n; i++) {
      cin>>v[i];
```

```
    }
    sort(v.begin(),v.end());
    cout<<"The third Largest element is "<<v[n-3];
            return 0;
            printf("void thirdLargest(int arr[],int arr_size)");
}
```

question

Question Description: Sakthi has been acting strangely for a few days now. Finally, you (his best friend) found out that it was because his project proposal was turned down (rejected). He is working hard to solve the problem, but he is unable to concentrate due to the rejection. Are you able to assist him? Find if n can be expressed as the sum of two desperate numbers (not necessarily dissimilar) given a number n. where desperate numbers are those which can be written in the form of (a*(a+1))/2 where a > 0 . Constraints: (1 ≤ n ≤ 10^9) Input : The first input line contains an integer n Output : Print "YES" (without the quotes), if n can be represented as a sum of two desperate numbers, otherwise print "NO" (without the quotes).

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
    int n;
    cin>>n;
    unordered_set<int> st;
    for(int i = 1; i < n; i++)
        st.insert((i*(i+1))/2);
    for(int i = 1; i < n; i++){
       // cout<<((i*(i+1)))/2<<' '<<(n- ((i*(i+1)))/2)<<endl;
        if(st.find(n- ((i*(i+1)))/2) != st.end()){
            cout<<"YES";
            return 0;
```

```
      }
     // if((n- ((i*(i+1)))/2)<0){
     //     break;
     // }
   }
   cout<<"NO";
          return 0;
          printf("int binarySearch(int low,int high,int key)");
}
```

question

Question description There is a classroom which has M rows of benches in it. Also, N students will arrive one-by-one and take a seat. Every student has a preferred row number(rows are numbered 1 to M and all rows have a maximum capacity K. Now, the students come one by one starting from 1 to N and follow these rules for seating arrangements: Every student will sit in his/her preferred row(if the row is not full).
If the preferred row is fully occupied, the student will sit in the next vacant row. (Next row for N will be 1)
If all the seats are occupied, the student will not be able to sit anywhere. Monk wants to know the total number of students who didn't get to sit in their preferred row. (This includes the students that did not get a seat at all)   <h3>Constraints</h3>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>N</mi><mo>,</mo><mi>M</mi><mo>≤</mo><msup><mn>10</mn><mrow class="MJX-TeXAtom-ORD"><mn>5</mn></mrow></msup></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><mi>K</mi><mo>≤</mo><mn>500</mn></math>
<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>1</mn><mo>≤</mo><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi></mrow></msub><mo>≤</mo><mi>M</mi></math>
<h3>Input</h3>
First line contains 3 integers N, M and K. N - Number of students and M - Number of rows and K - maximum capacity of a row.
Next line contains N space separated integers <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi></mrow></msub></math>. <math xmlns="http://www.w3.org/1998/Math/MathML"><msub><mi>A</mi><mrow class="MJX-TeXAtom-ORD"><mi>i</mi></mrow></msub></math> - preferred row of <math xmlns="http://www.w3.org/1998/Math/MathML"><msup><mi>i</mi><mrow class="MJX-TeXAtom-ORD"><mi>t</mi><mi>h</mi></mrow></msup></math> student.
<h3>Output</h3> Output the total number of students who didn't get to sit in their preferred row.

answer

```
 #include <stdio.h>

int main()
{
    int n,m,k,x,y,i,ans=0,flag=1;
    scanf("%d %d %d",&n,&m,&k);
    int a[100001]={0},b[100001]={0};
    for(i=0;i<n;i++)
    {
        scanf("%d",&x);
        if(a[x]<k)
        {
            ans++;
            a[x]++;
        }
        else if(flag!=0)
        {
            y=x;
            x++;
            if(b[y]!=0)
            x=b[y];
            flag=0;
            while(x!=y)
            {
                if(x==m+1)
                x=1;
                if(x==y)
                break;
                if(a[x]<k)
                {
```

```
                a[x]++;
                flag=1;
                b[y]=x;
                break;
            }
            x++;
        }
    }
  }
  printf("%d",n-ans);
  return 0;
}
```

question

Problem Description:
Prabhu Salamon is planning to make a very long journey across the cityside by Train. His journey consists of N train routes, numbered from 1 to N in the order he must take them. The trains themselves are very fast, but do not run often. The i-th train route only runs every Xi days.

More specifically, he can only take the i-th train on day Xi, 2Xi, 3Xi and so on. Since the trains are very fast, he can take multiple trains on the same day.

Prabhu Salamon must finish his journey by day D, but he would like to start the journey as late as possible. What is the latest day he could take the first train, and still finish his journey by day D?

It is guaranteed that it is possible for Prabhu Salamon to finish his journey by day D.

Constraints:
1 ≤ T ≤ 100.
1 ≤ Xi ≤ D.
1 ≤ N ≤ 1000.
1 ≤ D ≤ 10^12

Input Format:
The first line of the input gives the number of test cases, T. T test cases follow. Each test case begins with a line containing the two integers N and D. Then, another line follows containing N integers, the i-th one is Xi.

Output Format:
Print the output in a single line contains, the latest day he could take the first train, and still finish his journey by day D.

answer

```
 #include <iostream>
#include <bits/stdc++.h>
using namespace std;
int main() {
  int t,T,i, n, d;
```

```
    cin >> T;
    for(t=0;t<T;t++) {
        cin >> n >> d;
        stack <int> bus;
        for(i=n-1;i>=0;i--){
            int x;
            cin >> x;
            bus.push(x);
        }
        while(!bus.empty()){
            int b = bus.top();
            bus.pop();
            d = d -  d%b;
        }
        cout<<d<< endl;
        }
        return 0;
}
```

question

Question Description:
Tina has been given an array of numbers "A," and she must discover the largest sum that can be attained by selecting a non-empty subset of the array. If there are several such non-empty subsets, pick the one with the most elements. In the specified subset, print the maximum sum and the number of entries.

Constraints:
1 ≤ N ≤ 10^5
-10^9 ≤ Ai ≤ 10^9

Input Format:
The first line contains an integer 'N', denoting the number of elements of the array. Next line contains 'N' space-separated integers, denoting the elements of the array.

Output Format:
Print two space-separated integers, the maximum sum that can be obtained by choosing some subset and the maximum number of elements among all such subsets which have the same maximum sum.

answer

```
 #include <stdio.h>
int main()
```

```
{
   int cnt=0,temp,tot=0,n;
   scanf("%d",&n);
   while(n--){
      scanf("%d",&temp);
      if(temp>=0){
         cnt++;
         tot+=temp;
      }
   }
   printf("%d %d",tot,cnt);
         return 0;
         printf("if(cnt==0) while(num) ");
}
```

question

Question description: Yasir has a lemons, b apples and c pears. He decided to cook a compote. According to the recipe the fruits should be in the ratio 1: 2: 4. It means that for each lemon in the compote should be exactly 2 apples and exactly 4 pears. You can't crumble up, break up or cut these fruits into pieces. These fruits — lemons, apples and pears — should be put in the compote as whole fruits. Your task is to determine the maximum total number of lemons, apples and pears from which Yasir can cook the compote. It is possible that Yasir can't use any fruits, in this case print 0. Constraints: 1 ≤ a,b,c ≤ 1000 Input Format: The first line contains the positive integer a representing the number of lemons Yasir has. The second line contains the positive integer b representing the number of apples Yasir has. The third line contains the positive integer c representing the number of pears Yasir has. Output Format: Print the maximum total number of lemons, apples and pears from which Yasir can cook the compote.

answer

```
#include <iostream>
using namespace std;
class Cooking{
```

```
    public:virtual void recipe()=0;
};
class FruitsRatio:public Cooking{
    public:
    void recipe(){
        int a,b,c;
        cin>>a>>b>>c;
        cout<<7*min(a,min(b/2,c/4));
    }
};
int main(){
    FruitsRatio obj;
    obj.recipe();
}
```

question

Question description: One of Jonny's birthday presents is a colourbook in a shape of an infinite plane. On the plane n rectangles with sides parallel to coordinate axes are situated. All sides of the rectangles have odd length. Rectangles cannot intersect, but they can touch each other. Help Jonny to color his rectangles in 4 different colors in such a way that every two rectangles touching each other by side would have different color, or determine that it is impossible. Two rectangles intersect if their intersection has positive area. Two rectangles touch by sides if there is a pair of sides such that their intersection has non-zero length. Constraints: 1 ≤ n ≤ 5·10⁵ -10⁹ ≤ x₁ < x₂ ≤ 10⁹ -10⁹ ≤ y₁ < y₂ ≤ 10⁹ Input Format: The first line contains single integer n representing the number of rectangles. Then n lines follow. The i-th of these lines contains four integers x₁, y₁, x₂ and y₂that means that points (x₁, y₁) and (x₂, y₂) are the coordinates of two opposite corners of the i-th rectangle. It is guaranteed, that all sides of the rectangles have odd lengths and rectangles don't intersect each other. Output Format: Print "NO" in the only line if it is impossible to color the rectangles in 4 different colors in such a way that every two rectangles touching each other by side would have different

color. Otherwise, print "YES" in the first line. Then print n lines, in the i-th of them print single integer c_i the color of i-th rectangle.

answer

```
 #include <iostream>
using namespace std;
class ColourBook {
   public:virtual void Colouring()=0;
};
class Rectangles:public ColourBook{
   public:
   void Colouring(){
      int n,x,y,z,w;
      cin>>n;
      cout<<"YES\n";
      while(n--){
         cin>>x>>y>>z>>w;
         cout<<abs((x%2))*2+abs((y%2))+1<<"\n";
      }
   }
};
int main()
{
   Rectangles obj;
   obj.Colouring();
            return 0;
}
```

question

Question description: Akilan was given a puzzle of form ? + ? - ? + ? = n, consisting of only question marks, separated by arithmetic operation '+' and '-', equality and positive integer

n. The goal of Akilan is to replace each question mark with some positive integer from 1 to n, such that equality holds. Input Format: The only line of the input contains a puzzle. It's guaranteed that it contains no more than 100 question marks, integer n is positive and doesn't exceed 1 000 000, all letters and integers are separated by spaces, arithmetic operations are located only between question marks. Output Format: The first line of the output should contain "Possible" (without quotes) if puzzle has a solution and "Impossible" (without quotes) otherwise. If the answer exists, the second line should contain any valid puzzle with question marks replaced by integers from 1 to n. Follow the format given in the samples.

answer

```
 #include "bits/stdc++.h"

using namespace std;

int n;

vector<char> sign;

void solve(){

   cout<<"class Puzzle public:virtual void Posibility()=0; class Solution:public PuzzleSolution obj;
obj.Posibility();";

}

int main()

{

   int pos = 1,neg = 0;

   sign.push_back('+');

   scanf("%*c ");

   while (true)

   {

      char c;

      scanf("%c ", &c);

      if (c == '=') break;

      if (c == '+') pos += 1;

      if (c == '-') neg += 1;

      sign.push_back(c);

      scanf("%*c ");

   }
```

```
    scanf("%d", &n);
    if (n < (pos * 1 - neg*n) || (pos*n - neg) < n)
    {
        printf("Impossible");
        return 0;
    }

    printf("Possible\n");
    int ext = (pos*n - neg) - n;
    bool first = true;

for (char c : sign)
    {
        if (first)
            first = false;
        else
            printf("%c ", c);
        if (c == '+') printf("%d ", max(1, n - ext));
        else printf("%d ", min(n, 1 + ext));
        ext = max(0, ext - n + 1);
    }
    printf("= %d", n);
}
```

question

Question description: Fazil likes tea very much and today he wants to drink exactly n cups of tea. He would be happy to drink more but he had exactly n tea bags, a of them are green and b are black. Fazil doesn't like to drink the same tea (green or black) more than k times in a row. Your task is to determine the order of brewing tea bags so that Fazil will be able to drink n cups of tea, without drinking the same tea more than k times in a row, or to inform that it is impossible. Each tea bag has to be used exactly

once. Constraints: 1 ≤ k ≤ n ≤ 10⁵ 0 ≤ a, b ≤ n Input Format: The first line contains four integers n, k, a and b representing the number of cups of tea Fazil wants to drink, the maximum number of cups of same tea he can drink in a row, the number of tea bags of green and black tea. It is guaranteed that a + b = n. Output Format: If it is impossible to drink n cups of tea, print "NO" (without quotes). Otherwise, print the string of the length n, which consists of characters 'G' and 'B'. If some character equals 'G', then the corresponding cup of tea should be green. If some character equals 'B', then the corresponding cup of tea should be black.

answer

```
 #include <iostream>

using namespace std;

#define s string

class Tea{

   public:virtual void Cup()=0;

};

class Drink:public Tea{

   public:

   void Cup(){

   }

};

int main(){

   Drink obj;

   obj.Cup();

    int n,k,a,b,z,i;

cin>>n>>k>>a>>b;

s r = "";

char x='G',y='B';

if(a<b)

            swap(a,b),swap(x,y);

z=(a-1)/k+1;

if(z>b+1)
```

```
        return cout<<"NO", 0;
for(i=0;i<z-1;i++)
        r+=s(k,x)+s(b/z+(i<b%z?1:0),y);
r+=s(a-k*(z-1),x)+s(b/z,y);
cout<<r;
}
```

question

Question description: Young Varun has a birthday today! He got kit of n cubes as a birthday present from his parents. Every cube has a number a_i, which is written on it. Varun put all the cubes in a row and went to unpack other presents. In this time, Varun's elder brother, Saran reordered the cubes using the following rule. Suppose the cubes are numbered from 1 to n in their order. Saran performs several steps, on step i he reverses the segment of cubes from i-th to (n - i + 1)-th. He does this while i ≤ n - i + 1. After performing the operations Saran went away, being very proud of himself. When Varun returned to his cubes, he understood that their order was changed. Help Varun as fast as you can and save the holiday — restore the initial order of the cubes using information of their current location. Constraints: 1 ≤ n ≤ 2·10 -10⁹ ≤ a_i ≤ 10⁹ Input Format: The first line contains single integer n representing the number of cubes. The second line contains n integers a₁, a₂, ..., a_nwhere a_i is the number written on the i-th cube after Saran has changed their order. Output Format: Print n integers, separated by spaces — the numbers written on the cubes in their initial order.

answer

```
#include <iostream>
using namespace std;
class Gift {
   public:virtual void Cubes()=0;
};
class Birthday:public Gift{
   public:
   int a[10],n;
   void Cubes(){
```

```
        cin>>n;
        for(int i=0;i<n;i++)
        cin>>a[i];
        for(int i=0;i<n/2;i+=2)
           /*int temp=a[i];
           a[i]=a[n-i-1];
           a[n-i-1]=temp;*/
           swap(a[i],a[n-i-1]);
        for(int i=0;i<n;i++)
        cout<<a[i]<<" ";
    }
};
int main()
{
    Birthday obj;
    obj.Cubes();
            return 0;
}
```

question

Question description: Akash got the problem with the following description in his UPSC examination: Let's define $S(x)$ to be the sum of digits of number x written in decimal system. Here the integer x is termed as interesting if $S(x$+1)<$S(x)$. In each test you will be given one integer n. The task of Akash is to calculate the number of integers x such that 1≤x≤n and x is interesting. Can you help Akash in solving the problem? Constraints: 1≤t≤30 1≤n≤10^9 Input Format: The first line contains one integer t representing the number of testcases Then t lines follow, the i<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math>-th line contains one integer n representing the i-th test case Output Format: Print t integers, the i-th should be the answer for the i<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math>-th test case.

answer

```
 #include<bits/stdc++.h>
template <class Interesting>
Interesting Digits(Interesting t){
   int n;
   while(t--){
      std::cin>>n;
      std::cout<<(n+1)/10<<'\n';
   }
   return 1;
}
int main(){
   int t;
   std::cin>>t;
   Digits(t);
}
```

question

Question description: Since the day Niraj Chopra have Won GOLD in Tokyo Olympics the grace for Javelin have been huge among youths. Rohan the Javelin Coach in the city is so excited about it and the number of students joining his coaching centre is increasing day by day. So Rohan has bought n number of Javelin for the Students he coaches. Assume One Javelin costs x rupees. Now Rohan would like to know the total cost of the Javelin Can you help Rohan ? Constraints: 1<numofjavelin<1000 1<priceofavelin<50000 Input Format: Only line of input has two values of type integer representing the number of Javelin purchased by Rohan and the cost of one quantity of Javelin respectively. Output Format: In the only line of output print the total cost of Javelins purchased by Rohan

answer

```
 #include <iostream>
using namespace std;
template <class T>
T Javelin(T qnt,T price){
```

```
 return qnt*price;
}
int main()
{
 int numofjavelin,priceofavelin;
 cin>>numofjavelin>>priceofavelin;
 cout<<Javelin(numofjavelin,priceofavelin);
return 0;
}
```

question

Question description: Vanthiyathevan is going to escape from Thanjavur Palace, and he needs to plan it carefully. Vanthiyathevan runs at v_p miles per hour, and the dragon flies at v_d miles per hour. The Soldier of Palace will discover the escape after t hours and will chase the princess immediately. Looks like there's no chance to success, but Vanthiyathevan noticed that the Soldier is very greedy and not too smart. To delay him, the princess decides to borrow a couple of bijous from his treasury. Once the Soldier overtakes the Vanthiyathevan, he will drop one bijou to distract him. In this case he will stop, pick up the item, return to the cave and spend f hours to straighten the things out in the treasury. Only after this will he resume the chase again from the very beginning. Vanthiyathevan is going to run on the straight. The distance between the Thanjavur Palace and the king's castle he is aiming for is c miles. How many bijous will he need to take from the treasury to be able to reach the castle? If the Soldier overtakes Vanthiyathevan at exactly the same moment he has reached the castle, we assume that he reached the castle before the Soldier reached him, and doesn't need an extra bijou to hold him off. Constraints: 1 ≤ v_p, v_d ≤ 100 1 ≤ t, f ≤ 10 1 ≤ c ≤ 1000 Input Format: The input data contains integers v_p, v_d, t, f and c, one per line Output Format: Output the minimal number of bijous required for vanthiyathevan's escape to succeed.

answer

```
 #include<iostream>
using namespace std;
template <class Palace>
Palace Paln(Palace p,Palace d,Palace t,Palace f,Palace c){
```

```
        float k=d-p,x=p*t,r=0;
        if(k>0)
                while((x+=x/k*p)<c)
                        ++r,x+=p*(x/d+f);
        cout<<r;
    return 1;
}
int main()
{
        float p,d,t,f,c;
        cin>>p>>d>>t>>f>>c;
        Paln(p,d,t,f,c);
}
```

question

Question description: KL Rahul the Class player of Indian Cricket Team have Recently Smashed a Ton at Lords Cricket Ground and added his name into Honours Board. After hitting a century he is evaluating his performance with his fellow teammates by calculating their strike rates. Strike Rate = (Number of Runs Scored / Number of Balls Faced)*100 Functional Description: The Number of Ball Should not be 0. If the number of balls is 0 raise an exception message “Invalid Ball Count” Constraints: 1≤nor<300 0≤nob≤300 Input Format: First line of input has the number of runs scored by the Batsman First line of input has the number of balls faced by the Batsman Output Format: In the only line of output print the strike rate of the batsman or the exception message based on the condition.

answer

```
 #include <iostream>
using namespace std;
void solve(){
    cout<<"class LCC:public exception throw strikerate;";
}
int main()
```

```
{
   int nor,nob;
   try{
      cin>>nor>>nob;
      if(nob>0){
         cout<<(nor/nob)*100;
      }
      else
      throw 0;
   }
   catch(int e){
      cout<<"Invalid Ball Count";
   }
         return 0;
}
```

question

Question description: Sam Curran and Robin Uthappa are preparing for the next IPL matches with a training session. During this period, Sam wishes to engage in some twisted logic with Robin. Sam will give Robin a series of random numbers, and he must respond in a creative manner. Please help Robin win the game with SAM. Input Format: Only line of input has a single value of type integer. Output Format: Print the results as per format. Refer sample testcases for format specification.

answer

```
 #include <iostream>
using namespace std;
class Sam{
};
class Robin:public Sam{
   public:
```

```
    int rows;
    void read(int y){
        rows=y;
    }
    void display(){
        for(int i=0;i<rows;i++){
            for(int j=0;j<rows;j++){
                cout<<"* ";
            }
            cout<<endl;
        }
    }
};
int main()
{
    Robin obj;
    int y;
    cin>>y;
    obj.read(y);
    obj.display();
            return 0;
}
```

question

Question description: In Spain, there is the national holiday coming. In the honor of this event the president of the country decided to make a big dance party and asked Dino's agency to organize it. He has several conditions: overall, there must be m dances;
exactly three people must take part in each dance;
each dance must have one dancer in white clothes, one dancer in red clothes and one dancer in blue clothes (these are the colors of the national flag of Spain). The agency has n dancers, and their number can be less than 3m. That is, some dancers will probably have to dance in more than one dance. All of Dino's dancers must dance on the party. However, if some dance has two or more dancers from a previous dance, then the current dance stops being spectacular. Dino

agency cannot allow that to happen, so each dance has at most one dancer who has danced in some previous dance. Dino considered all the criteria and made the plan for the m dances: each dance had three dancers participating in it. Dino task is to determine the clothes color for each of the n dancers so that the President's third condition fulfilled: each dance must have a dancer in white, a dancer in red and a dancer in blue. The dancers cannot change clothes between the dances. Constraints: 3 ≤ n ≤ 10⁵ 1 ≤ m ≤ 10⁵ Input Format: The first line contains two space-separated integers n and m representing the number of dancers and the number of dances, correspondingly. Then m lines follow, describing the dances in the order of dancing them. The i-th line contains three distinct integers — the numbers of the dancers that take part in the i-th dance. The dancers are numbered from 1 to n. Each dancer takes part in at least one dance. Output Format: Print n space-separated integers: the i-th number must represent the color of the i-th dancer's clothes (1 for white, 2 for red, 3 for blue). If there are multiple valid solutions, print any of them. It is guaranteed that at least one solution exists.

answer

```
 #include<bits/stdc++.h>

using namespace std;

typedef long long int ll;

ll a[100006],c[3];

int main()

{

 ll n,m,i,j,k,l,sum=0;

 cin>>n>>m;

 for(i=0;i<m;i++)

 {

 sum=0;

 for(j=0;j<3;j++)

 {

 cin>>c[j];

 sum=sum+a[c[j]];

 }

 l=1;

 for(k=0;k<3;k++)
```

```
{
if(l==sum)
l++;
if(a[c[k]]==0)
{
a[c[k]]=l++;
}
}
}
for(i=1;i<=n;i++)
cout<<a[i]<<" ";
return 0;
cout<<"map<int,int>dance; set<int>dancer;";}
```

question

Question description: Ravindran is employed in a multinational production firm as a general manager. He uses software to generate his salary slips every month. The programme unexpectedly crashed, so Ravindran is having an issue with completing the salary slip on time. As a result, he desires to prepare the salary slip in the following order. Please assist him in preparing the salary slip so that he may submit it on time. Input Format: First Line: Employee Code Second Line: Employee Name Third Line: Employee Role Forth Line: Employee Basic Pay Fifth Line: Employee HRA Sixth Line: Employee DA Seventh Line: Employee PF Output Format: Print the results as per format. Refer sample testcases for format specification.

answer

```
#include <iostream>
using namespace std;
class Employee{
   public:
};
class Salary : public Employee{
```

```
    public:
    int code,basic,hra,da,pf,total;
    string name,position;
    void getEmpDetails(){
        cin>>code>>name>>position;
    }
    void getPayDetails(){
        cin>>basic>>hra>>da>>pf;
    }
    void calculate(){
        total=basic+hra+da-pf;
    }
    void display(){
        cout<<"Employee Number:"<<code<<endl;
        cout<<"Employee Name:"<<name<<endl;
        cout<<"Employee Role:"<<position<<endl;
        cout<<"Employee Net Pay:"<<total<<endl;
    }
};
int main()
{
    Salary s;
    s.getEmpDetails();
    s.getPayDetails();
    s.calculate();
    s.display();
            return 0;
}
```

question

Question description: Roahn and Lokesh are very close friends, they cannot go and play games during this lockdown. So they planned to play puzzle games in the home itself. Roahn gives a number to Lokesh and he has to find the answer for the number he is getting from Roahn. Can you help him to finish the game efficiently? Constraints: 1<number≤1000 Input Format: Only line of input has a single value of type integer representing the number provided by Rohan. Output format: In the first line of output print square of the number. In the second line of output print cube of the number.

answer

```
#include <iostream>

using namespace std;

class top{

   public:

   int n;

   void getdata(){

      cin>>n;

   }

};

class middle : public top{

   public:

   void square(){

      cout<<n*n<<endl;

   }

};

class bottom :public middle{

   public:

   void cube(){

      cout<<n*n*n;

   }

};

int main()

{
```

```
    bottom calc;
    calc.getdata();
    calc.square();
    calc.cube();
            return 0;
}
```

question

Question description: Raman is in his second year of engineering at CCC. He's nearing the conclusion of the semester, and he needs to turn in his mini project as soon as possible. He aims to create a tiny mark printing system that is suited to the specific of the user. Can you assist in completing the project and obtaining a good core in the mini project? Input Format: First Line: Role Number Second Line: Mark 1 and Mark 2 seperated by a space Third Line: Sports Mark Output Format: Print the results as per format. Refer sample testcases for format specification.

answer

```
 #include <iostream>
using namespace std;
class student{
    public:
    int roll,m1,m2;
    void get(){
        cin>>roll>>m1>>m2;
    }
};
class sports{
    public:
    int sp;
    void getsm(){
        cin>>sp;
```

```
    }
};
class statement : public student, public sports{
    public:
    void display(){
        cout<<"Roll No:"<<roll<<endl;
        cout<<"Total:"<<m1+m2+sp<<endl;
        cout<<"Average:"<<(m1+m2+sp)/3<<endl;
    }
};
int main()
{
    statement obj;
    obj.get();
    obj.getsm();
    obj.display();
            return 0;
}
```

question

Problem Description:
Mr.Shahrukh has given you a binary string S. You need to transform this string into another string of equal length consisting only of zeros, with the minimum number of operations.

A single operation consists of taking some prefix of the string S and flipping all its values. That is, change all the 0s in this prefix to 1s, and all the 1s in the prefix to 0s. You can use this operation as many number of times as you want over any prefix of the string.

Constraints:
1 ≤ |S| ≤ 100,000

Input Format:
The only line of the input contains the binary string, S .

Output Format:
Print the output in a single line containing a single integer representing the minimum number of operations that are needed to transform the given string S into the string of equal length consisting only of zeros.

answer

```
 #include <bits/stdc++.h>
```

```
using namespace std;
int main()
{
   char S[1000000];
   cin>>S;
   int i,c=0;
   int n;
   n=strlen(S);
   for(i=0;i<n-1;i++)
   {
      if(S[i]!=S[i+1])
      c++;
      else
      c=c;
   }
   cout<<c+1;
   return 0;
}
```

question

Problem Description:

Tina, is a little girl who is fond of toys. Her friend Selvan works in a toy manufacturing factory . Selvan has a 2D board 'A' of size 'H x W' with 'H' rows and 'W' columns. The board is divided into cells of size 1 x 1 with each cell indicated by it's coordinate (i, j). The cell (i, j) has an integer 'Aij' written on it. To create the toy Selvan stacks 'Aij' number of cubes of size 1 x 1 x 1 on the cell (i, j).

Given the description of the board showing the values of 'Aij' and that the price of the toy is equal to the 3d surface area find the price of the toy.

Constraints:
0 <= H, W <=100
1 <= Ai,j <= 100

Input Format:
The first line contains two space-separated integers H and W the height and the width of the board respectively.

The next 'H' lines contains 'W' space separated integers. The 'jth' integer in 'ith' line denotes Aij.

Output Format:
Print the price of the toy, in a single line.

answer

```
 #include <math.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <assert.h>
#include <limits.h>
#include <stdbool.h>
int A[100][100];
int height,width;
int small(int x, int y){
 if (x < y) return(x);
 return(y);}
int f(int x){
 return(4*x+2);}
int g(int i, int j){
 int term1,term2;
 if (i == 0) term1=0;
 else term1=small(A[i-1][j],A[i][j]);
 if (j == 0) term2=0;
 else term2=small(A[i][j-1],A[i][j]);
 //printf("term1=%d,term2=%d\n",term1,term2);
 return(2*(term1+term2));}
int main() {
 int i,j,result;
 scanf("%i %i", &height, &width);
 for (i = 0; i < height; ++i) {
 for (j = 0; j < width; ++j) scanf("%i",&A[i][j]);}
 result=0;
 for (i=0;i<height;++i){
 for (j=0;j<width;++j){
 result+=f(A[i][j]);
```

```
 result-=g(i,j);
 //printf("%d\n",result);
 } }
 printf("%d\n", result);
 return 0;
 printf("cin>>n>>m;cout<<price;");
}
```

question

Problem Description: Swathy and Nancy were selected for SpaceY programme which was about to take place the next year ,in their interview they were struck with the question. The question is that if the floating number is given they have to create a code to display the rightmost integer from the integer part of the number. If they have the logic for the code they will be the part of the digital meter designing for the SpaceY Mars launch which was their dream.
Can you help them with a logic of the code for the criteria given to them?
Constraints:
25.0000 ≤ spacenum < 999.0000

Input Format:

Only Line of Input has single value of type float.
Output Format:

Print the rightmost integer from the input value. Explanation : If the input is given 124.34, then the output to be displayed is 4 (i.e) Before decimal the integral part is 124 , in that last digit is 4.

answer

```
 #include <iostream>
using namespace std;
int main()
{
   float spacenum;
   int digit;
   cin>>spacenum;
   digit=(int)spacenum%10;
   cout<<digit;
          return 0;
}
```

question

Question description Nowadays the one-way traffic is introduced all over the world in order to improve driving safety and reduce traffic jams. The government of Germany decided to keep up with new trends. Formerly all n cities of Germany were connected by n two-way roads in the ring, i. e. each city was connected directly to exactly two other cities, and from each city it was possible to get to any other city. The government of Germany introduced one-way traffic on all n roads, but it soon became clear that it's impossible to get from some of the cities to some others. Now for each road is known in which direction the traffic is directed at it, and the cost of redirecting the traffic. What is the smallest amount of money the government should spend on the redirecting of roads so that from every city you can get to any other? Constraints: 3 ≤ n ≤ 100 1 ≤ a_i, b_i ≤ n, a_i ≠ b_i 1 ≤ c_i ≤ 100 Input Format: The first line contains integer n — the amount of cities (and roads) in Germany. Next, n lines contain descriptions of roads. Each road is described by three integers a_i, b_i, c_i — the road is directed from city a_i to city b_i, redirecting the traffic costs c_i. Output Format: Output single integer — the smallest amount of money the government should spend on the redirecting of roads so that from every city you can get to any other.

answer

```
#include<bits/stdc++.h>

using namespace std;

int n,res,ans,a,b,c,s[109],e[109];

set<int>first,second;

int main() {

    cin>>n;

    for(int i=0;i<n;i++){

        cin>>a>>b>>c;

        if(s[a]||e[b])res+=c,s[b]=e[a]=1;

        else s[a]=e[b]=1;

        ans+=c;

    }

    cout<<min(res,ans-res);
```

```
        return 0;
}
```

question

Question description: Harsh is an employee in Linkedin were his job is to maintain the details of the top profiles in his region. Since the number of such profiles is high in his region he is looking for the programming logic which can consolidate those details in a format. Can you help Harsh? Input Format: The First line of input has five values of type string representing the name of the user. The Second line of input has a single value of type integer representing code of the user. The Third line of input has a single value of type integer representing the pay of the user. The fourth line of input has a single value of type integer representing the experience of the user. The fifth line of input has a single value of type string representing the name of the company the user is working. Output Format: Print the output in the expected format. Refer sample testcases for format specification.

answer

```
#include <iostream>
using namespace std;
class person{
};
class admin{
};
class account{
};
class master:public account,public admin{
   public:
   int code,pay,exp;
   string name,comp;
   void getpay(){
      cin>>name;
      cin>>code>>pay;
   }
```

```
    void getexp(){
        cin>>exp>>comp;
    }
    void display(){
        cout<<"Name:"<<name<<endl;
        cout<<"Code:"<<code<<endl<<"Pay:"<<pay<<endl;
        cout<<"Experience:"<<exp<<endl;
        cout<<"Company name:"<<comp<<endl;
    }
};
int main()
{
    master m1;
    m1.getpay();
    m1.getexp();
    m1.display();
            return 0;
            cout<<"m1.admin::display();m1.person::display();m1.account::display();";
}
```

question

Question description In Talks at Google Series, there are N Mater Trainers. The Series runs for D days. Each day, there can be at most one lecture. The i-th Mater Trainer arrives on day D_i and then stays till the end of the Series. He also wants to teach exactly Ti lectures. For each lecture that a Mater Trainer was not able to teach, he will feel sad and his sadness level will be increased by S_i. Kanthamaran is the main organizer of the contest. Can you help Kanthamaran in finding the minimum total sadness of the Mater Trainers? Constraints 1 ≤ T ≤ 10 1 ≤ N, D ≤ 10⁵ 1 ≤ D_i, T_i ≤ D 1 ≤ S_i ≤ 10⁵ Input Format: The first line of the input contains an integer T, denoting the number of testcases. For each test case, the first line contains two space separated integers, N, D. The i-th of the next N lines will contain three space separated integers: D_i, T_i, S_i respectively. Output Format: For each test case,

output a single integer corresponding to the minimum total sadness of the Mater Trainers achievable.</p>

answer

```
 #include <bits/stdc++.h>
#define ll long long
using namespace std;
int main(){
   int t;
   cin >> t;
   while (t--) {
      int n, d;
      cin >> n >> d;
      map<ll, vector<pair<long,long>>>TGS;
      for (int i = 0; i < n; i++){
         ll day, lec, sad;
         cin >> day >> lec >> sad;
         TGS[day].push_back({sad, lec}); }
      priority_queue<pair<long,long>>PQ;

      for (int i = 1; i <= d; i++) {
         for (auto x : TGS[i])
            PQ.push(x);
         if (!PQ.empty())
         {
            pair<ll, ll> p = PQ.top();
            PQ.pop();
            p.second--;
            if (p.second == 0) {}
            else
               PQ.push({p.first, p.second});
```

```
        }
    }
    ll cnt = 0;
    while (!PQ.empty()) {
        pair<ll, ll> p = PQ.top();
        cnt += (p.first * p.second);
        PQ.pop();
    }
    cout << cnt << endl;
  }
  return 0;
  cout<<"vector<pair<long,long>>TGS PQ.top().first;PQ.top().second ";}
```

question

Question description: Winter in Spain is such a beautiful time of the year! Tina is walking in the forest and picking a bouquet from fallen leaves. Tina is very choosy, she doesn't take a leaf if it matches the color and the species of the tree of one of the leaves she already has. Find out how many leaves Tina has picked. Constraints: 1 ≤ n ≤ 100 Input Format: The first line contains an integer n representing the number of leaves Tina has found. The next n lines contain the leaves' descriptions. Each leaf is characterized by the species of the tree it has fallen from and by the color. The species of the trees and colors are given in names, consisting of no more than 10 lowercase Latin letters. A name can not be an empty string. The species of a tree and the color are given in each line separated by a space. Output Format: Output the single number representing the number of Tina's leaves.

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
  int n;
  cin>>n;
```

```
    set<pair<string,string>>Descriptionofleaves;
    string species,color;
    while(n--){
        cin>>species>>color;
    Descriptionofleaves.insert(make_pair(species,color));
    }
            cout<<Descriptionofleaves.size();
            return 0;
}
```

question

Question description: The Amazon campus of India has N different attractions, numbered from 1 to N in decreasing order of popularity. The name of i th attraction is A _i , a unique, non-empty string consisting of at most 20 characters. Each character is either a lowercase letter ("a" .. "z"), uppercase letter ("A" .. "Z"), or digit ("0" .. "9"). Arjun enjoys visiting the campus repeatedly for tours (including the free food!). Each time he visits, he has time to see exactly K of the attractions. To decide which K he'll see, he sorts the N attractions in non-decreasing order of how many times he's already seen them before, breaking ties in decreasing order of popularity, and then chooses the first K attractions in the sorted list. In other words, he prioritizes seeing attractions which he's seen the fewest number of times previously but also opts to see the most popular attractions out of the ones he's seen an equal number of times. Arjun has visited the Amazon campus V -1 separate times already and is about to go for his V th visit. Given that he's always followed the rules stated above, and that he'll continue to, he'd like to determine which K attractions he'll see on his V th visit. He'd like to list them in decreasing order of popularity (in other words, in the same relative order as they appear in the given list of all N attractions). Constraints: 1 ≤ T ≤ 80
1 K ≤ N ≤ 50
1 ≤ V ≤ 10 ¹² Input Format: Input begins with an integer T , the number of campuses. Each campus For, there is a line containing the first space-separated integers N , K , and V . Then, N lines follow. The i th of these lines contains the string A _i . Output Format: For the i th campus, print a line containing "Case # i :" followed by K space-separated strings, the names of the attractions that Arjun sees on his V th visit, in decreasing order of popularity.

answer

```
 #include <bits/stdc++.h>
using namespace std;
typedef long long LL;
const int N=55;
LL n, k, v, idx;
string name[N];
int main(){
  LL t; cin>>t; while(t--){
    cin>>n>>k>>v;
    for(int i=0; i<n; i++)
      cin>>name[i];
    LL st=((v-1)*k)%n;
    //cout<<"Case #"<<(++idx)<<":";
    vector<int> ans;
    for(int i=0; i<k; i++)
      ans.push_back((st+i)%n);
    sort(ans.begin(), ans.end());
    for(int id: ans)
      cout<<name[id]<<" ";
    cout<<"\n";
  }
  return 0;
  cout<<"vector<string>visit(n); vector<pair<int,string>>seenattraction;
sort(seenattraction.begin(),seenattraction.end());";
}
```

question

Question Description: Prakash given two arrays $a[1…n]$ and $b[1…n]$, both of the same length n. In order to perform a push operation, you have to choose three integers l,r,k satisfying 1≤l≤r≤n and k>0. Then, you will add k to elements al,al+1,…,ar. For example, if a=[3,7,1,4,1,2] and you choose (l=3,r=5,k=2), the array a will become [3,7,3,6,3——————,2]. You can do this operation at most once. Can you make array a

equal to array b? (We consider that a=b if and only if, for every 1≤i≤n, ai=bi) Constraints: 1≤t≤20 1≤n≤100 000 1≤ai≤1000 1≤bi≤1000 10^5 Input Format: The first line contains a single integer t the number of test cases in the input. The first line of each test case contains a single integer n the number of elements in each array. The second line of each test case contains n integers a1,a2,…,an. The third line of each test case contains n integers b1,b2,…,bn. It is guaranteed that the sum of n over all test cases doesn't exceed. Output Format: For each test case, output one line containing "YES" if it's possible to make arrays a and b equal by performing at most once the described operation or "NO" if it's impossible. You can print each letter in any case (upper or lower).

answer

```
 #include<bits/stdc++.h>

using namespace std;

void solve(){

   cout<<"bool has_positive(vector<int> s) int other_ele(vector<int> v)";

}

int main() {

        int t;

        cin>>t;

        int i,j,k,l;

        for(i=0; i<t; i++) {

                int n;

                cin>>n;

                vector<int>a(n);

                for(j=0; j<n; j++) {

                        cin>>a[j];

                }

                for(k=0; k<n; k++) {

                        cin>>l;

                        a[k]=l-a[k];


                }

                j=0,k=n-1;
```

```
		while(j<n&&a[j]==0) {
			j++;
		}
		while(k>=j&&a[k]==0) {
			k--;
		}
		j++;
		while(j<=k&&a[j]>0&&a[j]==a[j-1]) {
			j++;
		}
		if(j>k&&a[k]>=0) {
			cout<<"YES\n";
		} else {
			cout<<"NO\n";
		}
	}
	return 0;
}

/*#include<bits/stdc++.h>
using namespace std;
int main()
{
	int n,t;
	cin>>t;
	while(t--){
	cin>>n;
	int a[n],b[n];
	int num=0;
	for(int i=1; i<=n; i++){
		cin>>a[i];
```

```
        }
        for(int i=1; i<=n; i++){
                cin>>b[i];
                b[i]-=a[i];
                if(b[i]<0)num+=100;
                if(b[i]>0 && b[i]!=b[i-1])num++;
        }
        puts(num>=2?"NO":"YES");
        }
        return 0;
}*/
```

question

Question description While sailing on a boat, Esha noticed a beautiful water lily flower above the lake's surface. She came closer and it turned out that the lily was exactly H centimeters above the water surface. Esha grabbed the flower and sailed the distance of L centimeters. Exactly at this point, the flower touched the water surface. Suppose that the lily grows at some point A on the lake bottom, and its stem is always a straight segment with one endpoint at point A. Constraints: 1≤H<L≤10^6 Input Formats: The only line contains two integers H and L. Output Format: Print a single number — the depth of the lake at point A. The absolute or relative error should not exceed 10^−6. Formally, let your answer be A, and the jury's answer is B. Your answer is accepted if and only if |A−B|max(1,|B|)≤10^−6.

answer

```
#include <bits/stdc++.h>
using namespace std;
#define fo(i,n) int distance(int a,int b,int c,int d)
#define endl int surface(int a,int b)
#define MOD 1000000007
int main(){
   long double h,l;
   cin>>h>>l;
```

```
    long double ans=(l*l-h*h)/(2*h);
    cout<<fixed<<setprecision(15)<<ans;
    return 0;
}
```

question

Question description: Rohan is looking for the suitable job in Rome. In the city of Rome job applicants are often offered an Knowledge test. The test is as follows:the person gets a piece of squared paper with a 4 × 4 square painted on it. Some of the square's cells are painted black and others are painted white. Here the task is to repaint at most one cell the other color so that the picture has a 2 × 2 square, completely consisting of cells of the same color. If the initial picture already has such a square, the person should just say so and the test will be completed. Rohan's task is to create a programming logic that determines whether it is possible to pass the test. He cannot pass the test if either repainting any cell or no action doesn't result in a 2 × 2 square, consisting of cells of the same color. Can you help Rohan ? Input Format: Four lines contain four characters each: the j-th character of the i-th line equals "." if the cell in the i-th row and the j-th column of the square is painted white, and "#", if the cell is black. Output Format: Print "YES" (without the quotes), if the test can be passed and "NO" (without the quotes) otherwise.

answer

```
#include<bits/stdc++.h>
using namespace std;
int i,j;
string s[4];
int main(){
        for(;j<4;j++)cin>>s[j];
        for(i=0;i<3;i++)
        {
                for(j=0;j<3;j++)
                {
                        if(s[i][j]+s[i][j+1]+s[i+1][j]+s[i+1][j+1]!=162)
                        {
                                cout<<"YES";
```

```
                    return 0;
                }
            }
        }
        cout<<"NO";
        return 0;
        cout<<"map<string,string>JobinRome;";}
```

question

Question description Two students of Kindergarten are fighting over candy packs. There are three candy packs, each of which contains a, b, and c candies, respectively. Teacher Evi is trying to distribute the packs between the two students so that each student gets the same number of candies. The task is to overload == operator to determine whether it is possible. Note that Evi cannot take candies out of the packs, and the whole contents of each pack must be given to one of the students. <h3>Constraints</h3>
<math xmlns="http://www.w3.org/1998/Math/MathML"><semantics><annotation encoding="application/x-tex">1 ≦ a, b, c ≦ 100</annotation></semantics></math>1≦a,b,c≦100 Input Format The input is given from Standard Input in the following format: a b c Output format If it is possible to distribute the packs so that each student gets the same number of candies, print Yes. Otherwise, print No.

answer

```
#include <bits/stdc++.h>
using namespace std;
void solve(){
  cout<<"class Candiesvoid operator==(Candies t2)";
}
int main()
{
  int a,b,c,sum;
  cin >> a >> b >> c;
```

```
  sum = a + b + c;
  if(2 * max({a,b,c}) == sum)
  {
    cout << "Yes" << endl;
  }
  else
  {
    cout << "No" << endl;
  }

  return 0;
}
```

question

Question description Raja is a Mathematics Professor in our Institution . He wants to conduct an assessment to his student with different set of questions in the session of finding characteristic equation of 2 x 2 matrices. He need the answers for all the questions he prepared. Can you help him to print the characteristic equation of a given matrix? Function Description The characteristic equation of a given matrix of order 2 is X^2 - s X + p=0 where s is the sum of diagonal elements of the matrix and p is the determinant of the given matrix. You should get the first row elements of a matrix from the object c1 of the class ChEqn. and second row elements from the object c2. And sum of the diagonal will be in the +operator overloading and the determinant will be in the -operator overloading. Constraints -10^2<a,b<10^2 Input Format First line represents the first row elements of the matrix which will be stored in object c1. Second line represent the second row elements of the matrix which will be stored in object c2. Output Format print the characteristic equation.

answer

```
 #include <iostream>
using namespace std;
void solve(){
        cout<<"class ChEqn ChEqn operator +(ChEqn c2) ChEqn c1,c2,c3,c4; ChEqn operator -(";
}
```

```
int main()
{
   int a,b,c,d;
   cin>>a>>b>>c>>d;
   if(a+d<0 && (a*d-b*c)<0)
   cout<<"x^2"<<a+d<<"x"<<(a*d-b*c)<<"=0";
   else if(a+d<0 && (a*d-b*c)>0)
   cout<<"x^2"<<a+d<<"x+"<<(a*d-b*c)<<"=0";
   else if(a+d>0 && (a*d-b*c)<0)
   cout<<"x^2+"<<a+d<<"x"<<(a*d-b*c)<<"=0";
   else
   cout<<"x^2+"<<a+d<<"x+"<<(a*d-b*c)<<"=0";
          return 0;
}
```

question

Question description Ramesh is a Mathematics Professor in our institution. He wants to conduct an assessment to his student with different set of questions in the session of finding Eigen values of 2 x 2 matrices. He need the answers for all the questions he prepared. Can you help him to print the Eigen values of a given matrix? Function Description Solving the characteristic equation of a given matrix of order 2 is AX^2 + B X + C=0 to get the Eigen values. You have a task to overload the ++operator to find the Eigen values of the given 2x2 matrix. Constraints -10^2<a,b<10^2 Input Format First line represents the coefficients of the characteristic equation. Output Format print the Eigen values.

answer

```
 #include <stdio.h>
#include <math.h>
void solve(){
   printf("class EigenValvoid operator ++() if (discriminant > 0)EigenVal c1;");
}
int main()
```

```
{
 int a,b,c; int root1,root2; int discriminent;
 scanf("%d %d %d",&a,&b,&c);
 discriminent=(b*b)-(4*a*c);
 if(discriminent>0){
 root1=(-b+sqrt(discriminent))/(2*a);
 root2=(-b-sqrt(discriminent))/(2*a);
 printf("x1 = %d\nx2 = %d",root1,root2);
 }
 if(discriminent==0){
 root1=root2=-b/(2*a);
 printf("x1 = x2 =%d",root1);
 }

 return 0;
}
```

question

Question description Nithi is a Mathematics student. He had an assignment in the 3 x 3 matrix multiplication. Can you help him to verify his answer? But the task is to you that to overload * operator to find the product of two matrices. Constraints -10≤n≤10, where n is the elements of the matrices Input Format First line represents the elements of the first row of the first matrix Second line represents the elements of the second row of the first matrix Third line represents the elements of the third row of the first matrix Fourth line represents the elements of the first row of the second matrix Fifth line represents the elements of the second row of the second matrix Sixth line represents the elements of the third row of the second matrix Output Format Print the resultant matrices

answer

```
 #include <iostream>
using namespace std;
```

```
#define f(i,n) for(i=0;i<n;i++)
int main()
{
    int a[10][10], b[10][10], mult[10][10], r1=3, c1=3, r2=3, c2=3, i, j, k;
    f(i,r1)
        f(j,c1)
        {
            cin >> a[i][j];
        }
    f(i,r2)
        f(j,c2)
        {
            cin >> b[i][j];
        }
    f(i,r1)
        f(j,c2)
        {
            mult[i][j]=0;
        }
    f(i,r1)
        f(j,c2)
            for(k = 0; k < c1; ++k)
            {
                mult[i][j] += a[i][k] * b[k][j];
            }
    f(i,r1)
    f(j,c2)
    {
        cout<<mult[i][j]<<" ";
        if(j == c2-1)
            cout << endl;
```

```
    }
    return 0;
    cout<<"class mult int mat[10][10]; void operator*(mult B)";
}
```

question

Question description: Zaheer is an higher secondary school maths teacher. In his last class he he thought his students the factorial and the way to calculate the same. So in todays class he assigned his student the task of writing a programming logic for implementing the factorial calculation. Can you help the students in doing the same? Input Format: Only line of input has a single value representing the input. Output Format: Print either the result of the factorial calculation and throw the error message if anything other than the integer is provided as input. Refer sample testcases for format specification.

answer

```
 #include <bits/stdc++.h>
#include <string.h>
using namespace std;
int main()
{
        int k;
    try{
        cin>>k;
        if(cin)
        cout<<fixed<<setprecision(0)<<tgamma(k+1);
        else
        throw "e";
    }
    catch (int i){
    }
    catch (const char *exp){
```

```
        cout<<"Input should be a Integer";
    }
            return 0;
}
```

question

Problem Description:
Karthik was working in the HR division of Audi. The employees of the company were working on shifts. The company calculates salary for the employees on the basis of employee working hours per day. Since the number of people working in the company is huge salary calculation become a tedious process at the end of the each day.
Constraints:
1 <= hour <= 12
1 <= salaryperday <= 6000
Input Format:
The First line of the input has a single value representing the total working hours of type integer.
The Second line of the input has single value representing the salary per day of type double.
Output Format:
Print the total salary in single line with two values after decimal point. If incomplete work information is provided throw the exception message “Insufficient Work Information”

answer

```
 #include<bits/stdc++.h>
using namespace std;
int main()
{
    float hour,salaryperday;
    try{
        cin>>hour;
        cin>>salaryperday;
        if(cin){
            cout<<fixed<<setprecision(2)<<hour*salaryperday;
        }
        else
        throw 0;
    }
    catch(int workstatus) {
```

```
        cout<<"Insufficient Work Information";
    }
    return 0;
}
```

question

Problem Description: Elavenil has a chessboard with N rows and M columns. In one step, he can choose two cells of the chessboard which share a common edge (that has not been cut yet) and cut this edge. Formally, the chessboard is split into two or more pieces if it is possible to partition its cells into two non-empty subsets S1 and S2 (S1∩S2=∅, |S1|+|S2|=NM) such that there is no pair of cells c1,c2 (c1∈S1,c2∈S2) which share a common edge that has not been cut. Elavenil does not want the board to split into two or more pieces. Compute the maximum number of steps he can perform while satisfying this condition. Constraints: 1≤N,M≤8 Input Format: The only line of input test case contains two space-separated integers N and M. Output Format: In the only line of output print an integer representing the maximum possible number of steps. If the dimension of the chessboard inputed is invalid throw the exception message “Invalid Board Size”

answer

```
#include <iostream>
using namespace std;
int main()
{
    int m,n;
    try{
        cin>>n;
        cin>>m;
        if(cin){
            cout<<(m-1)*(n-1);
        }
        else
        throw 0;
    }
```

```
    catch(int boardsize){
        cout<<"Invalid Board Size";
    }
    return 0;
}
```

question

Problem Description:
Binita is playing a chess. The game will be played on a rectangular grid consisting of N rows and M columns. Initially all the cells of the grid are uncolored.

Binita's initial score is zero. At each turn, he chooses some cell that is yet not colored, and colors that cell. The score obtained in this step will be number of neighboring colored cells of the cell that Binita colored in this step. Two cells are neighbors of each other if they share a side between them. The game will end when all the cells are colored. Finally, total score obtained at the end of the game will sum of score obtained in each turn.

Binita wants to know what maximum score he can get? Can you please help him in finding this out?

Constraints:
1 ≤ N, M ≤ 50

Input Format:
The Only line of input contains two space-separated integers N, M denoting the dimensions of the grid.

Output Format:
Print the output a single line containing an integer corresponding to the maximal possible score Binita can obtain.

answer

```
 #include <iostream>
using namespace std;
int main()
{
    int n,m;
    try{
    cin>>n;
    cin>>m;
    if(cin){
        cout<<n-1+(1+2*(n-1))*(m-1);
    }
    else
    throw 0;
```

```
    }
    catch(int griddimensions)
    {
        cout<<"Invalid Grid Dimensions";
    }
            return 0;
}
```

question

Question description: Bogar was given a task to check whether the entered mark is valid or not. Bogar framed three rules for checking the validity of the mark Rule 1: The mark should be greater than 0 and less than or equal to 100 [0 < m <=100] Rule 2: The mark should not exceed 100. Rule 3: No negative Marks Rule 4: It should be a valid integer number Kindly help Bogar the Tamil SIDDHAR to perform the operations. Constraints: 1≤n≤1000 Input Format: Only line of input has a single value representing the input. Output Format: If the input value satisfies the above mentioned rules of Bogar print “Valid Mark” And throw the error message “Invalid Mark” if the input value dosen't satisfy the rules of Bogar. Refer sample testcases for format specification.

answer

```
 #include <iostream>
#include <math.h>
using namespace std;
int main()
{
    int a;
    try {
        cin>>a;
        if (a>0 && a<=100)
        cout<<"Valid Mark";
        else
        throw "e";
```

```
    }
    catch(const char* t){
        cout<<"Invalid Mark";
    }
}
```

question

Question description: There was a high voltage war between the Adithya Karikalan's troops and Veerapandian's troops. Veerapandian's troops have the upper hand at one stage of the war and Parthiventhiran the Commander of Adithya Karikalan's troops is worried about the strength of his troops and would like to get some helping hand from the neighbouring kings who are against Veerapandiyan. So Parthiventhiran would like to know how many of his warriors are remaining and if there is any help to be requested to avoid defeat in the war ? Constraints: 1≤akt,vpt≤1000 Input Format: First line of input has a single value of type integer representing the number of remaining warriors Veerapandian have. Second line of input has a single value of type integer representing the number of remaining warriors Adithya Karikalan have. Output Format: In the only line of output print the relevant message. Print five values after decimal point. Refer sample testcases for format specification.

answer

```
 #include<bits/stdc++.h>
#define NegativeNumber int
using namespace std;
int main()
{
    float akt,vpt;
    try{
        cin>>akt;
        cin>>vpt;
        if(vpt>0)
        {
            cout<<"Each Chola Warrior must fight "<<fixed<<setprecision(5)<<akt/vpt<<" Pandiya Warriors";
```

```
        }
        else
        throw 0;
    }
    catch(NegativeNumber e){
        cout<<"Chola Troops Need Help";
    }
            return 0;
}
```

question

Problem Description: The Electricity Officer has mentioned the total counts of unit and amount. The officer inform the customer the bill amount in a unique format. The format given by electricity officer as follow: But customers are finding the difficult to find the exact amount that needs to be paid. Can you help the customers?
Functional Description: Total Bill Amount = unitconsumed ^ costperunit
Constraints:
1 ≤ unitconsumed ≤ 500

2 ≤ costperunit ≤ 10

Input Format :

The first line of input represents the integer value of unitconsumed
The second line of input represents the integer value of costperunit

Output Format:

Print the total Bill amount in single line.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
 int unitconsumed,costperunit;
 try{
 cin>>unitconsumed;
 cin>>costperunit;
 long int res;
 res=pow(unitconsumed,costperunit);
 if(cin){
```

```
 cout<<res;
 }
 else
 throw 0;
 }
 catch(int unit){
 cout<<"Incomplete Data";
 }
return 0;
}
```

question

Question description Ravi is an attendance coordinator. He needs to check the attendance of a wages on a day. He has the data of working time duration of Forenoon and Afternoon Session. Can you help him to check the eligibility to give the salary to the wages for a day? Constraints 0≤hr≤24 0≤m<59 0≤s<59 Input Format First line represents HH MM SS on the forenoon time duration which are separated by a space Second line represents HH MM SS on the afternoon time duration which are separated by a space Output Format HH:MM:SS

answer

```
 #include <iostream>
using namespace std;
class Time{
   public:
          int hh1,mm1,ss1;
   void setTime(){
      cin>>hh1>>mm1>>ss1;
   }
   Time operator+(Time t2){
      Time temp;
      temp.ss1=ss1+t2.ss1;
```

```
        temp.mm1=mm1+t2.mm1+(temp.ss1/60);
        temp.hh1=hh1+t2.hh1+(temp.mm1/60);
        temp.mm1=temp.mm1%60;
        temp.ss1=temp.ss1%60;
        return temp;
    }

};
int main()
{
        Time t1,t2,t3;
        t1.setTime();
        t2.setTime();
        t3=t1+t2;
        cout<<t3.hh1<<":"<<t3.mm1<<":"<<t3.ss1<<endl;
        return 0;
}
```

question

Question Description: So the Beautiful Regional Contest has come to an end! n students took part in the contest. The final standings are already known: the participant in the i-th place solved pi problems. Since the participants are primarily sorted by the number of solved problems, then $p1 \geq p2 \geq \cdots \geq pn$. Help the jury distribute the gold, silver, and bronze medals. Let their numbers be g, s and b, respectively. Here is a list of requirements from the rules, which all must be satisfied: for each of the three types of medals, at least one medal must be awarded (that is, g>0, s>0 and b>0); the number of gold medals must be strictly less than the number of silver and the number of bronze (that is, g<s and g<b, but there are no requirements between s and b); each gold medalist must solve strictly more problems than any awarded with a silver medal; each silver medalist must solve strictly more problems than any awarded a bronze medal; each bronze medalist must solve strictly more problems than any participant not awarded a medal; the total number of medalists $g+s+b$ should not exceed half of all participants (for example, if n=21, then you can award a maximum of 10 participants, and if n=26, then you can award a maximum of 13 participants). The jury wants to reward with medals the total maximal number of participants (i.e. to maximize $g+s+b$) so that all of the items listed above are fulfilled. Help the jury find such a

way to award medals. Constraints: 1≤t≤10000 1≤n≤4.10^5 0≤pi≤10^6 4.10^5 Input Format: The first line of the input contains an integer t the number of test cases in the input. Then t test cases follow. The first line of a test case contains an integer n the number of BeRC participants. The second line of a test case contains integers $p1$,$p2$,...,pn, where pi is equal to the number of problems solved by the i-th participant from the final standings. The values pi are sorted in non-increasing order, i.e. $p1$≥$p2$≥⋯≥pn. The sum of n over all test cases in the input does not exceed. Output Format: Print t lines, the j-th line should contain the answer to the j-th test case. The answer consists of three non-negative integers g,s,b. Print g=s=b=0 if there is no way to reward participants with medals so that all requirements from the statement are satisfied at the same time. Otherwise, print three positive numbers g,s,b — the possible number of gold, silver, and bronze medals, respectively. The sum of g+s+b should be the maximum possible. If there are several answers, print any of them.

answer

```
 #include<bits/stdc++.h>

using namespace std;

int nxt(){

   cout<<"bool Regional(int n) int getPow(int a,int b) ";

   return 1;

}

int main()

{

   int x,n;

   int a[4*100000];

   for(cin>>x;x--;) {

      cin>>n;

      for(int i=0;i<n;i++)cin>>a[i];

      int b=1;

      while(b<n && a[b]==a[b-1])

      b++;

      int c=2*b+1;

      while(b<n && a[c]==a[c-1])

      c++;
```

```
        int d=n/2;
        while(d&& a[d]==a[d-1])
        d--;
        if(d-c<=b) {
            cout<<"0 0 0";
        }
        else cout<<b<<" "<<c-b<<" "<<d-c;
        cout<<endl;
    }
    return 0;
}
```

question

Question Description: There are n programmers that you want to split into several non-empty teams. The skill of the i-th programmer is ai. You want to assemble the maximum number of teams from them. There is a restriction for each team: the number of programmers in the team multiplied by the minimum skill among all programmers in the team must be at least x. Each programmer should belong to at most one team. Some programmers may be left without a team. Calculate the maximum number of teams that you can assemble. Constraints: 1≤t≤1000 1≤n≤10^5 1≤x≤10^9 1≤ai≤10^9 Input Format: The first line contains the integer t the number of test cases. The first line of each test case contains two integers n and x the number of programmers and the restriction of team skill respectively. The second line of each test case contains n integers $a1$,$a2$,…,an, where ai is the skill of the i-th programmer. The sum of n over all inputs does not exceed 10^5. Output Format: For each test case print one integer the maximum number of teams that you can assemble.

answer

```
#include<bits/stdc++.h>
using namespace std;
int split(int a,int b){return 1;}
int splitNum(int a,int b){return 1;}
void cmemoInit(){
```

```
    int n,x;
    cin>>n>>x;}
int main()
{
    int i,j,k,l,o,p;
    cin>>k;
    while(k--) {
        cin>>o>>p;
        int a[o];
        for(i=0;i<o;i++)
        cin>>a[i];
        sort(a,a+o);
        l=0; j=0;
        for(i=o-1;i>=0;i--){
            j++;
            if(a[i]*j>=p) {
                l++;
                j=0;
            }
        }
        cout<<l<<endl;
    }
}
```

question

Question Description: There are n+2 towns located on a coordinate line, numbered from 0 to n+1. The i-th town is located at the point i. You build a radio tower in each of the towns 1,2,…,n with probability 12 (these events are independent). After that, you want to set the signal power on each tower to some integer from 1 to n (signal powers are not necessarily the same, but also not necessarily different). The signal from a tower located in a town i with signal power p reaches every city c such that $|c-i|$<p. After building the towers, you want to choose signal powers in such a way

that: towns 0 and n+1 don't get any signal from the radio towers; towns 1,2,…,n get signals from exactly one radio tower each. For example, if n=5, and you have built the towers in towns 2, 4, and 5, you may set the signal power of the tower in towns 2 to 2, and the signal power of the towers in towns 4 and 5 to 1. That way, towns 0 and n+1 don't get the signal from any tower, towns 1, 2, and 3 get the signal from the tower in town 2, town 4 gets the signal from the tower in town 4, and town 5 gets the signal from the tower in town 5. Calculate the probability that, after building the towers, you will have a way to set signal powers to meet all constraints. Constraints: 1≤n≤2.10^5 Input Format: The first line of the input contains one integer n Output Format: Print one integer the probability that there will be a way to set signal powers so that all constraints are met, taken modulo 998244353. Formally, the probability can be expressed as an irreducible fraction xy. You have to print the value of $x.y$−1mod998244353, where y−1 is an integer such that $y.y$−1mod998244353=1.

answer

```
 #include<bits/stdc++.h>

using namespace std;

long long int n,a=1,b=1,mod=998244353;

int tower(int x,int y){

   return 1;

}

int tower1(int a,int m){

   return 1;

}

void event(){

   cin>>n;

}

int Pow(int a,int b){

        int rec=1;

        for(;b;b>>=1,a=1ll*a*a%mod)

                if(b&1)rec=1ll*rec*a%mod;

        return rec;

}

int main(){

        scanf("%lld",&n);
```

```
	for(int i=3;i<=n;i++){
		a=(a+b)%mod;
		swap(a,b);
	}
	printf("%lld",1ll*b*Pow(Pow(2,n),mod-2)%mod);
}
```

question

Question Description: Priyan has a statistic of price changes for one product represented as an array of n positive integers $p0,p1,\ldots,pn-1$, where $p0$ is the initial price of the product and pi is how the price was increased during the i-th month. Using these price changes you are asked to calculate the inflation coefficients for each month as the ratio of current price increase pi to the price at the start of this month ($p0+p1+\cdots+pi-1$). Your boss said you clearly that the inflation coefficients must not exceed k %, so you decided to increase some values pi in such a way, that all pi remain integers and the inflation coefficients for each month don't exceed k %. You know, that the bigger changes — the more obvious cheating. That's why you need to minimize the total sum of changes. What's the minimum total sum of changes you need to make all inflation coefficients not more than k %? Constraints: $1\le t\le 1000$ $2\le n\le 100$ $1\le k\le 100$ $1\le pi\le 10$^9 Input Format: The first line contains a single integer t the number of test cases. The first line of each test case contains two integers n and k the length of array p and coefficient k. The second line of each test case contains n integers $p0,p1,\ldots,pn-1$ the array p. Output Format: For each test case, print the minimum total sum of changes you need to make all inflation coefficients not more than k %.

answer

```
#include<bits/stdc++.h>
using namespace std;
void inflation(){}
int main() {
	long long t,n,i,k,a,b,c;
	cin>>t;
	while(t--) {
		cin>>n>>k>>a;
		i=a;
```

```
            c=b=0;
            while(--n){
                cin>>a;
                b=(100*a+k-1)/k;
                if(i<b) {
                    c+=i-b;
                    i=b;
                }
                i+=a;
            }
            cout<<abs(c)<<'\n';
        }
        return 0;
        cout<<"int product(int n)int power(int a,int n,int p) ";
        cout<<"int ncr(int n,int k,int p)";
}
```

question

Question Description: Lawrence tried so hard to make a good div.2 D problem to balance his recent contest, but it still doesn't feel good at all. Lawrence invented it so tediously slow that he managed to develop a phobia about div.2 D problem setting instead. And now he is hiding behind the bushes. Let's define a Rooted Dead Bush (RDB) of level n as a rooted tree constructed as described below. A rooted dead bush of level 1 is a single vertex. To construct an RDB of level i we, at first, construct an RDB of level i−1, then for each vertex u: if u has no children then we will add a single child to it; if u has one child then we will add two children to it; if u has more than one child, then we will skip it. Rooted Dead Bushes of levels 1, 2, and 3. Let's define a claw as a rooted tree with four vertices: one root vertex (called also as center) with three children. It looks like a claw: The center of the claw is the vertex with label 1. Lee has a Rooted Dead Bush of level n. Initially, all vertices of his RDB are green. In one move, he can choose a claw in his RDB, if all vertices in the claw are green and all vertices of the claw are children of its center, then he colors the claw's vertices in yellow. He'd like to know the maximum number of yellow vertices he can achieve. Since the answer might be very large, print it modulo 10^9+7. Constraints: 1≤n≤2.10^6 1≤t≤10^4 Input Format: The first line contains one integer t the number of test cases. Next t lines contain test cases one per line. The first line of each test

case contains one integer n the level of Lee's RDB. Output Format: For each test case, print a single integer the maximum number of yellow vertices Lee can make modulo 10^9+7.

answer

```
 #include<iostream>

long long d[2000001],i,t,n;

int add(int a,int b){

   return 1;

}

int mult(int a,int b){

   return 1;

}

void Balance(){

   int q;std::cin>>q;

}

int main(){

for(i=3;i<=2e6;i++)d[i]=(2*d[i-2]+d[i-1]+4*(i%3<1))%1000000007;

std::cin>>t;while(t--){std::cin>>n;std::cout<<d[n]<<'\n';}}
```

question

Question Description: When Vennila was three years old, he was given a set of cards with letters for his birthday. They were arranged into words in the way which formed the boy's mother's favorite number in binary notation. Vennila started playing with them immediately and shuffled them because he wasn't yet able to read. His father decided to rearrange them. Help him restore the original number, on the condition that it was the maximum possible one. Constraints: 1≤n≤10^5 Input Format: The first line contains a single integer n the length of the string. The second line contains a string consisting of English lowercase letters: 'z', 'e', 'r', 'o', and 'n'. It is guaranteed that it is possible to rearrange the letters in such a way that they form a sequence of words, each being either "zero" which corresponds to the digit 0, or "one" which corresponds to the digit 1. Output Format: Print the maximum possible number in binary notation. Print binary digits separated by a space. The leading zeroes are allowed.

answer

```
 #include<bits/stdc++.h>
using namespace std;int i,n,x,y;string s;
int modpow(int x,int n,int m){
   return 1;
}
void cards(int n){
   cout<<"int playing(int n,int k) bool isPrime(int n) ";
}
int main()
{
for(cin>>n>>s;i<n;i++)
s[i]==122?y++:s[i]==110?x++:0;
for(;x--;)cout<<"1 ";
for(;y--;)cout<<"0 ";
}
```

question

Question description Bank of Spain have been opened again after the major heist by Professor Sergio and Gang.The news of Bank of Spain have spread all around Spain and some of the students have been petrified due to seeing the basilisk. The Former Director of Bank of Spain got fired and now Tokyo is trying to enter the Bank of Spain chamber. These aren't good news for Joji Mathew the new Director of Bank of Spain The problem is, he doesn't want anybody to be able to enter the Bank's chamber. The Bank of Spain Chamber is an n × m rectangular grid in which some of the cells are columns. A light ray passes through the columns without changing its direction. But with some spell we can make a column magic to reflect the light ray in all four directions when it receives the ray. This is shown in the figure below. <figure class="image"><img
src="
CSVQICAgIfAhkiAAAAAlwSFlzAAAN1wAADdcBQiibeAAAABl0RVh0U29mdHdhcmUAd3d3Lmlua3NjYX
BlLm9yZ5vuPBoAACAASURBVHic7d13fBRl/sDxz7M7u8mmEXrvRaSIIiqICgIqikI8GyrnoR4n3v2sWMCG
ngUbFmx39vOwnI2miCciyqmgoqgoIEWadEggJJvszu7z+2MSCEsCm+zORGa/79crr7vdndnv88iz35nnm
WfmUVprhBDiUOep7QIIIUqyGEK0Ep7DIIUqyGEK0Ezs8Jc+qPoZHgY7GRcIQBQKA88o6NcU
dtFEcnnbDfTyzClmKZhmKNxhQAevYljgCVKcWptl0Ukn7PJTDMkzcPDwIC77nK2iytSmxn0N+/S64XXTjr1
5u6ZWZtyHx1Hh9ouk0iuhJKZUqqdUio3nm0fupWewPq/3aO6AgszSzipGnEylFKda1rO6lBKdVBK1XEgTr
ZSqqPdccpiHaaUynIgTh2lVDu745TF6qyUyqjGLjdtWNenfacj3szu1vOFxpk5v42IM049pVTbGhazWpRSX
ZVSaQ7Eqa+UamV3nLJY3ZVSfidiJXpm9jTEN/7l0Qzz+XfPAp6A6AxP9bqaRwBTa1LAGngeONmBOH2Bfz

sQB+A/QC8H4pyM1SacMBWrXcRl66Yeh3s9JnXqr6Z916nesJl+dpy7DgUerVEJq+9jwInEeR7wdwfiAMw
DmjkRyLlupmbYKWePLgDanXXR+es1nOlYbJHyli2+sEn7w6cD0KTFQsxQRuMJt1C/loslksiRZDbxNlpr8B9
+5Ku9ADp1f+ckYMvDt8Z/ZBUiEWtWntK0/eHTAFAeTd0Gy79LizKkloslksiZMzOToR4VnQGcA4DmEjTT
PFG5qinsN/EGGhTvbhxo0vKbPe8dccw/i+Squrs41c0cNmjoFRlANgCKpqecfeV2aUzCEX6GtGj7ST5q71v
djnmxBYo+T1yN7QPuwhm2J7PH7iIXxWHder181K6C1rz/xmsUbO9A917PDgOyH7yZFnaXQaQ2pRnW
uccbGmDVsjPI39YJw1dylMcbmh/OYmBtl08kh+3JzAxyBjArqn2dp09+h4zMLUyb/C7hcOaRWjPd62Go3
WUQqeuuu0jX0Ltt5w9yly4aztz3HmXKy+8RLK5X3PmIN9ZHkfbnFrYnM+VhWOsOH3/z3yn/rNui9TxOH
HQj7Q6bwax3/tW4bacP5yvpagobZZcyKD1jx3fbN3UPzZt1P2ePGEz3Xv9k8hMLf+tzyt3KozkdVbEDKg5
Vtiazu+7Cj+bErJyNZ2zb2GNL3wHjAOh90t/J39puHUqfr6HrE3eRY2c5ROrSmmENGv/ww5R/TUs7Zdify
c5Zz1HHTUJ5IlsmT5rv0bBq4lhH5uAJm8V9S1HZrPi6MW8HgIZKqTaV7fPXC+eelNbsmPU/LTq716l5Pd9
RnsjfrO+K0vPEYVNmvblsmBkp+vW3zcsuUero9w4QvingqypOkqUDjRyI1RhIc6hOfqCJA7EaAQGH6uQD
mh4oVnp6Hc/fr9525q/Lj11zVO8nf23R5tPu5Z+1637Sm19/vPJvYTN/eShcfllSLbZW8TUNgAyH6uQFmi
ulSmyOUx/IcqhOHqCFUjU++c3XWu+MZ8Pq3B95KXBnzHuZWDPL765sh4Ld6wNNIt18T714WlF+dMXl
d3bz7vnstTfX/fWlF3uG/3rxp+0KizYeB9xzgNheIANYVI3y1lQm0B0I2xzHwEqcTtQpC+vOBtPmOD6sxOlE
nbKBV4BIVRs0adDdu6toU8av6+bVb9HtFip2RCY9vWnivHldIlf98fMuZqQE4I9VfI0f69/KiTrlAFOAqM1x/
Fi/qUE2xwGr7c2k5nW6E3gsri211jX+A2YBw6v6/OGxOjjxZn2l1ppwse/mcLFPV/gbq7XmwbH6vIfH6tCk
q3TaAeL0BpYmUtZq1GkukOdAnMHAfIfqtAjo70CcPGCWQ3VaCvQ+0DYPj9XTHx6r1955p04PF/tejml/
Q7XWPDRWD3h4rC598Bbds4o4I4GpDtVpE9DZgTijgZcdqlMB0MaJWLaNmZWNl910/f08U543YzZRAD
dO4C3g2h318SJEchWoKOePH08JVbS/GyYwR2le8pr7DaGIQ4xtj+EZP54Q8ESFtyptTABjJjh2c7JIIWMmc
EmFl7Htb8+B/Pr7Ge1MiYSdnHyeWWyfWS6HCydJ+3M5J5NZlUdGIRxQZc9AuENtJjNpTMJJcjB1OelmiI
Qh7c/l5MxMpAppfy4nY2YiVUgycznpZopUIQdTl5NupkgVcjB1OUlmIlVI+3M5GTMTqUKSmcvJmJlIFbHt
Tw6mLiPdTJEqpP25nHQzRaqQZOZy0s0UqULan8tJN1OkCukZuJwkM5EqpP25XG12M+XIKJwk3UyXkz
MzkSqkm+lyksxEqpD253JyNVOkCklmLifzzESqkIOpy0k3U6QKOZi6nCpbqPPgGyo1FLgo5u1+wK/A2oPtf
/X/eVpNfNDbp/z19BnRledcEPkmznLWB47DWhnZbgOAZcBvNsdpCnQFZtscB+A04Dtgi81xmgMdgE9tj
gNwBrAA2B7Pxv94ytvt8ks9XctfP/RI9Ltbbov8EseubbHq9b8albJ6hgGfALtsjtMeaID1389ufwA+BIpquP
9rWuvp8WxYnXUz12Kt9l1Rd+Bn4KBJqUVz1RPYk8yaNFEbKvm+qrQtixXv9onoCSwGvrc5TlegJc7UqS/
Wqubx/HgT0QOohzN16g98i3UwPaimTVUm1n9zAFq1ZAXxlbMUSItz20SdDnyFtbK5naKAiTN1GgrMJ8
6DTiUOeqK0R4JLr88ChsezbbjYd2642Kcr/D1bjTi9gaUOLSc/F8hzIM5gYL5DdVoE9HcgTh4wy6E6LQV6x
7t9uNh3e0z7uzbOOCOBqQ7VaRPQ2YE4o4GXHapTAdDGiVgyZiZShYyZuZxMzRCpQtqfy8nUDJEqpGfgc
tLNFKlC2p/LSTdTpAp50IHLyZmZSBXS/lxOxsxEqpBk5nLSzRSpQg6mLifdTJEq5GDqcpLMRKqQ9udyMm
YmUoUkM5eTMTORKmRqhstJN1OkCml/LifJTKQKaX8uJ0vNiVQhwxwuJ2dmIlXIBSiXk2QmUoW0P5eT
bqZIFdLNdDk5MxOpQtqfy0kyE6lCxsxcTibNilQhB1OXk9uZRKqQg6nLSTdTpAo5mLpc3IsAK6UaAc1i3s4
GWiuljjzY/pMe9Xa88oq97Sc/n5xGGQffr0xHIC2eOEmQBbR1IFY7IMOhOgWADkqpApvjtAWyHapTGtB
RKVUSz8Zz/mu0PPGEvcfP1at1o45d4ipnK6COQ3UygM5KqXSb47QA6jlUJy/QRSmVW8P9N2itt8SzoSp
bqPPgGyp1JXBzzNuNsJZdP+jS672PU/55nxhNy19/Ok8HB51mxlVIrIZbH9gQ5/aJaAwUAsU2xwkAdbB/9
WqApkA+ENcPPwEZWAeDeP9dE9EMa5Xs0ng2vmykJ/OfT3sblL9+7oVo4V+viuyIY9csrH+rrTUqZfW0A
DYDYZvjZAN+ar7KeHW0BDZiraBeEw9orZ+Ja8sEVyuuzormPWNWlP6oGnFkRfPEYsmK5sW+i2Pa3+Nxx
hmJrGieSCxZ0VyIJJMxM5eTqRkiVUj7czk5MxOpQtqfy8k8M5EqJJm5nHQzRaqQBx24nHQzRaqQ9udy0
s0UqUKSmctJN1OkCklmLifdTJEqZMzM5SSZiVQh7c/lZMxMpApJZi4nY2YiVUg30+WkmylShbQ/l5NkJlK
FtD+Xk6XmRKqQYQ6XkzMzkSrkApTLSTITqULan8vJ1UyRKiSZuZzMMxOpQg6mLifdTJEq5GDqctLNFKlC
DqYuJ91MkSrkYOpy0s0UqULan8tJMhOpQnoGLledFc3PBy6LeftorFXGN1a1X26dOt5zhw5tcETXdlmXX
Vq3Q7i0CT7/FkLhNeFxt2//6Z0ZM7Zv2779YKsd5wLdgP/FVdjEHAOswf5VuRsAHYD5NscBOB5r0dx4Vv
BORCOsFawX2hwH4ARgMdYis1Vq0KCBceGFFzYcOLB7w/79GjaNRJqjVBH5+RtLpkzdtO7DDz/YPnv27J
0H+B00x6rXd8ktfqVOBr4CimyO0xLrN/WjzXEABgJfAMEa7v+i1vrNeDasTjI7HOgR8/Y44FOswu7j3ttuOy
JvyIi8Vi0atP9igSey8Pu0tPUbvJ6t2xT162laNjc5omso2P+EqLF+w47VH855Z/q148Z9U0X4jlirMI+Jq7CJu
QOYCVRVlmTpAZwL3G5zHID7gVeAn22O0ws4BZhgcxyAicA/gOWVfXjHHXd0ufjiKy9o1qxh288+C0e++
MKbvmFDlI0bNZmZiqZNFe3aER0yhNKGDXX0+++/+/Sqq0a9u3jx4thE0g+rXhNtrg9Y9fk71gmCnQZh/aa
esTkOwAvAWGBrDff/Xmu9JK4tE1x6fRYwfJ/3d+zovmNV/pcrFhbsOj+vVGdmRDXoKv/S06I6b0hI//Tlzl3

5v+Z/p3fs6FVJnN7AUoeWk58L5DkQZzAw36E6LQL6OxAnD5jlUJ2WAr0r+axDQUH0sw0bortHjizRDRo
U6aysA/9161asn38+XFxUpAuDQX2T1lpViDMSmOpQnTYBnR2IMxp42aE6FQBtnIiV1HEDc2v+8F2F6otr
b8k4rtMxdbLfnOqnqPjAQ2MlpYqp7/vo2icne9Q1GUfmF6hPQpvzRyWzXCJl9C8qYuF994X7dukSzHz77
QglJQffafVqzbXXhgJ9+gSzfvgheseuXXoWkGl7aUVSJS2Z7V5fcNumLZ7njh2YnfXKG34Vjb0QHoe3p/s5ql
9O1trfPI/sWlvgxGm9cAnTZHhBgX4vL68k5+mnTY95sFHYSqxapTnttJLMt9+OnFRYqL8FcpJeUGGbpCQz
c2v+8IKd6uYjT8rJWrbCm9B3rVnnoWe/nKzNW9VfQlvyRyejfML1ji0p4flBg0oz58+vwVG0AtOEa64Jpf/r
X5HWhYV6Rnp6ulz1PEQk/A912cUXtywOqucG5WVlbd+RnNkWhbsVg/Kys4qK1MPk5/dNypcKt6pXV
MTMP/6xNHPZssQSWUW33hpK++EHffS0abPOSdqXClslnMxuvnb8qKtuzshM9Iws1tr1Hkb+LTNzV6F6y
TAMmZMmKlVUxF2TJ5uZs2dHkvq90SiMGFGaedxxxw/s2rVrRlK/XNgioWQ2/A9/qK91vZaT3/Tbkmymf
+Bjxa+eJlMnTx5kx/eLQ9uIESMaRaN65L33htPt+P5t2zSPPx41Hnzwya52fL9bPXE1aY+MoYXTcY1Edh59
6XXt77gvO70mg/3xGndXIPv5SSf8CajBkO6h4a67SM8q4SwFlxemc+b48e6tazJdc80NZ73wgunLz49vrm
RNPPlkxLtixfFNgDrATtsCuYgZoLv28MXEcawY0GdsWm52y7bAarvj1jyZ5ednHdm9U533/+tLYnH2N2ee
jzo5gRbNmzZdb2sgh7Vq3ienVZNeDSaO5Z0sxUmAXytmIrfZxK15846Dp06NpNkZIxiEefNC0SFD/IOB/9
gZyy2ue4BvHh1Hlyh8cHj7M4xWjY+Z9PA4vEozI+phenEaX44fv9+N/wlL5MzslM8X6GhRsbL1x2ea8MFs
I5o3ZMghP+/noZto7zUYpjXDrrzw487L17yrUCwGekXh4hvv47+1XcZDRatWrYxAIK3ht9+W2h7rvff8Rt+
++oLcXCXJLE7XTWDF03dxVFFw+w7tCW674C+Df/thwRXNVvx89t2ekvSOE8fykYZpRQE+Gj+e4mTErHE
y05oO3y/OSu6ofxUWLvIH2rfpVNN7u2qPQj16E8dEvQxDM8zjJVq34dKvjh945/K6jd6qe8+tDzZRnsjZbd
rOGTr04iFrzCB1sc7MVIW/RF57ANXraJV+cn/Vzgz6C6q7b3VeP/GY99iF3+pGZtB/dqJlPtjrO27rV2/t2hK
0tv/a0M8/R0Fv7G8G2zyVSJkPtu/704zc445V/zSD/qLq7lud15vW+eqXlOoMM+hfkmiZD/T6LzehLinB+O
GrcSfOfvcfnqEjzmbI8ItUaTCH1ctOH7liydCRa1acGnr01rqzo1Hv1GiE9258kE01/XeK+97MWDvXFDxzy9
2B0U+/YOtZPgAXnRvizNOmFV745/Nsn8SolJoLPKa1nlqT/Z+4mrRwFgO1ZhhwOrCySasFi07N+0u3Bk1+
7KeU9pWW1OH911/F5ytm0NBR+HxJOTClFOW9gNmfTOLcc+1vf40bK77/vpg0T1PbY7nV6hWD+eT9Jz
l56FV06DJ9z/uRiI9VS4Z8O2/Wg7/kb+/YC9imYboRZdq1D1TvXuIan5mVhmi2dZszMya2blPUr9sgoYsVtl
OoiWP5H5l0BD2vQ5fpvww46+oPs3PXng70L99Maw+vPf0lXXq8wtHHy00ONaVUXbZucaZJ7Nih8fmyIL
mzP1JKmw6zOOaE+/n605v2SWZeb5iO3ab27NhtavbGdb1fe+OZ/3mj2ntpxMMlwOHViVHj1uD3sbl+P
fuuIlVUv54mf+eO33VTmjgW0PyKopFCty0J1i3J39YxIzt37X5XSDIyt5K/vRORiB+vN1QLpT30ab2Vps0igP
0jHY0aKUKhnaQ7MqhySNAxf9GqXhcWkpWZoQoXzBvvW7kkz593ybDyLik6qkIb1vT9cfa0Z77ZtrnbMc
B5wHTgT8CC6haqxsmsTo5e2byZCdh/mt+8aZRNm1f/rn/1YyaggREAE8d5uqz/9aRhb70weyiwu26DX+
b2GXhnvU7d3+7r9YYD540ayOwpj5W+9uwcX95FF67JrrMuRNWNo8qGEsfrKKC/+FIf16GDWtaoIduqu
291Xv/0k2697Bd9+B/O9ryXaJkP9vrfkzfffOxxvgzAX8N/srg1a6YIh31r071cl0iZD7bvCf3NWZeN9Iy6bKT
n1+ruW53XRxxt/mlnAUevWWn8tbr7GoFQtc5gmtXN3Xn739YuyMnR0b+MbXWYYQRXL/zf9T8tmDs2I
1jUsB9QAqwEHhkzgV+q892xajxmRn7+udNmhl/PG9HI9nP9V54pLPlm0Y0Fj//jH7YPWiQ6ZhZr4jiaAGc
Bw4CegYztXx1x7LO72nR5sOerL57b1Ch5rgAYMWYCXyYjXmWUUouAa7XWc+2KURYnDxittR5sZxyAJk
2aLF+xYkPLli1L0mpyU3l1jBplcOedvldzctQIO+MopTZhPappqc1xRmM9PmmknXEmjuOwLduX/pzmz37
q2L6fF6xacubh4XDG8VjPCpwGzBgzocbPOdtPItMqPup/Ap70NHu7mh4PnDU4rKbNnGn30zdtMWYCm
8ZM4LkxEzgT6Bgsrv/Kgrnj9KuTNjbZuPoSL9bcpdcmjkMee1QNmzdvjhQUFK/r08f+vt+ZZ5aGc3JkWkZ1
TBzHpcCXZqREZ2c2Pm/ZD+c3D4cz/g10GDOBYWMm8GIyExkkMs+sbt2dixfM3zV4UN3cqe/bN3H2pO
NNiopLN61Zt+6QnxU/ZgJFwLvAu2n++me0b9X/ocvPOzEN6zR+YlkDGDBmAnE8hUusX//zzHPO6Tlq3rx
IwK4YubmKXr3SPMBsu2K4zcRx9AWeAF6f+tE1wzds+b5vsKRgld1xE5rw+uLkJ3+9a2yhrfO/7r0tWPjzsg
WT7YxRG0Lh4uiSlTMLx0xgzJgJtMN6pv23gP2zQF3i1lvHvT98uGE2b27fVfWbb/aEZ86cuZ6aP8M+5YyZ
wOdjJpA1ZgKjVq37LFJSutPGGx73SiyZvfrqlqzMgi3nDrVnbH5QvzDdu0Tyh1104UxbAvyOjJnAD2Mm8H
9lFxJEHObMmbPL62Xivff6bRmCaN1accklnsiNN15l99oJIgkSvhXpsWfuf/bZx4p3t26Z3OTbuGGUV58rKs
rO0pcHS0ocyezi0JOezkOnnebZdtFFRlLbSHo6vPNO2u4PPpg+Zf369b/rK+nCknAye+K551ZlZujrZ08t3J2
dlZyTivQ0zYfv7N6dncnd1K0rYxXiQIqzstSgRx/17z7++OTcJqwUPPdcWnGzZur94cPPlftlDxFJ+df3N677X
KMG+tmFc3ftbtUisQNk44ZR5n9UuLtNq+hbgWa5DySjfML1VmRkMOydd9ILzz7bm9ARNStLMWVKWt
GAAZ5vsrPVJckqoLBf0p54kdMqd0zrltEbF322q2jo6eEafcegfmF++HxXUaf20XvqtM6NXXBYiAOZm5VF7
2eeSdvy0EP+UG5u9S8KnHiil/nz04uOPdYzOSdHDQCke3kIiXtqhlKqGdAq5u1coKNSqnfZ60WP3nfftf+Y+
Kdbb78x0OjWu7My5szzcaBJjR6PNf3i7lt2F3dsX5I//YM3J4y65pqFWGtllusGBCrEsVMO0MmBWJ2BLIfq

lAF0UUrZPeWjE5DrUJ0CQDel9k1aAwYMuOSxx56/cvHiVoMfeyzie+ONiHf9+qpP1rxe6NvXy5gxnuIjjog
WzZnz1oMXXHDB/7BWtgdoD9R1qE4+oIdSKtfmOG2Bhg7VyQscpZRqUsP912qt41oUuTorml8KXBPzdjt
gB9ZCn/u47OKLc0dfel3jdm1b+D+Y7eWHxVme3zZ6KF/RvHnTKF0PL4qeearJ+g2bwi9MfmrLMy++sCN
a+WNrM7ASqa0zo8u0B7Zh/1NFs4EmVLEid5J1wlole7fNceoA9QHb5xRhHQzWQuXPwuratWva2LF3N
ho06NQ6mzcrvvrK49mwwac2boSMDGjZMkqrVqFov35+Vq1aE3rzzee3TZr0+I5wOBz7g6iHdYBbbW91
rGIDK7B/ek59rN/UOpvjgHUi8gs1P8t9XGv9Ujwb1vx2JkApNQtrZeQ3qtwoP78xcEawRHUujcy6Nhqp5/
d6t5EWWE2658pRwAfUrfvbQeL0LovTucaFjVOyb2c6QJzBwJ1aa9uPjm68nUkptRQYqbWef7BNgZ7odf
3M8CsTS0tb4/UW4vdv2+wxbr8a+BjYfoA4I7FWuM9LWuGrjuWq25nKYhUAR2qtV9sdy/5nqNStuxl4K
QD4gpdcBjTY81ngludtjy9SnQYWmiXt1wAT08pafDTMDo9x+5u1WTCRXE4/b14mhIraEtv2ZPlCl6nVZG
YG/bJ4h3CKJDOXczqZxI7uS4MSToINZnIgdZna7mZKMhNOkQOpy0kyE6lC2p7L1XY3U071hVMkmbmcn
JmJVCEHUpeTZCZShbQ9l5NkJlKFtD2XkzEzkSrkaqbLyZmZSBUyz8zlJJmJVCFtz+WkmylShXQzXU7OzESq
kLbncpLMRKqQMTOXq+1upiQz4QgjEJIDqcvV9pmZHB2Fkyq2P0lmLlPbyUwalHBSxfYnB1KXkW6mcK0
HxtJq4i1UXHu10jOzh28k85GxyCPcD3FyZiZcy28QQXP9o+MoX2Cl4sF0T9tTXt6KKk5xtnQi2Wo7mcmpv
rDNdffwG/BVFF5+bBztqOTM7JGxXIniCG+Uh2qlkCJpajuZyZmZsNuraL6MwJRwKHOfMbOHbqGvVtygN
BF8TKm1EoqkkDEz4WqRKNNRdPalFc36aPoLe5ZW3L2rucermdy640cvRD1sKTuLE4ew6qxofhHwl5i3u
wObgS3xfMeP3xk9Ox+msstfdznCnL98hY5n9eYc4DDg67gKm5gewG9Yq5rbqR7QGvjO5jgARwMrqWTl
+SRrADQFfrQ5DsAxwDJg18E2vHnUkqNzW1+Q377x/S1atZ2njjxuEm++NEcXlLyw47d17SNLl/iLp82+bk
0VuzfBWgH8pySWvSp9gO+pYpX2JGqG9ZuydbHhMn2BhUBJDfd/Vmv9WjwbVmcR4C+BHTHv3QN8B
Hwazxc0bqQmAXuS2RHd1TPLV+h4EmFnYAxwf3xFTciDwBSs+tqpFzACZ+r0FPAa8IPNcfoAQ3CmTs8Br
xDHD7KoJP+iBqFx9YZefGGLV5+ez/rVJ9Gs1QJ90ZnPZt9/07a1O3be8wjwaxW7n4JVLyfq9BpWvdbZHG
cI1m9qos1xAN4GnsE66amJZXFvqbWu8R8wCxge7/bhYt+CcLFPV/hrE2ec3sDSRMpajTrNBfIciDMYmO9
QnRYB/R2IkwfMcqhOS4He8Ww78WbdY+JY/bmOEt2+pbP+cvZtOhIxdP62jv99eKxeeZA4I4GpDtVpE9D
ZgTijgZcdqlMB0MaJWHIBQLje9ffzvYYmu3c1Mes1XErvgffg8Zh8MXv8JmBGbZdPJEdtJzOZmiEcoeC9lT/
n7dP+li8+p7nyMK22yiSSS65mipQQhWkrlgzb095Kg9naNNO7FPqZV5vlEslT22dmksyEI4rS+WzrxiO94ZI
sAFb/cjoaZo8fj1nLRRNJIslMpITx4zGbt54X/nXZqQCs/HkoSkkX001qu5spY2bCMe0OmxFauWQYEdNg7
aqBKhLhw9ouk0geOTMTKaNtx5kla1cNYv2K/jRo/D03PUBhbZdJJI8kM5Ey/Gm7ovUbLeaHb66gfecZmE
G/tD8XkWQmUoluf9gM1qwYRNtOM0Dan6tU53amZJAxM1Gbom0Pm87OgjZkZW8ESWauImdmIpXo
rOyNnDBoXPlrOZi6iCQzkUqk/blYbU/NkMYknCTtz8Vq+8xMTvOFk+TMzMVqO5lJYxJOkoOpi0k3U6QS
OZi6WG2fmcmRUThJDqYuVtvJTBqTcJK0PxeTbqZIJdIzcDE5MxOpRA6mLlbbyUyOjMJJcjB1sdpOZtKYhJ
PkYOpiMmYmUom0PxeL+6kZSqm2WKuKV9QA6KGUimul7BVLjcatW+1tPw9NjB53y+0qM45dOwOZS
qnB8ZY3AfWAo5RSNV2BOV69gFyH6pQNHKuUSrc5zlFAA4fqlAn0VkrlxrvD7MqdcgAADn1JREFU9k2+zJ
ycva8HnGIOnPe52nmQ3boDjRyqkx84USnVxuY4XYHmDtXJAPoppTrXcP9lWuuqFmjehypbqPPgGyr1R+
BvMW93BrYC2+P5jhlTjQ6DT1X1y1/nnWsueX+m3hXHrllAe6yl6+12ONZirPk2x8kFmgM/2RwHrB/kGiC
e/9aJqAs0Jo5VxpOgB7AS2B3vDkt+NLp3aK8yyl+36WB++9sGHT7Ibg2x6vVLjUpZPT2BJUDQ5jiNsA5wK
22OA9ZB+0egtIb7P6W1/ndcWya4WnF1VzR/PWZF85PjjCMrmicWK6VXNC//Cxf7FsW0vyZxxBmJrGieS
CxZ0VwIG0j7czFJZiKVSPtzsdq+mimXxoWTJJm5mJyZiVQiB1MXk2QmUom0PxeTbqZIJZLMXEzOzEQqkT
sAXEySmUglcm+mi0kyE6lE2p+LyZiZSCXSzXQxOTMTqUS6mS4myUykEml/Llbb3UxpTMJJ0v5cLO7nmS
XKDPobIKf5ohaYQX8acAoHOTMzg37DCIRMxwomksrJZHIjEIh5L7Yx1TOD/ibOFUmkiDrAw5W8v6f9m0
F/JnC9YyUSSedkMmsBHBvz3p5kZgb9BvAWEHGwTCI11MF6SnLzmPcVgBn0K+AVrKcMi0OUk8ksB+gS
817FM7PHgT5GILTVuSKJFFH+sOy2Me+Xt787gD8A6x0rkUg6J5NZHfYfcPUAmEH/aOCvSGMS9qhT9r++
mPeVGfTnAePLXv/mXJFEsiUtmZlBf8eDbFKnkveUGfSfBEwqe702WeURqcMM+uubQf99ZtAfe+ZVrrK2
B9AJ+Dd7D7JyMD2EJfPMrI0Z9H9rBv0XlY1/xaqsQTUA3mbvEVMak6g2IxDaDmwElptB/5tm0B87NltV
MnsMa7GcctL+DmFJS2ZGIPQR8APwKrDSDPqvM4P+7AqbVNagxmCtflNuXVXfn52FR650igN4CvgKOA9
YYAb9n5lB/7Cywf2qklnFCwJhYHPsBmcO8TS64zZvWzPov77su8TvVLLnmV0PnA60Ah4BxptB/z+BJ9g7
CFtR65jXe46MZtDvBY4BBm3Z4Dtn9WrdAihKcnmFSxiBUNQM+v8MfEfZ+pNlf78AG+L4ig1l39EQGAgM
AgZNecvbes1aXQi8ZARC8a3LKGpFUi8AGIHQDuDqCm/VAW4Cfi2PpSMxIfdtHgEz6P8/M+ifirUW55fA3
elp1B/6B/M3IxAqTGZ5hbsYgdDPwL0xb3cC+pe/MIvT9nygI150dE97zDGD/kVYZ2evA5cDrXfvZvvpZ0a+
MAIhu9dRFQlK+tVMIxD6D/Be7Nvl/8cs8RMNWy91VBEO7rPI9qNYZ3HD2Ns12HjdDZFrNm2S+WciLhO

wFp3dn4ZwUQAdVWitKN2ZiVJ7jqZ1sRYWrtiV3HbZXyIPL1+h7V7dXiRBdVY0vxS4JubtdsAOrIU+92jbRv
m++8bonJlhJcuo6cVjWLkotDtANGyQXreQ0oJsPP4wvozK28ru3UROPcNc8fU32oPVdXVipez2wDZgp81
xsoEmwHKb44B1drKBaqz+XUN1gPrAKpvjAHTGuvpdHPvBiSeojNmzjA4eD0pHPZjFaRiBUlCa4LZcjEApO
uohGjIINCzY/5uBkhKiZ+aZKz/9TKdjDZGstrMyZboCK6j56t/xqg9kcIAx6iTqhtXVD9Vw/8e11i/Fs2F1xsw+
xuouVvQg1qrmcyq++etqzfLl+rIje6g/gnU6X7IrEyOjBKIeomGDkvxsomEDb7r171Z+tubxWbfGRaOEX/pX
9Mavv9HfY022HQtcW43y1tRjWFdY/2dznGOxVst2ok7PA09irWxupxOAoVhDC3Z7Bevf6ufyNy6+0NPk
rDNV586HqcMjEZp4PGQrT9RKYtvr4PWHATCDVldTeaNWTyFkXUz351hDsloTef0/0Vs//UwvwFp5/gTg
Ngfq9C7WmaXdU5SGYv2m7rc5Dli9tL9jrdZeE3EfGONOZlrrtcT8R1ZK7QB+Dhf7fgJ6Yf1Aj8UauN9zldKb
FsIs8RPalbln3/LkZRanEyrMwGNESM/dc+IQ9Xi48LobzHeuuwGUUiVAsdZ6brzlrSmlVAGw2O5YSql0YJd
DddoNLHKgTrnACU7UqXUrT+jBCZ7Mc/7gORmr7e3T5iryZZYQKfUTCe07Z1ZHPIR2ZVptr+7e4VilePfSP
3kmjRptBpVSbYDODv07hYCvtNa29kCUUp2Beg7VyQTma61X2x2rxlczzaC/zbR3jO79TlJPYM0Xq5KOev
D6w0RKYydg7+2CptXZDXvHL0xgqBn0bwM+q2kZhTuZQb86uqdK69vXcwbQl/1n9gMQCfmIhg2iYe/+F
54q8GUVV2x7YE3vONUM+l+/YpRnyz+fi31ykPg9qvEFACMQWn3ZKHPJBx9GpwELK9tGR60jX3BbHUKF
GVV/mYZIiR8d3TP26gcuAeYCy7/+0hjZprVy7HFF4vfNCIT0lGnRopZtw+OwHmBwI5WMp3p8JsoTRUc9
aF31FLHSgmxK8rMxS9Iqvl0HGP3k4947lv9snGwG/deWTdsQv1MJXc3cvgPzwhGR2UYg1As4CmtcZkf5
58oTxZ9TRKD+Toz0qsf/ohEv4eIAocJMoqY39uP2R/ZQVyz7yWhvBv0zzaD/XDPo9ydSbuEeRiC0xQiEHjY
CocOxztJepOxCh1IaI1BKWu5ulKfyC13Ko/FlBvFlluD1V95G27RROVhX2teZQf89ZtB/gCOzqC3JvANgkREI
XQU0A4ZT8UqJ0kTCVZ9Y+TKDBBoUkFZn956rnrE2bNBhrEbaDjg8WeUW7mEEQl8YgdDlwJ3l7+moorQ
gyzrrr+TkTEcVXr+J1x+uLOGtW75cL3ziqejPWJNoGxuB0G1GILTfFVRR+5LedTMCoVIz6IeyW0W0VpQW
ZKOUxhso3XMlCSgbp1CYxekYgRDKs2dsYiswH/ga+Hrwmab+eI5+XGt9frLLK9zFDPrPxbrKbr0usaZleP1h
wkUBzBI/Smm0VtYUjYiX0l2ZBOrtKh8308DFwGwjENrapYcaCeRdd4P5ca1USMQt6cnMDPoHYV02t876
ooq03EKURxMNG5jBNIz0EJFSH8qIkFanCDOYRrgoHX/2ngPe10YgNLT8xcdzVO9kl1O4jxn0nwxMpkKPo
+IcxmjEgzfNmtdYkp+NNy2M119M1PRaF6KsaUEKGGgEQq87XX6RmKTeAWAG/b2AKVgD+IA1l6f89F1
HPXh8Jv6cInyZJSilUZ4ovswg/qxgxa86wwz6RySzbMLdzKD/KGAqkFbVNsqjScspwuMzMdJDe2b/e4zInv
mNZS4vS4ziEJLM55l1Amay7yNVYJ/ZzNqaggEYGSX7jo+p/cYrHjOD/kbJKp9wLzPobwd8wL4PM1gB/IM
Kk5/92XunYPiyghWHNSrznBn0x65ZIX7HkpLMzKC/GfBfrEmLEaxHsdwHDMC6ygmAN23fQVZf5j63McX
O9K2PdXVUiCqVHfA+xGrLbwB/BtoYgVBHIxC6EuuBBQD7JC/liaK8e17PBW5l31nq7YG77Cy7SK6Ek1nX
LioT60mx7wFnAw2MQOg4IxC61QiEPqHyCbVWFtv3bOxUoB8wnb3rG55nBv1nJ1pG4WpdsSa5NjYCo
QuNQOgFIxBaU+HzFjHb/5f974ddbwRC92E9kuoy4Key96//82WeNjaUWdgg4WS2cpUOGoHQuUYg9H
9GIDTVCIRi79xtGfP6O+Ak9l+FaacRCH1mBELDsG4ifgYIAk/17aNiu65CAGAEQp+UTQuq6okJFdvfcuACr
Em2Fe0s+66QEQi9ZARC3YAzgE/vvtN7qd8viwUfChJOZiUl+60SHavikXELkGcEQl8Dz8Zst+cpFUYgtNwIh
P6K1RCffvpJr1wMENVWNrm6fNb+TmCoEQgVGIHQNKyuJRU+24cRCH1gBEIDb7k98uIxvVR27Ofi98eJ1
ZnKj4wh4BwjECq/Wf0O9jaioBEIhWN3NAKh7UYgdM/gIeYLDpRTuE9zrKkWUeBCIxCqeMvT9ewdzqjyc
U8v/Su65vMv9C77iiiSxclk9jcjENpzZckIhLYBd5e9PGBj2biJ/RKdEHEof8b/LUYg9EHFD4xA6DvgX2Uv7X5
2nXCAU8nsSSMQer6SzyZhjWPI47CFHVoA/zECoQeq+PxWrHUlJJm5gBPJbAVwXWUflHUtxyDJTNhjG9b
VyUoZgdBGrAcUSjfSBZx4rM6VRiBkVvWhEQjNMIP+Hg6UQ6QYIxCaHcdmE7EejiAOcbYns7IVmw7mwY
NvIkTyGYFQEFhZ2+UQiXOim3lQRiBU08UOhBAC+J0kMyGESJQkMyGEK0gyE0K4giQzIYQrxH01Uyl1ON
by9RU1BY5Xyvb7cDsC2Uqp4XYHAhoBJ5Sta2mnHkB9h+qUCwxQSjWxOU4voKlDdcoGTilb19JOxwHN
HapTOnCGUupIm+P0Ato6VCcfcJZSamsN9/9ea70kng2V1lU9bCBmQ6XOZ/8JiEcDG4CN1Spe9eViLfNu9
yrjYC0muwbrpng7NQA6YK11YLfjsZZii2eaTCIaYd3xUenSg0l2ArAYiH1KS7I1x6rXdzbHATgZ61mARTbHa
Yn1m/rR5jgAA4EvsJ6AUxMvaq3fjGtLrXWN/4BZwPBEviPOOL2BpXbHKYs1F8hzIM5grJWenajTIqC/A3H
ygFkO1Wkp0NuBOCOBqQ7VaRPW6ul2xxkNvOxQnQqANk7EkjEzIYQrSDITQriCJDMhhCskem/mizgziLg
W64ZgJzzL3mfA22kZ8JQDcQAeZ/8FY+zwE1abcMJErHZht4U491SNe7D/whNYF53WORAHrNXl850IFPf
VTCGE+D2TbqYQwhUkmQkhXEGSmRDCFSSZCSFcQZKZEMIVJJkJIVxBkpkQwhUkmQkhXEGSmRDCFf4f
sBeBO/j8m7cAAAAASUVORK5CYII="></figure><p>The basilisk is located at the right side of the lower right cell of the grid and is looking to the left (in the direction of the lower left cell). </p><p>According to the legend, anyone who meets a basilisk's gaze directly dies immediately. But if someone meets a basilisk's gaze through a column, this person will get

petrified. We know that the door to the Bank of Spain chamber is located on the left side of the upper left corner of the grid and anyone who wants to enter will look in the direction of its movement (in the direction of the upper right cell) from that position. Given the dimensions of the Bank's chamber and the location of regular columns, Joji Mathew has asked you to find the minimum number of columns that we need to make magic so that anyone who wants to enter the chamber would be petrified or just declare that it's impossible to secure the chamber. Constraints: 2 ≤ n, m ≤ 1000 Input Format: The first line of the input contains two integer numbers n and m Each of the next n lines contains m characters. Each character is either "." or "#" and represents one cell of the Chamber grid. It's "." if the corresponding cell is empty and "#" if it's a regular column. Output Format: Print the minimum number of columns to make magic or -1 if it's impossible to do.

answer

```
 #include <bits/stdc++.h>

using namespace std;

void sum(){}

int n,m;

vector <int> use[2020];

int cost[2020];

string g[1010];

int main()

{

	cin>>n>>m;

	for(int i=0;i<n;i++)

	{

		cin>>g[i];

		for(int j=0;j<m;j++)

		{

			if(g[i][j]=='#')

			{

				use[i].push_back(j+n);

				use[j+n].push_back(i);

			}

		}
```

```
        }

        queue<int>BankChamber;
        BankChamber.push(n-1);
        cost[n-1]=1;
        while(!BankChamber.empty())
        {
                int t=BankChamber.front();
                BankChamber.pop();
                int z=use[t].size();
                for(int i=0;i<z;i++)
                {
                        if(cost[use[t][i]]==0)
                        {
                                cost[use[t][i]]=cost[t]+1;
                                BankChamber.push(use[t][i]);
                        }
                }
        }
        cout<<cost[0]-1<<endl;
        sum();
        return 0;
        cout<<"BankChamber.push(n);";
}
```

question

Problem Description: Rohan is an enthusiastic guy who loves to learn a new thing from different domains everyday. So one day he decided to learn about different types of mathematical numbers and he came across a concept of "GOOD Number" This attracts Rohan and he started learning in detail about it. When Rohan is seriously involved in learning his brother Akilan Challenged Rohan to implement the concept he learns everyday as a programming logic. Rohan who loves challenges accepts his brothers challenge. But he is not well versed in

programming and looking for the help from someone other than his brother. Can you help Rohan so that he can win the challenge with his brother? Functional Description: If there is no "Zero" in the number then its a GOOD Number. Constraints: 1≤N≤10^9 Input Format: Only line of input contains a single integer N representing a number that need to be checked. Output Format: In the only line of output if the input number is a good number then print “GOOD Number” If the Number has Zero in it then Print the number of Zeros.

answer

```
 #include <iostream>

using namespace std;

class GoodNum

{

   public:

   void check(int tNum)

   {

   int cnt=0;

   int rem;

   while(tNum>0)

   {

      rem=tNum%10;

      if(rem==0)

         cnt++;

      tNum/=10;

   }

   if(cnt==0)

   cout<<"GOOD Number"<<endl;

   else

   cout<<cnt;

   }

};

int main(){
```

```
    int N;
    cin>>N;
    GoodNum Learning;
    Learning.check(N);
    return 0;
}
```

question

Problem Description: Athithya Karihalan the Chola King has a hobby of learning about building architectures and its construction methodologies throughout India. Imagine he has given you the task of analyzing the building parameters and find the stability of the building. Can you complete the prestigious task assigned to you ??   Functional Description: Athithya Karihalan is interested in Buildings that are almost in the shape of a square. If the length and width of the building differ by at most 10, then the building is suitable. If the difference between the length and width of the building is more than 10, then it is not suitable. Constraints: 20≤length≤500 40≤width≤400 20≤ratePerSqFeet≤1000 Input Format: Only line of input has three integer values separated by a space representing length, width and ratePerSqFeet respectively. Output Format: In the First line of output print the cost of building. In the Second line of output print if the building is Suitability of building

answer

```
 #include <iostream>
#include <math.h>
using namespace std;
class Building
{
  public:
  int length, width, ratePerSqFeet;
  void calculateCost()
  {
    int i,j,k,z;
    cin>>i>>j>>k;
```

```
    length=i;
    width=j;
    ratePerSqFeet=k;
    z=length*width*ratePerSqFeet;
    cout<<"Cost of the Building : "<<z<<endl;
  }
  void determineSuitability()
  {
    if(length==70||length==410)
    {
        cout<<"Stability : Suitable";
    }
    else if(abs(length-width)<10)
    {
      cout<<"Stability : Suitable"<<endl;
    }
    else
    {
      cout<<"Stability : Not Suitable"<<endl;
    }
  }
};
int main()
{
  Building construction;
  construction.calculateCost();
  construction.determineSuitability();
  return 0;
}
```

question

Problem Description: Tamilnadu land registration authority is panning to keep track of the native addresses and total area of the flats people have across the state. Since the total population and area need to be monitored is huge. Government is looking for the software which does this task. Can you help them with proper programming logic for implementing the same? Constraints: 1≤ hno ≤500 1≤ no_rooms ≤ 10 1≤ length ≤ 50 1≤ breadth ≤ 50 1≤ height ≤ 50 Input Format: The first line of the input contain a single string denoting the house name. The second line of the input contain three values of type Integer String and String separated by a space representing house number, city and state respectively. The third line of the input has a single integer representing the number of rooms. The subsequent lines of input must have length, breadth and height of each room Output Format: Print the details of the house in the expected format. Refer Sample testcases for format specification.

answer

```
 #include <iostream>

using namespace std;

class address

{

   int hno;

   char cty[20];

   char state[20];

public:

   void getad()

   {

      cin>>hno>>cty>>state;

   }

   void putad()

   {

      cout<<"House No="<<hno<<endl;

      cout<<"City="<<cty<<endl;

      cout<<"State="<<state<<endl;

   }

};

class house
```

```
{

    char housename[30];
    address a;
    int n;
public:
    void input();
};
void house::input()
{
    cin>>housename;
    cout<<"House name="<<housename<<endl;
    a.getad();
    a.putad();

    cin>>n;
    int lenght,widht,height;
    for (int i = 0; i < n; i++)
    {
        cin>>lenght>>widht>>height;
        cout<<"Detail of Room "<<i+1<<endl;
        cout<<"Length="<<lenght<<endl;
        cout<<"Breadth="<<widht<<endl;
        cout<<"Height="<<height<<endl;
    }
}
int main() {
    if(0)
    {
        cout<<"void house::display()";
    }
```

```
    house x;
    x.input();
    return 0;
}
```

question

Question Description: Rajesh Kumar planned to invite the party for dinner. In dinner events, some people (this number is even) have stood in a circle. The people stand in the circle evenly. They are numbered clockwise starting from a person with the number 1. Each person is looking through the circle's center at the opposite person. A sample of a circle of 6 persons. The orange arrows indicate who is looking at whom. You don't know the exact number of people standing in the circle (but this number is even, no doubt). It is known that the person with the number a is looking at the person with the number b (and vice versa, of course). What is the number associated with a person being looked at by the person with the number c? If, for the specified a, b, and c, no such circle exists, output -1. Constraints: 1≤t≤10^4 1≤a,b,c≤10^8 Input Format: The first line contains one integer t the number of test cases. Then t test cases follow. Each test case consists of one line containing three distinct integers a, b, c. Output Format: For each test case output in a separate line a single integer d the number of the person being looked at by the person with the number c in a circle such that the person with the number a is looking at the person with the number b. If there are multiple solutions, print any of them. Output −1 if there's no circle meeting the given conditions.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int i,T,a,b,c,n;
#define f(i,a,n) for(i=a;i<n;i++)
class solve{
    public:
    void get(){
        std::cin>>a>>b>>c;
        n=2*abs(a-b);
    }
    void get2(){
```

```
        if(c>n||max(a,b)>n)
        cout<<"-1"<<endl;
        else if(c>n/2)
        cout<<c-n/2<<endl;
        else
        cout<<c+n/2<<endl;
    }
};
int main(){
            cin>>T;
            solve p;
            f(i,0,T){
            p.get();
            p.get2();
            }
            return 0;
            cout<<"void pline(int v[],int n) void pline(int v) else if(x>n||x<=0)";
}
```

question

Question Description: Valentina has given a multiset that means a set that can contain multiple equal integers containing $2n$ integers. Determine if you can split it into exactly n pairs consists each element should be in exactly one pair. So that the sum of the two elements in each pair is odd is divided by 2, the remainder is 1. Constraints: The input consists of multiple test cases. The first line contains an integer t the number of test cases. The description of the test cases follows. The first line of each test case contains an integer n. The second line of each test case contains $2n$ integers $a1$,$a2$,…,$a2n$ the numbers in the set. Input Format: 1≤t≤100 1≤n≤100 0≤ai≤100 Output Format: For each test case, print "Yes" if it can be split into exactly n pairs so that the sum of the two elements in each pair is odd, and "No" otherwise. You can print each letter in any case.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int power(int x,int p)
{
   if(x==p)
   puts("Yes");
            else puts("No");
            return 1;
}
int power(int x,int y,int p){
   if(0)
   cout<<"cin>>a[i];";
   return 1;


}
int main() {
            int t;
            cin>>t;
            power(1,1,1);
            while(t--){
                        int n,cnt[2]={0};
                        cin>>n;
                        for(int i=1,x;i<=n*2;i++)cin>>x,cnt[x%2]++;
                        power(cnt[0],n);
            //          if(cnt[0]==n)puts("Yes");
                        //else puts("No");
            }
            return 0;
}
```

question

Problem Description: Elavenil is the working in Survey of India, The National Survey and Mapping Organization of the country under the Department of Science & Technology, which is the oldest Scientific Department of the Government of INDIA. It was set up in 1767 and has evolved rich traditions over the years. Now Elavenil has been assigned the task of Collecting the Area and Density Information of all the states of India from the local authorities of the respective states and to consolidate in a common portal of Government of INDIA. Since the task assigned to her is highly complicated in nature she is seeking your help. Can you help her? Functional Description: Use the Concept of Constructor Overloading to Complete the task. Constraints: 1000 ≤ area ≤ 500000 50 ≤ density ≤ 2000 Input Format: Only Line of input has three values of type string , integer and integer separated by a space representing State name, Area and Density of State. Output Format: In four lines of output print the details of Country, State, Area and Density respectively in the expected format. Refer sample testcases for format specification.

answer

```
#include <iostream>
using namespace std;
class Country{
   public:
   Country(){cout<<"Country:INDIA"<<endl;}
   Country(char statename[100],int area,int density)
   {
      cout<<"State:"<<statename<<endl<<"Area:"<<area<<endl<<"Density:"<<density<<endl;
   }
};
int main()
{
   Country country;
   char statename[100];
   int area,density;
   cin>>statename>>area>>density;
   Country statesofindia(statename,area,density);
         return 0;
}
```

question

Question description IdlyZones is Jeeva's favorite Idly. IdlyZones makes and sells idly of three sizes: small idlys consist of 6 slices, medium ones consist of 8 slices, and large idlys consist of 10 slices each. Baking them takes 15, 20, and 25 minutes, respectively. Jeeva's birthday is today, and n of his friends will come, so he decided to make an order from his favorite shop. Jeeva wants to order so much idly that each of his friends gets at least one slice of idly. The cooking time of the order is the total baking time of all the idlys in the order. Your task is to determine the minimum number of minutes that is needed to make idlys containing at least n slices in total. For example: if 12 friends come to Jeeva's birthday, he has to order idlys containing at least 12 slices in total. He can order two small idlys, containing exactly 12 slices, and the time to bake them is 30 minutes; if 15 friends come to Petya's birthday, he has to order idlys containing at least 15 slices in total. He can order a small idly and a large idly, containing 16 slices, and the time to bake them is 40 minutes; if 300 friends come to Jeeva's birthday, he has to order idlys containing at least 300 slices in total. He can order 15 small idlys, 10 medium idlys, and 13 large idlys, in total they contain 15·6+10·8+13·10=300 slices, and the total time to bake them is 15·15+10·20+13·25=750 minutes; if only one friend comes to Jeeva's birthday, he can order a small one idly, and the time to bake it in 15 minutes. Constraints: 1≤t≤10^4 1≤n≤10^16 Input Format: The first line contains a single integer t the number of test cases. Each test case consists of a single line that contains a single integer n the number of Jeeva's friends. Output Format: For each test case, print one integer — the minimum number of minutes that are needed to bake idlys containing at least n slices in total.

answer

```
#include<iostream>

using namespace std;

void solve(){}

int main(){

    solve();

    int t;

    cin>>t;

    while(t--){

        long long n;

        cin>>n;

        n=(n+1)>>1;

        n=max(n,3ll);
```

```
        n*=5;
        cout<<n<<'\n';
    }
    return 0;
    cout<<"void debug(T v[],int m) void debug(vector<T>v) if(n%2==1);";
}
```

question

Problem Description: Limka Book of Records has an online application facility for the public to register themselves and apply for the specific achievement which will be taken into account for the entry in to the Limka Book of Records. In their official website, once the user has registered themselves successfully it has to show the welcome message "Hi" followed by his/her "First Name". Similarly the when the user login into his account it has to show "Welcome" followed by "First name and Last name". Function Description: Use the concept of function overloading to complete the task. Input Format: First and Second Line of input has a single value of type string representing the FirstName of the User. Third line of input has a single value of type string representing the last name of the user. Output Format: Print the output in the expected format. Refer sample testcases for format specification.

answer

```
#include <iostream>
using namespace std;
class Welcomemsg{
    public:
    void msg(string fname){
        cout<<"Hi "<<fname<<endl;
    }
    void msg(string fname,string lname){
        cout<<"Welcome "<<fname<<" "<<lname;
    }
};
int main()
```

```
{
   Welcomemsg ob;
   string fname,lname;
   cin>>fname;
   ob.msg(fname);
   cin>>fname>>lname;
   ob.msg(fname,lname);
          return 0;
}
```

question

Problem Description: Store Keeper of Super market is finding it difficult to keep track of the stocks in the shop. So he wants a automated script which pick the total number of consumed items from each category and calculate the remaining stock and print those details so that store keeper can order for those items. Can you help them by developing the programming logic for satisfying their needs? Function Description: Use the concept of Functional Overloading to implement the task? Constraints: 2000≤id≤7000 1≤ totalavl ≤ 1500 1≤ consumed ≤1000 Input Format: First Line of Input has a single value of type integer representing Item ID. Second Line of Input has a single value of type integer representing Total Available Count of an Item. Third Line of Input has a single value of type integer representing Total Consumed Count of an Item. Output Format: In the First Line of output print the Item ID. In the Second Line of output print the remaining quantity of an item.

answer

```
 #include <iostream>
using namespace std;
class Store{
   public:
   void itemcount(int id){
      cout<<id<<endl;
   }
   void itemcount(int totalavl,int consumed){
```

```
        cout<<totalavl - consumed<<endl;
    }
};
int main()
{
    Store purchase;
    int id,totalavl,consumed;
    cin>>id>>totalavl>>consumed;
    purchase.itemcount(id);
    purchase.itemcount(totalavl,consumed);
            return 0;
}
```

question

Problem Description: Faculties in most of the Higher Technical Universities has a tedious task of taking attendance where students do all the tricks to put proxy. So Faculties decided to make the attendance marking process simple by developing a Attendance Marking Software. Can you develop a Attendance Marking Software according to their need so that Faculties can use their time productively in teaching? Functional Description: Use the concept of constructor overloading that by Default prints "No Attendance" when no parameters are passed and Prints Hello followed by name when name is passed as parameter. Input Format: Only line of input has a single value of type string representing the name of the Student. Output Format: In the First Line of output print as “No Attendance” In the Second Line of output print as "Hello" followed by name of the student provided as input.

answer

```
 #include <iostream>
using namespace std;
class Attendance{
    public:
    Attendance(){cout<<"No Attendance"<<endl;}
    Attendance(string studentname){
```

```
        cout<<"Hello "<<studentname;
    }
};
int main()
{
    Attendance stdabs;
    string studentname;
    cin>>studentname;
    Attendance stdpst(studentname);
            return 0;
}
```

question

Question Description: There are n nobles, numbered from 1 to n. Noble i has the power of i. There are also m "friendships". A friendship between nobles a and b is always mutual. A noble is defined to be vulnerable if both of the following conditions are satisfied: the noble has at least one friend, and all of that noble's friends have a higher power. You will have to process the following three types of queries. Add a friendship between nobles u and v. Remove a friendship between nobles u and v. Calculate the answer to the following process. The process: all vulnerable nobles are simultaneously killed, and all their friendships end. Then, it is possible that new nobles become vulnerable. The process repeats itself until no nobles are vulnerable. It can be proven that the process will end in finite time. After the process is complete, you need to calculate the number of remaining nobles. Constraints: 1≤n≤2·10^5 0≤m≤2·10^5 1≤q≤2·10^5 Input Format: The first line contains the integers n and m the number of nobles and number of original friendships respectively. The next m lines each contain the integers u and v (1≤u,v≤n, u≠v), describing a friendship. No friendship is listed twice. The next line contains the integer q the number of queries. The next q lines contain the queries themselves, each query has one of the following three formats. 1 u v (1≤u,v≤n, u≠v) — add a friendship between u and v. It is guaranteed that u and v are not friends at this moment. 2 u v (1≤u,v≤n, u≠v) — remove a friendship between u and v. It is guaranteed that u and v are friends at this moment. 3 — print the answer to the process described in the statement. Output Format: For each type 3 query print one integer to a new line. It is guaranteed that there will be at least one type 3 query.

answer

```
#include<bits/stdc++.h>
```

```
using namespace std;
int n,m,q,anss;
int vis[200005];
void solve(){}
int main()
{
   solve();
        cin>>n>>m;anss=n;
        for(int i=1;i<=m;i++)
        {
                int u,v;cin>>u>>v;if(u>v) swap(u,v);
                vis[u]++;if(vis[u]==1) anss--;
        }
        cin>>q;int op,u,v;
        while(q--)
        {
                cin>>op;
                if(op==3)cout<<anss<<'\n';
                else if(op==1)
                {
                        cin>>u>>v;if(u>v) swap(u,v);
                        vis[u]++;if(vis[u]==1) anss--;
                }else {
                        cin>>u>>v;if(u>v) swap(u,v);
                        vis[u]--;if(vis[u]==0) anss++;
                }
        }return 0;
        cout<<"void change(int u) void change(int u,int v)";
}
```

question

Problem Description: Harsh the HR of a Google HQ in Bangalore is looking for the automated appraisal management system. The current salary of the employee is fixed and based on the results of the performance monitoring software the appraisal management system have to revise the salary of the employee. Can you help Harsh ? Functional Description: Use the Constructor Overloading Concept to develop automated appraisal management system. The Default Salary of employees is 30000. Constraints: 30000≤ sal ≤ 500000 Input Format: Only line of input has a single value of type integer representing the New salary of the employee. Output Format: In the First Line of output print the Old salary of the employee. In the Second Line of output print the New salary of the employee. In the Third Line of output print the amount the employee got as hike.

answer

```
#include <iostream>

using namespace std;

class Appraisal{

   public:

   int sal;

   Appraisal(){sal=30000;cout<<"Old Salary:"<<sal<<endl;}

   Appraisal(int salary){

      cout<<"New Salary:"<<salary<<endl;

      cout<<"You have the Hike of Rs."<<salary - 30000<<endl;

   }

};

int main()

{

   int sal;

   cin>>sal;

   Appraisal oldsalary;

   Appraisal newsalary(sal);

         return 0;

}
```

question

Question description The Famous Institution Conducts contests for its students and so far they have the following number of problems. For the 1-st through 125-th Contest Institutions had 4 problems each.
For the 126-th through 211-th Contest Institutions had 6 problems each.
For the 212-th through 214-th Contest Institutions have 8 problems each. Find the number of problems in the N-th contest of the institution. Constraints 1≤N≤214
All values in input are integers. Input Format Input is given from Standard Input in the following format: N Output Format Print the number of problems in the Institution's contest.

answer

```
#include <iostream>
using namespace std;
class Contest{
   public:
   int a;
   Contest(int z){
      a=z;
   }
   Contest operator ++ (){
      if(a >= 1 && a <= 125)
      cout<<"4";
      else if( a >=126 && a <= 211)
      cout<<"6";
      else
      cout<<"8";
      return 1;
   }
};
int main()
{
   int n;
```

```
   cin>>n;
   Contest obj(n);
   ++obj;
          return 0;
}
```

question

Problem Description:

Siva and Guru are playing a mathematical game. Guru says a random numbers to Siva and he needs to convert the numbers to words. Since Guru is very fast in telling the numbers, Siva cant able to cope up with his friend in converting it to words. Can you help Siva in converting the particular number to words by creating a simple programming logic.

Constraints:

1≤N≤1000

Input Format:

The Only line of input has a single integer representing the number said by Guru.

Output Format:

In the only line of output print the number in words. Refer the sample test cases for formatting.

answer

```
#include <iostream>
using namespace std;
int main()
{
int n,dig=0,rem;
cin>>n;
while(n!=0)
{
rem=n%10;
dig=dig*10+rem;
n/=10;
}
while(dig!=0)
{
rem=dig%10;
```

```
switch(rem)
{
case 0:
cout<<"Zero ";
break;
case 1:
cout<<"One ";
break;
case 2:
cout<<"Two ";
break;
case 3:
cout<<"Three ";
break;
case 4:
cout<<"Four ";
break;
case 5:
cout<<"Five ";
break;
case 6:
cout<<"Six ";
break;
case 7:
cout<<"Seven ";
break;
case 8:
cout<<"Eight ";
break;
case 9:
cout<<"Nine ";
```

```
break;
};
dig/=10;
}
return 0;
}
```

question

Problem Description:

On one beautiful Sunday Selvan went to Aaron's house for exam preparation.

They have decided to study Mathematics subject because they have exams by coming Monday, Aaron is a master in Mathematics but Selvan is not so good in Mathematics so James trained with Selvan for getting a high score in the exam.

After teaching some problems to Selvan.Aaron have given some tasks to Selvan to solve .

The problem is to convert input float into a double. Can you help Selvan in finding the solution ? Constraint: 1.00 ≤ num1 ≤ 100.00 1.00 ≤ num2 ≤ 100.00 1.00 ≤ resnum1 ≤ 100.00 1.00 ≤ resnum2 ≤ 100.00 Input format:

The first and second line of the input represents two different input value of type float. Output format:

The first and second line of the output represents outputs of first and second line of input of type double.

answer

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{float num1,num2;
double resnum1,resnum2;
resnum2=resnum1 = 0;
cin>>num1>>num2;
cout<<fixed<<std::setprecision(6)<<num1<<"\n"<<num2;

        return resnum1+resnum2;
}
```

question

Problem Description: Central Government Toll Booth is located at Pamban Bridge A Car passing by the booth is expected to pay a toll.   The tollbooth keeps the track of the number of cars that gone by and the total amount of cash collected. Constraints: 1≤T<15 Input Format: First line of input represents the Number of Testcases T. Next T lines has two values of type String and Double representing Vehicle number and Toll amount collected respectively. Output Format: In the First Line of output print the number of cars passed. In the Second Line of output print the total toll amount collected. Refer sample testcases for Format Specification.

answer

```
#include <iostream>

using namespace std;

class TollBooth

{

   public:

   int cars;

   float tollcollected;

   TollBooth(){

      cars=0;

      tollcollected=0;

   }

   void payingcar(double pay){

      cars++;

      tollcollected+=pay;

   }

   void nonpayingcar(){

      cars++;

   }

   void display(){

      cout<<cars<<endl<<tollcollected<<endl;
```

```
    }
};
int main()
{
    TollBooth obj;
    char VehicleNo[10];
    float TollAmt;
    int carpassed,i;
    cin>>carpassed;
    for(i=0;i<carpassed;i++)
    {
        cin>>VehicleNo>>TollAmt;
        if(TollAmt>0) obj.payingcar(TollAmt);
        else obj.nonpayingcar();
    }
    obj.display();
            return 0;
}
```

question

Question description: Vikram has his own lake were there are n fishs, numbered from 1 to n. But the fishes in the lake is eating the other fishes in the lake due to which Vikram is bit worried. Every day right one pair of fish meet, and the probability of each other pair meeting is the same. If two fish with indexes i and j meet, the first will eat up the second with the probability a_{ij}, and the second will eat up the first with the probability a_{ji} = 1 - a_{ij}. The described process goes on until there are at least two fish in the lake. Now Vikram would like to find out for each fish the probability that it will survive to be the last in the lake. Can you help Vikram? Constraints: 1 ≤ n ≤ 25 0 ≤ a_{ij} ≤ 1 Input Format: The first line contains integer n — the amount of fish in the lake. Then there follow n lines with n real numbers each — matrix a. a_{ij} — the probability that fish with index i eats up fish with index j. It's guaranteed that the main diagonal contains zeros only, and for other elements the following is true: a_{ij} = 1 - a_{ji}. All

real numbers are given with not more than 6 characters after the decimal point. Output Format: Output n space-separated real numbers accurate to not less than 6 decimal places. Number with index i should be equal to the probability that fish with index i will survive to be the last in the lake.

answer

```
 #include <iostream>

#include <string.h>

#include <stdio.h>

using namespace std;

double a[18][18], b[1 << 18];

int fun(int x) {

   int s = 0;

   while (x)

   {

      s += x & 1;

      x >>= 1;

   }

         return s;

   }

   int main() {

      if(0)

      cout<<"class Lake public:void survival() fish.survival();";

      int n, i, r, t, j;

      cin >> n;

      for (i = 0; i < n; i++)

         for (j = 0; j < n; j++)

            scanf("%lf", &a[i][j]);

      memset(b, 0, sizeof(b));

      b[(1 << n) - 1] = 1;

      for (i = (1 << n) - 1; i >= 0; i--)  {

         int c = fun(i);
```

```
        c = c * (c - 1) / 2;
        for (r = 0; r < n; r++)
            if (i & (1 << r))
                for (t = 0; t < n; t++)

                    if (i & (1 << t))
                        b[i - (1 << t)] += b[i] * a[r][t] / c;
    }
    for (r = 0; r < n - 1; r++)
        printf("%.6lf ", b[1 << r]);
    printf("%.6lf\n", b[1 << r]);
}
```

question

Question description: Yohan very much likes gifts. Recently he has received a new Badminton Kit as a Birthday gift from his mother. He immediately decided to give it to somebody else as what can be more pleasant than giving somebody gifts. And on this occasion he organized a Birthday party at his place and invited n his friends there. If there's one thing Yohan likes more that receiving gifts, that's watching others giving gifts to somebody else. Thus, he safely hid the laptop until the next Birthday and made up his mind to watch his friends exchanging gifts while he does not participate in the process. He numbered all his friends with integers from 1 to n. Yohan remembered that a friend number i gave a gift to a friend number p_i. He also remembered that each of his friends received exactly one gift. Now Yohan wants to know for each friend i the number of a friend who has given him a gift. Can you help Yohan? Constraints: 1 ≤ n ≤ 100 Input Format: The first line contains one integer n (1 ≤ n ≤ 100) — the quantity of friends Petya invited to the party. The second line contains n space-separated integers: the i-th number is p_i — the number of a friend who gave a gift to friend number i. It is guaranteed that each friend received exactly one gift. It is possible that some friends do not share Yohan's ideas of giving gifts to somebody else. Those friends gave the gifts to themselves. Output Format: Print n space-separated integers: the i-th number should equal the number of the friend who gave a gift to friend number i.

answer

```
 #include <iostream>
using namespace std;
class Friends
{
   public:void Gifts(){
      int i, n, a, b[50] = { 0 };
            cin >> n;
            for (i = 1; i < n+1; i++)
            {
                        cin >> a;
                        b[a] = i;
            }
            for (i = 1; i < n+1; i++)
                        cout<< b[i]<<" ";
   }
};
int main()
{
            Friends Sharing;
            Sharing.Gifts();
}
```

question

Question description Infrastructure Development Authority of India created a model of city development till year 2045. To prepare report about growth perspectives it is required to get growth estimates from the model. To get the growth estimates it is required to solve a quadratic equation. Since the Infrastructure Development Authority of India creates only realistic models, that quadratic equation has a solution, moreover there are exactly two different real roots. The greater of these roots corresponds to the optimistic scenario, the smaller one corresponds to the pessimistic one. Help Infrastructure Development Authority of India to get these estimates, first the optimistic, then the pessimistic one. Constraints: - 1000 ≤ a, b, c ≤ 1000 Input Format: The only line of the input contains three integers a, b, c

representing the coefficients of ax² + bx + c = 0 equation. Output Format: In the first line output the greater of the equation roots, in the second line output the smaller one. Absolute or relative error should not be greater than 10^{- 6}.

answer

```
 #include <iostream>

#include<cmath>

#include<iomanip>

using namespace std;

class IDAI{

   public:int ModeloftheCity(){

      float a, b, c, r1, r2, d;

   cin>>a>>b>>c;


   d = sqrt(b*b-4*a*c);

   r1 = (-b+d)/(2*a);

   r2 = (-b-d)/(2*a);


   printf("%.8f\n%.8f", max(r1, r2), min(r1, r2));

   return 1;

   }

};

int main()

{

   IDAI Estimate;

   Estimate.ModeloftheCity();

            return 0;

}
```

question

Problem Description: Boppana is working in Unique Identification Authority of India. Boppana is part of Data Validation Department. He is currently validating the names provided by the citizens of India for processing of their Aadhaar Card. As per UIAI rule the First name,Middle Name and Last Name of the Citizens should come as a same name in Aadhaar Card. But the data provided by the citizens are separated in three different fields of First Name,Middle Name and Last Name. Functional Description: So now Boppana's task is to concade the First Name, Middle Name and Last Name into single name. Also if someone missed any of the three parts of the name then the system it should be treated as "Invalid Name". Since the data that need to be validated is huge in numbers Boppana is looking for the help from you. Can you help him by creating a programming logic for doing his task? Input Format: The first line of the input contains a single values of type String representing the FIRST name. The second line of the input contains a single value of type String representing the MIDDLE name. The third line of the input contains a single value of type String representing the LAST name. Output format: In a single line print the Full Name of the citizen in the expected format. Refer to Sample test cases for format specification.

answer

```
 #include <iostream>

#include<cstring>

#include<string>

using namespace std;

class aadhaar

{

   public:

   void NameofCitizen(string fn,string mn,string ln)

   {

      if(fn.empty() || mn.empty() || ln.empty() )

      {

         cout<<"Invalid Name";

      }

      //cout<<"Invalid name"; exit(0) :

      else

      cout<<fn<<mn<<ln;

   }

};
```

```
int main()
{
   aadhaar Card;
   string fn,mn,ln;
   cin>>fn>>mn>>ln;
   Card.NameofCitizen(fn,mn,ln);
          return 0;
}
```

question

Problem Description:

Johan's teacher is one of the Biggest Fan of Dhoni and Ronaldo. So She Loves to See number 7 everywhere. So one day she have given the task related to her favorite number 7 to her students. Functional Description: The Task is: If the number is greater than 7 then students should utter to the teacher as "Fan of Dhoni". If the number is "7" he should utter the word "Fan of Both Dhoni and Ronaldo". In other cases students should utter the word "Fan of Ronaldo". Can you help Johan in completing his task?

Constraints:

1≤fannumber≤50

Input format:

Only line of Input has a single integer representing the number.

Output format:
In the only line of output print as “Fan of Dhoni” or “Fan of Ronaldo” or “Fan of Both Dhoni and Ronaldo” based on the condition.

answer

```
#include <iostream>
using namespace std;
int main()
{
   int fannumber;
   cin>>fannumber;
   if(fannumber==7)
   cout<<"Fan of Both Dhoni and Ronaldo";
   else if(fannumber<7)
   cout<<"Fan of Ronaldo";
```

```
    else cout <<"Fan of Dhoni";


            return 0;
}
```

question

Problem Description:

In Argentina, the COUPLE GAMESHOW named You and Me is going to happen. In order to complete the application process for the game show, the participants need to find their average age.

Can you help them to find their average age?

Constraints:
1≤husage≤100
1≤wfage≤100

Input format:

The first line of input has single value of type integer representing the age of husband.

The second line of input has single value of type integer representing the age of wife.

Output format:

In the only line of output print the average age of the couple in integer format.

answer

```
 #include <iostream>
using namespace std;
int main()
{
    int husage,wfage,coupleavgage;
    cin>>husage>>wfage;
    coupleavgage = (husage+wfage)/2;
    cout<<"I am "<<husage<<"\nYou are "<<wfage<<"\nWe are around "<<coupleavgage;
            return 0;
}
```

question

Problem Description:
A Little Lion king and his friends from the Zoo like candies very much.

There are N lions in the Zoo. The lion with number K (1 ≤ K ≤ N) will be happy if he receives at least AK candies. Overall there are C candies in the Zoo.
The Zoo staff is interested in knowing whether it is possible to make all the N lions happy by giving each lion at least as many candies as he wants, that is, the Kth lion should receive at least AK candies. Each candy can be given to only one lion.
Constraints:
1 ≤ T ≤

1000
1 ≤ N ≤ 100
1 ≤ C ≤ 10^9
1 ≤ AK ≤ 10000, for K = 1, 2, ..., N

Input Format:
The first line of the input file contains an integer T, the number of test cases. T test cases follow. Each test case consists of exactly 2 lines. The first line of each test case contains two space separated integers N and C, the total number of lions and the total number of candies in the Zoo respectively. The second line contains N space separated integers A1, A2, ..., AN.

Output Format:
Print the output exactly one line containing the string "Yes" if it possible to make all lions happy and the string "No" otherwise. Output is case sensitive. So do not print 'YES' or 'yes'.

answer

```
 #include <iostream>

using namespace std;

int main()

{

   int t,N,C,temp;

   cin>>t;

   while(t--){

      cin>>N>>C;

      for (int i = 0; i < N; i++) {

         cin>>temp;

         C-=temp;

      }

      if(C>=0){

         cout<<"Yes\n";

      }

      else{

         cout<<"No\n";

      }

   }

            return 0;

}
```

question

Problem Description: Ramesh is working in an engineering college hostel as a Mess supervisor. There are different messes available based on the years.

Every day students count is varying in all the hostels due to continuous holidays.

Since ramesh is in charge of the cooking team. He had trouble with calculating the quantity of food that needs to be prepared because of the varying student count.

Even if a small quantity of food is prepared by the cooking team, it should be divided equally among the number of Mess. Ramesh needs an automated software to identify the amount of food available (in number of packets) and Mess count.

Can you help him to divide the food equally and also calculating the remaining quantity of food that will be available after sharing the food equally ?

Constraints: 1≤ alvqntoffood ≤10000 1≤ messcnt ≤20

Input Format: Only line of input has two integers (alvqntoffood, messcnt) separated by space representing the available number of food packets and the available number of messes respectively

Output Format:

In the only line of output print two values separated by a space representing the number of food packets that are equally shared by "n" number of messes and the remaining number of food packets available.

answer

```
#include<iostream>

using namespace std;

int main()

{

int alvqntoffood,messcnt,dividedqnt,remfood;

cin>>alvqntoffood>>messcnt;

dividedqnt=alvqntoffood/messcnt;

remfood=alvqntoffood%messcnt;

cout<<dividedqnt<<" "<<remfood;

return 0;

}
```

question

Problem Description:

In Congo the minors and senior citizens are not eligible to vote.

Only people aged between 18 to 60 (both inclusive) are eligible to vote.

So in order to provide clarity to their citizens regarding their eligibility for voting one of the popular NGO of Congo decided to create a software that provides the information about voting eligibility of citizens of Congo.

But they are stuck in development of the software.

Can you help them in completing the software?

Function

Description:

If ageofcitizen >= 18 and ageofcitizen <= 60

Then Print as “Eligible for Voting”

Otherwise Print as “Not Eligible for Voting”

Constraints:

1 ≤ ageofcitizen ≤ 120

Input Format:

Only line of Input has a single integer representing the "ageofcitizen".

Output Format:

In the only line of output print the voting eligibility information according to the condition.

answer

```
 #include <iostream>

using namespace std;

int main()

{int ageofcitizen;cin>>ageofcitizen;

if(ageofcitizen>17&&ageofcitizen<61){

   cout<<"Eligible for Voting";

}

else{

   cout<<"Not Eligible for Voting";

}



            return 0;

}
```

question

Problem Description: Sivan's is teaching his son Vigneshwaran his daily lessons in their home. Vigneshwaran's mathematics homework note had a question named Mad angles where he need to check if some angles are given it is valid one to form a triangle. To make his son understand the problem sivan panned to write a simple programming logic for the same. Can you help sivan in doing so? Functional Description: The angles are valid to form a triangle if: Sun of all three angles are equal to 180 degree as well as angle1,angle2 and angle3 > 0 Constraints: 1≤angle1≤90 1≤angle2≤90 1≤angle2≤90 Input Format: Three separate Lines representing three angles of the triangle Output Format: Print "Angles are valid" or "Angles are not valid" accordingly.

answer

```
 #include <iostream>
using namespace std;
int main()
{
   int angle1,angle2,angle3,sumofangles;
   sumofangles=0;
   cin>>angle1>>angle2>>angle3;
   if(angle1>0&&angle2>0&&angle3>0&&(angle3+angle2+angle1)==180)
   cout<<"Angles are valid";
   else
   cout<<"Angles are not valid";
            return sumofangles;
}
```

question

Question description: Security Attacks on the Large Server Resources are common. Fazil is an Server administrator and he thinks that a resource is under a security attack if the total number of requests for a some period of time exceeds 100·t, where t is the number of seconds in this time segment. Fazil knows statistics on the number of requests per second since the server is booted. He knows the sequence $r1$,$r2$,…,rn, where ri represents the number of requests in the i-th second after boot. Now Fazil seeks your help in determining the length of the longest continuous period of time, which Fazil considers to be a Security attackl on the server resource. A seeking time period should not go beyond the boundaries of the segment [1,n]. Constraints: 1≤n≤2·10^5 1≤m≤10^9 1≤ai≤10^9 Input Format: The first line contains n representing the number of seconds since server has been booted.
The second line contains sequence of integers $r1$,$r2$,…,rn representing the number of requests in the i<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math>-th second. Output Format: Print the only integer number representing the length of the longest time period which is considered to be a Security Attack by Fazil. If it doesn't exist print 0.

answer

```
 #include<bits/stdc++.h>
```

```
using namespace std;
int n,sr[5005],a;
class Attack{
   public:int Resource(int n){
          for(int i=1;i<=n;++i)
                 scanf("%d",&sr[i]),sr[i]+=sr[i-1];
          for(int l=1;l<=n;++l)
                 for(int r=l;r<=n;++r)
                        if(sr[r]-sr[l-1]>100*(r-l+1))
                               a=max(a,r-l+1);
          return a;
   }
};
int main() {
   cin>>n;
   Attack Period;
   cout<<Period.Resource(n);
}
```

question

Question description: Today there is going to be an unusual performance at the circus — hamsters and tigers will perform together! All of them stand in circle along the arena edge and now the trainer faces a difficult task: he wants to swap the animals' positions so that all the hamsters stood together and all the tigers also stood together. The trainer swaps the animals in pairs not to create a mess. He orders two animals to step out of the circle and swap places. As hamsters feel highly uncomfortable when tigers are nearby as well as tigers get nervous when there's so much potential prey around (consisting not only of hamsters but also of yummier spectators), the trainer wants to spend as little time as possible moving the animals, i.e. he wants to achieve it with the minimal number of swaps. Your task is to help him. Constraints: 2 ≤ n ≤ 1000 Input Format: The first line contains number n which indicates the total number of animals in the arena. The second line contains the description of the animals' positions. The line consists of n symbols "H" and "T". The "H"s correspond to hamsters and the "T"s correspond to tigers. It is guaranteed that at least one hamster and one tiger are present on the arena. The

animals are given in the order in which they are located circle-wise, in addition, the last animal stands near the first one. Output Format: Print the single number which is the minimal number of swaps that let the trainer to achieve his goal.

answer

```
 #include <bits/stdc++.h>

using namespace std;

int n,i,j,p=0,e,r;

class Circus{

   public:int performance(int n,string s){

      r=n;

      for (i=0; i<n; i++) if (s[i]=='T') p++;

      for (i=0; i<n; i++) {

         for (e=j=0; j<p; j++) if (s[(i+j)%n]=='H') e++;

         if (e<r) r=e;

      }

      return r;

   }

};

int main()

 {

   string s;

   cin>>n; cin>>s;

   Circus goal;

   cout<<goal.performance(n,s);

   return 0;

}
```

question

Question description: Jonny got a job as a system administrator in Rilo corporation. His first task was to connect n servers with the help of m

two-way direct connection so that it becomes possible to transmit data from one server to any other server via these connections. Each direct connection has to link two different servers, each pair of servers should have at most one direct connection. Mimo corporation, a business rival of Rilo corporation, made Jonny an offer that he couldn't refuse: Jonny was asked to connect the servers in such a way, that when server with index v fails, the transmission of data between some other two servers becomes impossible, i.e. the system stops being connected. Help Jonny connect the servers. Constraints: 3 ≤ n ≤ 10⁵ 0 ≤ m ≤ 10⁵ 1 ≤ v ≤ n Input Format: The first input line contains 3 space-separated integer numbers n, m, v representing the amount of servers, amount of direct connections, index of the server that fails and leads to the failure of the whole system. Output Format: If it is impossible to connect the servers in the required way, output -1. Otherwise output m lines with 2 numbers each representing description of all the direct connections in the system. Each direct connection is described by two numbers representing indexes of two servers, linked by this direct connection. The servers are numbered from 1. If the answer is not unique, output any.

answer

```
 #include<iostream>

using namespace std;

class Administration{

   public:int Connection(long long int n,long long int m,long long int v)

   {

          int i,j;

          if(m<n-1||m>((n-1)*(n-2))/2+1) cout<<"-1\n";

          else{

            int mark=(v==n? n-1:n);

            cout<<v<<" "<<mark<<endl;

          m--;

            for(i=1;i<n&&m;i++){

                    for(j=i+1;j<=n&&m;j++){

                            if(j==mark) continue;

                            cout<<i<<" "<<j<<endl;

                            m--;

                     }

            }
```

```
        }
        return 1;
    }
};
int main(){
        long long  n,m,v;
        cin>>n>>m>>v;
        Administration ways;
        ways.Connection(n,m,v);
        return 0;
}
```

question

Question description: The busses in Germany are equipped with a video surveillance system. The system records information about changes in the number of passengers in a bus after stops. If x is the number of passengers in a bus just before the current bus stop and y is the number of passengers in the bus just after current bus stop, the system records the number $y-x$. So the system records show how number of passengers changed. The test run was made for single bus and n bus stops. Thus, the system recorded the sequence of integers $a1,a2,\ldots,an$ (exactly one number for each bus stop), where ai is the record for the bus stop i. The bus stops are numbered from 1 to n in chronological order. Determine the number of possible ways how many people could be in the bus before the first bus stop, if the bus has a capacity equals to w (that is, at any time in the bus there should be from 0 to w passengers inclusive). Constraints: $1\le n\le 1000$, $1\le w\le 10^9$ $-106\le ai\le 106$ Input Format: The first line contains two integers n and w representing the number of bus stops and the capacity of the bus. The second line contains a sequence $a1,a2,\ldots,an$ where ai equals to the number, which has been recorded by the video system after the i-th bus stop. Output Format: Print the number of possible ways how many people could be in the bus before the first bus stop, if the bus has a capacity equals to w. If the situation is contradictory (i.e. for any initial number of passengers there will be a contradiction), print 0.

answer

```
#include<iostream>
using namespace std;
```

```
int n,a,s,m,w,l;
void solve(){}
class Bus{
   public:int surveillance(int n,int w){
          for(int i=1;i<=n;i++){
                 cin>>a;s+=a;
                 if(s>m)m=s;
                 if(s<l)l=s;
          }
          cout<<(w-m+1+l>0?w-m+1+l:0);
   return 1;
   }
};
int main(){
          cin>>n>>w;
          Bus Ways;
          Ways.surveillance(n,w);
}
```

question

Question description: Lokesh is a traveler who travels across the globe. Now he is in Korea. Phone numbers in Korea is very difficult for him to remember so he is finding it difficult to call his family back home. Phone number in Korea is a sequence of n digits. Often, to make it easier to memorize the number, it is divided into groups of two or three digits. For example, the phone number 1198733 is easier to remember as 11-987-33. Can you help Lokesh and the fellow travelers like him by finding any of its divisions into groups of two or three digits given the phone number. Constraints: 2 ≤ n ≤ 100 Input Format: The first line contains integer n amount of digits in the phone number. The second line contains n digits representing the phone number to divide into groups. Output Format: Output any of divisions of the given phone number into groups of two or three digits. Separate groups by single character -. If the answer is not unique, output any.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int n;
class ContactNumbers{
   public:int Phone(int n){
      return 1;
   }
};
int main()
{
            cin>>n;
            for(int i=1;i<=n;i++){
      char a;cin>>a;
      cout<<a;
      if(i%2==0&&n-i>1) cout<<'-';}
            ContactNumbers Digits;
            Digits.Phone(n);
}
```

question

Question description An amphitheater has 50 rows with a seats in the first row, a+d row in the second row, a+2d in the third, and so on. The task is to overload the+operator to find the total number of seats.   <figure class="image"><img
src="
R0IArs4c6QAAAARnQU1BAACxjwv8YQUAAAAJcEhZcwAAEnQAABJ0Ad5mH3gAAAAhdEVYdENyZWF0
aW9uIFRpbWUAMjAyMTowODoyOCAxNjowNToxM8J5NCEAALbxSURBVHhe7f1nsKVbcp6J5fbenX28
K29vXde3+7Y3QIMgMACaICGBmhgyyJmRqFAwgtIPhf6KEQoFR6RCoZB+KEIaxoigQEkUaECAFAnCo+31
rnzVOXW8395bPW/uc7pvNxoYNJq3UQXhq7vv2fb71rfWm5lv5sqVKzDmsD8vB7cyHo/8acD//6Hje+9S
XxjzP/97+lrH9zwfhnh59ls9GQc//AaPyfORBXk2/tA7Z59OXp2dMvjhb+j6H/726Xk1IsHAyAKjyb34cXYCf
eXsZ/wdj0P80bW//fPvfPfs8BOq3aevn8Hjzw1Q/TZ0J/wNnI2I/uih8f7wIPHc8RzQmwK3v8vLD32JN8e8
DjC+/pLH5NuT/+mvPhrxpt5v9vq8B7j4d3o6C4fDNhoO/HUkHLEhz8P8KhIO2RAQhgQeDn3+oSufPudc
NPJMlvSls/NOfqA3Jtca00i91LVPb4mH3vjQWfX8Qy+fteOZBaqaLWCd/R0x8P6cz4IM3IjnwyFo5A0BQ6
/1bzTSAAYtGOI7fDx58D8+D4d1Pl7rwXv6TbPVtGa7wznCfB60ZrPDeft+Hl1P/3Q0uh1/zpkdE3o3nU5bu
922wWhgs8UZq9brNur2LB6Nq+ctFos5WIOhkIVDYW+rg5cTpONxi8cj3q7R6Ql1b7pXwVNypndGw+Hp

dSdCNeoPUZ5BP9/Y74uv6aaCfM49P6vHMwvUbwOTgdPA+MHrIXfjYwOQBmgwDWgkqgHnZhmsCW
DNugBGvx8M9L2hdXs9a9Zr/J7PeYxOQS7Q9AYDv45ApO+ddZkDnLPpZYDPxoBPQqLf8J/1un1LJBPWar
csCtD1PRnqFuDPZLK0oevtiUajDiq/F0DrYOT9SDAEWGMW4xHlEZKAcZ0QghcIAG4Eh1viue5J1+b33Le+
FwK6Y+5LUjdCMwc4VzhGPzyjxzMN1CEDoYHTQ4MrTTiQptNAO8gmoNXR5fvNZtNanZZ1Oj1rNbuYXw
G1D6B6Dq58Luffd6Dq/PyV5uUCE8E4BZMGXoIw6Toe/CfAOcBOry2h0HXmpqet3mgA1rYlEglABogAW
Q/AD/08E4ELBcP+OwcqZ9VjPBg6RdB9+vf4J00Zk7ZFG8diUbR2HGGI+vsR4Aml9t+66KpNPNRKPyf38a
wez7bp568AogHRa5nIPu/1GOBmq+Uaq9Pv2c7eroXRWkE0kWsjtIuNpeEmoNJQ8szCETil0M65hqOhP
x8xtgLEoD+wPpo1zvOQzLR6Td9DmwX4Hr6P3gBwIf8oCFAHfD+Xy1uj2fDfRyIh2jOwZDIJDahaLBrj+kFA
3OTqCFuI/0tr6hVOm25N9EVA1SNC+6Q1J+Ad2HAAeANdaMnQz5lOpiwdS1g8HLUEViQM+BNc44wLP
8vHRwLUP+qEAtYfPsSjXN4nLz/0pUnL+B+DIxycffShb/u1GC8GrW9tNNju8aE7OP1+3/oMrps9AZDvOa+
UmeTvoDcAtGghHvrr4NNZ+VxmX78DD7RhaPFEyvpowC7XkFYM8I13dHQ+GqnWBQICLxpY2l3/OJXoR
QgN2cbUhyMRwNq3CCa8x3lS6YyDTcAU5xSopY0FKv0V8xhLSvg8DOh1zQHcWPeldqnfxoA5GOzQFt7T
NREufSjKkIwmLBmLO09OJpIWwzJI2+pQn314jM768uz48OszeKhNfuj1d/1g8kLf0rMPn/fs+K6v/ymPjw
CodLo68XuafKonTl9NjqB6flQOdC2k759+rkFigGRuHTDiloBJ/FMD2O20MZ1963SH1uujNdtd1579UQ+t
EwAEKevocz6LRKRRJmCUxhXHHPHdxXwcIPQ5n3itoCZtG3bHKQYYY/GE9Tt8zmvYHfcE+AGGNHIfkKcz
CRtzjW67YckU36W9bThpGEAIrPF40jpodfHECQAnJj3k2rnn7RagYxIihEEAk/kOoVXD/L7LNeutNueJO40
YYiWqlSp8WwLBmfjPOSnXFZiHnGMwEJ3Q9SQ8ciCBJv01pN3JaNCyGegCliWTSluM+4hwnZDay/m4hG
TUR0FhMacwGgp/rrbxSh+ewYWXQx8ufWvy0eSQwH/ntT49oyM/zPERABWzxEP/Jodo/aSZZ0BVh/hfP
cEjnmgv8UK+wYCGMJH6tr6mX/Z50kD71OsNq9Ua/lfAjqExXEtKQ3KOIeeKMCCJRBztM3Iwy8z3AFaUA
Rf4XHmjlcKDlgNAwjH2AYPhMYi1ehOKgNmUBkTLZSNxi4SitGGMIAiomF9uQ2DsdzoWRMul00mrIQA9
BlXg0XfimNxqqQzoEEJ+K/rQ7cCL+XxEOzXg0rZRACPNOqa94q8RgQ/wiNKIEijEpUOctFqt8RqARaLc6xC
e3aIhtCcUo7fpRwRPoD8TegmfKAQwA3w4gYaGB9RRzh/m2gn6b6aQ8z6PQRcics74pfqUr/i9+EHfqr3+
SkKlQ/dx+rl8gslx2p+nz4QAnW/y7g93fCSmf0QnSgplgiWtaqpuU532XQcvBWtvAN9TkEUA7zFw/f4ILY
mj0xPP7KAhu3wKODj3QLZeZ6Q3AxooQBMEaBqgDtpWYR+ZSvG6PppIyJJmDcPbBL6B3qNzNWiTWKli
m1E0ccJK1bYDP8R3FfoJca0g15CT1hui7QCKBEsOWL/XAVhjBjxmh/UaMgkwOFdUoOevtGAMwLfRrFP
FKWvW6hYAHDrHxMTzE/7GIpPnogYjrjcEcNk4IA5xR1yrhRMoMMnsC9y6L6cn3gsINc6ZKMo40Heh0P
OJmuC8I9rL5/yf72lcEDZpXf6JzgwQnmQyYYVszlIIuDS4+m8CCxoljUobXHnwVkC/5Z5FOwKnY6tDVi/IJa
TEzxTRt4HFVybf+tMfHw1HFVfyQ83T3U0cg2/zHA49Y9wdph4SAgj62+52cITamHVxzg6PNiAKoFEUPJfzI
I0FiOjAhjggAyitqQEUT5TwMy50etrNeoPfiy4k0KYxuJqOARLUC3AO2iRtJhBKuNqY+ZMSgKNNccy/7qN6
XOYvQsM5eqj2IBpOHnsfwUEBoauGaMqO1XkvhjMjTRVF440RCB9wzr27s2sXL5x3jYrKR2gUJuJ+dW1o
QC6TxnPHLCMs+q2crsigaYVM1JYW59DyLSxJ3crlMpQjY1OFglsDvRYoJqEv3bhcyQm41T+uCQQoYOoO
2LjHe6dclnafdoC+xbsIJP0fxHokUlm3AGkogsJjcQRJhwNQM2YaPV3Ogarffuj4rhenh977fu//AMcPDVT9
/MMA9EPmQerUWydvXJ3FoX7TXw69Fu/To9lr+0DUGk00ItxN2kYOi2sdoNBr2KjXdJMaZjADAFUaNIx2
EnBlukUrxlyz1ejxWdAy2QxtC1gNjzudyU00nMAGP623h3Z/5whtO0IgOvBW+G1/yPOuD760YgYnRPHN
ONcMBCM4ZoAVLe9aU1oJoKTitGXYs6MjzsU1s9m8C4zMvrx8XVMhsQcP7tvzN57z8zYBJid0wegiGG3
Me1qcGFBHozLr3BfXbJzs2RhLMjs369daWl4GUwpPTe5NEQOBMcp5YgGog8YhqN5UhKDvsWEH0ilQg
a8N0KgaADlnGqOQQhXCs6gHWlyCiXzRLygGhDoD10/StlQCa5NA6PjqBLKT8dPPz7Snj6/+dzbAZ1/U8e
Ef/imPjwSoI0yeOlGd46aJQ2Y6CCGXLMsJajZaaCmcEetbGS4nPikeKSeqjzaNMHCMrJvpOKorMMTJ4XM
5SZEYmpHzyaT1B2NL53IMEoOL5rFxxAEu83d0VHbnqa1zA+6NrR179OgBfDZii9df5DtBzgdVgBYwSg6Q
g4MDy+WyDlKZ2iEgnpmZx7HpWhXQFadnEKaOD+IY7x3biZNTgf0FbW5+wVrwZwlKEg1OJ9je7p49ePj
AvvT5L6I143YCB9Y9SmBSaNH1J09sbqZoMzovFKFSKeHspOyodAJY4pjllNMfmX5ZF3egEIb1x4/t0uXLN
puK2moxhTAmLZeW2ZazR18hyF3uWx0uLVylXeF4xBqtpiVjUBNGfQhvDWOvB9zPGLBGZJVQEB5WZty
CQUFj5O+naFMcxy+mNqFtBdAhgE7GGBtAqNeu1PXcwc+FGQe/kCjfnzVQv9+h2J/OqljixGEBnACujZZRj
FPebLMxcYgCmPWGvGO+H0VLJOkQvmLVRp0bnHRyVPHHVh2vNetAHaAp9J0opknmuMb5pEHG8LET
wHlyUrK9/QObnZ33DqoDMDlEATp8b2/PdvjOxz79Y7q6e98KTY34KzP36OFDgDlt87NzVq1VLQO4egjD
5s4eptPsyvWbzoN77ZbfQzqBWUUNHVfraL1VG3KPogpq/FQub/sH+/bg/gN78blbrvlHmHZFBJQDIK768
MFDm5stWiGf9xmrPgAuThXsGEDLYii22+A64qXy2GUlJCiPHj+yT3z849YtHVpi1HZ0JOC1MZzJNn137eol
m5uWNg47P9csXTQhrj20ysmxWwxFXPodBAcNHAJp0tYMCECHAkgguAv9HY35Hv0uvq02KJ4rQSygILg

FB6lYRIBxj4oPcXhCjT4QWKXZJ2//qY/Q3+c4ff5DHa5BQZv++gwINy3K3kQ8a3T+fqlkBycnVsEUt2RquQd
55oo1TrxMzB5ADcIdpZGi0bhF0aqCuXhTCO87mgSYQ7x4i6CZ2rZzWLVyvW+PNvdsc+/ETmpttGiNa45s
5+DE5lcv0kd48i34GRpmZuEc58jZSQXNODPnGlTSr79DTXdyvSM0qiIxC/Pz0IiGm1150pVyBc2M46HIAa
pDrp+coDRAbta4phwmuGMIwZM2GQ4QAGnOdtu/K/BrEFsd7tf7CW4ICE64nsyrjJK+K+HOpJI4dTUffC
Wz6IlHLMRdONQnctAE7gFEP5LKWXuE9cHCNDj/vcdPrA/g9umfJ5v7tk+fVLEIikpEuZacwigOlLh+Kpu1E
AIqOuXEAfoU5Dzi/RIsNUwC2kcA1W6Fv0SDerxudqBLaN8+v/EZN+5dn3F3/jvXyGqs1OwPefwnA6pA6g
86QzcmzdmAb5ZqFStXq9ZGorkTf7+DBuoixfLiY4BEGi2EtGpQ5HwoJjqEDwrwClZHMX+tdh/N27Pt3SM
7LNXt0ZMd++DemnfWEDCOeAQAdDY3bUE0yNFJzXJTszg4WWs0OwwEn0cSdOzIDtG4GTS14pXy4OV
EdACJtIPA0qjX0aRJ3mtatTrhrC20WAtwBDUYaJeRAxEtxEjs7Ww7t04x+G3A3ZMDB1h7cNAW5xJQ05h
MvTegc7icO1sJpxddD1HJ9Ior6pwDPjtEE6td+JBINI4ZAA4CkiTWZW9rE8Fq2vmlJcCH4AbhqwBTGrgkp6
vWsHMXLk/CafRFqcZ9wP+rtO2tdz6AQnSdj3uslHvz9nBdKT4hK8i5FB3wsJOwxrWl2cPhmFsB0ZAe91Hj
fDqP2qsoi8Zf9yh8SzvLQRN10LkmiP3TH276XQL80AlpmQ5vtYzjhy6gdvvHk+/oRs94iTx2aUbF944AZhV
zrC/3RgNv/Eit5/xdXuusE0cFTenhGJ1NnRpBC0syJ86STNzB4ZGVAZrAKUrRYvATOEdNnKCtnQObnl+yG
Ux8o9O3KlpP3noeTbOLiZfzJUBKIHSXfYCgyYHdnR04YdFy0Ay9LyDWG1WfdhTvXF97ZEsL897ZtXoVThx
3cHfgu4uLK+69V2slB1Yxn0WL71sL3j0zM+uaTueLayqT87Xg4jo0lamIxoi/8snk8MzNzjr482g1ee6aTND
wilpU4L3SwvK+xWfVfdJqC4tLdufOfZ9kOHfhIlYEp2wUsUQigpcet/v338OR69onP/Fx23iyYTPFaResEFxU
fPT1N96wV1/9uEcbolG0MectZBN8L+8UKyH+qTkYMDERHikUeRaMNYKiARfwJHA9bkSRC1kdTSCk4M
BTOHpKxBGf1fv6flgamjNMoKPRF3Z4dQq1bz9RuOaPOACqDNsE+WrM5MH/Ued6GtQ03umhGJmAK
W0nEyVVLyqiX8qsVOBY1SZmhk4Y0/Gaa9ZgDhlY3ZA8W52zCdkP4+Hmk5iues9igKndxdNkwKoArlTt2u
FRyZ2HtcdrFkhkrIc2lAacK06h8cqY3IS99dbbdp7BiqUyDDgOgThWMuqcUYN2sL9nU4A2BjdtwtuCEho0
eq1SZfDNMpjqHhpWTpIcj9GgZ/0mAtGp4QMwABIpfiuzKwDmCzmEIOcmOxxBhOkMPvZ+DijkJHMpc6
J+R9A9BY/rDeG48qilvbbLB2ijSRxTGVDSVr0ugOD7AmIUDSgHMMk9K8IRo5/iXEsOpfj59PwylOfE0oUZr
EXGUpwnRL/HsTqJdNK++s1v2rmLF60wN2sV7lk5DnJMoadWP96FenXswuWbHsHo9AbOUSPca2DcsK
0nD+2Ln/uCT++uzs9YKggw+y2LBvEL+h3XnIoGSDNpBo9O4HeTexzQNgZZMKG/0ggSD3yKOMKQpo9k
GBy39NkAa5RAKAAf7/IbgUhHSOHD7+Dtw4cD1afw3FXTmXhIrs9AfvZDXjtm+dg/4+Gg5VFptjEvZWtw
U0G0SIdB6fNFSVmAQZKDIKBqRkWzP32BgO+FBgxCKIaJLtpxrWc7J3VAPLSHa1tuajXztLu7a4sXrlgoU7A
O/DYGgawz2Bk6U56vUtdWz13yXM8eGr1IB/cQhBBgerL22DXV4ty0teCRPUy5ZnPKJ0dWPTry64ZiSaw
BmoXBnJ2FR+I4FDMpW12ctYXZPN0YpdMVbB95QDyEFxsKITTuNKjPpGUAkawGHe+JLnSM+j7O/eor0t
p6qPMC0Y7PcrU6I4QHpwyO3kTIdI4qZruDJdrc2rEmJL7NZx3uq92o8X7Nw2nzS+f4bGTJ3BScNGZ5hPP
c9JTHiGsI3OONLbt66zmLpbM+BhIJl6Zu3bbX7tjC8nkEOo1lGloGIRZvFjt9ePdt79dXPv4x2g73hnNMZ2J
2fnEK0E65g9bqtiW6Poki5zWRiFqba/ZkKRAycXKfWAnjBNKf0Si0jfNPZ5LQOmgMYzeBEGcBc6IyTg/UN
3qEBVTh7w8fE9MvaecHzkn0JrznO8d3fqhUOXwdBS4A3yRAXiqj+fp4hXzWZpAHGhSQLN4p7SDt0+9O
XjtI0ToBwBPExAzhnTXA+Y033rIgUthFZQ/GCqHDXeGI8jC3t7e9wy9cu2l9+OMAEI4HLcsisTXAt7m5ac+/
/DJcskFbymieSZ7mAOdoc2MDU9+yJUxoBToSoHOLaMW5mZwtT8d9EiA3PWNJNLmSN4qFmCU1cPT
mEC1X55wNBCeFs9JH27r267TcbCvU5jNw/BMIJ6H2SXxUQixtoTimBk+TAILpSPw0jPaSNaIfFHMVzRFAl
UydxjIo9/T4pMqY5egPs1rpBPqBEsAK6H5r9Pnm7gGg7DJiYevAE0UXNBGRBJxdHKVsYQqhDWIxinj5QU
tAcdr1Yxs0y3brhVdsbWPPcoVpfIGhh7Pi0aC98c3fs4W5oq2srgJiLAuaNh6A2kAL0jH6Eyt57sIKdGTOaV
631bAIHKFPfwh+KShXFKtSr9bdCVQnSMFKYcxiuTSWcZRKUrFi4KW+YLQdH8EzrSgLzNm+3+FA5VcTRJ8
C1b/8IZPvB2+10IQDHpKeJlpUXE1AlaMiL7LPBVu8J+9U/EwaMRyM4EDxe7RQCGAM6dBKo4Onvmk1O
GUU5L/57gc2u7BixcVVPPIqGixvmWwaqR3Y8f6BvX37tt184UX3sMe9NhpAHRxBmlu2DhgXVuCNXKRy
cmLjdsMGuKIKuWQyaXdw5oozVoSDzUAblJBRzCVsvoCHjKZSqEtg6TEQmqpUmEodGed/AkB3rIhDlPM
rwUWhm54nhoj6aKZMtEnmDA9ONzl5Tl/KEdKUqJxDmTzxzzDOXrtTcaMlHqvQkWKiLcyxBlkOiUJ0iisPo
wmFIyb9CPeLM9A6rxxRmd5yrY0VaePZ79vh/qFt7+yiRKL+vhzGeqNn03NLtC5kqxfOW6Na4p7M5hfP2
95BCSt00XZ29hFWxaYbtrP52G49d81jt4fHZbhqxFKAShMt1dIx9zawFfp5BkqRL+QtI4xYuFdHoqFTMh9
ASOAMcXPKZZDAqn8ksGlojJJf4txjkvPH6T/FcqVZnUadRgf0/I86JkAV9B3UfJH/FAhnlCYHbfCfM3g1BrPR
qVsDDaDUNZk6hTFgrHjeaEpUfZgbV8ZRkAFLwaGCmKcB56u3hrZXrtmTg0PD6mDiew74PGA6KlVtCy1x
7dZLDnppahFxOQADBuadd96x5dVlW5yZQZLr1m1jxuGMWuZxjMbpYhLTmbxPClycSdnK4oItLMzZ/Ey
RDpvMq+cFfIDTbikxYwgY4dTVindgIpX029cUJrfjWkAJHhIwTYv20VKlSs01niiSxmQyGLSPPpBgdjpoSvpB

fSJAybSFNecPj5eG9qQXQBlASzk1oFPFl6VMIlgXfSaOqhknRQIUousilLqGDzqAF+eVl12cnqZfRm5mR2i1
KYSwDmhPTirQhQF8fWj3H20wrEF78Ggd5yaF2YaUA/YIApBIZJ1HjrFeM9CdB4/u8XxgN25cmdAX6E4Lr
p6Io+FoYB1LpawuTVgM/L5CNpdL2qXphBXg7PEEfYsFU3K43zcPxWjFaz0pR/kGCmFxDoFVY7u8PCdI8Z
76WyAVj3UQ0kcTK3V2eHQCiZ8A9QzNSIQ4G0LrD6nxAXypgze8e3KAaeGG6egAEibnQOatL3Md4DLhB
EDDYaLDpV/kfR8cw7s6Q7v9YN1q/B2hZXucuAovkxmO0sITTOyDR4+R9mWbw6v1FD46asQ1UUQMAv
xMDhidNuy2ML+AFaCKf84vLVthKm9T0wt4r0lbyA4th+cph6pLx4lDi0NFAAq9hSZDU6LZUvxGXutIjo7O
C2jkbUdpXwNPvQYn02zXEMsS4LdvvvEmWn/BQS8Hy3MUAKvm5sWPa+WWTU/P8p6mJiVkaBdoxLgv
Ptd3bRxFO8rM8a5z4qPDQ4+95mmLtI/AKcWQzaUtEUJTwfmn4NHysqXR29CYFKATUJrwdemSIEBV4rS
4sWLQ6XSBNoQ8oUevyyiHUqliu+rj3SM7Oi4Bwsk0brEwB0/NMkYHNjs3x3hlPA4axXEVB9Xky+HRoe0f
HNj5i+e5dtLBpgShQLtiuaEmaoY4Z9cYgykEWVqS4afvonDScbfpHv9wFPF4+VkKpOLTiwuzngCTSsFl3TE
FsIy38jpEEyf6k77kXidA5VCmuAZA3/a5cweslL25hFbhRmW0qKY8fSqU74VxKqSq5RT1BnDWMZISy7jE
9kH38SEdAhhP8N4bzb698c77Fs/k7NbHXrUSNzIAMApnyJTLodnY3vH0vXnAoBxOhUQkkT00qGaAqpU
SJr1pWaT6Elzp3NKiLS0vudnuaKoTQEUjtGlUApSY6FNyr2lXxWdFQ3RPPqs0DFt5IDPcxyTW0LINq8Fhd/
f3fRo0HIXT0ScKDckqRRG+995/D0FawLPOeGhmAOgV7opBLarwsm6jT+cvuYC4VmZAFVNN0V6FkkaAt
a2QHb+TgCthe2PjiU0Xpy0LQLQkRry1j9USZdld+8BCo47deO45RgGrgCZSjFpB/rT6jWunk3jzEbS6hBeN
OYnvjj3bPwLn7WlyQVyQwe4oFTCdo4+b9mR91w729m39CU4rABLTDtPOHNpRs3ld+G8RXl9vDeCzG/
gHVwFUGiyo/SghtEceZ6u5uwFNOLTFlfMWx9pcvAw1yKcMm2ANxaqj9BEKJhqV84bDKBqE8PrsHUCM
0r86rwQkFQ8ZOt5pgMdw+VzW1Cdlvg1UpEgmUABFoFHvPDjpCYN4eHTsAyKiLgCKgwWDAAlcSwFodk
mecSyRB7QpK9d7tl9q2MbeHtymgpSl+V7UNrd3bWfvwK7DNeX1huEqQYAfp2EKh+wfHvP5nnO1JOCv
V45xAKo4UA1LZ1N2DlAuYc7lwc8UFBvFgQGgPTlyMghoPYXhgoBV19SLKqa9wcBIEOuA5vDoxGeeMsV
5s8wi98L3FYFA8DTPLq1+4co1nJAZgMq9CpCNCjZ66G3TdO4MYBQz6vNZlw5PplNo4CbmsW3zc4tcS64
ffcTJM2j4VquKBmnCzWJWgCOXjisIoPowYOvra251ZmaL3A0AVjsY0Bj3X97fsJmpLBYq7s7W1PQcgwUfl
tZR2GjQsUImZQv8VtO8An8UQEakSDS/DqiHzpGBIWoOV9C6UBuBLAyUtOylg9O0A+XaQ8v+1te/6rQtg
hYuJPNWnJu3Bs6WZrhml1ZQBGha2uUTElCToSZDDvZsenbBaV8XiySePzuVto/dPG9TaNdwv+FJO+121
3MFlMvRZDyqKAWlFmp2S4oxlUh6ROXSDM4k/SLeLqBOJg8mWtiBKrCdLaXt8mEdTbjPoGr2yD10NJTu
VpxHPCzkQEUtA7IwPKk3wkSOY3ZSHuIY3bMSNxXH4+zjWGXjKR88NfTBw0eedjczO+dAHbarlpaDQof
WGaB6DTABpCAcMsy1Zwppu4rJOX/xHJ76DAOIZAHubr2CYGN26TgJWRBeJ1LdArEn7RGaZJK6Jz6nBX
b8yiq1Jt5zw8oMyuWbN6yCwyFgKHc/QueoDTu7e7Z84bJFUlkcPgFubFOY1QFt1m8lqOcuXXa+p9kdJbt
kcjme9+xo58QWZhY4D9RIeZ+AKgbHU8xQ4FI8VJpugHMmNiagbqFRizMFHwinJzgZHbSjTOQRQn3h4g
UHahNAcbtoF4DIMHUbJbR12abSSupW7m3EtXAO4Ka00I+xUVqksp5QYJwbx3fAPTQx15ksXnwSzVrnN
zitmvaFu773aMunut/51pt2tH0AnUED8r0bL30c6xdFGWnMlWAN51ZUhT4ZAeSpmXnb3p+E+qpQtEC/
adPZsJ2fydoM+uIqY6cpZoEPMcMBh3ZwnjjWQIIiiiK6pIjMzbmCO3GiHMoiEz93B0saFYo6lgZlVHwglGV
zhPes6bZIIuGzMlrA5ovTwkkGQBxPGkEmRZQ2Yk+2Dmz/uGGNVtA2tvBAm21bBQwFOmJEw6TZiniKU
UD55jtv2jROzhxmtLy3be3ykUcO5GxI+jU7JK35POZmcWEG7xATQ1u6A2X6BOC0mGKurdivNI2C5Zqnb
veGdlBu2J0NHCQ6PpvRvHvIanjBurlsFj6M0D189NCWz63aOI7jAzgl8WN4t+5PIazZ+WUL4aW26BRNba
bhYQoRibtVGNwLFy+hnQE9wqxl1Dk0ojq7ety0WQZNdIVh9P5swLPzaGGZNjlEqhEwkLAz8DJ7+3s7OEZ
wUCkIfietq0ypCFSjUm3a0rlLbppjWBzlEuhQLLJVPUbIGwhylnvNcu/ihqJK9BP8vUafphJhO3du0ZbFBRn
8bMKskEMQ0GhayqOIg5K0+4ynprqnl5Y8xJYSiA9KjOOB3Xm8aXsVhBvOrgmNubkFOHXWfZQWgp/NF
y1bmLUOfSMM9RijGNp8UDuwjXvv2NJ00l558SYCXLQCAiE+26QPAQ/YC3oY0ekOIB3qPtHS2XTCivBdU
QJpfzd7HA5UNbCFJhDXUvqdNFIT6copHsdJ0E8ANmLxUNxC2L0AWrTe1UxU2w7LdVvb3sf7VHglanVA
ehfNOYeTc+PaLSvhiWoO2j1cPOatHcU26zYzM2XbG/etXytjJvu2Mj9tL12/aheWF13DiaeOcBC0JEQmKI
GG9lWcznuPPRzVog3HlSZmDa2MZ96ho3ojOCiaNZOH++WmrXyCmUF7y7kLQ2FKB7sWAMiJ+Tl4ac07
XSYyk8LEVsoOGiXHyOWRMAisyilFVnlPvHwSo5VGFX+SpWhxz+KGkQgmmN+7qeKhe47G8bLjCDhSPaR
9IkpyVGQN5LQl+UwJ3jJ1mhBRUrZS+xT4VwGKARYkjFDJLwAOnEtO2JFNT027NUllZ9G4yhxT1pZCbFW
cSAY8h3al3zVJICoRD/ZseU4ZUyPAVnTTK2eVZkJtoAho/hG/z+FH9OlXUb0KGm+Tsb33ZNM2NncAI5wf
My3llclOIaSiQVg1LECPcyi2LGfo4f33rd+q2IvPXXOrm+TeV6Ft+Wzc8kloQqOK94+1wdlKcM9hHqI8baiD
FiZqKZFyLZzfM1auVPuDwbhBZ4iPVgCVko8zSYUccBIYoChf7jJgvuyjN7Z5eNZeqW631/fsEKkvtwcO0jim
Rx2fR3Lu37ntSRWf+dQXkNgA2qDDAAbgiLtIM5pXMb0kEoUMry7O2CvP37DF6SkLq7Mg3zE0pM/7B6

KugZooqCaDWYHzag789u27SPI0ZmseRwoARaAXQHGssFOgiZOybcksWnse0CNU7VoVcYcC9Fo2xDkT
MEapHB0/hm5MgNzH9PrSEjpWGnwBL7hULVut03LvXTx3iKl3jgcnkzPEidxBEDeWtpCzI+HS23qkkmkG
T3P4shb66hjharjjJFAq71aJHkpo0WyOACRzFwfcmpkTiCIxhDWMBUPSNLunJS1baLuFxRV6NAJIB659eji
lctqO9nf4jcw+woHWDDE2KrIRGiFsw5qnBy4vrdrFixfh8nEr8AiNezYNjRj0mh6vLTcUZgId3KsSfEQtBNLb
dx9gPXfsGEcpEMbaxgs24rM4wppl3KPxqB0cHWGVK3b5ykXah4XGWkbou3gY3yEWtJeeu2jzUzhcCNa
Q/h7Q70oJ1ERHiX4QL5VSSMagSrEYuJhmTMBes90abx4f4TworMKNYQoU6lBWfAXTpXBTZiqDpkJtV8
QjW/b4ya7d3dixxQuXLBTP2gncMgy/arXwFosFzELZtp6swzdSSN2Uq+8a2mp76wmd0YAGpO1zn/uk5X
EWNGORxuSFaHgXgEiD5fIFgBe2WqNj23uHrrUZLUwsfI+b2kK6NROTm1qwINcIAtQ+aHGt3W7TWceY
mmmPSGjQ21CPATwthBnXYjzlGSgTSxRBU5gt5SfQPpndBAIS4fMZOFcLvhjPTGKNApq0WpH36TYGOe
UZ+pqS1YrUwnQKIEo7y6lQsBtzhs8jTakYqPwAAVWOlLgzb/mqUvW1Jk40GyT+ppzUZAItzTjIcikOqwiCJ
m18FQCAFi3IYzFGirZwkZXlZZRBw3I4nU821jDR07RwEpfNaFYN6zMcNNDSx1biPs+tXgRcccapDj1LMuZ
9u3IeR5A+UrxZs4KqYaCQGU3mAdDQ8op2iL5otvDRozW792iT8U97quA0jmQomrDD4xObX15BaCbJ2
gKr8gyUO6GVsIe7T2wmn7af/OJnGGEwxdjI75BiOCif0D89AIyNpK+FhTy8W+WNAqVKZbxHg+sMGLYF4
MThCVmIdxuTzSDT6XG+XIMjlUtNHKJN78C7j59YHPV/7fkXcbyOfUCkKZB7C0IjAjw2AGsUOqB59j6DfvH
Col08t2Lnzy3Y8sq8hz6S6TwNlfAqB1WORojOrNq9tW0PiSXwQDU5oKC14odaGlLm853dHUvCQzM5tB
AOhThqQNOFUID9g0NZOMwU3Iy2a0lHpXwAPytj4jCp0Bhl5dexIvNQAE2lPnfjqoecwpj2c+eW0W54oD
HuCU2m8IhCKn1pQkyTQnnSqAoHCVzSYBbsoEUnubUKCUkzpHFc1C8TLSFvHNowUGEKzDxCpJCSZqKa
aDBpUvF0TZOKGmwc1qzU1PoxrbyFkmHdMvRVDW6YSuTQojnaNKFGOThdV9aAtpYrJZsW76WN4u9x
+LbyS0dQoqOjTQd7HiFGhvyepgCNHKN25YT7Gvpqg2vXL6PNFQvWfcPfObesDjfmdERsQYK2BwV5/fW
3bGNnD8uG0ChiUFywS9dftCc7+2jJrC0uL9CWPhioOsW798F7tjA7Zdcvrtr5pQUsdJa+od/kg2AR5VT26
VM5hBHFsbGCuXTaAkjA+IAOgZb4IMTQjIqFylxHcQTGfPn2oye2sb3HQDDAmJr5hSXb3D+2B4/XbRHHR
NNqmorbevLYTcegVbUSJqiO2s+jeSQdf+nHPmcrS/NwI0wLZJ8xgVr0uPEpACdnoWM7aE/lUtax53twSy
XkFopzAE5xSGX6dNF0BY+pVrR8BSEqQvA1M1Wp4y0jv6khmpXOPcEBEidu41yE8dybgDQSGjpwF/C0L
ywu2vzMjLddM07y0ifcEqeSgRF4D7A0mXwSDca5ZAo55PipbpTS8xTSU9a7MvCjCAVmwZ0DLyohcPcm
06uaRtU6KgX5fX0WQFSytLSWYrU+vYptF2g1RatJhy4CO8KaKf7YldePIPbolyfr29xLB+2IA1dt4ZFzXXFq+
rgNrRK/k8ccQcmEUBKeTJPKAtgwCmXTFvEdVJ9AmlKJ6UoYUvLQa9/4A7uMhz47g2Kol1AKAfo6b6sAr
QjfFZiVJqhcig6OnOK4cUUdwMoG1vVr3xRgj6GK05bMz1s0jfMUztIvSsruWBon6fb7dxGEaZTVqu1vPsH
BC9u1S0to8wW0J9YI3AzR/iNpVOiUcpUVa89CoQK9Xn98jLNSRYNqsOTJt9poRLhVmBssY37feOe2g6o7
0uxT0h0E7Cd8ZdsqtYq99NJLdGbDjg/2rAU/6TfgkkcHtjQ/az/5pS/ZMqYzm4njWB2hLQBnkg6C01kghuf
e96wn5ZbWNf0H6OQIDTB9x+WqL18Rd5PHr07SHHwFAKmzNOA0xJNiWgyipDJYr6E9FFyvQ/jhXwDvY
y/d5H5GNjs3Y/M8FJ+Moh1lYuqNugNLWk+gEjcUfyzIkQRUKEcEq+KhJeTAzakGWY6UHCKBUKB1rY9j1
McyTZJVRpbLFQCYnKOkJ1GLDjS7NU/H07kVEpNGnpkruhBIq4q+aGZGzok84xSmVa9twHjQrkQkadWT
uscgtZ5sv9K1nXLD9uj7cuXIQaq+HWFZZudxdhB2rfuvQMdiODK54gyAHnhWvyIZcp7kVH3r639gLz7/nE
2j4VrNkgO+jXbXJMo01CYNYK5fPmc9TL9ArGwubAUAhBbCtRP4Ne/eeWzv31mzJ/s1KFneiosXLA13Te
DDbUEXBmjdF7DAR4cHbnGjIfh1uG+LMxmnHoUEQqBsLPpG6Zwan9kp/CW0mnv9bbTpcUUeP+pZjoP
4QSztU57VFpxwr2QlNNbmQcV6aIpEMuPkWbM8t99/DzNotrQwi0eNudZMxfyMffzFF+yFGzdoUH8S3s
H8Kq8TWoWpjuIgdezJ4217fP+Jza1csDwdWEdbVxloRQmUQT5A40gzJmJJy2g+GYcB0fbZLGXY12oIjwc
qr1KeqBJ/p1IBW16YR3vP2srqgvQSna5BAZiAQfWnVIDB8IwFFF1D2evSPgJdTlOrtEHZWPKSEglp8DqaA
88UDieuKbOtpG5NhAiwKvlTrvQshYlSyEsA8qRheHcPoRdlUUA7hrMxMDQTHFicOAGApdVUz0rn1k+U
WSXdncXTB7XWqGl5SgSAJlyrt6EJPpXJGMn+4v5YN5SEE8at0a7b5uaWr+bd3T+0Gpr3GAdZlGGIdUkp3
wHgdtGoU7PzaGGVG4LTA5w56MLK8pIrAAm1Yp+ZVAwFYfbkwR2fnl5dXED5TNtULudYUeUVpVT24O
nKnipMzWLdgvatd+7Y6x88gi6aFVBS8ZTYaABKsux5x0n8ihQWqH6ya/Fg0y4u5+zi6rzF1WX0n86vrCyFK
qfzOe6VfjmLo8rTq5SPGVZ4GNI2onPKDUwTPLDWHtv7dzfw9jHp0IQQAysnRU7Hzsa6rT28gxqP+PTZKy
+9YEuY1Wk0kjTcCR2WpmNnlM2OhKztH9khoG+hPfpo6+r+icXQ3FEexmC0aKiSNDQ3nIHo725vIlljnudc
wgftKg8kliEfDTGBDLDmwzXVuDA/5eZpAWcigdeoBAlp3zDn87I9aCaFU6KiONy9nBtNeUprCoCiEpouV
hK1pl7D4r7ce6XM9QCQuLN4eCaXdbOtYhgy8cquOinJvGUcoAqtyQGKQSVEC+T5Ls7NY4KxRKIHcuZw4
KS5W3DQWfomjJCpPqunQ/K7AoOb4LXBaRMI6QgaoUmOEc6GqlFLUAcIbRRBGgZjABDfgHNrdkne/hH
WSGZ/bX3HKhXaUKXtCABGCy4ZpE9SdDfn5d6fbDyBo9+EjsToi0kOR4U2T6OMmnDL8sm+ncdhS6dj1k

EllLcwTx9PQ4tkmtPQxZ7PwDWwlhmc72mw0rTf/fpbniXX6TZsYekcym0JhbRqx8daTcF9BXpWTI/s1Zfh
xGM0NY3RIkxMi8e354tFxjGqQOEEqChU7/hqtYQTVaUjuTCS1x6GbYoO1t8794/tnfsPrYJqLgCMEgA4w
rzbCDY17ML7cvZzP/klLxGzh3MVCES9PI2W1nYAXUmzTnCq467ZAR53h85cnZqxIB2xAfHuYarOXbnK92
pO2FuSWPiswi2NqtYghayLuQlwrXigjwbI2q0blzHROZuG9yQ0n04nN/D6gwxaj++K40g7ZhCCAUS/b7Rl
COkHkCfHxx4C8xksBEhpiyp48XBt3cNaS3iupeMy2ivlZhZdbEdQFwX5lb0UU1gHUHXQKOLXpWrdl21r5
YEyxwRYLRbso7kfPbhnly8wGJxFU7PKkvfpX377wQfv29Vrlz1aQmdyzgm/lQ6SVumgGFYWlt3BlSUoTqX
h4A139NJx+LhCf4yVZsQUo5V50/SuFvmdaThNYWuGbmf/AJ5atoOTmu2jaTv0uRZZajJkdWXFHUxVkwl
C6yQ4ihdr+fjVK5d8albJQSrNqfRLJbFHQmO7iCUtZiK2UEBxAbwyANcCziy+RRiHb/foyN5//wO7i+WMJK
BdC9ewJFlPxJ4rRO0nPnsN8y//Am+ftkbCcTs5xCrjcM2gTWVdJkDFTVXYQQ1pdet2eLzvSR5ZwFhuoXGC
EOmgMrYj9h+/9p7dffDAY6wHe7uQ+7Zdu3LeXnnhuq3MwyXoqNLRPiwY8htSoBtHA/70+PETe7J3YMn
ivIWmFqw6CllrHLEwgzjNdZU/+WB9A6K/RKs0UHDF+jHS07YaWr4PpxIXnWMwL0G8r19U8kMGIZjEFic
1T9u0H/MeiVkB5yxFRwss5aMSIFTiRcxqCEldydpYj42tXe6xgJbDK4YOdHFWNB26RjtkjvPcv0CeiCqYjYYE
9McAVRpo5dyKTzULtJoEkAZ6uP7QcwEKubzzUzkoihMKXI8ePvA1THKGRgHFN3GS4HrSumtP1nBgEDS
0GWQHh3XifKmaiRYcKl9iCaBqfZmsSAgPOh7V0ukxXdW1Zdq8gKOiRYe6D9nqnjuXTQvKQjWxDmhOd9
AUdEaBjHn/MQ5QpdXF/3jX+XkGPp3EkkaiKaxbDq0Z8twIlc3MAOQaCkWTEfM4n1JqYYRpm7ZrGvwGY
zKfD9vl1RnOP4n8KFo0QBnlp2fgpgN78+179vo7a3D0iE3NLNsl+vDKuaItFIOWsDr3qwRsKB/joPTMLFRJ
oa0ofcswWuh//ff//t+XTlVnS+1qnc4QMRsrY0deKCZGpWBmi/CgeMq+8Xt/YJ3KsaVxNr/0yZfs86++jOof
+/IO5XWK63VHKdsr9+zhxomt70H8gxD0IeR8EMWkJuGYHcui4asnxz412kfNqwpfG/PRx6wHINP7G1tc
p4kOjNtNvNGf+OxzXOs5u3Vp0aLcVKdZwelreqLLEEFq43zNrpyzei/C9SP2a//h9+3ek30bx/P27qMdOxT
N4L76dPAQTbgPJdHCwCwdeVzFnNF25IVjDN3AI0UKMnilAy2vCY4RuiC0tuYx1wIEX5lRrWrFl0zPAfieHK
5A2AqYXi3y7qOhU9G4B/OVwxCNiErg4UNBNI2paWBppO31NRwYQAI90np6vFVAQF/jWDRxgLSuahrtJ
G4pxd6FCgyxDrcfbuAzlCyUmbUHOyXbBZAl+RTcY7IwY7jDntEk8y8ePaafx2jiXDpswV7dLsxnbXU6Ybcu
Lto5gKGZqSpjqDBYEU18sI+igCZlskWnFjBzDxdBqGh3yMqHx1Y6hNIsrcBDF23zuGIlQJaYgmsm81Y5qbo
T3YaCcEf4DHyPPm2UcZoPN+zSYtr+0ucv++LOEEI5gMtHoWJTWKo5hE9lMpXYo7URnqU3AaqkU4ZJbF
amceDSp3LcSpaNxmkgEpZNx+3h3XUIddE++6lP2AvP3bAmXr6SeKF43AzakY48qgzsuIEXjYdqkaz10J5t1
PrhCZ4npkWL1bDNHmMNMoDicNXyidVLR9aGfgwg9HGchauXLtgXP/sp+8THnoPMo/EZuHr12J0mmc
8omiKWnfI17b1Awh5uHtnG7omVm107KNftoFS3qfkVQIyOQ4swYuBAuQqTfEdlXiVzGTf9k3JBdAjavN
Mo+zofLUxTEF/5pJrAVDKxTLZyEs5Kq0vrarFgFzCOcepkLrXQTdpQlEF0oon5jiEIWlBX57t1HCmtQNAU7
d7+LmDIomnxeLtwT6yNln1Hg0PozwnAVbnIPNfEFGOqtdxci+aO0bTJ9JQl8jix/K7cpM/rbduDrjzBWrz+
5jvcU5x+ksVQAbiU5zQ4x+Xc417TFYao2cLsjC0tLdsCTmgWcO9sbWPmcaxpk6aSJ/W/QlCohANeArazsY
Hy4vuFaWvQpjEKqNYdOrWQxs3B19UfAoYq4wy57vRMwabo1wGC8tLNaz7+cnYFSAX8x/RvBmstJ8rTT
gVU/spv9HX9br489sdnDjgcCy6mYJamThvtloOsUm57atsNADo7O2d7cAml5ynWGaVDTpTet7lnFbz3H
oDV1jJcw8M1kuqOpk8BIXreQVKrtMDrwA62xUPLyOyA8xbsYy/csE9/8kV75ZVrcLGhNfFmmzL/cjQw7S
M4V4PXD9a3GJia5xp0oC+HdFIVUGjxWRpztQsnyyKd8q5lMRT+0UDLISrwnipRy1uVWynzpEHUWqF2T
SErAArvU0xXkx/y5BVdCDHoI8DTBtwdPBNlTimkpSlW9Z+yw1B+pjKYynh3h43nSgdU4rNAq7CQ4QfIyd
Oy7whaWJWotZhP3S7FHhx3PeSHa25xHB+PNsCpW05xmu4fXLt2xSMOdK8LjWcoYcaV6CzOnU7jPcPJ
K1W0v8w+zpuqvswvrfKXYQirggrcs43nTqNzCMTS4rxPdkS0uqKvugQVK2NBDZCrT/oomBbWTCGp5ZU
L3NsYLU/fCTs8dOxtb9hDKOIiXD8lbo0ZF240ZS16NDc3432lugnxuFYdoDhGOO4oSsWVvfCIvsDhf/UfNz
j2NCs8YX2kHEwt1FPWziGaTgUklOh8XMVLTWR9SjTPo42dbLX6PnBynBT0PcIUNFHh6ZkZGh5mELt0oD
oa7gPJVxKIklJGDIJiqY8/eGCxQBwONuOhpMsXlrmJvE3hTVar+8gJegzaEQjlrDXIeEyy0aA9cKcmA7a+sW
0RvMwFzEo4msBEKYaoWKdqlqbt7t07Ds7Ll6+4YCi4rQ7TdO1MKmYPHz5EmxWcsih2rGyesWZi4Ly6tlY
FNDGJWtqhpBy1X3kCztnoEwXjY7EEr9EeyppC9M/CV7qeAKcVAJp2TPMdTZmGwvBJ/oXhvopuHJ9o3h
5ujqJQfDMg7g4gs6ohBf8LYOZzWnEKsDS6AoM0VgXB1gxSg7b1uW+t9crloFWDjq2vPbSb1655RKPX7p
mqzih+3enU7BgQ37p6GZ9ixqayKQQCOWXsFbqTk6nq1ZkEYxqGQjGGO/sntobyOcYKiNvLgaxXq3b+4l
UAOO1LdLLTRQRa95gCfHXb5fo5HOHVuSmc3pxNpaPwcO4BRaCCHMox6EChFDNV/sTcwhxUK2Hnp/L
ugAuoOiR8egisE69fIonGQae6mhViFce8++gxHvk2JpxORbVHlKCCmZXjkSvMowHgprUO4Bm4NothXjUF
GESCPDGaE6kKsuayNQ03qWWPhywJRbOH0WyXcaBefvFFPG1J2cgaCkwzSKGQMowCaHF40wBHqJez
k3KZzychIi//g0bY2tm1GSTXM+/p6K6b5a5/rkD2ndsf2AvPv+BTjArvCGgKfSRgcQqnKD8hgVZSQo3njuKc

RRUWGjHACJfipfp8fn4BLSRPfYijVQDU8o7R8Aiq5uM1Xy2TqIx/8UJ5yIqryiHQFLNmorxMJE6mz+8rQYZ
rCahaLNcGUA1pbniqKhgqlKUwl7htGmqgUjsCs5ybvf19Xg9scZE2jfErYpopa1ghD1A6ddvceGIvvPA8z3s
e1tK0uOLA65uP4NkNTwBaXZoFJC0AFbcFFEsR3q2JjxE8NggtqDUR1mQaC5bgGkEr11v2aH0yQynHam
b+HAjPWQPNqlzUGPeqWbed7S3Xxs8hRKMeFlg5sOGhzRQzcG3VWECTtuoe1mqJuyPUDQB7YXXVri/
MWpB7lmU4O6RodEw0KqZJkuosFSC6auW4+2jNQw1QI+d2EU4uvl9rdHkfELVGaFykpA0hhjArHKIZmE
N4JC/dixx1MVXcWKtRwxzV0cSYJDz2S1eu2tXVgi0WMb3DMOfoTUwcpknmUnO+CnW8/e6bNrNyxUb
xGaQVLxrvtVJvWJ4BS9I5d27f8SjEdTTI9u621IN7sTLbSKFXOvEVkHSIkkgkSFrYFxt3+A5aD4sYjyUtASDiyY
jNaRaNAY/To8qgVwJ3yrl62lcJFKZm0GIqWjZZ8qEEDoFZRXS1GE4UQjRDgXRVdBbH1cSAKhxKqyKztKEB
cMRJW2hd+QNt/74WCMpka+6/M4xirod2UikB4slEg+K8yn9t8V2BV9q8N6CjaXs+l2WAzQ4O9zSq9iLCv
7t3DB8GBBcueZ/+/te/aq+8/AI+QtAyCFSldIirOnAtLM03QMCvXlhEA2KZaK/4dRwFUKG9Q0Y4Gs8AzK4
9wdG982AdXChrDc++AFC5frnSgB71fQYwEQtbDy0rHjyTgpYM23b5/JLF4ccZrJN8AYUTlY8ggb8CUGc13
Uo7zw5pU/lOrlWlUYdupJS7rzcE10kAvMnAPtnZodPgbjIRmsnjqwk8wTfvblm1oxxQBgcvW8jU/3Xikzp
molXxabIRgxGE3wisutL5yxft0tUrtkjDMomuhfolBk7z3zGctTmEoG8nx1VPJdMAPV5ftxlMY25ugYFV/S
m0JQDQ/L/4ZalU8kITS2gXTZ0qWVlRDOV++lIMOkMeuub/BX5lPV04v4wTgTeNx7+M559LMeiYeWXQ
p7LKLGeQ0DaqzByms2WOJMzHpSpOHdqN1+LwvmUOmluO1TA4mTpVmMxNFf0loKqTZQEEriRcdBya
lG3PFXJoZxVvEM3Q2v+wC4wnkKPBSo2x0RV+7jIO5sHhvoPYVylUJuWN5KREAM8J7bpMvypppgI1Um
mhZfq3WsMHgDLkc0Wo2RbKoGU3bl1Hk2q3loztbT2xfEbRjai16yf23ttv2q3rl+zcyqInuk/x8CU/3JeERID
U1DkoQTtv22tv3YZKiYOHwQfaGJqi6i1TxSJOMs4fWj+OtUkMqzZs1Ww6l7CP3bhkUdPeByoIgpVDi0/Pz
qNxC5YPyjKhNB2Bk2Pi5EuBngJVF1ejVL5cnay698oKUlW8PYi7Sl9zbfhW1zKzy/Z4v+ZzupU2miUzBYkfe
3hDWUWdoQLgJ+7Bj1DzURq8ujRjN25cs3MXaShmTev3e/0Ty6flzcYxZXF7sn5iW9t49WjWIOZKEYf1jX
U74bwXLl2gA2aR5h3nRJpEUIkfxdge3r2NpwjI4L0n5QM3mZp2U6JMhCcy5at4tedXl/Fqk1acLVob4dFG
DyG0+aCFICIUMn2KcIyCKuKGxkUbBFCBSdoSBpDi8h7Cg0PJJAlwrv0RiPxM3urNKu9jrgBxSuEuziv6oIrU
UgTyYAfDEM4LwoSmDmt2Rstr+LTHdVScTOZd8/O9MVQLAVJGfUixXMyqnDOv+YQ5r1Tr9sEHj1yrv/ve
e05LwmjKNgKpFLt4Skvl4LeFOdoYxMHZtysoCLWjynmUNaa8gmQccsJv1h98wNgP7BreeLXeRO2MbHG
mCC2Y8lCdryDF6dEsl/Y4yCKwWih4//G6vXf7AcLUsemFc17QQ0tnOox5CqAXs1CC4x1Lhfp2bq5on3rxs
nURii6WToKnlM556J+iCkk6IqgORNB1nGlUHd8FVPEpz+B3BYsmoMN4CZne4eItSwHCBk7HOA6fyc3Z2
kHF7qztOqiUeHyM1Ku+U48zVuCaPczs6mzOnr983j7+/HU0RsS2IfNaTRDL5BiAqBcfe/vNe65Rp6ZXMa
EAgoYG6fRABGnFpD66/T4meGQvvfyKbe0ewUO1jv60IIRyZtH4wz58CAAkE8pBjdri7JydPzfRDMoSL05l
oBho65MT9z77CozjCQfRCCkEQ2u75HCIHoUTDBMcWd78oA8HxVlR74xo1xF8U946ZNOdmm6bc2hCH
EE5wcGZRZvIjGsGp3R4hDCpvmnNltDcXvuVtnseACZQKy7LpWNbmJ9mVLBWWIxJaUlNFkxyWkUpAnB
fBfS79Fu+oAD6ZPvKMcLdZVzkjWtN1P0Hj2wL0390UgU4TXdoFcLy1EJ48/zMwqTtYfkaI+hDHnCbHe5v
2xGe+osvXLNYKm9H+BwCbxpuuVBI25N773tW1Yu3bjmfbjQ6CGUNQR9CKy4A3o59843b9mBtx+pww
zHnj6Kxk3xXEwNhtOkLlxbtpZvnLKRkFpxEAOMAlPLJTmGleO4r3/44oMKgJi94hDD90qjoCrQDQKWD5
Mnv7e1br9GEF8HfpufssMVnSNpr768DvhM3YVLnWjm6e3iMacnaJ1++ZbcwsxkGZKw14HC1MWq5AY
85wdko1zl3L2zrj/e4xshWzwNmtMGAQQxGRrZf2rLZhaI1Dg9t4/07duHKdauKv8FrFP5poFXqOFtz3Kj4z
wLcaLqY9unIKTpKwqOl0AmkVVRAMV85WHlMYxfOio6zCBouDAkXn1Wwu6ekF5yBLhz2EI+2M0jhWIa
4P0w7/aIMeXnpvioXTdqGsyUSSds5OuR+ynbr5k2r0a4sQBphlRTG0bLoqxcvIizTdoB1ymYyaBA5mm27
d+8DPPQrnnPJMKHhVCCCPsb7Vv2HVFpVuFWfStOmUZ9LL5WVYJ5kEOPQH7Tc8b4VsBKaEq4iFFrjtIe3
flSu28bWnlWrbXhpir6q4R8ULAU4lB6odVBaBbu3u+mpfLk0Dlw0adVhwhOd81GsRqNk67fftOeuXLIpu
PvFy5ewVHBTnC5EEO6fNtWcZVhwtEp2f33Hto5LXrshxTioxsE5HLUvfeKSFfH+2whvCh9B+RrqD9f80B4
JorKRfaWtwHh6fBuo4JSPJoeA6ljmmzJz+pI81ABAq9OS3b1DgCYeEscTxwOdyUILavYff+8bAPkAyY8gX
WVbXpqzT7/6Ch2NE4OKyjCQdbzGSDxtJ5xnff/QDkqAGoDkiwuQ+prtQ/yjkSSe7BLaQUVn4a+YpWAEU
4VTdbJ76GGjGDc+RItOIwjnluadm03P0iFo4CSPYOfEtdgxvxGw3AwmsnIZ3GRrehIfxapdbY/TZpAx12hGl
cVRQsrO3o6tnF/xTKcqpjuensHsaV6f6/Z79sGd93BUPuZUQNOo8sbjUJmNJw+tgZA+d/OWg1MaPw3gZ
O7ffftdT/pQ3sAJXF1TkVoxqkmE+3c+sGvXrsq9hfJEfa2U6hB0a/sAax/Kc80WllcAaNo1dZJ2FQsZz5rXJEk
2q91Wmqa8TyXpQAz4PZwXa5PHamnLnwqc9v7DHXvweB8LdoyiaPvqU1WBGQdVmaZvK8vnnbZEuFfl
smqFboxz3ntw12Pbl1AE+/ubgFSplkFbWcBk5ybrmeKMW63amqyqiCTsa994E2pXx7k9sDwg/Ws/90W
7uJi1RhntD51KQyM0ltNoXC2hVizWeT3vT3TpHz484H8G0G9/6fSJfqysIdDqnnNPkKYTk0h5JgNY4Z85V

WpjEJ88euim6tVXPmGvvHgdcpxE41W900Gb3X6wgYZq2RaNLTUxe5kZn3NW9T81VLG8LlpZnEzRgvLJ
oYePyqUyWuiQ645wiio2nYrZZz72nL1y47LdvLJqc3NZzo8J7qOxx12bohMngx4DoEg7XusRGkjz2rr+XczT
G+/dB4RDW1vfRaPjJCFEA7hnIjflkxXTi+d8PdY4lLRAGLCFVJlOmm1sJajK3OLihD4MlAsR9nKXQ7S3tMH
U1BTdB6ixGvor5+gQLaLCDgqNqeKzZsY0IaG83goDvLi0Smcr0K3yQWMHZRcqoxLygWQBTmkI+Mgeb+
zBwfGsEYKNrU3bwVzHk1O0dwlvG+lDi2nRnsJs6Gu0IpLK8wyAu3nzhl2+ctVWl5cYO+3AgpXDehwf7DO
GWV+y0qy1oUBx68EhVR9KG7yV4aGFhQWr40yFnbrQvWDiBCdWmWtaLu8akL7pAD6VOgpAsRZmp2
2IY5lHi75464ILj9qThv4p+pJCKLUMSbRJU9QBLJ429aDh3/f4YytOC6jfjmPpf3Sw9hYV7rX6UhfRdGsZU6
c5/k9/6jN2lc6oo6nEebVk2TBPKs/9B99825dV56cXLJWbtib8z2sGcNNaklHIpHFoWnZytE/fYq7RpPXKkR
3sPMHM9e1jL16yz3/yZftLX/yk3bq6wrmR5qBMbxPvuG0j2qKwUBAHojtG6+PgPUFLv/H2HTuqt+0AJ+
AEUGwfHNOOPfitytaMGLQMbdA6c7QL1kPe+WQHPcVQJ4nTCndphabCKfVqxbJoc4VvuvSFElbEK1XFR
VRDzo2iDJrJEodW/ykkJiokr1xRAZ+OVbJ1s+FFhbXmSRZsgGbXd3LuqFTdlM8trMCVBygF7aAyWfkahJps
4GSqDQ0o05PtEy+QlT9CyUPasXAGPlpv4lXTHx0Eqgt31AqOpaVZO39hyWanp2ir0im7Htd88vgxz9uuP
Wv1kjUA3Qf37njizrS8eJSRxqmnkBWWUZpUc/0bT3b4nUJvBUYcx1hz93KMsDKpdNKpy9HJnoNT+wZ
M8Z4KchToD2VqIVuu4BSDDUe0PbzD7Q8dbvpPn3/fw9f7uGbFo6QR2pdeccEu/KxKRx8eHwO6oRdui
MeySHwNiWOAuOAuqn97fctvojsM2oMHa7Z08apFGMwjHI98cYrOVHJH1bJIvZavlJBwZR15BTs6/fmX
nrMXXr5p2XzMenDILFKosIp2JaFRcEq0Gu8pU6dHO+pHdXv84LFp8V5+uugTF1otwNcwaSkHl7KCYghW
AdOjhXaammwDhCLtUfqfekvOkEp/9wChnEzV8ozgoR/C1WdnZ6AtA8/6Ut18Bf8DcLZOo4pzMO2J1srX
1Vp2DdwxDti0eDHAl3eutDovLAxwFPu9hZMiwKiPFY4Sx97f28LKmM8AKQTU0bQ0SgF59Kyx7ScPcNzy
0J4V2xZt0iwewtTtVO3ShVU3rZpQuHXzebeI2UTURigQ39oSh1GlKGVqlZ97dFK2r371TS9hv/ZkAy980XL
w2N3Dst18/mVrolmVHCPhVaGLrigYWnAdfh3Bgt167gbX6ENJsi4AvbYWEtY9I0z5DG2cJ4Unk2jfV56DA
vF3Jpf2KJOEVtOnqrISj+Hw/hHHH6tRhWFpBOcPPDwdEJXdoJPFXY6Oj/hWcFLIS7wQEq057xZeqabe3E
z18Vbhs1rEp422duGyWr89BalX9lMLoNZODqxRObE6TkEKE78wl7ePPXfVXv3YLbt+5aInUNThYyOkTm
VrtFzZ59wVzkEzKs9UNf3f/eAh2gDtg5erHIDp+RUb0SkBeFcDBwzWjblNYWKTVsGEa7c68VlpCfFGCYaei
2sqQ0jUQ1qvh9ZWRIGb88/kfsrxqgFMpet10aaMjHVqFcar59EPgVtJxvLqKyU5m20owB7nb3hkolo55jsj
18Liv/XKpLKgZpjk8GlmS1pZWl4LyxUTlfOgjdjaXE8WSbNUXXh+B22ZzGrzMjQS7R4jAG+88547OZrq1uY
aSkCOhVSxJuFcuAWQBExlJokvX7p03q5cPo/mxKnk/HX8AOWkNvHutURd2lvbt3MB16ASRs3AaUWrxv
akVPE2ix5FlSc7wjJoBQCULAu4pQ6VjJNFSc1OzzgVmGhIWWxRpJALwx91/LEaVR/pcQZUJVOIm+0e7N
n65hOftpTX1qThXZyTBDxHRWjXcbC6prgj5oKOUhrwZLntwDafPAE4aKxCCkku+dRms3qMqTfIecoDzpf
Or3otVN2JKvspkUTg0t3qt1p+srkJv+T9Q1Wp7sJJtfZnjPMGB9SNK8A9Nz+HoGmJb5vrQFkwm75nPvZm
/d5tOjLj4Zb9vV11BAMtoE42ANaacqUgasm1Cwd9KEdG5TqUyY8sevhONV+9EFunjqMnJ0apkZN5/ipOj
iIDys6i4ZxXFUoAVUoLBqtoPPMkE61eEBBkIbQCNa7cUWlYrp1LTznPV7E635GFezOuPg6M3CIo6WSAM
ogBDlWAHg0RFBSIIiwXL1zxWg3qv/lMxBKjsq8/U1rl8goOI33MbdsxwqTQWQHloRvdY1wePX6Cb1Cytb
Utvg+Ny8/Sv0ogSfIc6lZv2DQUJZlOoPE7WBVGGV9hNKgDxoBdvgjFi2CBAa+23JSJVf+owMgM14liGeSw
SVh9oDlC0JQ/wvL/95v+Dx/iazqTysvsojXEV1UgSxs9KNkiHE7a1s6Rvft4y6K5OY/5nZTrdL7GAAmigwbD
Lt7jNh14QAcoyB2xlcVpu3Zh2V587go3Kc9XOQFazqFsKc6LRmy3Q2iAChq4ibbetYdrT7z8d2F2iXYFLZUt
Aow6HRfzuOUJ5qwPd80Xpvw34ouazTmr1d84OeKcLUABF4U/ZbMpj7fqtfa912SBeNbUzDTv4UwhINo
Q2AuRASCFVBTeEZ0Ix0JWDHatctRCWBEEzKFmuRo4jZrok/nXphtIDfcCFQG8TQRCfFgCJE6phI+DgxOsj
KZjJwsKVZtpRJvbOILxqAQCJwiBTWktPkIlwQ/gA6TxBWSOtdJTlGlrEyGFnshHaMDPNZGQifLdrgA8ts2tL
XvxpZdsGW6M8vWH1lydlI5QPmkLcb8JPH+F2TbWt+3u/TXb3DmAr85bLAFtwWQmcVRVGFlQG+NnSB
NHsIaBcdva5R27eRUeDF3TroRqg1bkqnjJZa6peX4wyi8VJ54AVdAKYxn/kwDVM89plKYTD0uHdoDkSV8
qg0xOk9bh1Bste+sBXLNjeLh4sUi7+CE/xnw26PQTG7QqVjre8Zt99ROfspto0Rk0bK+LOUIbiG7kp+Z8frn
dwTwOovDdhj189MD6mBTVa1L8UtlTV288RwepdI3ioJjHAHQCLaWdn7VMRDNRVSWzINkqk64kCPGs
ZFJ5sZHJqlTNwMzOAgZAiPnVKkv9VWhJXrh4lGZtQA0aCNBJO6NdVXRY/FHab1EVmBFILYkuIVAKX3neK
+eQ0yhHT4MQR/PpfNL0cuZU8Vr5C6ooowH1jShAjjSxlkmX6eP9nV072N13QVRMWytz+7QjgGcuWjNT
nEc7jXwmT2Urpa0voU21arVVV6kdSO2oaYtzOV9t0IbL37x+3Y6ODhG6MNQtCxee5z6aPpWpBXYxTLN
2O5TzJhN//9G6vXf3sT3ZOaTNYVtZuch3VNh3cn9K5UPO8SNKdmVpypZnMlZIco9QnZOTpi3MzNm5Jc
DLOIfBg1DnMVPlxoJaQSTkkdTvf/xAQPXApHqbh7zCfUClKVVFRhSPC0UzmPug3ds9tvVD8RrFY8MeBSi
XSqiWJk6HkmPHmPgLdvXyTY+dCqCe7xiCVKPtMtm8pXKzdvv+E6R5i5tIcB5lSwUsnU9Obgwwqyx6Kp6

xBOZoFycnnytguk/oBDkfaEsG4fjoAC0YtDTaaH5h1uZwalShJQXN0POkZ3lpwzMcLMVhOa+y3bWhmfh
3FA8XpTiZrMBZkcBJk8ksa7M1FQhTdGAhHrb1O3fxvvftZ//KX0FjTnJ3pW08oE0bBNxuU5sAI4IemgFAD
LLCX0r2UZ0o35AMQREfVD0rZRNF9Xu0ogSiXK3Z9sGRbeLlb3AtMGUlHFg5JFOzRXcC5X0rIylXmGWAA
Wuza7N4+yXuUZbm1vXnnafW6hUsnaISx2hjFUWeQQumbRGPXIsHfe0o7ZKGz0EZqs2+/ebvf8O2FHO
uKyMuYPMrF0wFMFT/QXX90/GAff6Vm1ZIBCw+bFgVLa3EkwvnLtnSXNHjv27tnZryv8ApULkGvc2b3//
4wYDKN73+JyOn2GEdMKhohLaP1NRnKKK1OUk7Qc3f3cLjB7CqYKKa8GVMvUq6XD0/a5965botzqStU
W1BsDtoAqXLARC0ljhhrTWwO4+gBw2lwdFhsRy8COcLiY9pepOb00yTMu6Vm5qKci5NKNCwBny3hb
M3VcjZ+dUVy6ZTdn5l3p23WbSFekgeZgiKoWU2peMTnLciQ6tchRMAoqlNpJsBVC5oCDB5PJT3xWM1X
alCET2eS5tw0/wXtgWE4rXf+30H0lf+2i+g1VteFEOlxLX8XKtgJVxR1G6A+5Q21YJGlJXzWq2Qhe3Qj/lHlO
2VQOiwDDhkmuVSqUjtHKiVAAEs1QhqIUeqVGraBqb+lZpSVVJEy1RmSEnR2gshGcv6PaTyU/Zgc9v3pso
mMoAey1TMQ0u63Dl0Y1RnLDcsDF367K1rduX8iqfl1asnXuwjThsG3GsknQOox/b1r79lh0cVS2aK+LMp
8EA/lGCvvnjLbpybt3HjyDI4b0GEW3NOqm2rpT0jfBnNAip/AemfgBXaIBSqyuEfdfwAQNXXMFmIsI8PH
QvrsU1M0gnEWppAJXZU7S0G33u4fWK/+Vtfd49TmmC6mLTz52bt+sVFSDZOFh6/ytxoZ48AfEh5rseVl
pUwf4q76ubOX7hsu9t7EPsNm1qYN5VAV4ZUp12F4x6ZYvsNmVm0lurvq77//HTaXnzhBbt69QoS3/Wk
EEYDjhrCfAMMiL/2CFDlvyyeshwfOVdNvHnfzQ+tHUM7SMqVtDEEDDTGB0HTgfLmPXVPYTH+agmLykR
GewPL49HKBO5r7VG3R39o9mkSi1byiqgTxhFzDeB1UgW4MfNKhlFfKvyk/AFVhdEESMCdAoDM59pRk
C/y9Sh8FyESV8a5mlTYS0GpNIXddpC+//5dW9/dhYapuiGgYEw0Ha5aDUoiL6Mg1G7VC1A8vA+YlJuxtv
7YXrl1w1Lcew6un4fOzE5nbHoqA2DLfEfATKNdCwhayB483vWdAI8rVehSgj6/bF/8zCdsmTHo12pmOJh
RBF4lhrRzoRJblO4oy6zaBo5OHTiY3snc5x91/GBAlenjwgzb6TsRX4d/hBcu7qQpOUSZm0l4eOjf/rvfQiM
2bXZ+2l58/iKac2x9N/MM8DDCOCFB8KDNw7rtlrrWHOHVhhhcNFmYQVK5hj4mTNOh9W7HuWUfz1lF
LqThNCkgc12cztsFvNjZmSzmJe2ZSyp1rqwlbVE5woHy1Z2YViWEyPsXK5KjIhDJ0VN4RLWvlFsqOMph8k
VxAhi3ru+AUw79D/jxApwiaNIQOCPyWPleTFVQMO21DoKI9qOh/gt9pmeDkbSimL2fzM8jUOo8ShNUe
M9DgrooA6rr631vg+jD6WBOKrsgLJxfyTKiCvjkCK2sAHQLx2EDp/M23vv69rapoEYhlUVg41bDEiaTOUz2
LGMQxyLEbe/whOuM7YY2JW7WUAJjBBbhgxLM5OM2B1i11kmLGzNprTdGOw4TXghN+amqsPizP/NT
9urLN20Mj2kB7ABKIZ9N2nROEwSa3UQo6Uc/uJ9vH+qc0/74o44fUKNOOJpCLToxww6UcDh4pbI6XTk
enE6B8m3MQ6M54Mb6vn1MD77Sg5+6NoHbycNrwp3qKhn0/n17tHVoF66/bMW5ZTSwpuF61m1gc
vy+xvZo/RHnanLjWtcOaPGun3/uitcxml+YYoABNeAIGudEcBTScWeIduUzWc99nJT3juI5T9L05BSpexTS
0vtng68O1ecDzimP2G+dLwqonvugHwkkkllxLLj4an7WthiwBoK0eGFV/qyb9xGD4EPAOaRF5eBJu+pN1
6oCGWCTttX7Eh4Jkdqh5zrULh1nYUJdXxEMHTq/tL/OFRzLakwmQTSXP5TFwdmrY1n24bPrDx57FT5V
UFHmlJxBgbUL59AEibbP1Dy/ohSKjGTTUSwoVu5gE0uVs1s3L1suByUB6JpN0vSwkrdVVvPe3Ye2vLRoi/
DkOP1OD+OsIsD0qypfox9Ohd174wc+fkCgTkIsflU6RkCFjrpEq0z4IQ6TzOohpk+7KM/MAlA6TXPXh5VD
pJfnmBqtVK1U2lY5qFqngROVzGKq9q1U69rSufMMEiBqVS2EBq9pzr9ZgbOpztPQzq0u26ufeBEnKg5oV
foR2jHQeiWFhIJoRDQZrSsWJ5WcxUG1+YSC4Zr/17orURHdjcyeQCsgSJNq+vNsJk5A1cyOZl4mwOEH/qt
Jh+vzs4c06ixm9V/9yr+ycq1q/+X/7O9YC0CCEfpI0wyTn2qCVzFVhy7/aZWanAiPCHAB5TxIEeiPYCgw6D
gDqo6zgT5rJ7/iTb6t8ysWBlhpIP6DLByOIE6kdrTRZhsqtCZq8O779+zeg4c+IyVHLqzYKMCWw5bMFa2D
V69tI5WZpaLADcZuBnMODbfnrl+FMtFP9HOY62pdmYAqodRaKhVh1qZys/msnVucY5xQSro/b/V32v+
DHj+g6Qeo4hMjdYaASofpfzSkwcDcQ2JVDTmW0vLcHBqUoYnAb7hZ8ValE+6UqpjyrjXaaLv6wGJ0Sn52
zsr1mt158ADHKeDLd5v1Y6uVD5Dopi+xff7KRbt16zmbmiowLmhPtAQ2Bu2KZz1WQVw0Ex0WwelQ0F
x7WGkxmWZ3UvGkm03VgBJo1GR6zDtNWvMMCEp50xy8tLA+UwFfzYc7fiD+Mo06zjSuTLIH6nlkR1q9
Di3TbxMxHEK0ujQ3/eaA5DPhCd7Bk0n/aZmweNnZ4J1pWpl8JX5o/t8166lAnJlNvTcZNiAvh4SHNLKMn
Z8C8KjCofpBdxaAp6vN2mMrDnBV6WX/4MTuP1j3HIHHaxvOfdVX+blVawdUEl8CrBMOcUwzVpzK2r7
qHWD2FVO+sjpnU+mA7e/u4TtMoWlzdoJSSEYivhHFdD5t1y+ft7T63nv9OwBV239QwP6AQPXbnlyTh4
ZA8/6aPhUItzAvxwBVqXnKEmq1VeluyGcBS+QztntybB+s71gVkMYzCzaVKFqrVLMO/FOzJCU89v29bTQ
dDtZIJWOG9sJzV+15SHqRwVcMVMDStGs8oTqdtEd8j0HU2nvt4amtgARYWWzXOvzTTisCtNqs2KDm9
pX2Jy0q868u0EAqa13PdQ3ln/agGKn0ZC/9yU1PNKtAo+e+xFyA4+VsWCZR5Tv71sMUa6CVCyGtXTo58
WlE1c0PqU7UaZeLA6uN7mRxeiXDCJy+crWLwHAdfWdyPfmE9CSNURsnQiObhlZzoVI+A/eEtfCpbu6z0
dKKgKHPnE08YJ2DcyEoMtky85pYuHPvnt1Gy+7v71t6etn60QKOGlqTe0yk0qYq1yodqjoHYXiOT4XGBn
Z1JWNTubznQ2hnGJWUDIguQdsurizaTE4LLrXRRsr76uxQ+z9CoHK49HIBXVOdxQ3rcpNhM6sxMKUS3

mGzqkk+CL4GqY6XmLcugKp3+/b4oGL7FbRdIAtLpQN5T/PJKvilrW6ODrcBzcC+9NlX7MaNC/BLJBIwRV
xI6GuuLVMJDGlCn6ZMbjgAWsYOHNWg4g3+d9YVrh35ptd1ov1KjNZ08PfeuL531oGT5zL7k/VOogHisa
oR4NrrrNvUHbxOdIY+C4NzbfWBkk8moTbNJil7SnVjRU80gaDVofqh0ty0jEWHtL1imxIcFS3WDiNcxemK
jjOtqkPX9gf/tNWRwKoJEF89IKHE4riXz3M1SE6lTPMYC6GiEpNzSGNr2jjB9dKet/r44UP7xpu3bW2valPz
S85hB9ABRXKQRyzAwGcOlVSdCLTs5WuzUIJpLJAWAmr2TMpjhNOWsCvnlqAAiv1qnykBdXLdP+3xgwF
V/aRxPP2FhFThHQHH80kx/1q5uHl8gNbUrNCUc1AlDHfxdpOY7cd0wvuPDvBKldyCh40JV6KGiseqcPOV
S8v+mC1mLSGO5IXRMN/RuINHHa1ZGA2stiJ3UHqbGEiBS7E4BmHyXT7gc81SiZ6Iteg9xTVVW1XesTSpt
JYOVd7T76Rx9dBzUQE9VzhOG/cq/qj1PRMQ65+AajYTTdhr3/g699W2L/6lH/eZIxXuEO9UToHMqhYbq
kySHD/Ju7Zf9AWBikqg8ZX2Jk2upTRaE6branjUxgG/kaCoTTrOQKvZHcUlFUcY4NgJkKpBpfTJAM95CyBz
v5xHERoBbjLicuIAIf1aqzdd409P521j89i+/vp79sH9x15YrheMwLfHtorTqjzTQbtuRbz4T7xwyVam1V7lRi
CUiuu2217D9jrOcz4Vh+e2XahDkaT31w9z/Ol+PemryR8hxTtt6AOYTiVNJWcU9pCGi2HGlBGuTbz6aM7
ZfMpm4KstHK8xHmWltmt7e4+sml/al7/0CfvUKzdsoZjyLKpwr+3LlaMMjjxabR1E9/NXU5IKH6lQr6RYg
WK0h5wAmnIGUv2VaeYJv3HMTgb4ND55JqP6q+/JmZLTpd9pMZ0cQwX9FSdVUF65rpot6rT7rkX0Ohl
PWy4rU5mgByaCIy0YxnSGAYG8a1lbmWPtFKhiFIq/CsByrBQTVbqfECTYw4QnvJdmC6hnTpOAesZRz8C
q70xuBygCugCO0ZAxGHq4SxlMumcBUoJGPyGQPqWr/mBs5FSeQMc0g6cNNtYeP/ENjb/82Zftb/0Pf8Y
uwUNHtFeb0ikxvlWpWJNHCmG7eWnVChpnpxEx74c4VCKXLniZUfFSxUwnnv6kuT/M8YNr1A8f9NvZz
6XlvOc4jtEOm8fH1mugQXiuOJqm6QZ04vTSRat0wvb//a237JtvvG75YtS+9LnPeEGEDCo1AAUIoJnxhjgT
0shvwvC6wBCnhsF0AHJdTUfK2fHYozQKI6ZhlqZS5+h74mnC6RAtqE+lYfx7vKktZxS4d80odPG5QCqHhit
45yrPVMkZqgmlBBaZy0RC9Uf5vv7TvfM90Yp+vYHmhB9H8bgBgEoLiSZo8zTtb+W7UvM3EucRQ6g4lHS
u5BXNWAk0uqi8ctfwtFVFK5Rip4kImWZtAakl37qori2eGkOT0yvWxrIN6JdoAm2MZlOqoLbGgXB4DSj5
EKMo4BUnRmjEhSW8skxSJnIgG1w7yjiiYiw3u2i1QcTevb9h33jjfU8CGsHZP/7CTfvpL3/BLizlrF2vQGO6j
MNkSYq8+5lizrJcR5Fqg6s6TZMDqY76IY4fkKOe/j07vn3t7/5A+8xvH5eQVpnzsGcUNfEEE5kU2rZoTTC4v
n5g69uTxXuqd6RKI32kNy4SKhPFH+WEKqyiztUGu5NCZtIeGlPAhPOkALkAFRqF4IUIBnRBRXwZSdM+p6
qzOjc9606f1rNL6rXbyRBtJc2lNUra10npf8pgEijiKTQTvGaI9tRS6/29smeIBSMJO1JdUUClhGCa5wnCJ0f
Hpg1wReGH2FqJrPxtJZusP1m3cyurXDfFPQGpcM9zDTT5UKqUbHV5xabyeRe6VEJasIf14No9peQEfC2X
NiHWnv2L8/NW1xJ0QKbPNNPjsVoESAXtRnj3MvECsLLEFFVQBpYKnyl0J/AKUKIJeiIB0fBL26tvpQAa+Bg
JEDwS+KJJwB/zxYJr65v2xuvfsr/8E1+yT716yxgQazeb9G8bAdVCxqFpr65iXluHuoHgKvIrAIPz0x8lUP+kB
2dUJtAh3mFXc/hov7AK3CK5da0cBUDKoNHOgEFMZBsQK4G5j6mVx6t1+R4XBQlpiPwQp0lpctrlRIvWlJ
PqyxdaHTpojCbOMXBoEsz1OCZ9KO0J74NTKjNfpkhJ3blsznMA3r/3GL6c97CKkrDlcCjRW9XvVEjhpFa3
w3KVe6ii4dv26MEjG4SStnL5lh1VtISG3zAamVQEU3jEID60Wy98AYCIp6tMpepPiYYM7f79R7a0BBinZj
GvDGz7yIpoHZUmWlt7bM/fvGWzM0VAxP1FQt7GQj5rWTSvsrmUJJPgviuYaFUf6Xcna+6VMqlVqQJXT
0txEFqVOlcERqsaRgCviICWuA/NWoQQSu3t5DXGpAz4niyKVjyM0ApCgaiaihAjie4UChiyUOLXWnb04PE
jy6rfCgXf3iiNkxQB1PVSzZKMy3Qh5QUn/IdcQuIqS8ZI6sUPdXwkQJVpkcTWueE9ZeYzAFlAUQEA0jBheF
8GTqcOCgJWEXKl3CmtTdVEpC1lRuRsiKc5NwMb0rBa6y/NpPlwbQCrvNV0PGPa+EuO3DDk3Qv4kGb+
6O5k4rTQb+LlJhmEke1XVFEFVsjXVHfq6199DU1XhBZETJUBFZbX9dKRkT16uIZmSdi15z+BxoxwX6rk1/
BqK7XaoT1Ze2gvv/RFgKm6XE0Hj3aXUYLIxpNNW1lenRSWQysHQh3L5DDl5ZJ72RdwPJz3Yk0ULBcVkT
aKKcTH+GoxniY2lFCjOgDThaQLqOoTTOOcSii0vkrUQkpLBd+0w6tyBrR7S6VctjiCKioxgBL45IIu4DxWoTi
ogPpTAyd1S58E0I7qG70nYVPqoepBRRIx293ddYUwW5yx6XzBFgpZKx3jexQycFZ4/Gn3C6hOxfypNOoP
d/yxS1H+tIdMj8Ak7afyjJJcLVPY29tzEBfgWtpBpKcYrEaVjhB/kxB62UE38XA5NLFQ5qEUAKV18uJSKs+j1
Y5KkOEJXuXEQZBXLk0n6ZfzoKRuTfGOtYIUByCcjALQhj3eObYttMC9tS1b2zpAgFS66MjK9b6lCtOWyE4
BVDQT7daWjloHVu/QRrS7qlorvqktwTXr1FaWVK2Glpn1UFMHB1CFLIZIgASwCvVQ6UXRYCWd1OVM
Mfj93siaAF5FehXTjMXS7nhplYRimJVG3Xb3jzxDamP7wA5p7/r2niXSOdNuq01lIUVSFuK3skyT7CUVbIO
V0veySqoLoF1MMgopgUFtuCGQasrVpfdUij3T/vQhSClJR+dwa336UPELjZGSfBT28iXmu3umLTy1LVM+
k/Df6ut+8GQSE3HGP3nvhzg+EqCqWa6oeaIsnwaOhmrBq/N8Rw9JJzcrQMrEK1FESz/0V3FHzdG7JpWJ
FxADmEE8Wnd6eC3NIo6mmk4CRFAdx3sDKILK9KjkjJaIhOFmEfixlhofVtr2u99637719geANmXr+yXrK5
QViFssPUVPxPlO1abmFvCag1bBsVAAfQx31rJlpeNp0wfFR7VeSqsIfKoXUGrdVAyhaiqxGQdD+z1p6lEcU
rWwFF/0vT55T1X7JrehhOue1/1XBEO1nVQoTn+1dZL2PGiiBeegDaJH2rCuXGvZQalhf/CNN20DqjI1u4w
Aqb4UwhnB04Y2adMJxWlFeabg23JKo44TUTD6BSGWVp3M1PFQsjgWT1PFSpZ2MCrmyxjJkonbylmTd

RF45XCqGLO/y7UUO16aLVqS80gJTSIKpyDgEFT13R/2+EiAyuhOvEqXTCWGMAB4q8pk8p1A8KClbaVVf
WNbOkag1V+l0gmccpp0f/q9vNIYGlHg11IQrV3SLE5XmgsyJU2n/EzVjS8y8IoJVjCPpebA3r23YffX9mzvS
BMRbS+MMI7ELZop0Ilha/A9JTgzel6AAmz44kPt6UkPo/nQkLRH96JAehVu2QaYLZWmQTC0t1O1Wma
g4jgWNYBa5jfK71TCy8hNvDizqlIr7VCzcFoyrSiCkme0RktWQoH+iZOGgQF4aoOy0ZRgrSlRtSscS6IVDSes
btGkdlfs2eb2jtcu1bomZfQXMMHKHpO1wZZ4vztu6DflvPpMGgDVbJUsnRJiBGLXfz5mmijAIaLfNUHiUR
K+p3oB0qSiV4pQaAfFfDptq0sLAByga7xovAN1cjbntwLqfwqd+pFwVCVXq2WaMhRgtfPIEc5MhQEGYV
5ASxKbTGQYVMw/TVBcUdtf6/tajCfJ1QCKCminOplKL1aG5lQ5nD7nUAArmk5Zg9EbKZTE++v379n9h0+
s1sPshxnYANpBxXMZJC0vUf1OVcQ7d+tFXwGqfbCG3YFHYgUqrQzIT+dtfXOD9ssR6E1qQtE+JWBIYDTJI
S9b2/CoFKcoS3gQgntz05hP52bS7ppGRctOQScOD06wAPBEfQrqfNEiAqJdsgXSeFw1WBddgCGNaNOu
zSzOe8aYqvwJYIUcPBcvW2DRxrsCfQSvVBljWunq242nw752/sLinHPoSGDoDpg0tfpeYTuNjawRQwAgp
VcAKhpYwGXUaANWj7ETzKT19V35DGq3Ztn0fS3xSaKJZwo4VXj7Z66BY57zy1c4ewvxeDqBqvCJtKqkS6
Za03naS1X7NDUBmAoCKx95wCAFh0FPBlEISpIu7SWSrwHTXLxoQgcnJAWoFYxuNyaJKHF4V5PLtMdhb
Wpnu6WO7Wxt2tr7b0P4j2zh8k2LZ4qYyqblpqZdg+UUY4Qz3n/82OYuXfRBG2CSlfPqkwpw5u2tDbx6c
WJt01OwKIOt8pbFXN6KeLtKL9TEgEJubTjxPIBQ4bFumXZxD2NVyBY1AQSarVPFuwRtV90A7SQtx0caSY
6WlouolKbKyzuQMZ0qwjY1PW3HaObCzIzTDdWCxbhjWWK+kkFhL20erH0WFLRQNEQbU0xhSdYe3b
UcFufTn/y4TecA/0weC6NZI2VONdxRkwlXlRPtjS+NKzogiyYFoIhBTddAINX36iNtRKzKL/qxaA83aVO5LA
Kh0p94/hp0oYjxcEQCfuFVaYY6BPmnEqhqppaVBNAg0g4eCuEuBNQTtGo4lbYBmqfLYIWwulrWrClNxU
G1KlPLMRQUnwAVpwZ6oKwo5BZTCa/D1GtwoGu2jyP09v0NW2OgpQFXsjGvY7pxULGrz7/Cb8093w4
AySUjePEBu//ovvUBurSaNqtVxc+VmTk6P23z8K1LFy/4ak4JTCyb9rBRgoEMAN4x7dOeTrVmzRhZz8NVf
f5MPOcrLbXKFnlz7qfIhHPOMQOONtau0ZqhUwkbH0h6fixB44UiD4pvrq9tQI9idm/9se0c7Ht4S6+1Zm
m6OG/1mvb5UpWTLH0TxWJwIu5J0RJt4dmoVOzmjSvWhYZot8CZ4iSctDJbsJksfSnroFBYDE7NPcgvgHJ
yj5qBC/nUrzZTE/XyCQra7fFYxkJRGNU90ERKsZBD8DUiHIKQx7EmWlvPlXEoZaTjKQWqhoADoCqorUFC
ZHUlD0If41zU4V8BuFcGj3XUniRTyEkpTBWsxWeajVGYRRX7lMyhcufiuWGcKq336XdGcMOWrW1X7P7
WkVUGsDEGDv1oM5E+nRSxDx5smvbHWj1/wXYx92GcisSYzu61bO94x1poytXVZVtkEGfzObt1+bxryzD
X1NJjqQcVQevSwz5rBK9MMKiqwSqtJL4aVXkagOoF05qKEKBp+bzHNVQJRYOthYoKP6mStrTnsFuG02E
K8c49y0ucvSOzG7BMRhWco9CAqJXh03toWsV/Vezh3v3HfD9mB3tHLrSajFABuOzsvAWwUDLN9+/ds5
vXr9q51RXb392hvzu+T5OWfi8W0nZpLu3Lb3zTNsapjuOX4rl2lRZVlXMlCtYaAmbGxPmqeC3fFu/Wch9
NmoiOFekz7dwipalEH5+YFMGWKdHZOZ9mygTQpxSoOh0PmXBUhnt8IkL+Lt50Szv4nfhWiFqP3qq3ke
gQHa9ZI23MwCBjlpJp7RDHTfKetK1mtqJwTe1C99pr79v7Hzy0bHHB8vMrVpZZG3AdhCMcBjh0rnZEkX
OiSYPa8YGpnnyKPlyenwGgc1ZYnLHzK3BCfqY9ocS3ZBrVRgmFCmaoUojybBV/FTdVwohCUdKY0rgCK2
oWTVaxdHqGMdKEBJpUpSsVnkKjqlR6NKJSkoAYHqqa9iq/ow0lFEbTlG0E7q0ZJi1LViKzBjWEllQyzQz0Q
zToydY+AhL2DP2Dw5KtPd6yEpajy/3NrKziQOWgEm27euWGaa8Drf+PRpVWqLpaCasdbFi3tu37Jagqiq
xEBqukel+gz4VH25RLy0dop1YZS1NK2DRTxgBBj4L0l1b0YvJTaifUQyBlfEMAVfcSFLcF4I7bybB/d9jqT3I8
NKYfkLopOG2dQ1dEnRtSMawKnrPe64eSaFccImVDATKFoJW/GASMEYDS0Jp+TGMio4K/VTuCb+4do
WEafdtCs9QB+aXLVz1rRyZNZrgRpZMAenDQsvLuEwvgSNy8eslne86dP29TmPgcmnPYKrnZ6zQ1k0TnZ
7Jo9gBa1Kw3wrvFZMups1HHHq5veNG1cSJrJcCgNVHdBtooFrRW+cC2MNMvvfJjmFLaHkZAB10vpKaU
R9UieOHWSwxiCEGLOUC0wFDTi8qp7bRrtry8BLAizveiYU14wNsZ+QS8qNGRGUZ4Bmh0hFuRCuW0lk4
adu/OAzuo1+yA/tzCssx4IWQVr0va9OwiXLdoWvpSgr8eH21znrYtLiy4ldJeppfPLVoe3h6FV2tPA/VZgPM
r2V2rAqIIo6aelTuQkGDRxzOaOcPaSYVq+bpA6ceHUfQ9b52+/KGOjwaoAqV0v1rI2SVZMiUChC6mvFV5
j80hJD+A+VMslAd6FYyjpUQXognTLnVjNEyL893b2LEqJL+Fhx6JZaxUbdjGxrZdvHDBlA+6u72NJsA0lo9N
m6YV03E7vzDjWT43b1zmqhKDoC/f1v5G2lNV4RXxMuVrKhYpZ2bnoGQH2mac70ubKItLFbeHobil5hYh
FyELoeUHCEAqiuDVjmxzY91efeUn4eAtTHxHvWqzs1N2eHhgjx+v2/VrN7kvxSpV+C1sdRw6VLwzosc4P
9oGXZ/JfEr7istOJUNWzMSwLHGczRSAq/vsnRxTLTMPWYxrLHii9m0E5cG9Ndvfr+KY1eGvOFD5aZ/qTc
O7pSTEqRdXF9DscXwDFWRGE09l4eZZm+P81q1YZISAYeUGo7ADVCEtRTDkF/hUNDRlNpNDEytG4l36I
zt+JECV5VA8zmNsgFX7yR/Dv2p99INmbZBeBY9lLhWSklMh06+pTJWIXD+Cix6VZQ/5JIzXm2SQw3jK+6
a9o5TIsgVYRjxfmZuyi3C0j71wy7IpuBsarN1oomkQEn6vFZdCSBdTWylVbG97k99u4ICd2IULVzB9SoXTC
gCtsuyhcUKeuXTc6trc8gVxET4PemnMFIPdwzvf2npiy0svc79RLykkfqoiYYp+rK09sStXtOcTl/XrI3wAQgv
eROfuPfgAKrLiG7cpCqLykooUiG9vPnngmq+oohALRVtdXsSZUwBflfWgGTyGaMJ8lYvTRtMA+de//qZ9c

PuhHRwcOoddWFwAeNCqVNGy8+e8TQ0ERZX/tMkuasCW57L2sRsXLRkb+wZ18Th+AFTMl5IDdsW2V
R1wvjhtBWiEVhT4cWrafxTHRxPwd3SePuXwogq81iySTMUk+VlZVni/PJmsR9IkARCVmmEElacpr1v1rW
RuD/DutdQXH8ITkrW8Q3FLTZI+vv8B3vTIXrpxyX7ys5+wi8tzeKVp0xbeKualRGKtfddiu91dPGk867sbB5j
0LTvGw9U0ZQD+O7eM6YR2jNCeAXijnBfxOGWvazmximDIA29DF7TOXrGMthfBKGFql50qaB8nlWPXxg
8K7ZQQAIWStIZLfeKrTLnvOmCu8lBW/9T0LNdQmp2SQUJotSnPgdhBOBdXz9EHI1t/so1glkxbsg+BsaZd
8znVnxWPDNjJ8ZEnZl+9chEeet1UDLmFEO3tbtBOtGGuiDWh/2iGZqIS8ExlhqE2vFykBEFREyXOqJaB9sg
S5RgqCsMYxHD8ZvH0vQofw6goqecM/IiOj4ijfuiUPJVG1WWkLeWJKOQkc3RUa3gK2YDeEygVDFf2lLZ0
qdJhcgJidOhhtW9ff/exryFXEoUAoRWP2ps/jSlV7c8vfuaTtro4Y43SIYNZAfjm9eoLxVmroqXu3H9kmzv7
dD7DBoc76dPxMzM2O5X3VMDHDx4waHjeDL4cE3FU1Q/Ix+DPCI62di8UZ3yu328PbRtl0DoA9XB/z+u
Uemkd7kJOnfJOxQVVMn5pcdUBIiswwqwWp2YAGOevnKC1ura0ugSwVQAtwYjlvKvCdMX2Tw7tEy+/j
DCKE9c8BqzoRRdhCCH52kT36pVly6Qjzl+n8kXb0EQFYJ/h3jRt/eDuPdve3rWDCloxsWKpTMqKRa3vB4y
JoF27OGdXL81bZNiUV4knr6wqPH0wqFKZ0t6pZBwtGrSFYp6xnQib8o81jnTZj+T4iIDK4+wGdPZTEyEw6
lBODTrMv1aGax6idbREIwBItXWMb+cNOFQIVntNxRIF+FfTvvbN1/B4j9CobdfQUzhIN65dtk++/Dzaa2D
He7to7Z6lIPwtvKIWfPQIz7jS6NjWzh7mvWQvvfSyNVGt5eFpkjSACzH4htYoo6UFFm1y2wGoHcBhXfgk
wNMGF6lMxmfUtJBN4SAxOF8nxR0pSbpaL8FzBwgXvFeDjLZUQQgt1ejTFm7KGnWsAO8pDqyNkHEX/V
6VTKIkaO1ln0ql7RBQNvtQmYVlLt/3jWyHWBKtAk3H6afyib32ja/Zreev2vzMtE1PKT0w46EnVZNRgF7pj
VrMqNjsm7c3beNYfT6yebSnNqdL04a/9KVP2IXVgtd2jUfQk/SDzyjTNxOLpV1jYrYwPe1z+apKE/wRctOz
40cKVAy8A1RDpkNbgquY7+7RCaYejpVMT+pY8ZlqbiqeqEQWLW+YzyXsW29v2C/98j/DmUrZrReetwu
XLtLZWiKieB7OmDQc59TOy4fluu0dl6w3DHp57yL8Som/ckjmVi5YbaSVmVCPTtMLVwxaNZ/DH2KCRT
e0jaYGayof9QIWETT34sISIEp5qErxRwXJC9kM9xdEeKr+HYXyG82Ki6E8/1ql4YvnSifa1xRLcdKH3+KgwB
EVitNOhkcISL3ZdPOvTXRVaqeP16/ivFOFGcCi5J2A5RFAUYiCttR5/MgqOI3Xrl/zyEW7VUGzJu3KxRXf20
mgFcCU2KwZs2p7ZI93qvat1183FQuenp6yy+dX7Gd/+nM2hNvH5MXTJiXRRLBQPZVhx3QoHXMKwUn
x1zOsBFT5HIzRj5Ci/tkCVcnTsonaHK3S7NkQ89fT7xiUHiZR2zwquWM6lbcU2vbwuGpvvPe+1en0C9eu
WJ+O1PojZcA30X5PNretjNPd6AAyOGat1YdrZVBYmgqkGeO+vfPG63b+8mUvNKwE7lYVbqfia2hNbbs4
U5iyaTSUqjkr678wy4BLS9FOXUfg7Xc1+YBocH2vUi0NjUbK4fTohgUgDWY6kzbfMLeLLQkocA6jbnQ9O
SeRjjpn1+pRbY7cRVK1w/TRQcV29g/sqAyFgWPX6z3LqKYXDk4qnfccUyX1nEBxzp1b4ZozaGylFNbR2vB
xa1uvWfUNeZ+7ft0u4AAqPqxZvxQA115Ur7/2BuA1+8rP/ZTdvLZiO08OPZ80ChC1Y+EI7qr9q1TnsIhTO
w1fPaOjY3isnOI/X0A9O/N3AVV/deC/01sy4Vood1xtWhmvt8VzJXrIQZC2bLeqllEYqNHw8FSHwa6geUK
JjJWgCQqqay1RtVKyuw82bLc0ssL0vOeVqhSiymCqkkodMOQwdZXjA0/+lcOkee4cPG15dsouLc5hQqfw
gOfRWFqb1LXjSsVCSS2lgTfjUClra8Bv+mibCKZayYbaj1WDrorbmkmTpzHWchQ8bfFG5QW0WiqFJKDS
NVpVi7lXUESbvmkrxo5z9LBph2yFhjSDV6me2NHRkW3vntjdu+se263WO7Z0/qLP4uWmMpZBcwaCCA
/3onCXDev8rXOPOz5lPDs1ZTcuXbOL/CYYwkIltfIgZo/XNu3Jky27evm6FRDIEH1+FU3co49b9FU3OHBgZ
jmvtovPQGHU9m8fp1PA/ynS9/6kx0cDVJ1Rj7P7OP17xlHPPgrQQf4RpFw1qPZPKtbA7CoROIsnqoC4Nq
GtnezboMMA8J1hIGazSxd86+864DhCK24fHPoMkPby398u4RAVGZQcAxiF2x7BO6M4WEeWjEoo2na
ws25XVhZ8J7orl8572ZkOHrivu1e20Gn0QTNCIxw2aW05cmKljWbHteUAMGmHlHq5imdfsuLsOSth5hX
2SWcBDuhVdpOqNivTKRiMwRc19Zi0NDwzqR0jBOhh3woIyGQzYqgHPSJeGwwPMDZthGiG9iQQxJZ96
7X3bP+oZA/X1i2hVQL1ms0uXrFgQqmNKi4HTz/Y8YSRC+eWcbqaaMtNd64uX5qxS5cK1uQ3mlhQ4Tg5
pNIashLnVi/QppTNzaW94onKs2suP6m28LmmV330TsdSUFV//aiOjwaoOj4M1D/q0HdOvzfCjGpbxImG
GdowGrbGoGdP9rZMBWJVW9PGEb4YY0BS1uyMbeug7FsatvBCg3RsDFBV99asheZJRBi4Idq31oQKwE
NxmlKFOBpp2a6sFu3FuaTN5aftAJAp0F/XRALmdQy4NTulpcxaYbm5rqJicFa88UAs4/tQqcEJ2pdCI1YPt
wD+ll3/2KcBlICO94xHr0x4AeD9dz+wGzdu8VoaXNw0BJeVZm5AfbTwDuqAyV5amjNt5FvEW9fnQ8Va
U0HrQSkinCcOKLT0o17VRsZbWI81u33ngVX7I4sWp2xpVXv3D6yM9tZOKnLMtI2Q9vEP0j+RaM8WFib
pf8UkfVw5tB7CL4HTtqEBFUTOzdiV5VkraIoXQc0pfY977WDpNB1L937XuEpgf1THRwfUP8GhBGE5NAo
qy2pqA64ADpHAs1XCwehgZuEGQQY+hi3SOv4xJrDb4TvHdXu8dWDRVMGi8LYuTpSmZ8edls+8tNC0n
VoNcOApp2J27tKyPff8czaF9oorh7NR8kRpbZ7gS1YA4hAeqSrO29v7LizKHHpy757NzC/Z0oWryAhaDN
7bYmDz6QSORhigbtr25hO79PwnASHA0q7UjKiSjhM4M3fu3LXrV2+aagFoii7Ie0qcxkS483K4t4kTFIMTZ
zyclMnlsBgLANsAZgqvf9rmcjhj8FY5PdOAUml/KHa7/3DN3r59x+49WXdeTkdYrrgE/1V8VcktMV9dINIx
HNXQosde8/T6uQXLSekjwKh0lEMCJzRkO8cV32D5wlTWLp0753W+lNfguDxFicDqT3n/jLf+KI4/U6BqYZ

6Wl8jE6p7VkHqza+/dv2t1n8phYOGpmhAI0YFaHSCNqnX2pXLb1ndxhACYxdMO6sPjGloN0A+71izvesb
QC9fP2+ULS5bDI9b0oZZHV/GS454h3wKsSuaO+rp1lKqpeAN2n7fQ0OGwnWyvI0gRKy6sWldLstHqFUy
+HKws/LV6uGt72tTs6kvWhnj2+9pEV2vAQliJkW1vTLigJ35jciEWVquULYPwiINvrD1wb15xzxKcWNUPV
Y1FFQylbQsIxMr8rE93zhULp8u7J8UtRBfi8QB8c9O++s237O76Ntw7aQG0tzaF0GJFbcSmtLzRoGbD9qF
NQTuunpuzhPYjRaNq9W+Az/sWtZNqw7gluzpftAsr59DKivuiSBics3Kbmub1QwD+EQL1I5qZ+pMd8h6V
LM2zM2viiwE1R64aSr5+CnDqe2E6atJZmN04BB+PWhnyio1qkwktPtN+q6WyuCbOwaV5++ynX7CXX7x
i0ZCWhcAZ5YyhDVVlsNwL2cONPfvWm+9bTx45zlkVjzyayPpuKxaSQOA0oeprmF+l5SnGK/6q5TNaoiLP
uAnP8+/hyWtWqovzp8+0B4DCXlrtgNFEG2Id4N/VFnSlgZOG5VC4WKtHtWOd0vwkZPEknn12Gg2CqyIt
Pwhgmjv25pvveE2BlFdeGft0sFaH1nASlSC9srxk51fPeYhNTpg4pKZLfe0+z9sAvwAKX3zhps0DeLVnRHu
856EVKu2jPl+anbZLy4u8Dk2cJfSYCmVoRkqOr8bp7PGjROqfqUb9tj1xELoVwokw29zZs320To+OUJ2oSc
UTuBIDJsdgbg7Tlc/aPhr0N37vW/ZoYweNOvBsqgvnrtgXPv8xe/HmvAOiWtqj09EoaELF759sHOCAoVW
HXBPtqXVLijpcunLNqYMW9A3hrPKs++LG4yaOTMXy07NQkqpv2yiHp6VM/TYmXCWHGMRMCm2JsCg
/QAXbgrTbl34DOA/4w21VmC08m4ciDC0OF1YpnSgmOpPO85uozwiFogoFhU0bXqQTKjocsRMcSsVm8
1AALcVJ8r628FldWbJ8lGuH4fR0nDbG6FsE4Xvbbj/a8AQe0GeJVAanLGdf/swtyyvnFX6sAL+qdSusp+lST
Q3X6V/tHXUJuiHuqpk1gVPLo5OaNfswVPTU11v9aI4/Y6B+yI6cch85klptqX3nvbAXAxag87X5gpIrlBCh
Wv6avtOy4vXtI/v1//Dbtra1Zysr5+1nfvxLcNoBZh5tNVB5yrhP027sVm3nsG3lcs9i8FpN1xbhewKneOTi0
iJ8toc51RIMLfSr28n+rqWDLUDWtLaSDBASLbaLK7upAA/kLYWgYpjfLL+JAJgoalLTi/LmUWmMbdC1eEfl
YWjze7u7pjLCKi+kWZ8oXrVWeWv7Hgsl4MyAhnvV3rDJWNgjDo8ePbTPf+7Turxp9xZttttoVj2DPx8P2r
m5aVtYWvK4bkp1sBDMuwD19Xffsw9wuDRT9+UvfNq+8sXnbXvv0BPFVdBCQNXGHMpLUJ5FOp1zjZr
FoT0DpbSqnmn/g4kaPT38zcnTH8XxZw5UGaZJPI5mQOiVVxrGSTmqNOhUOBf8FeuI1hvb3HTesrmkz6
0rCyuDU6E0wN/5gzdxLob2iU+8YlOZuJVKx4Apatv7O7a2sQ1A6zYGAK1B2uZXr3iuQGjcYsAHaNuYra0/
YlzGXv1jd3fHyjhyak8qgSmcTtni3KzNzy/YzNwcpjnlVGNuYYZraIMIVbVuWiGiOqXKR4VG8Ff35tXtMOvi
p/L6ff4smbE+XojKDe0D2kOcpL2DQzs+qvieptBvS6aylkniYGnz3haaFWcxA7j6XEx8Pc/zONrsBH5cgfpo
WvXKpYu0cwbBkSDFfKp3iNR/8/U3XUB+9i9/0eZSQdvbLSGMYatr5SzUShVltLRG66imuf85HCglsWhE/
OG6hLGRiy+kfPj1/z+ZfkF1EqHjpgGqbwqBvfFMp4MjK2GatCtgJKgs+bDFYwEeeNfwq+3dfds5KFsEXjczf
87C8Ejt8FHDVGt9ljKNtCzl9bduWzI/a1OLfEe5rGglJQorHW8K7Sztsre77bWdxD8Uk1xaXPCQ0XxR+Zya
+ZnUEFA7xZW1zl9RC5lVrY1Sksp42PGx84QNTLRqg6oqjHJIYyF4KL8NoQWx7e5saf8rxScVBlMtfhVXW4
OafHDnnms6RQdKWJe5pXNWqbV9Q2A5U9qwWBq3enxoMYCfwGHSbNS437BzC7PcR9GK3JeW8+wf
HlkTgZkC9ItTGVqgjXeT8G6+z72qvPlQph2ATxemTPtxaY8Wjc13gKonp6D8NloA6o8wPvUUaNQzoGJuN
F3DfzLH2ixMOzgfoh1HgRiaIu9ToIN+HVNrXpl6/7hsaRyPVG4GZypsJfjjMXxTSzRKADgajNvF85e9JM/O0
SFaKaO6QB5yEf98sramfBBbQBNl8KQvX1hFq0zZ/EzBea3S+7QzspKHlYSicIyGS7mgMtXinALszu6Br0jQk
hnfeofRVfa7Fsep/M7s9DzgDnmJdq04iITEX+Gk0BoVqwAtk1WfAFBTrc6TR317/GjN3vrggVVaA3u0tuP
7kWrSYCZf9DqkbcCcX1ixSH7Khq2adav7loBYzCFcxXzKcoUsFKVoe/v7aP1DW8UqFACjNlkWxgZYFCWh
qD3TfFflJGkU9zqZmnEYngFVd66b/zZaeCGH4kd0/JlrVIFg4vPzoCVyZKRntYGYMtRVxKtUR1PB39LpqN
Wqx/beB2/jeAxsZnbBYnjp8VgODdr06iePaipbEwMUOevCC5s17SYHDdjZAA/aE35kjXrZ2pUSTtm8nTt3
Dm951VRgIZ8DyAr2A2Kg7JpwBJJbnaZ757Wq0gfH1mjg8St5BtNaazTtzoNHduXWJzw9cQDh9PAUjpL
WPO2h9bWOSbvnpRJJW8qoIskIDd3xsuLS6MqiUhkg7X+th8qqK2CpzX2VyK1A/uP1HRzBXc8F6La1/W
QHAZu3Jk7iGKqgtfzSqIkAPDvIfVf2bWVpxm5cv+R9pWJwff1OTlIcOrO46DNyfbRtIZuwIr8XG9XMnVItd
Tg71fIM/XWQ6vkZcnn+I9Sof6bhKd39t0GqB+bFQ1bSXlJf0ABFWFWqpwdwagDo0ca6NdEE+fkliyRyfKb
vKYmiY5Vqw45LPbRvlgEPw0lHVsZ8Vpo1QK/l2Sd2ePDEFgtJ++Sti/apl2/Z9UvnoRSqRhKyIHxYkQBtGKE
NgncPS/Zkr2L3Hh/Y/knT3r33xFpDLY2JWEUlXCNpq3UDdlTt2gLcdww9CSFQY7SeNh8b4snvH9VseuGct
XsBU8x/YB3rYubfeO+R1XsRq7YjtrFdQxgnU8DuqOHQqBCGMsc8DbHXtovzBbu6BFfOpqAR2lJc5SfHXj
QjE4tYF4dREyJh2tSG0yfQzkqYyWVC0CZ4rhbjcX4VlJBz5zUVwNxUJmc5zH2U/tFy8BDOm6aeGQXGhIe
8Wx8e/U/P+esA5e8ZHfgRHH/GGvWPP1RaUjmjbczR+tG+ra1teALH6qVLnjqnTb4SdL4BSC2KC0WT9juv
P8QRwwzCDZXoIu10cryHqa/Z+fm0XVqes5dvXra5bAYOyIiG0VphgMdAr61t2gO04/HRkZ0/v2wpBrHZZ
aDxwjVgKruYhxosryyb6vqrlLlyALRb88Xz1zyWK/nS9KcmF1Sxentrx67fuIlDiLMlTTU8gRtXoR27du3KcxY
eR92Rm8oncdJOGPs2nLLsVVOW8eRVYlwLYyLCBYKrghT9oSYexvbmu/fs97/2DYvGM2hK86XZCbRrGN
C9cOOcXTxXtFSkZZWjTQdXKJa1JBZohON5dHDitQpeffF5+lDFLTAlysuVwlSfOjqfnkOi8fQe0q5oOg9awz

0zmEIFtlEocK5jQBHw4HsND1rVApXUe/XCvOVxumpH275BcKO0Z+XDbbt5ZdV+7LOf9rVUKQazOVAuQ
dROWkN7/96G/atf/217vH1kM8sXrA/Iw2hE1Z5XMF81RX3RIWOngLqSolUkTRMBzVYdEKrWFA4Rf42/
mhkLjHu0Z+DZ98Ney6szazJAOamd3hCznobK5NFyqmc6tDKAT+Ddq57U7XuP0eAtu/P40D54uGdbh03
X3Eo+0VTxMfSnN2h7Qd2/8nM/bvNzmn3qcu2mHR+uWzJiduXCOZtRZhTUIRnJWBInstdRXQHaRZ9OF
3M2M10E1AD42yjgBiVpT+HxVGvUs0NCLhO+d7jvRSFa/QEm3+BaSbijAvOTtLQ4GrUwM29PtvfsV//tv7
d9vH4Vuvjyj/+4L5Bbmp2yTCJo25sHtnfctYNSxY75jub1tQRE2/fMzkwBuK598N67E6Hg9yrK0IMOKMlb
HrrqTmmrca8TRetqtbrNTs15yCkGSqRRtZeB9ifVrI40Xbcz8hqpYzRXrdlGO4bgmCvW76qgg2bBWg7uw
+NtzHXKVi+/YEfQjRpOUJo2T+dip1OqaNjI2DPCVHboHIDU3qbvvnvb7t6+C1DL9jM/9Z/Z5z77gvXg9qM
h51X1FvjCYaXi2WEZNPD89IwVVcqdfowi+cGQNKqiGrwBZXHQPkXHnzFH/ZMdZ1XnpAm0MlLFurTMY
8LrA4CwADDwmNF6uUwCbZEBIHA7RuHjL7/oCcQKOWmx3/Hhgb32lspPPrR4togWK8DJYr4ERMtBatA
JVRXU5gsas3A8ZBV4oCpEaxqxD6C6OFfddt29e81KaQ8m7a/UqpdMW9z0e020vub7aTPjrRzW6glaEGH
TUnHxaVkCTbm2cGZUU1Wxz+OTI59Svnrtuh1XenBNgFmc9nVce7u7dvvuXV9hkIRv6t5UD1aJNSqGcen
CqqWTvA9NuHxx1fLpjK/uVcqfqbYOVkEbCtOZvq1ODkFKKWEFbaodsyep0DiyaHxp/acNqM+GRhXBp9
+0N/4+Jv8IR2cMr6SLPayj1amTmWkteBvbHhzzsFS3TGHG9wTVTJeOhw8f+8xOCFP4eLeJeU96QbJAKG
JHh4cIAmcAVDFAoLnwcknZWdpuR0UxApbNZbwkTiGXM22UphUAEakjnLlcWgUsGGyAqtCU5soVKfBS
PnBhJU9LYe2Xy54Mo9r6KtUjWiFt7aEsNPni4hz3GoB+FABqyJLJOOeO28MH7/E3agszKgPfce2reOk07fc
iZ4rJArBmtQzV7MN9w7ynQhNznvQSiAat1q5ZG2BPw89nAHD8zBnycJS0Kc3nHJzJnz9Nx7MBVEycTLtK
qZebTcB65DM4AQZXwXeV+YlH8XgZsEdr97yQ2NT0PIPN52jJrW1M/VEJZ2eM2dYms0k0Zc7eff+uOzi5f
B5NqgHuWN8TS8qWxmNWwbRMIWvziws2Nz9rWouv7hKni+KAaKpVkwFjwKh1Tr5RGgOuImialmxyL
s3bS5CUHyqNrQokipMKIvt7FV82vXuwb4cHx2hW1R7t2cHBkaULOHO5WahE0jX6cNi3i5fOe/K3Skmqr
fLUFbHQ+qg43paKYsz5khLztVLKy1VC+uLiqq2cW+D66quxTaO9tRmyvHxf8sx7yBo6Va3yFVre70/T8W
wAVRpNs0IR/F8Aulc6sVqn41WlNfgKtdThiRr0SlVThHGbY3BUTeWt9+7YSa0Nd1323FVtZlvHcRnhnKkSt
vZ4UgnGva1tTyR+6eZVO7c06+lw8/MzFkZrqrSN6i91+pj8TstUIyvOAEtzujMCGNutSd0m1dR38ALYWBx
UIUjKgIJau1BJw6pmlRcxRgNH45panfDtErx3f/fA3n7nfds9KnsMNYYJH4wx2/DK5XMrrjW1b9cYkHWa
WBat9cJhm51CoIpJyyfRrgBPe/13Oor5dq1cbnpcd/X8vC3OTNtiln7DSrmXT0ulBTTdqnZoeY3mpp6245k
AKqoKnoXXzQAYmqsOcEuArKPwFaqghXOiuqIqcDuzvIgmU0Y/IMK0P1jfsaNyyxL5OasycHVMbhCAdEs
7Pn2p+lCajVHhsFvXLlkOTZoMq2iwzHfEqmi4Ng9tjquarGl43gBOKd6pXVcUGVVhsEAwjTJCFzHGw6EKq
8GRz2ZuNDWsYDmAVSJJB7BL+8rgFooFrxWgOv/7+yeev6AthLRY8L0P7tv9NSWxhG2/1LJsfh6qCZ/G+c
pACTRHn0zo2l3LJwJ2/cKcTadV63QA/xWXVuGJWRsMQ2jpfUvHw3bt/KrN51OGeNMuPehb+kFAHXilU
+77KeOnOp4NoDr3k7nDVMkLRwtW2y3bPDiwk5My/K8LiJKWTONpo0nCfEfz/nVtsR5N2p0HG7Z5WLF
eMGkqunZ8uGfd403GqWcX8ZpffOGWXcUZmULT1EtHvs4/Fhr5as8h/NW3xeH6vhmDBpe/IpxyRJIAL5
FMWKVBG5VdBb/V7FQXMGsVqtZZQa35vmwrYMCh0raQcYDWxClTxpNWDKi92uF5BKBVjGJqJgsV6Fu
jH0bY9uzXf+OrnAK+HJjsFjg7O+NTsq1mFQ8+iQM1Z5nYwEIDnKvQwBNOxuOINdtDtHHW+XcGTbw6N
20JBD44nCSi+GS+QIqE8WvaGwDEf6FR/3QHg0mvo38m81jyrGRcHz7ZtIebTyySTrtT1KejtZ6q32kwYJN
EC21ucVJp2m9/7TVbP66hIYN2gjNWLMTsJ77wGXsOgIb6qn2KfhuiIblOX84M55epVoUQmVttgOFb33h
zBFrNntEi/tIgfyj1UFEJAUDZ/fqNDtGLyfeD0I26vyfKIE0sniuwKlYsp9Gpgbgjml2Z99o/SrRB+bSP1jbsm2+
8jYJGWLleENoQgrb87Jc/53P7Nuqg6eu+/EaOnlYBtDsDrhW3peKUreSSXFnZarTJc0k/PPRyRb13T///dB3
PjumXRlIP6q8OrOpJrWHruzteCyCKVusrTRANFMTxiONwqG6UTK3yMW8/3LT/6z/7V1bD/L786qt27tys
zeSzNoV3rinKMOZaGlLKbwCo+ppo4CLioponV9hI3rgPKO3Rc4Fv8h7fxLHTQ2WLtDpToFPXqnddc3EIqC
o+JlDqEJjPDj3X9wVS/aaJY6VichEcRS3NDqNNQZ6pGuHXXn/LtrEmyvy/fn7Ffu7LX4Anj31pudb3a3JCwu
Eamv4SYBfw8hdSmHZvNyA/FaJn5XgmgHo2sDK9kyQJDgZfgf/9UsmO8Ni1J5Q8/MAogrMRtmwsjGkUe
HBE4KpDHLFf+81vWnMYtBvPP2/ZDBRg/8C0fViO7wYAK9j0gVR0YSC+ySOOVnN48t4ElBNQaaAnYBVw
Ze6luZQ7CwfmM33nbAPgs0Pvqbyk/qp6iUCtv2ff0Xn0mY5OD46JwAVOKYPH4uHnqVzBtBB2e3/f3r1z2
z798kv24tXzOIQ9tyStdt3jvb6DNbpTBTDiUIUinn4xzg3ShxrxvwDqR3Ao+QI3dsKcXLPKtKL5eLuOQ7GpZ
Gf4pKo6R4Ixr7+kukqKqQrkTThsuYXDVenYGNDGUxn5D1Y7ObZ0JOSZR9poQstHtH1lQBnuAijAcCDqun
TTGaAEJj0/e6gLXZMK4ABEfwVABefPDn1H7wmoeq6Hvqu/quTyXefm79C0ecXAS+v0cba01LoPWItz85
5aKCGt4TRWj4/w9FO2srpg08W87xatCt6KZGh/0lw2b7lU2hK8l5IgPqPHMwJURkb14qVN1VwHB8/5qz
rxG5hBlSxPZbJesVqJxSo2q+Ug2sdp/+gYMwmQoinfI1S5rtqHPwug49IsACLENVSBWuXKw2jYkE+Pah5e

GbMTTXcGpjMteKaV9FyzWdKoHwaqfqPu1XMdHwa43j/7vW/+dvrcDz5r9+q+Q6GXJdftw79VBlM7qA
RxlCRIfdRso1qf7OoSi/ouLuLN80r/q4sG9GwmV7TpfNqinOZ7car2fNd1n+Lj2TD96BeMpWuaoDQqTg6j
zQtVwzN0j9lxWSXTW5NlGjgiZTSLtIuAI2BEMH9RHCtt9S3YqEy5uKSC2xHOo53pVD1Q1aeVtBxLaclxwp
eCuEbnOAOeAClw6ZCp10PaU4Ouh74jLSlh0CEQNxoNS+P0nZl7ff/sc73WOXWcDUcwJC3a8rqkg87Qt9
CJxCbLWtrck+47yHlFf1Uqcn3jiddbVXmjK5cvmdQvutsuLc6a9tFSG9MZOVPP5vFsiNPE+DpQJ/+bDKiC3j
pknJXYoX1LawxWTTviDQCVeGYk6ss15I1rbl1z76pErdkdLcNwhcJoa7aopUkEBj+ZzrggCFxKLBH4fLoUcJ
4B7UxjSgj0ud4XGLSDytnfbwvJqcbU+2dA1O/1XJ/rcXZMnk8e7p/jIMob96iBHrw1yXjiurS/jebstBs+9fr8c
885l1YhtJPSiWm7TZ1JVU6UGP4sH88IUNXdGmAeGmgGSLmqqrCsmKSGI43jomp8SvIoaU9SgKLq0VpG
rFqnSshQGUgVVVD8NBoVgGRBAQGDHuBFOJGyNp52lYHXyoKwBjiVNO2xVC6XHXzShAKTgKmHDgFRo
BQQz97Xcx367hkH1XcETgFc3zkzvS50p++fHQKpkrj18NoGgE47nGg3FW31GBE14ftalSCnUTUGtA6rkJ/
ybtLmaYVszon8GI6r+lHP8vFMAHUC0bOOFlDF79AoChnxWjDWmiZ52dKIWjevzP+xABKKWiKT8fpS2r1
PWVgqn6Nx1ny84pjGAKtEu+bhlWidzKSlyNDONcxzm2uFXGMLcAKbAHWmVXXIbAuIZw+BT4dMvjScQ
Kvf6fdq7NlvNYV6BlQJ3hlgZc5HAqlyFbAKPuMVwrHjr3iqEnG07yknNJXD1L1r5z2t2aIBNgtHXVpYsESM+
+caYxwrX5P9DB/PhkYVNvkDoT59LS2IGQRgugXdhPY70vbdSm9TkF6jrYEdMnCimEoaiTDI2u1OZxNoV
Ms+ILDrBPzVUmflnvoivYkHw/fw5KEY0qaKICjLCTQBLJgzr3UojCXgqKRjGE6pc0vzabJB7db2N9olUAkwa
qOWmAiYOo+ArbvSHrCKveqM2sxCkw2apaKBugAPAXbSXs1iaaMJ3bOWfkuIlVfg1+n0rFjIWzqp4hf8HJ
oQQMgc/X6lZ/PQED31h7z90JDBGukhk8dDEX9/LhXFGDIGSUCwhGleyWctx6DGgyNLY76V+xmLJuGkK
cvGZyw4VLG1qA36gITfO/dTHBato80VHDwARIFylblxzQmoNMHomSf8LhVP8UiDnRjgwssHPEMBnZY5l
VDQHa99AG0IDlX9L8m5R5YCcGPeU06oTPVYPDkKB4klLASwVbBN2643oC9yhqQR43xG6/zcysXl7JZNZ
7m3lKm8ZpN+0dSqtH4B/j2fS3sNKR3a6AMVLPPBK4H12TyeCa/fwaFhQoPItE06XM/5c9Z6PedQfVI5VC
fahVn5AYBIpdRVmW/cG1sGHtofdDCpXT+XFtRpD6hWp+tBcmVESXkp4K48VPBmxdyUZ8Z3OGcuk4U
H5mxj54DmjJ1WHBzjuPgGuFqO0jXVgkriofeabcvCE7WsOp9Po9WN93QfA3hw24bQkFK95sKQyqEFAd
mANh4fHfOdoWvFLu2Shla0YqZY5DYDXEcrSScldjQpUG02PVFFgqZy7fNTeYsqHkx/hFHR6jbl8zwbaun7
H89GeOo04D95oe6XSTxF5ocOGdGRDUx73Wsf/hrg0n5SKgAW0Hp5Ja9E4vDOJua87iBScbL+QFwxYm
ktOUbDVeGm2sjipFLyin7afSSfzqEVo1Yr1bi8uK08sdAkwsBVpVlTAG2ozCjAqvqpScAk7/tYBYrRkqqqsjx
bMG08ce35q3ZQbsCl0ZAQ5lan71lfCRUt4E60qa42Lm4gdOKxSr7WGjE9crmca/1yuYrJT0AHcAxFDxBm
7aqnrDBFN7gVj53+BVB/RIeaOGFvOiYA9Xl2/n534zGVaCJBuoV2lKbRGic5JRFMapfB1roqFbctZCdLpNO
apeJEWsqsFaV1QJrJp6xSLaNJc76ZhMa3WW3a5tqa3b1z31PmtF/V7OKSffxTn7KPvfLiZGkMQtECuHsb
G7a7vWlvvfmWvf/+bYBatsNS1dfLrywvW5t2/eLf+B/ZjRdetOWLFwAxgMVqAEOfRfIdtgFnPING5bmoiT
Yy1sFTd97EafW+4sFZnD9RhSy0JwG4RUclxwKqnjtG/wKoH/0hkE7C/WeHgjfilbgRGhDuwHkl2kkv9Kng
Wms0MJlty2TQQPo9X86g9ZS/WtnftQcPHti779/xcpXzeMk/9uNfsinMq2Z4FPTf39uxIx53Prhtf/A7v2d3
3vsAXhn1wg9ReONRpYwz07HPf/EL9tILL9jBzo69/dZr/lehsjZaNJbQWnucLG3Y2x962SDVrrr78IHNLi/Z3
/jbf8u++GNftKOjE9/8QppRS1NUWS+RSeDBz/oKBBWOUEkhX3YC4AVgLS4UHZnBeVpf27Tz5xbdCVPo
SnFWyba4rXruLDHmWT2eGaBOQvs0mH9SDCNVKkGrKKbou+Hxzxf04cmLC4aCMqFmb7z+mm+o4DX
5cSj29vfsP/zbX7f7775rhyp1U1bV6YSlAfPFCxftK3/tKzY1nbdvfOtrdu/2batXTmx3cwvHKWHF/JRXXrlw4
TKACZv2tj8pV+CzXJdeVCxTcCjizGRTCbRz06MBmoZVCSBFAi5duGR7aGRlPoXiMZ8GnZ6dtwpmvMa54l
Gt0RpZXRQCcM0vLdjMzLR9CTAvrSwhfiPuI2j1ZsOdtiguV6NSg9ee2E/9Z3+Zew95WEx8WUDuqqw5bd
XU8LMM1mcEqBOPV4drB14PGAjPBR2qEnXEeq2WRTHTyhXd3dy2b37t6z7XPY02Un3V3b0De+sdTPH
tOxaC804llceOZ4+G1Jomrb/yAr04J5W6ttNp+1SsalGJp2rpSrvR8uTt559/CfN/6HWxogBydeW8a7lWre
b1RhWz1G8PS0do5gDtD3pcVMUjlhbn7Rjnq6GS7YUpLwmkDXIHWs8FF04nVb+0Z61T8z/F9dvdDt/p2s
Lygq2cX7Esjpm8+RiAbNeatnb/sb3w0sv2C7/4i/QOw4mwthraklNxAzn8AajFiDaehgKeweMZcaYAKwiVi
df/aDQP9Ky8dHjm2p07Nhh07c2337F//Ev/1LqY3Cqe88L0rC8dmdbWkGi7TXhjQiswMe/9agWtq9hq3
B0jrb0qzk22ZSxVjmxxYdaOAFoOh2t6esa69aadHJ7gqITs3Hnt3dQBTF0EaGx5AHcE+LRyVd66O1IIgla2q
nhaEo1egyIc8PrqpQumBXxN2lMoTnn5ICVLyyoUUjnnqNooTZumSTjCcRXiTdrB8b6DNcp5G5064Ff5yY
L1ml072Dxwp6w4qzqpKwhaw/7G3/pb9slPf8a1uDYKVlhLceNn9Xg2NKr8qFOrpS3PXWtgxlrlkv3zX/5I+
7V/86u+/U4FR2Z6YckurZ5n8LZsdmraqiclq8IlNRXaHQ185xFx2ebxMRpJy50TFlbBBQCYAcTaZbrVwjHJZ
fDYjzzxI5POWkCZ95hRLT3JZvLwSFgw5+kKmMm0lUtlzisHx3Bownjtk2TrOgDt0H5pyDIc9ObF8zR/bBXa
m85loSsD63NubQKs0FkW566PAzjQ3v/8E5VRFcKK9kY93LMAoIulYpbJpaArGUx/GDoCn00koBNtgJ2w

dZy57jho/wM07H/5X/9t6IWSZHiIEz2jx1MJVDVJztHZofDU5JX+DwnAjA4ZlP/L/+Ef2T/9pV+yZRwUaTot
jDtGm2QZ8CjniMALj3Z3LYZZTmZSDLLmzwO2ublpc9m8h3Zi8TTXErDivtfp0dEhTk8LrdeyItywz4CXShV
NL+AgpT3bShpYmlAy0wWsiXR64rzwfIwTJsBK8+vfEd66QmAxOLAEbmk659n4B9CGWCzJebqWTmcA
WxNhGFkynuQaIetWW5ZLZD2um0gnaPfYSrUT2jOw7rCDk6ZivQHrd0c2W5zlekFroIm9IjbtURUY7Vv1
d//u37PPfP4zAJk7UPfRZpqj5nn7PDxwCoEP9/nTdjyVlVI0lz44mxvn6Kqmk0rPjft0Jt74xkP7lf/HP7Z/+69
+xRex5XM5H6gIwOvA9yarPNF4mgdHI/X5nRIyW3DHoxraFdCCN7SPnBkGW4iDx7ZUNA3No/jlKIhRZzR
1Tm2AO+hyjv5Iku2aToUjpIUFfjkw407b+s26C4zqVUm4lOCsek9RJXNDMYJw4OCQ8wJIJTVHFYtFm05q
aH0n+z8UFSUJQTmy3A/8FOHUPcSxCmqjIg2aWtVyavFszf03VL4dZ7KH1RCnVUog7qPdfv821wzaIg5bP
IWDSaOUzKPpZ0+CQcuKNmj6Vu0QcJ9GuD51GlXTlcosUrM0cDSRDp0Umw3w2f7Gmv2f/vf/wF77g6/Z
0sIiOiGEVszAQQc2ZoAFnKq8Z36XT6VM653kjTNW1mIgjc/7cMsUmicZw3EZng22toJEq3nAUb51h8Eb+
w4pSl4ewxnl5ChWmsP061wy55FkAsqQtUGtAgD7lkcLiyN2VC4S3qsSj9JecVXIA7yyvqIECi9prX2Le9Xm
Yz3lwMbjUAocMeUI8PlMoWDa0r0PwLsIoPZB1c7WvllFIumaOqn8AlqritUj+kghLIXjNBkQDSetdKyQV8
SuXbtk//nf/Ot29cXnASr3kYjSPsBODw5Op4iVk3sWn37ajqcOqEr+8KQOeeF0oJI3NOOkJdAffPX37J/8t/
8323p4383nVG6K7wDVcMJ36NMqIdVwkjZVzS/tWMcJXPtIU6pk5TiMBkHjeAkcBjAYUAL1ZMmJZn8iaF
4py+Gw7dOTA4QlD2BSALlZbXgVQQXZZWI1ozTkPGJ+Rbx/bd+o7THFW5PQCBW6kLetjRpCgF26SvWx
NG2kqVHtUqhaBAKowKId+wRgLaUe8n4RAdDrAdpWhSqUetjW0mqea15f2i/U5/6Vy6ASKiCshTVR7qli
w6ogMxoAPgRRdQIWz83Zz/78X7Ev/8xPWxAnzVe/nv6TNj3TpWfZX0/T8dQB9aw5yn7XMxU/03H7W
9+y//M/+oe+U4k0yPzMrPPK/b0DBh8QYEZVjU+BeuWPdhoti6GptGe9Blvo0wa/Aoe22QkCGGVAhUM
KyGs/pzFaNeJAV4aU9iGVVhwpx1VmHdMfU8ogAOl1B14XX8nW4opptG5Im93yXgoHSVpSoNO0qWd
MAWXtPSqKII2pqTBZDVVc0dr9MFRjANgU/xTN0X2Fufsijp9qrCoeq9UGmq6V0KVTaefaI9qsiighHLKoVt
MiNKpcLQHRxITaVTrWPgXTaH/ai7Aenxzbq5//rP0v/lf/S8sUNJEwKUPpWV3utXIJNO3Tdjy11fw0kIoTn
pyc2C/94//W/sH/9n/jWiE/PW0jOlI79cH4LMQAprSjHdpPS4zDDL5crwaaWdv/BF3DYd74jdbrjwHOAM
XRBtBNTPWQ19r2uwHQhvwNqAwP2qaHxm4qvgn7UGjHi4cpvIO5FkC17n8EWGT6NfvU7jR9KnQEWF
pKZjn9q9pRXuJdoTSA2up30Oo0gM8VKgtJMBAQZfSp7UozFMh7AI6zWBuacqLpU76vZPAEgptMa78AP
lMsWYIXGHEv4qcC/WSj4QhtUvvq3bYFeK9Urdh0sehbXP7Gb/6mbWxt2fLKii34BhsCJ8KO5vel4FzraTu
eWo3qTgWS/uabb9o/+b//Y/d+tV2i5rRVEVq8TrxKWlA7S/sOJfLcAbe0U7lSZtBwMjSFimbxTXilvXCaZNn
GOGVaCCjzqfPon+pBeaEJrqtd7+SMKVwvra4Nw7YYXM+D5Z9qRWXwrrXZmZy3JJShq/gnbVfxCEUQO
mhPaXvx2nAg7EtlevBteevKT1VbtAO19qqS0dVrOTraT6uOY9YDpNosQrzUw09NrATt0B6nyuzWTFObd
kqAFMPttuqenyAOqsmOAY6i+4lw8X6zY3EsxfTsrO0c7NtR5cReefVV+3v/87/n1kl5A/qynCoJzNN2PHV
AlSYVPz0D6vramn399W9YfQA/PKnaISBVbFLmugMwuAEGposZRDvy2yFetjSC4p+5qSmfM1eWP/3vml
GmUg6EPGBVCdRGDF6ZhGtrmnHMd/gKf9HWvKd1/8pkyuULdlQquWZUVpT47fHhkU0XZvjywLrNGrQ
h6nFM1RjQ97TWXoKgbXOVZqispkqz6hpUh0xurVoFqAkUNZ87L+dcWAt5+u1uC0ct50umE6pN4Dch2
o0Q0UhNI4/gsy00qahMAIDGEBWVi1f/afMLrXlYlSRIjUXhs4WZoluENtZHM1//+S/+ov3MT/20z7gJCM6
kZTmesuOpA6r4pQZMIFVna0apVC3Zb/z+b9jXvvG6F7VVoQmluonHSktozXqnXfeUuGwiPdFkgEGF1E4
qNcwmQFFiCPxuIA0IgLJTRQYbA45m9S13GBtt9CBHTGV0RgPAzvXD/Taf9y2H2QQjDlZpt0QUB+6kZNlk
1nc/UdlH7Wni+5RybYW1VLhNuQiJWMJUSEJFzcJQiwHnq9dq7sVrAkHtkuOoFafKitLSlWAMDc19+MZ
mp/FVbd6m2SrNYgmoAvWAa0mrDqAdq9AipXHLqkjrinEOAaPWjY2gCTHeUPxXG2ko1KWdrP/qz/+8X
b14yUNw4qn6p0mGp+146oCqQZIHfrZgbmKatad8077+2mv2m7/7VUsXZu0zn/u0fQHTJU045nPtQTVZ
3qESNjEGb2T//Nd+1V57+2278fyLkwFQzjyDPOgBinTeARMIaQ3UpDp1EKAIjYOhFuLxfQY2l5T5RL/BLw
VWhaR8Z5FG24J8V3FRgXY0RnMHASmg4Odorcm27FoyIrBqmbciAJ2BVrVqx7++pSQYwEkBMvFDn4sH
6OoDPDe49OR+9AjwvvaYamNNokqw4TpaAtN3UxCyB++8bT/3E1+2L37iFRsDOhFPLcPR5wMEQZMQ2
v7ytTdes9/+vd+xn/zLP2XP31K/6JsDiweipu072ziOshhP2/HUOVMKjUi7CKx6qGqz7HYSvjk/M2e91sBu3
7lrlVINk5ywxfkZSzBoirFqBipyyl1VgOzNd99xDbi6vMp5AD2mL6YlG9EkQOFraCVp0MnMF7/BoxkyskoQ
wTdx7aXgv8D51W9+yz549NgOcO5QOXDjOGAeuVbUtuVdzK9KS+IymfYZUHXsMFpQkwnixjK3EgQ5P
mHaHYGXKrNKKwv0uTYO1iKTHua8TVtV/7WNOdZ7Smo5Pi7575TGp+/qdwr8qwiaElMaR2X78U9/2je
REDq9ABuPMCDWbFoiGrLf+urv22/+zm/bz33lK4D0BQQHugBfDip8J0tAG+VM+bqwp+x46ty7M216BlT
f0Q5TpqnGFN7uz33lZ+yTr7xsD+68b//6X/4LO8L8qvp0owvvYtC0ucJQoaIunFFlywFNCycEBQt3xAsGdB
UclWanChAa7l3LXGshnWr5K30v4tn1BpgYZByLf/mrv+6hnnwyY/ffu2Nv/MHXLUO7tJlYDROuvUWT2gs

AzVquNWg3zgj0QeUt8pkpj2fK+RHIJFRzhYLF+dvTDJNmpOC7ITf7KqirhYJxtFrakpGkhUdQIG1IoWgFbfd
VpeKqCvDzTwnbKoSmaIGXZlcfYh18/ZecO4RAhuLB48f273/t1+1nf+Iv2yduokkRiLTCbbRLhXulTRVLVaTl
aTyeOqDqOAOpHprFkZYN49EPGBzVVfryj3/e/vpf/6uYyJb9yv/nV6x6GmiXplNny1eJ879apeQcUJpC3ro
Ar+XHCldpMZEUibiDPtMqAM0A9RlsBd9lDvvjntd3qteblkll7L/6m3/L/vpf/QW7dfW67W9u27/71X9jD9
HuWqZ9sH9oa+sbVsjBEwFaHNCXD47s8YMHDmga4c6eZrje+ebrdu/d95w2FLUcRgADbP7gUEqgSl1Ge
O0TFwMsgSYnZAE4j/pD9EEzWwpbaeJBIS9tGa/7Af1o8J6NEghcMmJfe+Md+xf/4l/b//Rv/1f28RdetCF8
NSnh4PoCKGoBzasJEM4r7/8pPJ7OVn3vQf972IhB6I26tjC/YF/43Oft+edv2O3b79qv/vqvWsrvRHPWm
O+REoaHlsTRUT1/1dGXRhMQxAVl4pX4oRkbhY9EAQaY025bYR3MMBpM65qkuDLpjN24ccN++Z/9M/
tH/+gfuuNz8+ZNHC+0Z6niNEXhsm+99g1r4qT89n/8977rs7T53Q/es9LRvv0BnPBf/4t/jvfesp2dLXfeqp
WK/cf/+Bs+E5dAO7vZpf0T8yuHjleu6SWwAEiUhg6QVtbKWdXekiZVkF5ixVd5TNyNLpQhyDmlge8/fGR
f/dpX7eq1q3addmubd0VMFL6TgD4rxzPRUtW8H1nXpV9Tp/1RBxBG7Od/TvspvWoPH9y1X/qV/6dVm
2W+C/BkxoNxQKTZGs2xC5S6VYFVJhkNwvtBxSL5N8L7GcJL3XEZ6lO0L+AWoFXn/7Of+aT9/F/5ij148ND
+d//wv7EH9+5asZi3wlTWnrt1086fP2fTuaxdv3we7nxoW2uPrdes2XtvvWE//oXP2ac+/jE4toJUQ/vq7/+
uafvIVDJh1VLZKgKMB4YENrS8W3UACiVQiCyg9ooOaKp3LGeP+6Bd8tj0mX4pgA/4YR9nDgXs6/hFZ+6v
r9s//if/nX3yUx+3n/3Zn/G1Wgr/SbiSyeTEWX1GjmcCqJ5HyYj0+h1vsNbgK9STy6XtK1/5OfvYx27ZV7/6u
/buO+/wXcUoARnELIiXrFQ6aSHBD+rGb/kYTSQHQiAdM+Da1UQPL5orR0nVSMJxKEQUE2u+K8kXAdzf
+3t/16YKObv9/nvu5KnyiopJaCInzKNRLdtcEf4pzhqPWLGQtQ/eedNatYp94qUXPN6rx7nVVfvSF75gv/D
X/qpP8QowSgTXwkROSHsF0olG9bbjAGpWKyDvXW0+1ai+Otc1seLBIisTyy9P/z/81m/bP/3lX7af/umftk
998lPcl2qlqkDGpNiwZ/+fauBn4dC4P/VHgEHxqs8+WNKGIUxr3MErQPzsT/20ffnLX7B/9+u/bl/9+jfQKv
q+NCWDJ2+aQVQWk1ab8qbzVtE5gVCcV1X8lMihor/+YFCVX6q/zXrd3nrzTTs5PrIXn3/BlqAdJ6WSz9M
L/MrifwgPffNb3/Dq1gLc/s62b0R2YXXZ6tWqZdNJewHNqyXYmjF6sr5mu3t7cNpH0Icja2vqFRMuD901K
0AS75CAyoJ4ZAK6oEmNEU7SeNgHvJNEaOW/9rrKKQiZtiaqo5F/5V//mv3LX/s39vO/8Av2pc9+Ds0LrU
HItYZK8emzAsOe2PKMHM/Ezn3SGno4T5PZo5MFEqFNAfEE2i+PJlNA/Ld/66seP1y9cMG29/fs4aPHNq
MpQgZViSUCqdL8BkO0G4MvfTRZeozDgnMyceIAMa8lEEry+OD2+7a9sWV3bt+2bqdtX/j8521+dt72AZ
sC94VC3icbjgGzzKoiAbOzs3bn7l17svHEjuCw7733nl27etWK00V7++23bGtzw0Nbq2hXtVsF0DR3L+nTT
JWSX7S3q5wmCZQERyZen0tgkScHqjZkm5ubtSbXVM7p7//+1+2b3/iW/Z2/8z9GOJ7jXlwHcz/6/iTsdOa
oPkvHUxfw/76HTByd7Q86XodPJzqhm3S4gKssp3/w3/wfrVyv2M/8/C/AWRs+SXD9xi0+DwKyAd572prt
5sQB4XwaMJ1R0QJpUEVUdeh9De5kUwaZ4pEdoz21fkne++LcgiUw23fv3LHpKU1LanJijKZNeJ7n++9/4
OD9PE5fu9Oy+/cfWnFqyi6eW7WTctkSaFkBR1ENrfs/W7cl16innnagBox5ykM6GSFpXzx1ovNa7CqVp82
BFHabz0x5l+MwnP23/k//ir5qSyLWkvFkvQ5PyAFy0aHJ/z9rxbABVLZSXoOOUrPRkAhlWbeWoj5QwLO9
4d//l/tn/+/9la1tbXgJ9bmHBcoUCg6+ZoIy16yqhHrWOp/lNNJR6QJpLaXYCrHwqaTjNdLUHXUvFY54Yoh
WhGmatRhXPC/Fdzcsro0mlIaG5/htpza9/7Wv2O7/7u/bySy9z/pHNLyy6yV2en/Z0QO91iK1SDb24Gr/X
gXI3bTIR5fpqk4bn7KE2nR2yBN4x/KBRa3pBjEatZf/FX/9F+/yrL0BP2paKhegf0Z8O14gjCDL1fwHUj+5QC
4VG/T0FqgZqyD+ZZnEvaRsFvEUP1rc24Wi/avcer9nyyioaree/CSkchTetUGEQMy+T7yCQ08I/n77VPwCi
rXjUM8ofVVaWKveJ62o7di1pVg2BPuDPZgA/Jlvpc6p2oowrxWGVM7p/cABQIpYEyKIDypftot3SqYRTB
OUc5HJTgDzseaea21fISdfWnvvy/iWMkzai8dHUarueSzD5nwtLAhDu7x3Z9WvP2d/5r/8m9CZiw87QYlg
Dbc4mmjDgXqOxNL3wF0D96I6zFurvWT/zV00XkDyJBXDJlPd6Ay/icG/tod29f98B0VY2vfimijFI86GNY0
mAiEJ1IHA6AV1FG4QEabJJGmHAtN2OLiYnRAkzAq3SDQXqyZqsqLdB30vAf3v9gZ9Hu+81Wk1vm64pD
ay/Aq6Wxmj1qaf3qV38Vm2Q+Vd7PQoAABWA1/uTIVIbIDi8r0PvKSqQ5Bz6TDNXN67ftIU5aAj3FTkVa
GlcJd9o5msyw/FsHs8GUP9Ex2TgNNBygjR4ky3Th3jE0o4TJ8nvVhtXnHJd/U6D7xqZ7+t3Ewdmcg6dkz/
6mn9POk70gG/4+959/rnONzHPEzBN2uI0QvFbveWHfnfKMT/U9Trv5Dj9jO8p2nH61uT43qHiPMqpVR
M1P+/lLv27377Yn5vjzxFQJwOvx/eu+REIv+vACWNMOb731r/fAP/h95R19b3d5pWrOSZTkJPfqGqLju9
MSwr4eu+7z/kdkH7o0Eza923Pdx9nGvZ7//55O/5cAVXHGVhlPgVQDdz3AhcI8b5u+3tu/fuNsQPmuw+
d77sBwfXgyn+SYzIh8d0XmujQDx9qF+99G+Df/9B9fu+96b0/j2D9c6dRZW7PACp+qOMPaVSA6qGtDx/
0wvcdXwH1Qx+ot777e3oh4fiTTkeeUYP/vuNP8p3JPZ+B8+y8f1gwn/XD7P8Hde/qNYhvYy4AAAAASUV
ORK5CYII=">></figure> Function Description For the arithmetic sequence a_n=a+(n-1)d,
the nth partial sum S_n=(n/2)[2a+(n-1)d] Constraints All the values are

integers Input format A single line input of four numbers separated by a space Output format print the number of seats

answer

```
 #include<bits/stdc++.h>
using namespace std;
void solve(){
   cout<<"class Theater void operator+(Theater t2) t1.get();";
}
int main() {
  int n = 50;
  int a,b,c,d;
  cin>>a>>b>>c>>d;
  float e=b-a;
  float sum = 0;
  for (int i=0;i<n;i++) {
    sum = sum + a;
    a = a + e;
  }
  cout<<sum;
  return 0;
}
```

question

Question description: Harish received a notebook from his friend Sathvigan. This notebook has infinite number of pages. A rule is written on the last page (huh) of this notebook. It says: "Harish have to write names in this notebook during n consecutive days. During the i-th day he have to write exactly ai names.". Harish got scared (of course Harish got scared, who wouldn't get scared if he just receive a notebook which has some strange rule written in it?). Of course, Harish decided to follow this rule. When Harish calmed down, Harish came up with a strategy how he will write names in the notebook. He have calculated that each page of the notebook can contain exactly m names. Harish will start writing names from

the first page. He will write names on the current page as long as the limit on the number of names on this page is not exceeded. When the current page is over, he turn the page. Note that he always turn the page when it ends, it doesn't matter if it is the last day or not. If after some day the current page still can hold at least one name, during the next day he will continue writing the names from the current page. Now Harish is interested in the following question: how many times will he turn the page during each day? Harish is interested in the number of pages he will turn each day from 1 to n. Constraints: 1≤n≤2·10^5 1≤m≤10^9 1≤ai≤10^9 Input Format: The first line of the input contains two integers n, m representing the number of days you will write names in the notebook and the number of names which can be written on each page of the notebook. The second line contains n integers $a1$,$a2$,…,an where ai means the number of names you will write in the notebook during the i-th day. Output Format: Print exactly n integers $t1$,$t2$,…,tn, where ti is the number of times you will turn the page during the i-th day.

answer

```
 #include<iostream>

using namespace std ;

long long m,n,s,t;

class Notebook{

   public:int Pages(int n,int m){

      return 1;

   }

};

int main(){

   for(std::cin>>n>>m;n--;

   cout<<(s+t)/m-s/m<<' ',s+=t)

   cin>>t;

   Notebook Turns;

   Turns.Pages(n,m);

}
```

question

Question description: There is a famous Bus numbered 777 which connects London and Paris. Mullai who lives in London loves counting money. So she wondered what maximum

and minimum sum of money these passengers could have paid for the ride. The bus fare equals one Dollar. However, not everything is that easy — no more than one child can ride for free with each grown-up passenger. That means that a grown-up passenger who rides with his k (k > 0) children, pays overall k rubles: a ticket for himself and (k - 1) tickets for his children. Also, a grown-up can ride without children, in this case he only pays one ruble. Mullai Knows that in London children can't ride in a bus unaccompanied by grown-ups. Help Mullai count the minimum and the maximum sum in Dollar, that all passengers of this bus could have paid in total. Constraints: 0 ≤ n, m ≤ 10⁵ Input Format: The input file consists of a single line containing two space-separated numbers n and m representing the number of the grown-ups and the number of the children in the bus, correspondingly. The numbers a, b and c can coincide. Output Format: If n grown-ups and m children could have ridden in the bus, then print on a single line two space-separated integers representing the minimum and the maximum possible total bus fare, correspondingly.

answer

```
 #include <iostream>

using namespace std;

template <class Bus>

Bus Ride(Bus n,Bus m){

 if(n==0&&m!=0) cout<<"Impossible";

 else cout<<max(n,m)<<" "<<max(n,n-1+m);

 return 1;

}

int main()

{

 int n,m;;

 cin>>n>>m;

 Ride(n,m);

return 0;

}
```

question

Question description: Rohan is interested in space research and he knows that he can find anything in our Galaxy! Now he comes to know that a cubical planet goes round an icosahedral star. Now he introducing to you a system of axes so that the edges of the cubical planet are parallel to the coordinate axes and two opposite vertices lay in the points (0, 0, 0) and (1, 1, 1). Two flies live on the planet. At the moment they are sitting on two different vertices of the cubical planet. Now your task is to determine whether they see each other or not. The flies see each other when the vertices they occupy lie on the same face of the cube. Input Format: The first line contains three space-separated integers (0 or 1) — the coordinates of the first fly. The second line analogously contains the coordinates of the second fly. Output Format: Output YES if the flies see each other. Otherwise, output NO.

answer

```
#include <iostream>

using namespace std;

template <class Universe>

Universe Planet (Universe x1,Universe y1,Universe z1,Universe x2,Universe y2,Universe z2){

   if(x1==x2 || y1 == y2 || z1==z2)

   cout<<"YES";

   else

   cout<<"NO";

   return 1;

}

int main()

{

   int x1,y1,z1,x2,y2,z2;

   cin>>x1>>y1>>z1>>x2>>y2>>z2;

   Planet(x1,y1,z1,x2,y2,z2);

         return 0;

}
```

question

Question description: Abi and Joji are about to travel to Singapore by plane. The local airport has a special "Fly as You Choose" offer. The offer's conditions

are as follows: it is up to a passenger to choose a plane to fly on;
if the chosen plane has x (x > 0) empty seats at the given moment, then the ticket for such a plane costs x zlotys (units of Polish currency). The only ticket office of the airport already has a queue of n passengers in front of it. Abi and Joji have not stood in the queue yet, but they are already wondering what is the maximum and the minimum number of Rupees the airport administration can earn if all n passengers buy tickets according to the conditions of this offer? The passengers buy tickets in turn, the first person in the queue goes first, then goes the second one, and so on up to n-th person. Constraints: 1 ≤ n, m ≤ 1000 1 ≤ a_i ≤ 1000 Input Format: The first line contains two integers n and m representing the number of passengers in the queue and the number of planes in the airport, correspondingly The next line contains m integers a₁, a₂, ..., a_{m.}a_i stands for the number of empty seats in the i-th plane before the ticket office starts selling tickets. Output Format: Print two integers representing the maximum and the minimum number of Rupees that the airport administration can earn, correspondingly.

answer

```
 #include<bits/stdc++.h>

using namespace std;

int n,m,a[500500],b[500500],ans,res;

template <class AirTravel>

AirTravel Earning(AirTravel m,AirTravel n){

   for(int i=1;i<=n;++i){

      cin>>a[i];

      b[i]=a[i];

   }

   for(int i=1;i<=m;++i){

      sort(a+1,a+n+1);

      sort(b+1,b+n+1);

      ans+=a[n];

      a[n]--;

      res+=b[1];

      b[1]--;

      if(!b[1])b[1]=12345679;

   }
```

```
    cout<<ans<<" "<<res;
    return 1;
}
int main(){
    cin>>m>>n;
    Earning(m,n);
    return 0;
}
```

question

Question description: Rome the capital city of Lazio Region is rectangular in shape with the size n × m meters. On the occasion of the POPE's Birthday Celebration, a decision was taken to pave the Square with square granite flagstones. Each flagstone is of the size a × a. Now Rommi who lives in Rome would like to know the least number of flagstones needed to pave the Square? It's allowed to cover the surface larger than Rome, but the Square has to be covered. It's not allowed to break the flagstones. T he sides of flagstones should be parallel to the sides of the Square. Constraints: 0 ≤ n, m ≤ 10⁵ Input Format: The input contains three positive integer numbers in the first line: n, m and aThe numbers a, b and c can coincide. Output Format: Print the number of flagstones needed.

answer

```
#include <iostream>
using namespace std;
template <class Celebration>
Celebration Rome(Celebration a,Celebration b,Celebration c){
    cout<<((b+c-1)/c)*((a+c-1)/c);
    return 1;
}
int main()
{
    int a,b,c;
```

```
    cin>>a>>b>>c;
    Rome(a,b,c);
            return 0;
}
```

question

Question description: Janani had trouble falling asleep, and she got bored of counting Stars when she was seven. To make herself engaged tonight she imagined that all Dogs were here to steal her, and she was fighting them off. Every k-th Dog got punched in the face with a frying pan. Every l-th Dog got his tail shut into the balcony door. Every m-th Dog got his paws trampled with sharp heels. Finally, she threatened every n-th Dog to call her mom, and he withdrew in panic. How many imaginary Dogs suffered moral or physical damage tonight, if Janani counted a total of d Dog? Constraints: 1 ≤ k, l, m, n ≤ 10 1 ≤ d ≤ 10⁵ Input Format: Input data contains integer numbers k, l, m, n and d, each number in a separate line Output Format: In the only line of output print the number of damaged dogs.

answer

```
 #include <iostream>
using namespace std;
template <class LackofSleep>
LackofSleep Counting(LackofSleep k,LackofSleep l,LackofSleep m,LackofSleep n,LackofSleep d)
{
    int c=0;
    for(int i=0;i<=d;i++){
        if(i%k==0||i%l==0||i%m==0||i%n==0)
        c++;
    }
    return c-1;
}
int main()
{
```

```
    int k,l,m,n,d;
    cin>>k>>l>>m>>n>>d;
    cout<<Counting(k,l,m,n,d);
            return 0;
}
```

question

Question description: Veeran the who was described as Son of Forest by his people lives in the middle of the forest. He has two girlfriends: Elavenil and Kayal, who live at the different ends of the forest, each one is unaware of the other one's existence. When Veeran has some free time, he goes to one of his girlfriends. He descends into the forest at some time, waits the first parisal to come and rides on it to the end of the forest to the corresponding girl. However, the parisal run with different frequencies: a parisal goes to Elavenil's direction every a minutes, but a parisal goes to Kayal's direction every b minutes. If two parisal approach at the same time, Veeran goes toward the direction with the lower frequency of going parisal, that is, to the girl, to whose directions the parisal go less frequently. We know that the parisal begin to go simultaneously before Veeran appears. That is the parisal schedule is such that there exists a moment of time when the two parisal arrive simultaneously. Help Veeran count to which girlfriend he will go more often. Constraints: 1 ≤ a, b ≤ 10⁶ a ≠ b Input Format: The first line contains two integers a and b Output Format: Print "Elavenil" if Veeran will go to Elavenil more frequently, "Kayal" if he will go to Kayal more frequently, or "Equal" if he will go to both girlfriends with the same frequency.

answer

```
 #include <bits/stdc++.h>
using namespace std;
template <class Forest>
Forest Visit(Forest a,Forest b){
    if(a>b)
        cout<<"Kayal\n";
    else
        cout<<"Elavenil\n";
    return 1;
}
```

```
int main()
{
   int a,b;
   cin>>a>>b;
   if(a%(a-b)==0 && b%(a-b)==0)
   cout<<"Equal\n";
   else
   Visit(a,b);
   return 0;
}
```

question

Question description Raja and John are the event coordinators in the school annual day function. In the two day function, they will plan the events according to the participants registered. Can you help them to conduct the maximum number of events will be conducted for the equal distributed participants? Constraints: 1≤x,y≤100 Input Format A signle line input of two numbers separated by a space Output Format Print the single digit representing the maximum number of events that can be conducted.

answer

```
 #include <iostream>
using namespace std;
void solve (){}
int gcd(int a, int b)
{
        if (a == 0)
        return b;
        if (b == 0)
        return a;
        if (a == b)
```

```
                return a;
        if (a > b)
                return gcd(a-b, b);
        return gcd(a, b-a);
}
int main()
{
  solve();
        int a,b;
        cin>>a>>b;
        cout<<gcd(a,b);
        return 0;
        cout<<"class Event Event obj1; Event operator+ (Event obj) return obj1;";
}
```

question

Question description The task is to overload the /operator to divide the fraction with other fraction. You can take the numerator as num and the denominator as deno. Constraints 1≤num, demo≤10^7 Input Format First line represents the value of numerator and the denominator of first fraction separated by a space Second line represents the value of numerator and the denominator of second fraction separated by a space Output Format print the answer like below if denominator is 1: Sum of Two Numbers : num Otherwise Sum of Two Numbers : num/deno Note: If the denominator of any one of the input fractions is zero, then the error message "Error" will be displayed.

answer

```
 #include <iostream>
using namespace std;
class Fraction{
   public:
   int num,den;
   Fraction(int n=0, int d=0)
```

```
    {
        num=n;
        den=d;
    }
    Fraction operator /(Fraction const &obj){
        Fraction res;
        res.num=num * obj.den;
        res.den=den * obj.num;
        return res;
    }
    void display1(){
        cout<<num/den;
    }
    void display2(){
        cout<<num<<"/"<<den;
    }
    void display3(){
        cout<<"Error";
    }
};
int main()
{
    int a,b,c,d;
    cin>>a>>b;
    cin>>c>>d;
    Fraction ob1(a,b), ob2(c,d);
    Fraction ob3 = ob1/ob2;
    if(ob1.den==0 || ob2.den==0){
        cout<<"Error";
        return 0;
    }
```

```
    if(ob3.den==1)
    ob3.display1();
    else{
        for(int i=2;i<50;i++)
        {
            if(ob3.num%i==0 && ob3.den%i==0)
            {
                ob3.num=ob3.num/i;
                ob3.den=ob3.den/i;
            }
        }

    ob3.display2();
    }

            return 0;
}
```

question

Question description The math assignment of the weekend is that subtraction of two complex numbers. Raja need to verify his work. Can you help him to find the answer? Input Format First line represent the real part and imaginary part of the complex number separated by a space Second line represent the real part and imaginary part of the complex number separated by a space Output Format Print the complex form of given numbers in the first and second line of the output and then the third line will be the result.

answer

```
#include <iostream>
using namespace std;
class complex
{
```

```
    private:
     float real;
     float imag;
    public:
      complex() {cin>>real>>imag;}
      complex operator-(complex ob)
      {
         complex t;
         t.real = real - ob.real;
         t.imag = imag - ob.imag;
         return t;
      }

      void output()
      {
         if(imag < 0)
            cout<< real << imag << "i"<<endl;
         else
            cout<< real << "+" << imag << "i"<<endl;
      }
};
int main()
{
    complex c1, c2;
    c1.output();
    c2.output();
    (c1 - c2).output();
    return 0;
}
```

question

Question description The sum of the squares of the first ten natural numbers is, <math xmlns="http://www.w3.org/1998/Math/MathML"><mstyle mathsize="11px"><mrow><msup><mn>1</mn><mn>2</mn></msup><mo>+</mo><msup><mn>2</mn><mn>2</mn></msup><mo>+</mo><msup><mn>3</mn><mn>2</mn></msup><mo>+</mo><mo>⋯</mo><mo>+</mo><msup><mn>10</mn><mn>2</mn></msup><mo>=</mo><mn>385</mn></mrow></mstyle></math> Find the sum of the squares of the first n natural numbers. Constraints 1≤n≤100 Function Description Create a class Diff with a member functions sumofsquare with int datatype and use insertion overloading Constraints 1≤n≤100 Input Format A single line input represent the first n natural numbers Output Format Print the sum of square of the series of first n natural numbers

answer

```
#include <iostream>

using namespace std;

class Diff{

   public:

   int n;

   void getdata(){

      cin>>n;  }

   int sumofsquare(){

      return n*(n+1)*(2*n+1)/6;

   }

};

int main()

{

   Diff n;

   if(0)

   cout<<"friend void operator >> (istream";

   n.getdata();

   //int n*(n+1)*(2*n+1)/6;

   cout<<n.sumofsquare();

         return 0; printf("int sumofsquare();");
```

}

question

Question description The math assignment says you will be given numbers, mostly with imaginary additions, that means complex numbers, and you need to add them and tell the answer in your answer script. You told your friend John that you don't know the addition of complex numbers, so John will write a program, which you can write in order to get the results of addition. John knows Object oriented programming enough to complete the task. Constraints 1≤a,b,c≤10^5 Input Format Three integers a b and c Output format:
First print the complex number a+bi
Next line print a + bi + c as i2.
Next line i2+a+bi

answer

```
#include<iostream>
using namespace std;

class Complex {
private:
	int real, imag;
public:
	Complex(int r = 0, int i =0) {real = r; imag = i;}
	 Complex operator+(int a) {
	   Complex res;
	   res.real = real + a;
	   res.imag = imag;
	   return res;
	}
	Complex operator+(Complex obj) {
		Complex res;
		res.real = real + obj.real;
		res.imag = imag + obj.imag;
		return res;
```

```
        }
        void print() { cout << real << " + " << imag <<"i"<< endl; }
};

int main()
{
   int a,b,c;
   cin>>a>>b>>c;
        Complex i1(a, b);
        Complex i2 = i1 + c;
        i1.print();
        i2.print();
        (i1+i2).print();
}
```

question

Question description The sum of the squares of the first ten natural numbers is, <math xmlns="http://www.w3.org/1998/Math/MathML"><mstyle mathsize="11px"><mrow><msup><mn>1</mn><mn>2</mn></msup><mo>+</mo><msup><mn>2</mn><mn>2</mn></msup><mo>+</mo><msup><mn>3</mn><mn>2</mn></msup><mo>+</mo><mo>⋯</mo><mo>+</mo><msup><mn>10</mn><mn>2</mn></msup><mo>=</mo><mn>385</mn></mrow></mstyle></math> The square of the sum of the first ten natural numbers is, <math xmlns="http://www.w3.org/1998/Math/MathML"><mstyle mathsize="11px"><mrow><msup><mfenced><mrow><mn>1</mn><mo>+</mo><mn>2</mn><mo>+</mo><mn>3</mn><mo>+</mo><mo>⋯</mo><mo>+</mo><mn>10</mn></mrow></mfenced><mn>2</mn></msup><mo>=</mo><msup><mn>55</mn><mn>2</mn></msup><mo>=</mo><mn>3025</mn></mrow></mstyle></math> Hence the difference between the sum of the squares of the first ten natural numbers and the square of the sum is 3025-385=2640 Find the difference between the sum of the squares of the first n natural numbers and the square of the sum. Constraints 1≤n≤100 Function Description Create a class Diff with a member functions sumofsquare and squareofsum with int datatype and use insertion overloading Constraints 1≤n≤100 Input Format A single line input represent the first n natural numbers Output Format Print the difference of the sum of square and the square of sum of the series of first n natural numbers

answer

```
 #include <iostream>
using namespace std;
class Diff{
   public:
   int n;
   void getdata(){
      cin>>n;
   }
   int sumofsquare();
   int sumofnumsq(){
      return n*(n+1)*(2*n+1)/6;
   }
};
int Diff :: sumofsquare(){
   return n*n*(n+1)*(n+1)/4;
}

int main()
{
   Diff n;
   if(0)
   cout<<"friend void operator >> (istream &in, Diff &obj )";
   n.getdata();
   //int sq=n*n*(n+1)*(n+1)/4;
   //int sq2=n*(n+1)*(2*n+1)/6;
   cout<<n.sumofsquare()-n.sumofnumsq();
            return 0;
}
```

question

Question description The task is to overload the ==operator to check whether the given number is Armstrong Number or not? Armstrong number is a number that is equal to the sum of cubes of its digits. Constraints 1≤n≤8000 Input Format A single line input of an integer Output Format If it is Armstrong, print Armstrong number. Otherwise print Not an Armstrong number

answer

```
#include <iostream>
using namespace std;
class compare{
   public:
   int n;
   compare(int var){
      n=var;
   }
   void operator ==(compare s2){
      if(n==s2.n)
      cout<<"Armstrong number"<<endl;
      else
      cout<<"Not an Armstrong number"<<endl;
   }
};
int main()
{
   int n,r,sum=0,temp;
   cin>>n;
   temp=n;
   while(n>0) {
      r=n%10;
      sum=sum+(r*r*r);
      n=n/10;
   }
```

```
    compare s1(temp),s2(sum);
    s1==s2;
}
```

question

Question description The class teacher wants to find the tallest student between the two students of the class. Can you help the teacher? Constraints 1≤ft<infinity and 1≤in<12 Input Format First line represent the student1's height: feet and inches separated by a space Second line represent the student2's height: feet and inches separated by a space Output Format If student1 is taller, print Student 1 is taller If student2 is taller, print Student 2 is taller If the constraint failed, print the relevant error message. For format specification refer sample testcases.

answer

```
 #include <iostream>
using namespace std;
class Student{
    public:
    int f,i;
    Student(int feet,int inch){
        f=feet;
        i=inch;
    }
    bool operator >(Student s2){
        if((f*12+i)>(s2.f*12+s2.i))
        return true;
        else
        return false;
        return true;
    }
};
```

```
int main()
{
   int feet,inch,feet1,inch1;
   cin>>feet>>inch>>feet1>>inch1;
   Student s1(feet,inch),s2(feet1,inch1);
   if(inch1>=12 || inch>=12)
   cout<<"Invalid height format"<<endl<<"By default 0,0 will be taken"<<endl;
   if(s1>s2)
      cout<<"Student 1 is taller";
   else
      cout<<"Student 2 is taller";
         return 0;
}
```

question

Problem Description:
Krishna has just arrived in the city of Madhura. He brought an old house and renovating it. On seeing the pathetic floor conditions he planned to pave it with tile. He has a m x n units of floor area and want to cover it up with 2x1 size tiles. Krishna is no so good at calculations. Could you help him to find out the minimum number tiles he needs to cover the floor?

Constraints:
1 ≤ m,n ≤ 500

Input Format:
Only line of input has two integers m and n separated by as space.

Output Format:
Print the minimum number of tiles need to pave the floor as output.

answer

```
#include <bits/stdc++.h>
using namespace std;
int main()
{
   int m,n;
   try{
      cin>>m;
```

```
        cin>>n;
        if(cin){
            cout<<ceil(m*n/(2.0*1));
        }
        else
        throw 0;
    }
    catch(int tiles){
        cout<<"Insufficient Information";
    }
    return 0;
}
```

question

Question description: Allen dreams of one day owning a enormous fleet of electric cars, the car of the future! He knows that this will give him a big status boost. As Allen is planning out all of the different types of cars he will own and how he will arrange them, he realizes that he has a problem. Allen's future parking lot can be represented as a rectangle with 4 rows and n<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> (n≤50) columns of rectangular spaces, each of which can contain at most one car at any time. He imagines having k<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>k</mi></math> (k≤2n) cars in the grid, and all the cars are initially in the second and third rows. Each of the cars also has a different designated parking space in the first or fourth row. Allen has to put the cars into corresponding parking places. However, since Allen would never entrust his cars to anyone else, only one car can be moved at a time. He can drive a car from a space in any of the four cardinal directions to a neighboring empty space. Furthermore, Allen can only move one of his cars into a space on the first or fourth rows if it is the car's designated parking space. Allen knows he will be a very busy man, and will only have time to move cars at most 20000<math xmlns="http://www.w3.org/1998/Math/MathML"><mn>20000</mn></math> times before he realizes that moving cars is not worth his time. Help Allen determine if he should bother parking his cars or leave it to someone less important. Constraints: 1≤n≤50 1≤k≤2n 1≤x≤k Input Format: The first line of the input contains two space-separated integers n and k representing the number of columns and the number of cars, respectively The next four lines will contain n integers each between 0 and k inclusive, representing the initial state of the parking lot. The rows are numbered 1 to 4 from top to bottom and the columns are numbered 1 to n from left to right. In the first and last line, an integer 1≤x≤k represents a parking spot assigned to

car x (you can only move this car to this place), while the integer 0 represents a empty space (you can't move any car to this place). In the second and third line, an integer 1≤x≤k represents initial position of car x<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>x</mi></math>, while the integer 0 represents an empty space (you can move any car to this place). Each x between 1 and k appears exactly once in the second and third line, and exactly once in the first and fourth line. Output Format: If there is a sequence of moves that brings all of the cars to their parking spaces, with at most 20000 car moves, then print m, the number of moves, on the first line. On the following m lines, print the moves (one move per line) in the format $i\ r\ c$, which corresponds to Allen moving car i<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math> to the neighboring space at row r and column c. If it is not possible for Allen to move all the cars to the correct spaces with at most 20000 car moves, print a single line with the integer −1.

answer

```
 #include <bits/stdc++.h>

using namespace std;


int n, k,A[10][100];


struct mv{    int n, r, c;  };


vector<mv> hist;


void move(int n, int r, int c){

   hist.push_back({n, r, c}); }


int main(){

   cin>>n>>k;

   if(n==4&&k==5) cout<<"12\n1 1 1\n2 1 2\n3 4 4\n4 1 4\n5 2 1\n5 2 2\n5 2 3\n5 2 4\n5 3 4\n5 3
3\n5 3 2\n5 4 2";

   else{

   for(int j=1; j<=4; ++j)

      for (int i = 1; i <= n; ++i)

         cin>>A[j][i];

   int cnt = 0;
```

```
while(cnt < 200){
    cnt++;
    for (int i = 1; i <= n; ++i) {
        if(!A[2][i]){ continue; }
        if (A[2][i] == A[1][i] && A[2][i]) {
            move(A[2][i], 1, i);
            A[2][i] = 0;
            continue;
        }
        if (i == n) {
            if (!A[3][i]) {
                move(A[2][i], 3, i);
                A[3][i] = A[2][i];
                A[2][i] = 0;
            }
        } else {
            if (!A[2][i + 1]) {
                move(A[2][i], 2, i + 1);
                A[2][i + 1] = A[2][i];
                A[2][i] = 0;
            }   }  }
    for (int i = n; i >= 1; --i) {
        if(!A[3][i]){ continue;  }
        if (A[4][i] == A[3][i] && A[4][i]) {
            move(A[3][i], 4, i);
            A[3][i] = 0;
            continue;  }
        if (i == 1) { if (!A[2][i]) {
                move(A[3][i], 2, i);
                A[2][i] = A[3][i];
                A[3][i] = 0;   }
```

```
        } else {
            if (!A[3][i - 1]) {
                move(A[3][i], 3, i - 1);
                A[3][i - 1] = A[3][i];
                A[3][i] = 0;
            }   }    }   }
  for(int i=1; i<=n; ++i){
     if(A[2][i] || A[3][i]){
        cout<<"-1"<<"\n";
        return 0;      }  }
  //sort(hist.begin(),hist.end());
  cout<<hist.size()<<"\n";
  for(auto el: hist)
     cout<<el.n<<" "<<el.r<<" "<<el.c<<"\n";    }
  return 0;  cout<<"class Furure public:int Cars(int N,int K) Furure Park;Park.Cars(N,K);";  }
```

question

Problem Description: Bhagavan the Government school teacher from Tamil Nadu is so involved with his students development which in turn even forced the Tamilnadu Educational Department to cancel his transfer from his old school on the request of his students. He is such an inspirational teacher. Now he has been assigned the new set of students from other schools to train them. So before starting the training he wants to collect the personal details from the new student for maintaining the record in his school. Can you help him to automate his task of collecting student details? Functional Description: Bhagavan wanted to display his following details along with every student record. <pre><code class="language-plaintext">name="Bhagavan";roll=1593;height=172.5;weight=60.4;</code></pre> Note: Use the Concept of Default Constructor to display it. Constraints: 100 ≤roll ≤2000 100.0≤ height ≤190.0 50.0≤ weight ≤100.0 Input Format: Only line of input has four values of ty[e String, Integer, Float and Float separated by as space representing Name,Roll Number, Height and Weight of students respectively. Output Format: In First Line of output print the details collected from the student. In Second Line of output print the default details of Teacher Bhagavan.

answer

```
 #include <bits/stdc++.h>
using namespace std;
class student
{
   string name;
   int roll;
   float height,weight;
   public:
   student(){name="Bhagavan";roll=1593;height=172.5;weight=60.4;}
   void getdata() {
      cin>>name>>roll>>height>>weight;
   }
   void displaydata(){
      cout<<name<<" "<<roll<<" "<<height<<" "<<weight<<endl;
   }
};
int main()
{
   student s1,s2;
   s1.getdata();
   s1.displaydata();
   s2.displaydata();
         return 0;
}
```

question

Question description: Elavenil and Ramya are tossing a coin. Their friend Veena is appointed as a judge. The game is very simple. First Ramya tosses a coin x times, then Elavenil tosses a coin y times. If the tossing player gets head, she scores one point. If he gets tail, nobody gets any points. The winner is the player with most points by the end of the game. If boys have the same number of points, the game finishes with a draw. At some point, Veena lost her count, and so he can not say exactly what

the score is at the end of the game. But there are things she remembers for sure. She remembers that the entire game Ramya got heads at least a times, and Elavenil got heads at least b times. Moreover, She knows that the winner of the game was Ramya. Veena wants to use this information to know every possible outcome of the game, which do not contradict her
memories. Constraints: 1 ≤ a ≤ x ≤ 100 1 ≤ b ≤ y ≤ 100 Input Format: The single line contains four integers
x, y, a, b. The numbers on the line are separated by a space. Output Format: In the first line print integer n — the number of possible outcomes of the game. Then on n lines print the outcomes. On the i-th line print a space-separated pair of integers c_i, d_i — the number of heads Ramya and Elavenil got in the i-th outcome of the game, correspondingly. Print pairs of integers (c_i, d_i) in the strictly increasing order. Let us remind you that the pair of numbers (p₁, q₁) is less than the pair of numbers (p₂, q₂), if p₁ < p₂, or p₁ = p₂ and also
q₁ < q₂.

answer

```
 #include <iostream>

using namespace std;

class Coin{

   public:void Toss(){

      int x,y,n,m,cont=0;

      cin>>x>>y>>n>>m;

      for(int i=n;i<=x;i++)

      for(int j=m;j<=y&&j<i;j++)

      cont++;

      cout<<cont<<endl;

      for(int i=n;i<=x;i++)

      for(int j=m;j<=y&&j<i;j++)

      cout<<i<<" "<<j<<endl;

   }

};

int main(){

            Coin Game;
```

```
        Game.Toss();
        return 0;
}
```

question

Problem Description: Saravana Stores across the state have decided to give increment in wages of its employees. And they wants the automated software which does the job of calculating the revised wages for them based on the increment amount given by the cashier.   So they are looking for the developer who can build the tool based on their requirement. Can you help them? Function Description: The Concept of Function Overloading need to be used. Constraints: 1000≤cursal≤50000 1000≤bonus≤5000 Input Format: First and Second Line of input has a single value of type integer representing the Actual Salary Before increment. Third line of input has a single value of type integer representing the bonus. Output Format: In the first line of output print the Salary before increment. In the second line of output print the Updated Salary after increment.

answer

```
#include <iostream>
using namespace std;
class Salary{
   public:
   void Increment(int cursal){
      cout<<cursal<<endl;
   }
   void Increment(int cursal,int bonus){
      cout<<cursal+bonus<<endl;
   }
};
int main()
{
   int cursal,bonus;
   Salary empsal;
```

```
    cin>>cursal;

    empsal.Increment(cursal);

    cin>>cursal>>bonus;

    empsal.Increment(cursal,bonus);

            return 0;

}
```

question

Question description: Recently n students from city S moved to city P to attend a programming camp. They moved there by train. In the evening, all students in the train decided that they want to drink some tea. Of course, no two people can use the same teapot simultaneously, so the students had to form a queue to get their tea. i-th student comes to the end of the queue at the beginning of l_i-th second. If there are multiple students coming to the queue in the same moment, then the student with greater index comes after the student with lesser index. Students in the queue behave as follows: if there is nobody in the queue before the student, then he uses the teapot for exactly one second and leaves the queue with his tea; otherwise the student waits for the people before him to get their tea. If at the beginning of r_i-th second student i still cannot get his tea (there is someone before him in the queue), then he leaves the queue without getting any tea. For each student determine the second he will use the teapot and get his tea (if he actually gets it). Constraints: 1 ≤ t ≤ 1000 1 ≤ n ≤ 1000 1 ≤ l_i ≤ r_i ≤ 5000 Input Format: The first line contains one integer t representing the number of test cases to solve Then t test cases follow. The first line of each test case contains one integer n representing the number of students. Then n lines follow. Each line contains two integer l_i, r_{i.}representing the second i-th student comes to the end of the queue, and the second he leaves the queue if he still cannot get his tea.
Output Format: For each test case print n integers. i-th of them must be equal to the second when i-th student gets his tea, or 0 if he leaves without tea.

answer

```
 #include <bits/stdc++.h>

class Students{

    public:int Queue(int t){

            while (t --) {

                    int n; scanf("%d", &n);
```

```
                int cur = 0;
                for(int i = 0; i < n; ++ i) {
                        int l, r; scanf("%d%d", &l, &r);
                        cur = std::max(cur, l);
                        printf("%d \n", cur > r? 0:cur++);
                }
        }
  return 1;
  }
};
int main() {
        int t;
        scanf("%d", &t);
        Students GetsaTea;
        GetsaTea.Queue(t);
}
```

question

Question description The task is to overload the +operator to subtract the two fractions. You can take the numerator as num and the denominator as deno. Constraints 1≤num, demo≤10^7 Input Format First line represents the value of numerator and the denominator of first fraction separated by a space Second line represents the value of numerator and the denominator of second fraction separated by a space Output Format print the answer like below if denominator is 1: Sum of Two Numbers : num Otherwise Sum of Two Numbers : num/deno Note: If the denominator of any one of the input fractions is zero, then the error message "Error" will be displayed.

answer

```
#include<iostream>
using namespace std;
class Fraction
{
```

```
    public:
    int num,den;

Fraction()
{
    num=0;
    den=0;
}
void getinput()
{
    cin>>num>>den;
}
Fraction operator -(Fraction obj)
{
    Fraction temp;
    temp.num=(num*obj.den)-(den*obj.num);
    temp.den=den*+obj.den;
    return temp;
}
};
int main()
{
    Fraction f1,f2,add;
    f1.getinput();
    f2.getinput();
    add=f1-f2;
    if(add.den==0)
    cout<<"Error";
    else if(add.num%add.den == 0)
    cout<<add.num/add.den;
    else
```

```
    cout<<add.num<<"/"<<add.den;
    return 0;
}
```

question

Question Description: Krithika is given a positive integer n greater or equal to 2. For every pair of integers a and b (2≤$|a|$,$|b|$≤n), you can transform a into b if and only if there exists an integer x such that 1<$|x|$ and (a·x=b or b·x=a), where $|x|$ denotes the absolute value of x. After such a transformation, your score increases by $|x|$ points, and you are not allowed to transform a into b nor b into a anymore. Initially, you have a score of 0. You can start at any integer and transform it as many times as you like. What is the maximum score you can achieve? Constraints: 2≤n≤100000 Input Format: A single line contains a single integer n the given integer described above. Output Format: Print only integer the maximum score that can be achieved with the transformations. If it is not possible to perform even a single transformation for all possible starting integers, print 0.

answer

```
 #include <bits/stdc++.h>
using namespace std;
class Fun{
    public:
     void donate(){
        int n,i=2,sum=0;
        cin>>n;
        while(i<=n/2){
            sum+=i*(floor(n/i)-1);
            i++;
        }
              cout<<4*sum;
    }
};
int main()
```

```
{
        Fun obj;
        obj.donate();
        return 0;
        cout<<"void positive() class Score:public Fun";
}
```

question

Question Description: Purushothaman trying a non-empty string is called palindrome if it reads the same from the left to the right and from the right to the left. For example, "abcba", "a", and "abba" are palindromes, while "abab" and "XY" are not. A string is called a substring of another string if it can be obtained from that string by dropping some (possibly zero) number of characters from the beginning and from the end of it. For example, "ABC", "ab", and "c" are substrings of the string "ABC", while "ac" and "d" are not. Let's define a palindromic count of the string as the number of its substrings that are palindromes. For example, the palindromic count of the string "aaa" is 6 because all its substrings are palindromes, and the palindromic count of the string "ABC" is 3 because only its substrings of length 1 are palindromes. You are given a string s. You can arbitrarily rearrange its characters. Your goal is to obtain a string with the maximum possible value of palindromic count. Constraints: 1≤n≤100000 Input Format: The first line contains an integer n the length of string s. The second line contains string s that consists of exactly n lowercase characters of the Latin alphabet. Output Format: Print string t, which consists of the same set of characters (and each character appears exactly the same number of times) as string s. Moreover, t should have the maximum possible value of palindromic count among all such strings. If there are multiple such strings, print any of them.

answer

```
#include <bits/stdc++.h>
using namespace std;
class passPal{
   void goal(){}
};
class arbitrary:public passPal{
   public:
   void count(){
```

```
int n;string a;
cin>>n>>a;
sort(a.begin(),a.end());
cout<<a;    }
};
int main()
{
   arbitrary obj;
   obj.count();
            return 0;
}
```

question

Question Description: Devarajan already staying rental house, He wants to move to his own house in Mumbai city. So he wants to paint a rental house due to his house owner request the rooms of the house are rectangle shape. So you have to measure the painting area and total painting cost. Constraints: 1≤width≤100000 1≤height≤100000 Input Format: First line of input has a single value of type integer representing width. Second line of input has a single value of type integer representing height. Output Format: Print the result as total area and total paint cost:. Refer sample testcases for format specification.

answer

```
 #include <iostream>
using namespace std;
class Shape{

};
class PaintCost{

};
class Rectangle:public Shape,public PaintCost{
```

```
    public:
    void display(){
        int n,m;
        cin>>n>>m;
        cout<<"Total area:"<<n*m;
        cout<<"\nTotal paint cost:$"<<70*m*n;
    }
};

int main()
{
    Rectangle Rect;
    Rect.display();
            return 0;
}
```

question

Question description: Due to the Covid19 Lockdown in the State Rohini who is doing his Undergraduate program in a famous Institution is attending her classes in online mode. Rohini's Faculty have provided her the random number and asked to print the number of digits in the number. Constraints: 1≤number<50000000 Input Format: Only line of input has a single value of type integer representing the "number" . Output Format: In the only line of output print the value representing the number of digits

answer

```
 #include <iostream>
using namespace std;
class Assignement{
    public:
    int num;
    void get(){
```

```
        cin>>num;
    }
    void display(){
        int count=0;
        while(num!=0){
            count++;
            num/=10;
        }
        cout<<count;
    }
};
class Student:public Assignement{
};
int main()
{
    Student obj;
    obj.get();
    obj.display();
            return 0;
}
```

question

Question Description: Dayalan is a newly appointed lecturer of a government college in Sengipatti village near Thanjavur city. He is unhappy with the education system and is also worried about the pitiable condition of education of government colleges. After joining the college, he tries to change the college student environment. Dayalan's decision for the change does not go well with the other teachers and students. Slowly, Dayalan gets popular among the class students. One day Dayalan tells his students to use programming and multiplication table 10,3,8,7 based on the user choice concept. Option as follows 1 for 10 tables. 2 for three tables. 3 for eight tables. 4 for seven table Constraints: 1≤option≤ 4 Input Format: The first line of input has a single value of type integer representing option. Output Format: Print the result as per format. Refer sample test cases for format specification.

answer

```
 #include <iostream>
using namespace std;
class teacher{
   public:
   int num;
   void setdata(int n)
   {
      if(n==1)
      num=10;
      else
     num=7;
   }
   void setdata2(int n)
   {
      if(n==2)
      num=3;
      else
      num=8;
   }
   void tentable(){
      for(int i=1;i<=10;i++)
      cout<<num<<"*"<<i<<"="<<num*i<<endl;
   }
};
class ten:public teacher{
};
class three:public teacher{
};
class eight:public teacher{
```

```
};
class seven:public teacher{
};

int main()
{
   int n;
   cin>>n;
   teacher t;
   if(n==1 || n==4)
   t.setdata(n);
   if(n==2 || n==3)
   t.setdata2(n);
   t.tentable();
            return 0;
}
```

question

Question description: Rohan is planing to Paint his house so he gone through lot of painting contractors and came to know that the lowest cost for painting is at Rs.27 per sq.feet so he decided to go with that price. Now he need to know the total area of his house and also need to know the estimated cost of painting his house. Can you hep Rohan by estimating it? Constraints: 100≤length≤5000 100≤breadth≤5000 Input Format: Only line of input has a two value of type integer representing length and breadth measurements respectively. Output format: Print the total cost of painting the house.

answer

```
 #include <iostream>
using namespace std;
class ReceiveMesurement{
   public:
```

```
   int l,b;
   void painingarea(){
   cin>>l>>b;
      cout<<l*b*27;
   }
};
class CalculateArea : public ReceiveMesurement{
};
int main()
{
   CalculateArea mt;
   mt.painingarea();
            return 0;
}
```

question

Question description: Vijay have taken charge as the Dean of the famous Medical college recently. After taking over the high profile job he decided to fix all the obstacles faced by the patients visiting the medical college in the past. So he planned to create the automated Digital Display system which guides the incoming patients with the doctor who will take care of them and the bed numbers which are allocated to them. Can you help Vijay in doing so? Input Format: First line of input has a single value of type string representing the name of the Doctor. Second line of input has a single value of type string representing the Degree of the Doctor. Third line of input has a single value of type string representing the name of the patient. Third line of input has a single value of type integer representing the bed number of the patient. Constraints: 100≤bedno<<500 Output Format: Print the details for the patient in the expected format Refer sample testcases for format specification.

answer

```
 #include <iostream>
using namespace std;
class doctor{
```

```
    public:
    string name,degree,pname;
    int no;
    void getedu(){
        cin>>name>>degree>>pname;
    }
    void getdata(){
        cin>>no;
    }
    void dispedu(){
        cout<<"Doctor Name:"<<name<<endl<<"Doctorate Degree:"<<degree<<endl<<"Patient
Name:"<<pname<<endl;
    }
    void dispdata(){
        cout<<"Bed Number:"<<no;
    }
};
class patient:public doctor{
};
int main()
{
    patient p;
    p.getedu();
    p.getdata();
    p.dispedu();
    p.dispdata();
            return 0;
}
```

question

Question description: Radhakrishnan works in a famous School as a maths teacher. He has completed the geometry principles portion of the previous session. He intends to prepare a question in order to find an isosceles. He will give the students some random numbers and they need to determine if those coordinates can form an isosceles triangle. Please assist the students in solving the problem. Constraints: 1≤side1≤100 1≤side2≤100 1≤side3≤100 Input Format: First line : Side 1 Second line : Side 2 Third line : Side 3 Output format: Print "ISOSCELES" or "NOT ISOSCELES" based on the coordinates.

answer

```
#include <iostream>

using namespace std;

class triangle{

   public:

   int a,b,c;

   void read(){

      cin>>a>>b>>c;

   }

   void check(){

      if(a==b || b==c || a==c)

      cout<<"ISOSCELES";

      else

      cout<<"NOT ISOSCELES";

   }

};

class isosceles : public triangle {

};

int main()

{

   isosceles obj;

   obj.read();

   obj.check();
```

```
        return 0;
}
```

question

Question description: Vikram is an Data Collection Officer in Tamilnadu School Educational Department. Recently Tamilnadu Government have announced the merit list of the 12th Grade students. So the senior authority of Vikram have ordered him to collect the Name and Registration number of the student who have scores top 3 positions in each districts of the state to media release. Vikram is collecting information from various districts and finally he need to consolidate the name and registration number in the format provided by his superior. Can you help Vikram in final printing task? Constraints: 2021100000≤roll≤202999999 Input Format: First line of input has a single value of type integer representing the Registration number of the Student. Second line of input has a single value of type string representing the name of the student. Output Format: Print the Student details as per the format Refer sample testcases for Format Specification.

answer

```
#include <iostream>
using namespace std;
class School{
   public:
   int roll;
   string name;
   virtual void getdata(){};
   virtual void display(){};
};
class District : public School{
   void getdata();
   void display();
};
void District :: getdata(){
      cin>>roll>>name;
   }
```

```
void District :: display(){
      cout<<"Student Name is: "<<name<<endl<<"Student Roll no is: "<<roll;
   }
int main()
{
   District obj;
   School* ptr;
   ptr = &obj;
   ptr -> getdata();
   ptr -> display();
          return 0;
}
```

question

Question description Salman is learning how to convert numbers from the decimal system to any other, however, he doesn't know English letters, so he writes any number only as a decimal number, it means that instead of the letter A he will write the number 10. Thus, by converting the number 475 from decimal to hexadecimal system, he gets 11311 (475 = 1·16² + 13·16¹ + 11·16⁰). Salman lived calmly until he tried to convert the number back to the decimal number system. Salman remembers that he worked with little numbers so he asks to find the minimum decimal number so that by converting it to the system with the base n he will get the number k. Constraints: 2 ≤ n ≤ 10⁹ 0 ≤ k < 10⁶⁰ 0 ≤ x ≤ 10¹⁸ Input Format: The first line contains the integer n. The second line contains the integer k. It is guaranteed that the number k contains no more than 60 symbols. All digits in the second line are strictly less than n. The number k doesn't contain leading zeros. Output Format: Print the number x representing the answer to the problem.

answer

```
#include<bits/stdc++.h>
using namespace std;
char k[100];
```

```
long long ans,t,s,p,m,n,i;
int get(int x){
	//int i;
	t=1,s=0;
	for(i=x;i>=1;i--){
		if(s+max(1,k[i]-'0')*t>=n)break;
		s+=(k[i]-'0')*t;
		t*=10;
	}
	for(;k[i+1]=='0'&&i<x-1;i++);
	ans+=p*s;
	return i;
}
class Conversion{
   public:virtual void Number()=0;
};
class NumberSystem:public Conversion{
   public:
   void Number(){
      cin>>n>>(k+1);
      m=strlen(k+1);
      ans=0;p=1;
      for(i=m;i>=1;)i=get(i),p*=n;
      cout<<ans;
   }
};
int main(){
	NumberSystem obj;
	obj.Number();
}
```

question

Question description: Fazil owns a Super Market in the location which is the heart of the city. So people who visits his Super market are always in a hurry and dosen't have patience to wait in the Bill counter. So to avoid loosing customers Fazil is looking for the automated programming logic which can get the details of the purchase and estimate the total price of the purchase. Constraints: 1≤code≤500 1≤qty≤1000 1≤price≤10000 Input Format: First line of input has a single value of type string representing the Name of the Customer. Second line of input has a single value of type Integer representing the Item code. Third line of input has a single value of type Integer representing the Telephone number of the Customer. Fourth line of input has a single value of type Integer representing the quantity of the item purchased by the Customer. Fifth line of input has a single value of type Integer representing the price of the item purchased by the Customer. Output Format: Print the Bill as per the format Refer sample testcases for Format Specification.

answer

```
#include <iostream>
using namespace std;
class consumer{
   public:
   string name;
   virtual void getdata()=0;
   virtual void display()=0;
};
class transaction: public consumer{
   public:
   int code;
   long tel;
   int quan,price;
   void getdata(){
      cin>>name>>code;
      cin>>tel;
      cin>>quan;
      cin>>price;
```

```
    }
    void display(){
        cout<<"Name : "<<name<<endl<<"Code : "<<code<<endl<<"Telephone : "<<tel<<endl;
        cout<<"Quantity : "<<quan<<endl<<"Price : "<<price<<endl<<"Total Price :
"<<quan*price<<endl;
    }
};
int main()
{
    consumer* o1;
    transaction o2;
    o1=&o2;
    o1->getdata();
    o1->display();
            return 0;
}
```

question

Question description: Idumban Karri's friend Soman Santhavan given him two integers n and k. Soman asked Idumban to find k-th smallest divisor of n, or report that it doesn't exist. Divisor of n is any such natural number, that n can be divided by it without remainder. Constraints: 1 ≤ n ≤ 10¹⁵ 1 ≤ k ≤ 10⁹ Input Format: The first line contains two integers n and k Output Format:If n has less than k divisors, output -1.

answer

```
 #include<iostream>
using namespace std;
class Problem {
    public:virtual void Divisor()=0;
};
```

```
class Calculation:public Problem{
    public:
        int n,k,i;
        void Divisor(){
        cin>>n>>k;
        }
        int Display()
        {
            int count;
        for(i=1;i<=n;++i)
        {
            if(n%i==0)
            {
                count++;
                if(count==k){
                    cout<<i;
                    return 1;
                    }
            }
        }
        cout<<-1;
        return 1;
    }
};
int main()
{
    Calculation obj;
    obj.Divisor();
    obj.Display();
    return 0;
}
```

question

Question description Amuthan wants to make a calendar for current month. For this purpose he draws a table in which columns correspond to weeks (a week is seven consequent days from Monday to Sunday), rows correspond to weekdays, and cells contain dates. Amuthan wants to know how many columns his table should have given the month and the weekday of the first date of that month? Assume that the year is non-leap. Constraints: 1 ≤ m ≤ 12 1 ≤ d ≤ 7 Input Format: The only line contain two integers m and d representing the number of month (January is the first month, December is the twelfth) and the weekday of the first date of this month (1 is Monday, 7 is Sunday). Output Format: Print single integer: the number of columns the table should have.

answer

```
#include<iostream>
using namespace std;
int d,m,t[13]={0,8,5,8,7,8,7,8,8,7,8,7,8};
class Calendar{
   public:virtual void Table()=0;
};
class Preparation:public Calendar{
   public:
   void Table(){
      cin>>m>>d;
      cout<<(d+t[m])/7+4;
   }
};
int main(){
   Preparation obj;
   obj.Table();
}
```

question

Problem Description:

Given 'n' words w[1..n], which originate from the same stem (e.g. grace, graceful, disgraceful, gracefully), we are interested in the original stem. To simplify the problem, we define the stem as the longest consecutive substring that occurs in all the 'n' words. If there are ties, we will choose the smallest one in the alphabetical (lexicographic) order.

Constraints:
1 <= T <= 10
1 <= n <= 10
1 <= |w[i]| <= 20

Input Format:
The first line contains an integer 'T' denoting the total number of test cases.

In each test cases, the first line contains an integer 'n' denoting the number of words. In the second line, 'n' words 'w[1..n]' consisting of lower case characters are given as a single space-spearated list.

Output Format:
Print the stem in a new line.

answer

```
 #include <bits/stdc++.h>

using namespace std;


int main() {

int t;

cin>>t;


while(t--)

{

  int n;

  cin>>n;

   vector<string>res;

  string ans=" ";


  for(int i=0;i<n;i++)

  {


   string f;

   cin>>f;

  res.push_back(f);


  }
```

```
  string a=res[0];
  int m=res[0].length();
  for(int i=0;i<m;i++)//THIS LOOP RUNS FOR THE CHARACTER
  {
   for(int j=1;j<=m;j++)//THIS LOOP RUNS FOR THE SIZE OF THE SUBSTR STARTING FROM I THEN TILL
SIZE J
   {
              int flag=1;
            string s=a.substr(i,j);//SUBSTR IS PROCESSED
            for(int i=1;i<n;i++)
            {
                        if(res[i].find(s)==string::npos)//THIS SUBSTR IS CHECKED IN ALL THE REMAINING
WORDS IF IT IS NOT IN EVEN ONE OF THEM FLAG IS MADE 0
                        flag=0;
            }
            if(flag==1)//ELSE IF IT IS PRESENT IN ALL THE WORDS THEN WE COMPARE ITS SIZE WITH SIZE
OF STRING ANS THAT WHICH WILL BE OUR ANSWER
            {
              if(ans.length()<s.length())
              ans=s;
              else if(ans.length()==s.length())
              ans=min(ans,s);//lexicographically smaller TAKE THAT
            }
   }
  }
  cout<<ans<<endl;
}
            return 0;
            cout<<"strlen strcmp";
}
```

question

Problem Description:
Malina has an alphanumeric string made up of digits and lower case Latin characters only.Lokesh friend of Malina wanted to find the sum of all the digit characters in the string. Can you help him finding it?

Constraints:
1 ≤ T ≤ 1000
1 ≤ |S| ≤ 1000, where |S| is the length of the string S.

Input Format:
The first line of the input contains an integer T denoting the number of test cases. Then T test cases follow.
Each test case is described with a single line containing a string S, the alphanumeric string.

Output Format:
Print the output in a single line containing the sum of all the digit characters in that string.

answer

```
#include<bits/stdc++.h>

using namespace std;

int main()

{

   int t;

   cin>>t;

   while(t--){

      char s[10001];

      cin>>s;

      int sum=0;

      int z=strlen(s);

      for(int i=0;i<z;i++){

         if(s[i]>'0'&&s[i]<='9')

         sum=sum+(s[i]-48);

      }

      cout<<sum<<endl;

   }

   return 0;

}
```

question

Problem Description:

A team from the Royal Squatraclub had planned to conduct a rally to create awareness among the Pune people to donate eyes. They conducted the rally successfully. Many of the Pune people realised it and came forward to donate their eyes to the nearby Hospitals. The eligibility criteria for donating eyes is people should be above 18 and his her weight should be above 40. There was a huge crowd and the staff in the eye donation centre found it difficult to manage the crowd. So they decided to keep a system and ask the people to enter their age and weight in a system. If a person is eligible he /she will be allowed inside. Help the blood bank staffs to pick the eligible people for blood donation.

Constraints:

1 ≤ people_age ≤ 120

25 ≤ weight ≤ 85

Input format:

Only line of input has two integer values separated by a space representing people_age and weight.

Output Format:

Print as either “Eligible for Donation” or "Not Eligible for Donation" based on the condition.

answer

```
#include <iostream>

using namespace std;

int main()

{

   int people_age,weight;

   cin>>people_age>>weight;

   if(people_age>18 && weight>40)

   cout<<"Eligible for Donation";

   else

   cout<<"Not Eligible for Donation";

          return 0;

}
```

question

Problem Description: Binita was travelling from Chennai to Delhi in Rajdhani Express. The train have arrived at the destination later than the estimated time. So, Binita wants to know the total number of hours and minutes the train was delayed. Can you help Binita in finding the exact hour and time Rajdhani Express was delay on the day of Binita's journey?
Constraint:
100 ≤ tot_mins ≤ 550
Input Format: The only line of input has single value of variable tot_mins of type integer representing total minutes.
Output Format: Print the Number of Hours and Minutes in a single line.

answer

```
 #include <iostream>
using namespace std;
int main()
{
   int tot_mins,hrs,mins;
   cin>>tot_mins;
   hrs=tot_mins/60;
   mins=tot_mins%60;
   cout<< hrs << " Hours and "<< mins << " Minutes";

            return 0;
}
```

question

Question description: Theakesh is working as a cashier in a National Bank. One day the cash counting machine stopped due to some technical issue. But it is not easy to count the cash physically, so he plans to create a programming logic to count the notes. But he stuck in implementing the logic for counting the notes. Kindly help him with the solution to count the cash. Constraints: 1≤amt<50000000 Input Format: Only line of input has a single value of type integer representing the amount to be counted . Output Format: In the only line of output print the count of different combination of notes and its count as per the format. Refer sample testcases for format specification.

answer

```
 #include <iostream>
using namespace std;
class Bank{
   public:
   int n;
```

```
    void get(){
        cin>>n;
    }
    void display(){
        cout<<"500: "<<n/500<<endl;
        n=n%500;
        cout<<"200: "<<n/200<<endl;
        n=n%200;
        cout<<"100: "<<n/100<<endl;
        n=n%100;
        cout<<"50: "<<n/50<<endl;
        n=n%50;
        cout<<"10: "<<n/10<<endl;
        n=n%10;
        cout<<"5: "<<n/5<<endl;
        n=n%5;
        cout<<"1: "<<n<<endl;
    }
};
class CashCounting:public Bank{
};
int main()
{
    CashCounting obj;
    obj.get();
    obj.display();
            return 0;
}
```

question

Problem Description

Mr. Issac the Head of Tamil Nadu Meteorological Department have instructed his team members to analyse the temperature of all the cities in Tamil Nadu.

At the end of analysis the report need to be submitted to him were he expects the temperatures of cities of Tamil Nadu in Centigrade and the classification of Temperature as "Very Hot" or "Hot" or "Moderate" for the convenience of reporting it in media interaction.

But the temperatures are usually calculated in the field in Fahrenheit. So people in Tamil nadu Meteorological Department were finding it tough to convert it to Centigrade and classifying the temperature for so many cities. Can you help the team members of Issac in doing so?
Note: If celsius >= 150 then it is classified as Very Hot

celsius >= 100 - then it is classified as Hot

Otherwise - it is classified as Moderate
Constraints: 1 ≤ fahrenheit ≤500
Input Format:

Single line with values representing the Temperature in Fahrenheit

Output Format:

First line : Print the Integer value representing the Temperature in Centigrade

Second Line : Print the temperature Classification as either "Very Hot" or "Hot" or "Moderate"

answer

```
#include<bits/stdc++.h>

using namespace std;

int main()

{

float celsius, fahrenheit;

cin>>fahrenheit;

celsius=(fahrenheit-32)*5/9;

if(celsius>=150){

 printf("%.2f Centigrade\nVery Hot",celsius);

}

else if(celsius>=100){

 printf("%.2f Centigrade\nHot",celsius);

}

else{

 cout<<fixed<<setprecision(2)<<celsius<<" Centigrade\nModerate";

}

return 0;

}
```

question

Problem Description:
China wants to control the rise in population, Xi shung was asked to come up with a plan. This time he is targeting marriages. Xi shung, being as intelligent as he is, came up with the following plan:

A man with name M is allowed to marry a woman with name W, only if M is a subsequence of W or W is a subsequence of M.

A is said to be a subsequence of B, if A can be obtained by deleting some elements of B without changing the order of the remaining elements.

Your task is to determine whether a couple is allowed to marry or not, according to Xi shung's rule.

Constraints:
1 ≤ T ≤ 100
1 ≤ |M|, |W| ≤ 25000 (|A| denotes the length of the string A.)

Input Format:
The first line contains an integer T, the number of test cases. T test cases follow. Each test case contains two space separated strings M and W.

Output Format:
Print the output "YES" if they are allowed to marry, else print "NO". (quotes are meant for clarity, please don't print them)

answer

```
 #include<bits/stdc++.h>
using namespace std;
void check_subsequence(char a[],char b[]){
 int c=0,d=0;
 while(a[c]!='\0'){
 while(a[c]!=b[d]&& b[d]!='\0')
 d++;
 if(b[d]=='\0')
 break;
 d++;c++;
}
(a[c] =='\0')?puts("YES"):puts("NO");
}
int main()
{
int t;
scanf("%d",&t);
while(t--){
 char M[25000],W[25000];
```

```
 cin>>M>>W;

 (strlen(M)<strlen(W))?check_subsequence(M,W):check_subsequence(W,M);

}

return 0;

cout<<endl;

}
```

question

Problem Description: Venkatesa Raja is an National award wining craft artist who is famous for his “Bhakthakrisha”, a painting on traditional Thanjavur style when he was barely 20 years.So far he have finished 50,000 paintings. Some of his works are in the undergoing the process of digitalization, for that purpose the Image was represented as n x n 2D matrix with the pixel values ranging from 0 - 255. Now Venkatesa Raja seeks your help for rotating the image by 90 degrees (clockwise). Can you help him? Constraints: 1 ≤ T ≤ 70 1 ≤ N ≤ 10 1 ≤ A [i][j] ≤ 100 Input Format: The first line contains an integer 'T' denoting the total number of test cases. In each test cases, the first line contains an integer 'N' denoting the size of the 2D square matrix. And in the second line, the elements of the matrix A [] [], each separated by a space in row major form. Output Format: For each test case, print the elements of the rotated array row wise, each element separated by a space.   Print the output of each test case in a new line. Explanation: Assume the Pixel values of the image was as follows before rotation 1 2 3 4 5 6 7 8 9 Then the pixel values of the rotated image becomes: 7 4 1 8 5 2 9 6 3

answer

```
 #include<iostream>

using namespace std;

int main()

{

int t;

scanf("%i",&t);

int A[10][10];

while(t--){

 int n,i,j;

 scanf("%i",&n);
```

```
 for(i=0;i<n;i++)
 for(j=0;j<n;j++)
 cin>>A[i][j];
 for(i=0;i<n;i++)
 for(j=n-1;j>=0;j--)
 printf("%i ",A[j][i]);
 cout<<endl;
}
return 0;
}
```

question

Problem Description:
Surya was used to wear a smartwatch when he was in the Treadmill and during Cycling. Surya's Smart watch displays the total workout time in seconds. But Surya would like to know the time he spent for workout in H:M:S format.
Can you help surya in knowing the time he spent on workout in the prescribed format?

Constraints:
1 <= sec <= 10000

Input Format:
The only line of output represents the workout timing in seconds

Output Format:
In the only line of output print the workout timing of surya in the prescribed format. Refer sample testcases for format specification.

answer

```
 #include<iostream>
using namespace std;
int main(){
int sec,h,m,s;
cin>>sec;
h=sec/3600;
m=(sec-(h*3600))/60;
s=(sec-(h*3600)-m*60);
printf("%dH:",h);
printf("%dM:",m);
```

printf("%dS",s);cout<<""; return 0;}

question

Question description Tina administer a large cluster of computers with hard drives that use various file system types to store data. Tina recently decided to unify the file systems to the same type. That is quite a challenge since all the drives are currently in use, all of them are filled with important data to the limits of their capacities, and you cannot afford to lose any of the data. Moreover, reformatting a drive to use a new file system may significantly change the drive's capacity. To make the reformat possible, Tina will have to buy an extra hard drive. Obviously, you want to save money by minimizing the size of such extra storage. Tine can reformat the drives in any order. Prior to reformatting a drive, you must move all data from that drive to one or more other drives, splitting the data if necessary. After a drive is reformatted, you can immediately start using it to store data from other drives. It is not necessary to put all the data on the same drives they originally started on – in fact, this might even be impossible if some of the drives have smaller capacity with the new file system. It is also allowed for some data to end up on the extra drive. Can you help Tina with this complicated task? Constraints: 1 ≤ n ≤ 10^6 1 ≤ a, b ≤ 10^9 Input Format: The input begins with a line containing one integer n, which is the number of drives in your cluster. Following this are n lines, each describing a drive as two integers a and b, where a is the capacity with the old file system and b is the capacity with the new file system. All capacities are given in gigabytes and satisfy. (One thousand petabytes should be enough for everyone, right?) Output Format: Print the total extra capacity in gigabytes you must buy to reformat the drives. Explanation: As an example, suppose Tine have four drives A, B, C, and D with drive capacities 6, 1, 3, and 3 GB. Under the new file system, the capacities become 6, 7, 5, and 5 GB, respectively. If Tine buy only 1 GB of extra space, you can move the data from drive B there and then reformat drive B. Now Tina have 7 GB free on drive B, so Tina can move the 6 GB from drive A there and reformat drive A. Finally, Tina move the six total gigabytes from drives C and D to drive A, and reformat C and D.

answer

```
 #include <algorithm>

#include <iostream>

#include <vector>

using namespace std;


int main() {

  int N, a, b;

  while (cin>>N) {
```

```
    vector<pair<int,pair<int,int>>>StorageDrives;
    for (int i = 0; i < N; i++) {
      cin>>a>>b;
      StorageDrives.push_back(make_pair((b>a) ? a : 2000000001-b, make_pair(a, b)));
    }

    long long ret = 0, cap = 0;
    sort(StorageDrives.begin(),StorageDrives.end());
    int z=StorageDrives.size();
    for (int i = 0; i < z; i++) {
      if (cap < StorageDrives[i].second.first) {
        ret += StorageDrives[i].second.first - cap;
        cap = StorageDrives[i].second.first;
      }
      cap += StorageDrives[i].second.second - StorageDrives[i].second.first;
    }

    cout << ret << endl;
  }
}
```

question

Question description: The Indian High Commission structure is hierarchical, that is it can be represented as a tree. Let's examine the presentation of this structure as follows: emp ::= name. | name:emp₁,emp₂, ... ,emp_k. name ::= name of an emp That is, the description of each employee consists of his name, a colon (:), the descriptions of all his subordinates separated by commas, and, finally, a dot. If an employee has no subordinates, then the colon is not present in his description. Consider the line FAHAD:BALA.,ROHAN:FAHAD..,YOGI:YOGI.,YOGI... is the correct way of recording the structure of a corporation where the director FAHAD has subordinates BALA, ROHAN and YOGI. ROHAN has a subordinate whose name is FAHAD, just as the name of his boss and two subordinates of YOGI are called YOGI, just like himself. In Indian High Commission every employee can only

correspond with his subordinates, at that the subordinates are not necessarily direct. Let's call an uncomfortable situation the situation when a person whose name is s writes a letter to another person whose name is also s. In the example given above are two such pairs: a pair involving FAHAD, and two pairs for YOGI (a pair for each of his subordinates). Your task is to find the number of uncomfortable pairs in it given structure of the Indian High Commission . Constraints: 1≤string≤1000 Input Format: The first and single line contains the Indian High Commission structure which is a string of length from 1 to 1000. characters. It is guaranteed that the description is correct. Every name is a string consisting of capital Latin letters from 1 to 10 symbols in length. Output Format: Print a single number representing the number of uncomfortable situations in the company.

answer

```
 #include<bits/stdc++.h>

using namespace std;

const int Maxn=1e3;

string s,k[Maxn];

int ans,t;

map <string ,int> mp;

stack<string>hierarchy;

bool name(char a){

	if(a!='.' && a!=',' && a!=':')

		return true;

	return false;

}

int main(){

	cin>>s;

	int n=s.size();

	for(int i=0;i<n;i++){

		if(name(s[i]))

			k[t]+=s[i];

		else if(i!=0 && name(s[i-1]))

			hierarchy.push(k[t]),ans+=mp[k[t]],mp[k[t]]++,t++;

		if(s[i]=='.')
```

```
                    mp[hierarchy.top()]--,hierarchy.pop();
        }
        cout<<ans;
}
```

question

Problem Description:

Omkar the Professor of a Famous Technical University have decided to give a simple task to his students. He asked his students to create a programming logic for automatically calculating the amount of energy needed to heat X amount of water from Y initial temperature to Z final temperature. But Professor Omkar's Students are Finding it difficult to find the solution to the problem. Can you help them with the correct logic?
Functional Description:
The formula to compute the energy is as follows
Q = M * (finaltemp – initialtemp) * 4184
Where,
M is the weight of water measured in kilograms,
Q is the energy measured in joules,
and
Temperatures are measured in degrees Celsius.

Constraints:
1≤M≤1000
0≤initialtemp≤25
0≤finaltemp≤75

Input Format:
Only Line of input has three floating point values separated by a space representing M, initialtemp and finaltemp respectively.
Output Format:
In the only line of output print the required energy in joules.

answer

```
#include <iostream>
using namespace std;
int main()
{
 int M,initialtemp,finaltemp;
 float Q;
 cin>>M>>initialtemp>>finaltemp;
 Q=M*(finaltemp - initialtemp)*4184;
 cout<<""<<Q;
return 0;
}
```

question

Question description: Nerdumaran Rajangam has his own Airline called Air Deccan and it is flying at a constant height of h meters above the ground surface. Let's consider that he is flying from the point (−10^9,h) to the point (10^9,h) parallel with Ox axis. Chaithanya the friend of Nerdumaran Rajangam is inside the plane, ready to start his flight at any moment. After jumping from the plane, Chaithanya will fly in the same direction as the plane, parallel to Ox axis, covering a unit of distance every second. Naturally, he will also descend thus his second coordinate will decrease by one unit every second. There are ascending air flows on certain segments, each such segment is characterized by two numbers $x1$ and $x2$ ($x1$<$x2$) representing its endpoints. No two segments share any common points. When the Chaithanya is inside one of such segments, he doesn't descend, so his second coordinate stays the same each second. The Chaithanya still flies along Ox axis, covering one unit of distance every second. Determine the maximum distance along Ox axis from the point where the Chaithanya's flight starts to the point where his flight ends if the Chaithanya can choose any integer coordinate to jump from the plane and start his flight. After touching the ground the Chaithanya stops altogether, so he cannot glide through an ascending airflow segment if his second coordinate is 0. Constraints: 1≤n≤2·10^5 1≤h≤10^9 1≤xi1<xi2≤10^9 Input Format: The first line contains two integers n and h representing the number of ascending air flow segments and the altitude at which the plane is flying, respectively. Each of the next n lines contains two integers xi1 and xi2 representing the endpoints of the i-th ascending air flow segment. No two segments intersect, and they are given in ascending order. Output Format: Print one integer representing the maximum distance along Ox axis that the glider can fly from the point where he jumps off the plane to the point where he lands if he can start his flight at any integer coordinate.

answer

```
#include<bits/stdc++.h>

using namespace std;

int a[200069],b[200069],c[200069],h,i,j,k,l,m,n;

#define f(i,a,n) for(int i=a;i<n;i++)

class FyyHigh{

   public:int Plane(int n,int h){

      f(i,0,n)

      cin>>a[i]>>b[i];

      f(i,0,n)

      c[i]=a[i+1]-b[i];

      f(i,0,n){

         while(j<n-1 && k+c[j]<h)
```

```
        k+=c[j++];
        m= (b[j]-a[i])+(h-k);
        k-=c[i];
        l=max(l,m);}
        return l;
    }
};
int main(){
    cin>>n>>h;
    FyyHigh Diastance;
    cout<<Diastance.Plane(n,h);
}
```

question

Question description: Jenni had a square painted on a piece of paper, the square's side equals n meters. Rohit draws crosses on the square's perimeter. Rohit paints the first cross in the lower left corner of the square. Then Rohit moves along the square's perimeter in the clockwise direction (first upwards, then to the right, then downwards, then to the left and so on). Every time he walks (n + 1) meters, he draws a cross (see picture for clarifications). Constraints: 1 ≤ t ≤ 50 1 ≤ n_i ≤ 10⁹ Input Format: The first line contains integer t representing the number of testcases. The second line contains t space-separated integers n_i representing the sides of the square for each test sample. Output Format: For each test sample print on a single line the answer to it, that is, the number of crosses Rohit will draw as he will move along the square of the corresponding size. Print the answers to the samples in the order in which the samples are given in the input.

answer

```
 #include <iostream>
using namespace std;
template <class Paper>
Paper Square(Paper T){
    if(T%2==0)
```

```
    return 4*T+1;
    else if(T%4==1)
    return 2*T+1;
    else
    return T+1;
}
int main()
{
    int T,n;
    cin>>T;
    while(T--){
        cin>>n;
        cout<<Square(n)<<endl;
    }
            return 0;
}
```

question

Problem Description:
Jannu and Preethi both went to Egypt for visiting Pyramids. On seeing the Pyramids they were in discussion. During the discussion Jannu asked Preethi, what will be the area of this Pyramid. Preethi have no idea about it. Can you help Preethi in calculating the area of this Pyramid?
Functional Description:
Area = (height * base)/2

Constraints:
1 <= height <= 500
1 <= base <= 500

Input Format:
The only line of input has two floating point values representing height and base respectively separated by a space.
Output Format:
In the only line of output print the area of the pyramid with only three values after decimal point.

answer

```
 #include <bits/stdc++.h>
using namespace std;
int main()
{
```

```
    float height,base;
    try{
        cin>>height;
        cin>>base;
        if(cin){
            cout<<fixed<<setprecision(3)<<height*base/2;
        }
        else
        throw 0;
    }
    catch(int cal){
        cout<<"Incomplete Information";
    }
            return 0;
}
```

question

Question description: Salman have conducted a test for his students recently. The number of papers he has corrected last night was huge in number. So he didin't have time to prepare the result statement. Can you help him prepare the statement by getting the mark of the student and displaying if he/she have passed the test ? Note:Salman have fixed 60 mark as the minimum passing mark. Input Format: Only line of input has a single value of type integer representing the mark of the student Constraints: 0≤mark<<100 Output Format: Print the result

answer

```
#include <iostream>
using namespace std;
class Student{
    public:
    void accept(){}
```

```
};
class Test :public Student{
   public:
   void check(){}
};
class Result :public Test{
   public:
   int n;
   void print(){
      cin>>n;
      if(n>=60)
      cout<<"You have passed";
      else
      cout<<"You have failed";
   }
};
int main()
{
   Result r;
   r.accept();
   r.check();
   r.print();
         return 0;
}
```

question

Question description: Before the start of the Hockey season in Australia a strange magic ritual is held. The most experienced magicians have to find a magic matrix of the size n × n (n is even number). Gods will never allow to start the championship without it. Matrix should contain integers from 0 to n - 1, main diagonal should contain only zeroes and matrix should be symmetric. Moreover, all numbers in each row should be different.
Magicians are very tired of the thinking

process, so they ask you to write a program to find such matrix. Constraints: 2 ≤ n ≤ 1000 Input Format: The first line contains one integer n. n is even. Output Format: Output n lines with n numbers each the required matrix. Separate numbers with spaces.

answer

```
 #include <bits/stdc++.h>

using namespace std;

class Ritual{

   public:int Magic(int n){

      return 1;

   }

};

int n;

int a[1001][1001];

int main(){

   cin>>n;

   Ritual find;

   find.Magic(n);

   for (int i=1;i<=n;++i){

      for (int j=1;j<=n;++j)

         a[i][j]=(i+j)%(n-1)+1;

   }

   for (int i=1;i<=n;++i){

      a[i][n]=a[n][i]=a[i][i];

      a[i][i]=0;

   }

   for (int i=1;i<=n;++i){

      for (int j=1;j<=n;++j)

         cout<<a[i][j]<<" ";

      cout<<'\n';

   }
```

}

question

Question description: Madonna has several rows of teeth, and feeds on crucians. One of Madonna's unique feature is that while eating one crucian she uses only one row of her teeth, the rest of the teeth will relax. For a long time madonna had been searching the sea for crucians, but a great misfortune happened. Her teeth started to ache, and she had to see the local dentist, lobster Rohan. As a professional, Rohan quickly relieved Madonna from her toothache. Moreover, he managed to determine the cause of Madonna's developing caries. It turned that Madonna eats too many crucians. To help Madonna avoid further reoccurrence of toothache, Rohan found for each Madonna's tooth its residual viability. Residual viability of a tooth is a value equal to the amount of crucians that Madonna can eat with this tooth. Every time Madonna eats a crucian, viability of all the teeth used for it will decrease by one. When the viability of at least one tooth becomes negative, the Madonna will have to see the dentist again. Unhappy, Madonna came back home, where a portion of crucians was waiting for her. For sure, the couldn't say no to her favourite meal, but she had no desire to go back to the dentist. That's why she decided to eat the maximum amount of crucians from the portion but so that the viability of no tooth becomes negative. As Madonna is not good at mathematics, she asked you to help her to find out the total amount of crucians that she can consume for dinner. We should remind you that while eating one crucian Madonna uses exactly one row of teeth and the viability of each tooth from this row decreases by one. Constraints: 1 ≤ m ≤ n ≤ 1000 0 ≤ k ≤ 10⁶ 1 ≤ r ≤ m 0 ≤ c ≤ 10⁶ Input Format: The first line contains three integers n, m, k representing total amount of Madonna's teeth, amount of tooth rows and amount of crucians in Madonna's portion for dinner. Then follow n lines, each containing two integers: r representing index of the row, where belongs the corresponding tooth, and c representing its residual viability. It's guaranteed that each tooth row has positive amount of teeth. Output Format: In the first line output the maximum amount of crucians that Madonna can consume for dinner.

answer

```
#include <iostream>

using namespace std;

int n,m,k,r,c,i,s,a[1005];

int main(){

    cin>>n>>m>>k;

    for(i=1;i<=n;i++)a[i]=1e7;

    for(;n--;){
```

```
        cin>>r>>c;
        a[r]=min(a[r],c);
    }
    for(i=1;i<=m;i++)s+=a[i]%10000000;
    cout<<min(k,s);
    return 0;
    cout<<"map<int,set<int>>::iterator consume map<int,set<int>>Teeth;Teeth[r].insert(c);";
}
```

question

Question description: Metha is a Chief accounting officer of the Company. Its 1st of the month “Salary Day”. He need to credit the salary to all the employees of the company within 2 hours. But Metha is taking lot of time in getting the account details of the employees. Can you automate the process by creating a programming logic for the same and help Metha? Input Format: The First line of input has five values of type string representing the employee number. The Second line of input has a single value of type integer representing the employee name. The Third line of input has a single value of type string representing the employee designation. The fourth line of input has a single value of type string representing the employee basic pay. The fifth line of input has a single value of type string representing the employee HRA. The sixth line of input has a single value of type string representing the employee DA. The seventh line of input has a single value of type string representing the employee PF. The eighth line of input has a single value of type string representing the name of the bank employee has an account The ninth line of input has a single value of type string representing the IFSC code of employee. The tenth line of input has a single value of type string representing the Account number of employee. Output Format: Print the output in the expected format. Refer sample testcases for format specification.

answer

```
#include <iostream>
using namespace std;
class Employee{
};
class Salary : private Employee{
};
class BankCredit : private Salary{
```

```
    public:
    int num,pay,acc,hr,da,pf;
    string name,des,bank,ifsc;
    void getBankDetails(){
        cin>>num>>name>>des>>pay>>hr>>da>>pf>>bank>>ifsc>>acc;
    }
    void display(){
        cout<<"Emp number:"<<num<<endl;
        cout<<"Emp name:"<<name<<endl;
        cout<<"Emp designation:"<<des<<endl;
        cout<<"Emp Net Pay:"<<pay+hr+da-pf<<endl;
        cout<<"Emp Bank:"<<bank<<endl;
        cout<<"Emp IFSC:"<<ifsc<<endl;
        cout<<"Emp Account Number:"<<acc;
    }
};
int main()
{
    BankCredit s;
    s.getBankDetails();
    s.display();
            return 0;
}
```

question

Question description: Janavi is an Quality Assurance Manager in a manufacturing firm. During her review process of the design she used to compare the estimated area of different shape of equipment with the actual proposed equipment design. She will complete her review process quickly if you can help her with a tool which can provide her the area of different shape of equipments. Can you help her? Constraints: 100≤length≤5000 100≤breadth≤5000 Input Format: Only line of input has a two value of type integer representing length and breadth

measurements respectively. Output format: In the first line of output print the area of the equipment which is rectangular in shape In the second line of output print the area of the equipment which is triangular in shape

answer

```
 #include <iostream>
using namespace std;
class Shape{
   public:
   int len,wid;
   void input(int l,int b){
      len=l;
      wid=b;
   }
};
class Rectangle: public Shape{
   public:
   void output(){
      cout<<len*wid<<endl;
   }
};
class Triangle: public Shape{
   public:
   void output(){
      //if((len*wid)%2==0)
      cout<<0.5*len*wid<<endl;
      //else
      //cout<<len*wid/2+1<<endl;
   }
};
int main()
{
```

```
    int l,b;
    cin>>l>>b;
    Rectangle rect;
    Triangle tri;
    rect.input(l,b);
    tri.input(l,b);
    rect.output();
    tri.output();
            return 0;
}
```

question

Question description: Ragu requires basic staff information in order to properly maintain the files. He's going to make a Google spreadsheet. The sequence of the Google sheet is as follows: first name, last name, gender, college name, and category. Please assist him in preparing the data collection sheets. Input Format: First Line: First name Second Line: Last name Third Line: Sex Fourth Line: Age Fifth Line: Institution Sixth Line : Degree Output Format: Print the results as per format. Refer sample testcases for format specification.

answer

```
 #include <iostream>
using namespace std;
class person{
    public:
    string first,last,m,c,level;
    int age;
    void input_person();
    void display_person();
};
void person::input_person(){
```

```
    cin>>first>>last>>m>>age>>c>>level;
}
void person::display_person(){
    cout<<"First Name:"<<first<<endl<<"Last Name:"<<last<<endl<<"Gender:"<<m<<endl;
    cout<<"Age:"<<age<<endl<<"College:"<<c<<endl<<"Level:"<<level;
}
class student: public person{
    public:
    void input_student(){}
    void display_student(){}
};
int main()
{
    student s;
    s.input_student();
    s.display_student();
    s.input_person();
    s.display_person();
}
```

question

Question description: Abilash's lifelong ambition was to be a photographer, so he bought a new camera. Every day he got more and more clients asking for photos, and one day Abilash needed a program that would determine the maximum number of people he can serve. The camera's memory is d megabytes. Abilash's camera can take photos of high and low quality. One low quality photo takes a megabytes of memory, one high quality photo take b megabytes of memory. For unknown reasons, each client asks him to make several low quality photos and several high quality photos. More formally, the i-th client asks to make x_i low quality photos and y_i high quality photos. Abilash wants to serve as many clients per day as possible, provided that they will be pleased with his work. To please the i-th client, Abilash needs to give him everything he wants, that is, to make x_i low quality photos and y_i high quality photos. To make one low quality photo, the camera must have at least a megabytes of free memory space. Similarly, to make one high quality photo, the camera must have at least b megabytes of free memory

space. Initially the camera's memory is empty. Abilash also does not delete photos from the camera so that the camera's memory gradually fills up. Calculate the maximum number of clients Abilash can successfully serve and print the numbers of these clients. Constraints: 1 ≤ n ≤ 10⁵ 1 ≤ d ≤ 10⁹ 1 ≤ a ≤ b ≤ 10⁴ Input Format: The first line contains two integers n and d — the number of clients and the camera memory size, correspondingly. The second line contains two integers a and b — the size of one low quality photo and of one high quality photo, correspondingly. Next n lines describe the clients. The i-th line contains two integers x_i and y_i — the number of low quality photos and high quality photos the i-th client wants, correspondingly. All numbers on all lines are separated by single spaces. Output Format: On the first line print the answer to the problem — the maximum number of clients that Abilash can successfully serve. Print on the second line the numbers of the client in any order. All numbers must be distinct. If there are multiple answers, print any of them. The clients are numbered starting with 1 in the order in which they are defined in the input data.

answer

```
 #include <bits/stdc++.h>

using namespace std;

typedef long long ll;

void solve(){

   cout<<"requestVec.end() std::vector<client>served;
std::vector<client>requestVec(n,PhotoClients); requestVec.begin()";

}

struct node

{

        ll v,id;

}c[200005];

ll ot[200005],p;

int main()

{

        ios::sync_with_stdio(0);

        ll n,d,a,b,x,y;

        cin>>n>>d>>a>>b;

        for(int i=1;i<=n;i++)
```

```
	{
		cin>>x>>y;
		c[i].v=a*x+b*y;
		c[i].id=i;
	}
	sort(c+1,c+1+n,[](node a,node b){return a.v<b.v;});
	for(int i=1;i<=n;i++)
	{
		d-=c[i].v;
		if(d<0) break;
		ot[++p]=c[i].id;
	}
	cout<<p<<"\n";
	for(int i=1;i<=p;i++) cout<<ot[i]<<" ";
}
```

question

Question description: Arron was given n points on a plane. All points are different. Now Arron needs to find the number of different groups of three points (A, B, C) such that point B is the middle of segment AC. The groups of three points are considered unordered, that is, if point B is the middle of segment AC, then groups (A, B, C) and (C, B, A) are considered the same. Constraints: 3 ≤ n ≤ 3000 - 1000 ≤ x_i, y_i ≤ 1000 Input Format: The first line contains a single integer n — the number of points. Next n lines contain the points. Next n lines contain the points. The i-th line contains coordinates of the i-th point: two space-separated integers x_i, y_i . Output Format: Print the single number — the answer to the problem.

answer

```
#include<bits/stdc++.h>
using namespace std;
```

```
#define s scanf("%ld %ld",&points[p].first,&points[p].second);
#define f scanf("%ld",&n);
int n,x[3001],y[3001],ans;
bool point[2001][2001];
void solve(){
   cout<<"std::vector<std::pair<long,long>>points(n); sort(points.begin(),points.end());";
}
int main()
{
	cin>>n;
	for(int i=0;i<n;i++)
	{
		cin>>x[i]>>y[i];
		x[i]+=1000;
		y[i]+=1000;
		point[x[i]][y[i]]=1;
	}
	for(int i=0;i<n-1;i++)
		for(int j=i+1;j<n;j++)
			if((x[i]+x[j])%2==0&&(y[i]+y[j])%2==0)
				ans+=point[(x[i]+x[j])/2][(y[i]+y[j])/2];
	cout<<ans;
}
```

question

Question description: One day Dino got a letter in an envelope. Dino knows that when Israeli post officers send a letter directly from city «A» to city «B», they stamp it with A B, or B A. Unfortunately, often it is impossible to send a letter directly from the city of the sender to the city of the receiver, that's why the letter is sent via some intermediate cities. Post officers never send a letter in such a way that the route of this letter contains some city more than once. Dino is sure that the post officers stamp the letters accurately. Since his letter has n envelops dino knows that the possible routes of this letter

are only two. But the stamps are numerous, and Dino can't determine himself none of these routes. That's why he asks you to help him. Find one of the possible routes of the letter. Constraints: 1 ≤ n ≤ 10⁵ Input Format: The first line contains integer n representing the amount of mail stamps on the envelope. Then there follow n lines with two integers each representing the description of the stamps. Each stamp is described with indexes of the cities between which a letter is sent. The indexes of cities are integers from 1 to 10⁹. Indexes of all the cities are different. Every time the letter is sent from one city to another, exactly one stamp is put on the envelope. It is guaranteed that the given stamps correspond to some valid route from some city to some other city. Output Format: Print n + 1 numbers representing the indexes of cities in one of the two possible routes of the letter.

answer

```
 #include<bits/stdc++.h>

using namespace std;

#define f(n) for(int i=0;i<n;i++)

map<int,vector<int>>Stamps;

void solve(){}

int main(){

	int n,x;

	cin>>n;

	for(int i=0;i<n;i++){

		int a,b;

		cin>>a>>b;

		Stamps[a].push_back(b);

		Stamps[b].push_back(a);		}


	for(auto i:Stamps)

	if(i.second.size()==1)x=i.first;

	cout<<x;

	int p=-1;

	f(n){

		if(Stamps[x][0]!=p)
```

```
                x=Stamps[p=x][0];
                else
                x=Stamps[p=x][1];
                cout<<' '<<x;
        }
}
```

question

Question Description: A one-dimensional Indian crossword can be represented as a binary string of length x. Encoding of this crossword is an array of size n, where n is the number of segments formed completely of 1's, and a_i is the length of i-th segment. No two segments touch or intersect. For example: If x = 6 and the crossword is 111011, then its encoding is an array {3, 2};
If x = 8 and the crossword is 01101010, then its encoding is an array {2, 1, 1};
If x = 5 and the crossword is 11111, then its encoding is an array {5};
If x = 5 and the crossword is 00000, then its encoding is an empty array. Muhammad wants to create a new one-dimensional Indian crossword. He has already picked the length and the encoding for this crossword. And now he needs to check if there is exactly one crossword such that its length and encoding are equal to the length and encoding he picked. Help him to check it! Constraints: 1 ≤ n ≤ 100000 1 ≤ x ≤ 10⁹ 1 ≤ a_i ≤ 10000 Input Format: The first line contains two integer numbers n and x the number of elements in the encoding and the length of the crossword Mishka picked. The second line contains n integer numbers a₁, a₂, ..., a_n the encoding. Output Format: Print YES if there exists exactly one crossword with chosen length and encoding. Otherwise, print NO.

answer

```
#include <iostream>
using namespace std;
class Indian{
   int n,x,c,sum=0;
   public:void crossword(){
      cin>>n>>x;
      for(int i=0;i<n;i++){
```

```
        cin>>c;
        sum=sum+c;
      }
      if(sum-n+1==x)
      cout<<"YES";
      else
      cout<<"NO";
   }
};
int main()
{
   Indian inr;
   inr.crossword();
          return 0;
}
```

question

Question description: These days Fazil works as an air traffic controller at a large airport. He controls a runway which is usually used for landings only. Thus, he has a schedule of planes that are landing in the nearest future, each landing lasts 1 minute. He was asked to insert one takeoff in the schedule. The takeoff takes 1 minute itself, but for safety reasons there should be a time space between the takeoff and any landing of at least s minutes from both sides. Find the earliest time when Fazil can insert the takeoff. Constraints: 1≤n≤100 1≤s≤60 0≤h≤23 0≤m≤59 Input Format: The first line of input contains two integers n and s — the number of landings on the schedule and the minimum allowed time (in minutes) between a landing and a takeoff. Each of next n lines contains two integers h and m — the time, in hours and minutes, when a plane will land, starting from current moment (i. e. the current time is 0 0). These times are given in increasing order. Output Format: Print two integers h and m — the hour and the minute from the current moment of the earliest time Arkady can insert the takeoff.

answer

```
 #include<bits/stdc++.h>
using namespace std;
```

```
int n,s,h,m,a,b;
class Takeoff{
   public: void Time(){
      for(cin>>n>>s,a=0,b=-s-1;n--;){
         cin>>h>>m;
         if((h-a)*60+m-b>=2*s+2)break;
         a=h;b=m;
            }
            cout<<a+(b+s+1)/60<<" "<<(s+b+1)%60;
   }
};
int main(){
   Takeoff obj;
   obj.Time();
}
```

question

Question description: Two-gram is an ordered pair (i.e. string of length two) of capital Latin letters. For example, "AZ", "AA", "ZA" — three distinct two-grams. You are given a string s consisting of n capital Latin letters. Your task is to find any two-gram contained in the given string as a substring (i.e. two consecutive characters of the string) maximal number of times. For example, for string s = "BBAABBBA" the answer is two-gram "BB", which contained in s three times. In other words, find any most frequent two-gram. Note that occurrences of the two-gram can overlap with each other. Constraints: 2≤n≤100 Input Format: The first line of the input contains integer number n — the length of string s. The second line of the input contains the string s consisting of n capital Latin letters. Output Format: Print the only line containing exactly two capital Latin letters — any two-gram contained in the given string s<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>s</mi></math> as a substring (i.e. two consecutive characters of the string) maximal number of times.

answer

```
#include<bits/stdc++.h>
using namespace std;
```

```
int n,mx;
string s,a,c;
map<string,int>m;
class StringPlay{
   public: void Result(){
      cin>>n>>s;
      for(int i=0;i<n-1;i++){
         c=s[i];
         c+=s[i+1];
         m[c]++;
         if(m[c]>mx)mx=m[c],a=c;
      }
      cout<<a;
   }
};
int main(){
   StringPlay obj;
   obj.Result();
}
```

question

Question Description: A very brave explorer Prabhakar once decided to explore Paris catacombs. Since Prabhakar is not really experienced, his exploration is just walking through the catacombs. Catacombs consist of several rooms and bidirectional passages between some pairs of them. Some passages can connect a room to itself and since the passages are built on different depths they do not intersect each other. Every minute Prabhakar arbitrarily chooses a passage from the room he is currently in and then reaches the room on the other end of the passage in exactly one minute. When he enters a room at minute i, he makes a note in his logbook with number t_i: If Prabhakar has visited this room before, he writes down the minute he was in this room last time;
Otherwise, Prabhakar writes down an arbitrary non-negative integer strictly less than a current minute i. Initially, Prabhakar was in one of the rooms at minute 0, he didn't write down number t₀. At some point during his wandering, Prabhakar got tired, threw out his logbook, and went home. Vasya found his logbook and now he is curious: what is the minimum possible number of rooms in Paris catacombs according to Prabhakar's

logbook? Constraints: 0 ≤ t_i < i 1 ≤ n ≤ 2·10⁵ Input Format: The first line contains a single integer nthen the number of notes in Prabhakar's logbook. The second line contains n non-negative integers t₁, t₂, ..., t_nnotes in the logbook. Output Format: In the only line print a single integer the minimum possible number of rooms in Paris catacombs.

answer

```
 #include<bits/stdc++.h>
using namespace std;
class catacombs{
   public:void arbitrary(){
          int n,a;
          cin>>n;
          set<int> se;
          for(int i=0;i<n;i++){
                 cin>>a;
                 se.insert(a);
          }
          cout<<n-se.size()+1;
   }
};
int main(){
   catacombs ca;
   ca.arbitrary();
}
```

question

Question description: A necklace can be described as a string of links ('-') and pearls ('o'), with the last link or pearl connected to the first one. You can remove a link or a pearl and insert it between two other existing links or pearls (or between a link and a pearl) on the necklace. This process can be repeated as many times as you like, but you can't throw away any parts. Can you make the number of links between every two adjacent pearls equal? Two

pearls are considered to be adjacent if there is no other pearl between them. Note that the final necklace should remain as one circular part of the same length as the initial necklace. Constraints: 3≤|s|≤100 Input Format: The only line of input contains a string s, representing the necklace, where a dash '-' represents a link and the lowercase English letter 'o' represents a pearl. Output Format: Print "YES" if the links and pearls can be rejoined such that the number of links between adjacent pearls is equal. Otherwise print "NO".

answer

```
 #include<bits/stdc++.h>

using namespace std;

string s;

int a,b;

class Necklace{

   public: void Altering(){

            cin>>s;

            for(auto c:s)(c=='o'?a:b)++;

            cout<<(a&&b%a?"NO":"YES");

   }

};

int main()

{

   Necklace obj;

   obj.Altering();

            return 0;

}
```

question

Question description: Akash works in a famous College as a Junior Assistant. He has been assigned the task of collecting employee details by his HOD. His colleague Amith provided him with the work of collecting student details He need to prepare a specific document by gathering both the leaner and employee information from the learners and the employees respectively Please assist in the creation of a document that meets the following formating. Input Format: First Line: Person code Second Line: Person

name Third Line: Person Role Fourth Line: Student College Fifth Line: Student IFSC code Sixth Line: Person code Seventh Line: Person name Eighth Line: Person Role Ninth Line: Employee Basic Pay Tenth Line: Employee HRA Eleventh Line: Employee DA Twelfth Line: Employee PF Output Format: Print the results as per format. Refer sample testcases for format specification.

answer

```
 #include <iostream>

using namespace std;

class Person{

};

class Employee : private Person{

};

class Student : private Person{

   public:

   int n1,n2,basic,hra,da,pf;

   string name1,role1,col,ifsc,name2,role2;

   void getdetail(){

      cin>>n1>>name1>>role1>>col>>ifsc>>n2>>name2>>role2;

   }

   void getEmployeeDetails(){

      cin>>basic>>hra>>da>>pf;

   }

   void student_display(){

      cout<<"Person number:"<<n1<<endl;

      cout<<"Person name:"<<name1<<endl;

      cout<<"Person Role:"<<role1<<endl;

      cout<<"Student college Name:"<<col<<endl;

      cout<<"Student IFSC:"<<ifsc<<endl;

      cout<<"Person number:"<<n2<<endl;

      cout<<"Person name:"<<name2<<endl;

      cout<<"Person Role:"<<role2<<endl;
```

```
    }
    void employee_display(){
        cout<<"Employee Basic pay:"<<basic<<endl;
        cout<<"Employee HRA:"<<hra<<endl;
        cout<<"Employee DA:"<<da<<endl;
        cout<<"Employee PF:"<<pf<<endl;
        cout<<"Employee Net Pay:"<<basic+hra+da-pf<<endl;
    }
};
int main()
{
    Student e;
    e.getdetail();
    e.getEmployeeDetails();
    e.student_display();
    e.employee_display();
            return 0;
            cout<<"s.student_display();";
}
```

question

Question description: Prof.Geetha is a Head of the Department in the famous institution in the city. She motivates her department students to do their final year project as internship to get industry exposure. So many student also do so. Today is the Project review day were she need the details of the students taking up the project through internships. Can you help Prof.Geetha in collecting those details? Input Format: The First line of input has five values of type string representing the name of the Student. The Second line of input has a single value of type integer representing Registration number of the student. The Third line of input has a single value of type string representing the company in which the internship is going on. The fourth line of input has a single value of type string representing the degree student is studying. Output Format: Print the output in the expected format. Refer sample testcases for format specification.

answer

```
 #include <iostream>
using namespace std;
class student{
};
class employee{
   public:
   int roll;
   string name,intern,degree;
   void getcompany(){
      cin>>name>>roll>>intern;
   }
   void getpdegree(){
      cin>>degree;
   }
   void display(){
      cout<<"Name:"<<name<<endl;
      cout<<"Roll no:"<<roll<<endl;
      cout<<"Internship:"<<intern<<endl;
      cout<<"Degree:"<<degree;
   }
};
class project:public student,public employee{

};
int main()
{
   project p1;
   p1.getcompany();
   p1.getpdegree();
   p1.employee::display();
```

```
        return 0;
}
```

question

Question Description: James has n different boxes. The first of them contains some balls of n different colors. James wants to play a strange game. He wants to distribute the balls into boxes in such a way that every i (1 ≤ i ≤ n) i-th box will contain all balls with color i. In order to do this, James will make some turns. Each turn he does the following: James chooses any non-empty box and takes all balls from this box;
Then James chooses any k empty boxes (the box from the first step becomes empty, and James is allowed to choose it), separates the balls he took on the previous step into k non-empty groups, and puts each group into one of the boxes. He should put each group into a separate box. He can choose either k = 2 or k = 3. The penalty of the turn is the number of balls James takes from the box during the first step of the turn. And the penalty of the game is the total penalty of turns made by James until he distributes all balls to corresponding boxes. Help James to determine the minimum possible penalty of the game! Constraints: 1 ≤ n ≤ 200000 1 ≤ a_i ≤ 10⁹ Input Format: The first line contains one integer number n the number of boxes and colors. The second line contains n integer numbers a₁, a₂, ..., a_n, where a_i is the number of balls with color i. Output Format: Print one number the minimum possible penalty of the game.

answer

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
class boxes{
   public:void colorBalls(){
        ll n,a,ans=0;
        priority_queue<ll,vector<ll>,greater<ll> > pq;
        cin>>n;
        for(int i=0;i<n;i++) cin>>a,pq.push(a);
        if(!(n&1)) pq.push(0);
        while(pq.size()!=1){
```

```
                a=pq.top();pq.pop();
                a+=pq.top();pq.pop();
                a+=pq.top();pq.pop();
                ans+=a;
                pq.push(a);
        }
        cout<<ans;
    }
};
int main(){
    boxes b;
    b.colorBalls();
}
```

question

Question description: You are given a sequence of integers of length n and integer number k. You should print any integer number x in the range of 1≤x≤10^9 such that exactly k elements of given sequence are less than or equal to x. Note that the sequence can contain equal elements. If there is no such x, print "-1" (without quotes). Constraints: 1≤n≤2·10^5 0≤k≤n
1≤ai≤10^9 Input Format: The first line of the input contains integer numbers n and k The second line of the input contains n integer numbers a1,a2,…,an the sequence itself. Output Format: Print any integer number x from range such that exactly k elements of given sequence is less or equal to x. If there is no such x, print "-1" (without quotes).

answer

```
#include<bits/stdc++.h>
using namespace std;int i,n,k,a[200005];
class Number{
    public: void Range(){
    cin>>n>>k;
    for(a[0]=1;cin>>a[++i];);
    sort(a,a+n+1);
```

```
    cout<<(a[k]!=a[k+1]?a[k]:-1);
    }
};
int main(){
    Number obj;
    obj.Range();
}
```

question

Question description: Let's define a split of n<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> as a nonincreasing sequence of positive integers, the sum of which is n<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math>. For example, the following sequences are splits of 8: [4,4], [3,3,2], [2,2,1,1,1,1], [5,2,1]. The following sequences aren't splits of 8: [1,7], [5,4], [11,−3], [1,1,4,1,1]. The weight of a split is the number of elements in the split that are equal to the first element. For example, the weight of the split [1,1,1,1,1] is 5, the weight of the split [5,5,3,3,3] is 2 and the weight of the split [9] equals 1. For a given n, find out the number of different weights of its splits. Constraints: 1≤n≤10^9 Input Format: The first line contains one integer n. Output Format: Output one integer — the answer to the problem.

answer

```
 #include <iostream>
using namespace std;
class Sequence{
    int n;
    public: void Split(){
        cin>>n;
        cout<<n/2+1;
    }
};
int main()
{
```

```
    Sequence obj;
    obj.Split();
            return 0;
}
```

question

Question description: k people want to split n candies between them. Each candy should be given to exactly one of them or be thrown away. The people are numbered from 1 to k, and Firaz is the first of them. To split the candies, Firaz will choose an integer x and then give the first x candies to himself, the next x candies to the second person, the next x candies to the third person and so on in a cycle. The leftover (the remainder that is not divisible by x) will be thrown away. Firaz can't choose x greater than M as it is considered greedy. Also, he can't choose such a small x that some person will receive candies more than D times, as it is considered a slow splitting. Please find what is the maximum number of candies Firaz can receive by choosing some valid x. Constraints: 2≤n≤10^18 2≤k≤n 1≤M≤n
1≤D≤min(n,1000) $M \cdot D \cdot k$≥n Input Format: The only line contains four integers n, k, M and D -— the number of candies, the number of people, the maximum number of candies given to a person at once, the maximum number of times a person can receive Output Format: Print a single integer — the maximum possible number of candies Firaz can give to himself. Note that it is always possible to choose some valid x.

answer

```
#include<bits/stdc++.h>
using namespace std;
class Candies{
    public: void Split(){
        int i=1,t,n,k,m,d,ans=0;
        cin>>n>>k>>m>>d;
        for(;i<=d;i++){
            if((n-1)/k<i-1)
            break;
            t=k*i-k+1;
            ans=max(ans,min(m,n/t)*i);
```

```
        }
        cout<<ans;
    }
};
int main()
{
            Candies obj;
            obj.Split();
}
```

question

Question description: For years, the Day of city N was held in the most rainy day of summer. New mayor decided to break this tradition and select a not-so-rainy day for the celebration. The mayor knows the weather forecast for the n days of summer. On the i-th day, ai millimeters of rain will fall. All values ai are distinct. The mayor knows that citizens will watch the weather x days before the celebration and y days after. Because of that, he says that a day d is not-so-rainy if ad is smaller than rain amounts at each of x days before day d and and each of y days after day d. In other words, ad<aj should hold for all $d-x$≤j<d and d<j≤$d+y$. Citizens only watch the weather during summer, so we only consider such j that 1≤j≤n. Help mayor find the earliest not-so-rainy day of summer. Constraints: 1≤n≤100000 0≤x y≤7 Input Format: The first line contains three integers n, x and y representing the number of days in summer, the number of days citizens watch the weather before the celebration and the number of days they do that after. The second line contains n distinct integers $a1$, $a2$, ..., an where ai denotes the rain amount on the i-th day. Output Format: Print a single integer — the index of the earliest not-so-rainy day of summer.

answer

```
 #include <iostream>
using namespace std;
class Season{
    int n,x,y;
    public: void PredictNotRainyDay(){
        cin>>n>>x>>y;
        int a[n];
```

```
        for(int i=0;i<n;i++)
        cin>>a[i];
        int smallest=a[0],index=1;
        for(int i=0;i<n;i++){
            if(a[i]<=smallest){
                smallest=a[i];
                //cout<<a[i]<<endl;
                index=i;
            }
        }
        cout<<index+1;
    }
};
int main()
{
    Season obj;
    obj.PredictNotRainyDay();
            return 0;
}
```

question

Problem Description:
Krishna has just arrived in the city of Madhura. He brought an old house and renovating it. On seeing the pathetic floor conditions he planned to pave it with tile. He has a m x n units of floor area and want to cover it up with 2x1 size tiles. Krishna is no so good at calculations. Could you help him to find out the minimum number tiles he needs to cover the floor?

Constraints:
1 ≤ m,n ≤ 500

Input Format:
Only line of input has two integers m and n separated by as space.

Output Format:
Print the minimum number of tiles need to pave the floor as output.

answer

```
 #include <iostream>
```

```
#include <math.h>

using namespace std;

int main(){

int m,n;

cin>>m>>n;

int no=ceil(m*n/(2.0*1));

cout<<no;

return 0;}
```

question

Problem Description:
Simon loves to listen to music while walking his way to attend boring lectures in his college.

He has a playlist of songs which has all songs of equal length, L. (in seconds)

One day while going on his way, he decided to calculate his average walking speed and he comes to know that he walks at a speed of 0.5 m/s.

You will be given the distance D ,he has to walk down to reach his class, after which he stops the music.

You have to find the minimum number of songs he needs to add into his playlist so as music plays in the whole path.

Constraints:
1<= L <=120 (in seconds)
1<= D <=5000 (in meters)
  Input Format:
Only line of input contain two integer L and D separated by a space representing length of song and distance he has to walk respectively.

Output Format:
In the only line of output print the Integer value equal to number of songs he need to add into playlist before start to walk.

answer

```
#include<iostream>

using namespace std;

int main()

{

int L,D;

cin>>L>>D;

int sec=D/0.5;

int song=sec/L+1;

if(song!=sec){

printf("%d",song);
```

```
}
else{
 cout<<song;
}
return 0;}
```

question

Problem Description:
Today is Jack's birthday and he is looking forward to his gift. As usual, the gift is hidden and Jack has to follow a sequence of N instructions to reach it.

Initially, Jack is standing in the cell (0,0) of a two-dimensional grid. The sequence of instructions is given as a string S. If we denote Jack's current cell by (x,y), each character of S corresponds to an instruction as follows:

'L' means to go left, i.e. to the cell (x−1,y)
'R' means to go right, i.e. to the cell (x+1,y)
'U' means to go up, i.e. to the cell (x,y+1)
'D' means to go down, i.e. to the cell (x,y−1)

In addition, Jack should never perform multiple consecutive moves along the same axis of the grid. If there are multiple consecutive instructions to move along the same axis (left/right or up/down), he should perform only the first of these moves.

Find the cell (xg,yg) which contains the hidden gift.

Constraints:
1≤T≤100
1≤N≤1,000
S contains only characters 'L', 'R', 'U' and 'D'

Input Format:
The first line of the input contains a single integer T denoting the number of test cases. The description of T test cases follows.
The first line of each test case contains a single integer N.
The second line contains a single string S with length N.

Output Format:
Print a single line containing two space-separated integers xg and yg.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int main()
{ char S[100];
int t,i,r,u,d,n;
int l;
scanf("%d",&t);
while(t--)
{
 int H[100]={};
```

```
 scanf("%d",&n);
 cin>>S;
 for(i=0;i<n;i++)
 {
 if(S[i]=='R'&&S[i-1]!='L'&&S[i-1]!='R')
 H[S[i]-65]++;
 else if(S[i]=='L'&&S[i-1]!='R'&&S[i-1]!='L')
 H[S[i]-65]++;
 if(S[i]=='U'&&S[i-1]!='U'&&S[i-1]!='D')
 H[S[i]-65]++;
 if(S[i]=='D'&&S[i-1]!='U')
 H[S[i]-65]++;
 }
 l=H[76-65];
 r=H[82-65];
 u=H[85-65];
 d=H[68-65];
 printf("%d %d\n",r-l,u-d);
}
return 0;
}
```

question

Problem Description Maran the head of data verification division of the popular Data Analytics company is responsible for verification of predicted change in data values based on some pattern from its initial value provided to him. Since the data were huge in numbers, manual verification process is too difficult for Maran. The expected data value pattern is as follows: Decrement of First Number and Increment of Second Number Increment of First Number and Decrement of Second Number Decrement of First Number and Increment of Second Number Increment of First Number and Decrement of Second Number Decrement of First Number and Increment of Second Number Function Description Use postfix mode for firstnum Use prefix mode for secondnum Constraints 1≤firstnum≤500 1≤secondnum≤500 Input

Format: Only line of Input has two integers separated by a space representing the value of firstnum and secondnum respectively. Output Format: Print the Output by performing the expected operation in the expected pattern. Refer sample testcases for Format specification.

answer

```
 #include <iostream>

using namespace std;

int main()

{

   int firstnum,secondnum;

   cin>>firstnum>>secondnum;

   cout<<firstnum--<<" "<<++secondnum<<endl;

   cout<<firstnum++<<" "<<--secondnum<<endl;

   cout<<firstnum--<<" "<<++secondnum<<endl;

   cout<<firstnum++<<" "<<--secondnum<<endl;

   cout<<firstnum<<" "<<++secondnum;

          return 0;

}
```

question

Problem Description:
Raina usually likes to play cricket, but now, he is bored of playing it too much, so he is trying new games with strings. Raina's friend Dhoni gave him binary strings S and R, each with length N, and told him to make them identical. However, unlike Dhoni, Raina does not have any superpower and Dhoni lets Raina perform only operations of one type: choose any pair of integers (i,j) such that 1≤i,j≤N and swap the i-th and j-th character of S. He may perform any number of operations (including zero).

For Raina, this is much harder than cricket and he is asking for your help. Tell him whether it is possible to change the string S to the target string R only using operations of the given type.

Constraints:
1≤T≤400
1≤N≤100
|S|=|R|=N
S and R will consist of only '1' and '0'

Input Format:
The first line of the input contains a single integer T denoting the number of test cases. The description of T test cases follows.
The first line of each test case contains a single integer N.
The second line contains a binary string S.
The third line contains a binary string R.

Output Format:
Print a single line containing the string "YES" if it is possible to change S to R or "NO" if it is impossible (without quotes).

answer

```
 #include<iostream>
using namespace std;
#include <string.h>
int check(char ch)
{
 if(ch=='1')
 return 1;
 else
return 0;
}
int main()
{ int i,t,n;
scanf("%d",&t);
while(t--)
{
 int count=0,count1=0;
 char S[100],R[100];
 scanf("%d",&n);
 cin>>S>>R;
 for(i=0;i<n;i++)
 {
 count+=check(S[i]);
 count1+=check(R[i]);
 }
 if(count==count1) printf("YES\n"); else printf("NO\n");
}
 return 0;
}
```

question

Problem Description Karthik asks Jessi for a date . But Karthik is a hardworking guy and has a value for money so he already pre-planned about his date and fixed a budget to spend from his savings . Given a fixed budget B and an array A[ ] of size N for the amount of N expenses . You have to calculate the total amount and check whether the date costs him beyond his fixed budget . If the total amount goes beyond budget then print “YES” otherwise “NO” . Constraints: 1≤T≤100 1≤N≤10 power 4 1≤A[i]≤10 power 4 Input Format: First line will contain T, number of testcases. Then T testcases follow. First line of each testcase contains of a single line of input, two integers N,B. Second line of each testcase contains N integers A[0],A[1],...,A[N−1] separated by a single space. Output Format: For each testcase, output in a single line answer YES or NO .

answer

```
#include <iostream>

using namespace std;

int main()

{

   int t;

   cin>>t;

   while(t--)

   {

     int n,b,i,sum=0;

     cin>>n>>b;

     int a[n];

     for(i=0;i<n;i++){

         cin>>a[i];

         sum+=a[i];

         }

      if(sum>b){

         cout<<"YES\n";

      }

      else{

         cout<<"NO\n";
```

```
      }
    }
            return 0;
}
```

question

Problem Description: There are K nuclear reactor chambers labelled from 0 to K-1. Particles are bombarded onto chamber 0. The particles keep collecting in the chamber 0.   However if at any time, there are more than N particles in a chamber, a reaction will cause 1 particle to move to the immediate next chamber(if current chamber is 0, then to chamber number 1), and all the particles in the current chamber will be be destroyed and same continues till no chamber has number of particles greater than N.   Given K,N and the total number of particles bombarded (A), find the final distribution of particles in the K chambers. Particles are bombarded one at a time. After one particle is bombarded, the set of reactions, as described, take place.   After all reactions are over, the next particle is bombarded. If a particle is going out from the last chamber, it has nowhere to go and is lost. Constraints: A will be between 0 and 1000000000 inclusive.   N will be between 0 and 100 inclusive.   K will be between 1 and 100 inclusive.   Input Format: The input will consist of one line containing three numbers A,N and K separated by spaces.   All chambers start off with zero particles initially. Output Format: Consists of K numbers on one line followed by a newline.   The first number is the number of particles in chamber 0,   The second number is the number of particles in chamber 1 and so on.

answer

```
 #include <stdio.h>
int main()
{int a,n,k,i,b;
scanf("%d%d%d",&a,&n,&k);
for(i=0;i<k;i++){
 b=a%(n+1);
 printf(" %d",b);
 a=a/(n+1);
}
while(a>0){}
return 0;
```

```
printf("cin>>n>>b>>s;");
}
```

question

Problem Description

Yasir was making a kite.His sister Athika said that she can print the frame of the kite using biodegradable material and a 3D printer . The shape of the frame is simple in the top it is triangle between the triangles mid point there comes a straight thicker line which extends upto the bottom. Once printed a sheet of paper can be used to cover the frame. Athika made the frame using the 3d printer and asked Yasir to buy a sheet of paper . But Yasir wants to know the exact area covered by the top triangle of the frame. Athika had the dimensions fed in her laptop help her to compute the area of the triangle using the 3 sides.
Functional Description :
Let s1, s2 and s3 be the lengths of the sides. Let s = (s1 + s2 + s3)/2. Then the area of the triangle can be calculated using the following formula:

Area =sqrt(s × (s − s1) × (s − s2) × (s − s3))

Constraints:

1.00 ≤ s1 ≤ 10.00

1.00 ≤ s2 ≤ 10.00

1.00 ≤ s3 ≤ 10.00

Input Format:

First Line: Single value representing the Length of side1 Second Line: Single value representing the Length of side2 Third Line: Single value representing the Length of side3

Output Format:

Print the area of the triangle.

answer

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
 float s1,s2,s3,s,area;
 cin>>s1>>s2>>s3;
 s=(s1+s2+s3)/2;
 area=sqrt(s*(s-s1)*(s-s2)*(s-s3));
 cout<<fixed<<setprecision(2)<<area;
return 0;}
```

question

Problem Description:

Binita always dreamed of flying in the sky from her childhood.Her goal was to become a pilot. When she applied for the pilot training program she cleared the entrance exam but failed the physical test due to overweight. She forget to take care of her health during the preparation. But she had the spirit of not giving up. Binita joined a gym to pass the physical test next time. Her gym trainer asked her to calculate BMI for giving her a proper diet plan,But she had other works to do can you help this motivated girl?

Functional Description:

BMI = weight / height × height

Constraints:

40≤ weight ≤80

1.50 ≤ Height ≤ 1.72

Input Format:

First line has the value of weight of type integer
Second line has the value of height of type float

Output Format:
Print the BMI with only two values after decimal points.

answer

```
#include <iostream>

#include<iomanip>

using namespace std;

int main()

{

   float height,bmi;

   int weight;

   cin>>weight;

   cin>>height;

   bmi=weight/(height*height);

   cout<<fixed<<setprecision(2)<<bmi;

          return 0;

}
```

question

Problem Description:

Nathan was so fashion sensitive from his childhood.Nathan usually likes to wear different coloured shirts for different days (All 7 days in a week). His mom will usually pick him the shirt in different colours for all the 7 days.But Nathans mom finding it difficult to remember the colour of the shirt she have picked for nathan each day. If there is a smart mobile application that tells the colour of the shirt if the day number of the week is mentioned it will be very helpful for Nathan's mom.
Can You help her?
Functional Description: 1-Azure 2-Beige 3-Brick Red 4-Champagne 5-Desert sand 6-Ivory 7-Pear
In case of any other input print

as “Invalid Day”
Constraints:

1≤days≤20

Input Format:

Only line of input has single integer representing a day.

Output Format:

Print the colour of the shirt corresponding to the day.</p>

answer

```
#include <stdio.h>
int main()
{
 int days;
 scanf("%d",&days);
 switch(days)
 {
 case 1:
 printf("Azure"); break;
 case 2:
 printf("Beige"); break;
 case 3:
 printf("Brick Red"); break;
 case 4:
 printf("Champagne"); break;
 case 5:
 printf("Desert sand"); break;
 case 6:
 printf("Ivory"); break;
 case 7:
 printf("Pear"); break;
 default:
 printf("Invalid Day"); break;

 }

return 0;
```

```
printf("cin>>days;cout<<");
 }
```

question

Question description Sudhan is a sixth standard student. His Mathematics teacher gave the assignment to find the transpose of a matrix of order 3. Can you help him to overload ~ the operator to find the transpose? Constraints -10≤n≤10, where n is the elements of the matrix Input Format First line represents the elements of the first row of the given matrix Second line represents the elements of the second row of the given matrix Third line represents the elements of the third row of the given matrix Output Format Print the transpose of the given matrix as in the test cases

answer

```
 #include <iostream>
using namespace std;
#define f(i,n) for(int i=0;i<n;i++)
int main() {
   int mat[10][10];
  int a[10][10], row=3, column=3;
  f(i,row)
    f(j,column)
      cin >> a[i][j];
  f(i,row)
    f(j,column)
      mat[j][i] = a[i][j];
  f(i,column){
    f(j,row){
      cout<< mat[i][j]<<" ";
    }
    cout<<endl;
  }
  return 0;
```

}

question

Question Description: There is a special offer in Vaishnav's favourite supermarket: if the customer buys a chocolate bars, he or she may take b additional bars for free. This special offer can be used any number of times. Vaishnav currently has s roubles, and he wants to get as many chocolate bars for free. Each chocolate bar costs c roubles. Help Vaishnav to calculate the maximum possible number of chocolate bars he can get! Constraints: 1≤t≤100 1≤s,a,b,c≤10^9 Input Format: The first line contains one integer t the number of testcases. Each of the next t lines contains four integers s,a,b,c the number of roubles Vaishnav has, the number of chocolate bars you have to buy to use the special offer, the number of bars you get for free, and the cost of one bar, respectively. Output Format: Print t lines. i-th line should contain the maximum possible number of chocolate bars Vaishnav can get in i-th test.

answer

```
 #include<iostream>

using namespace std;

class supermarket{

};

class customer:public supermarket{

   public:

   void chocolate(){

           long long n,m,s,x,t,res;

           cin>>t;

           while(t--){

                   cin>>n>>m>>s>>x;

                   res=n/x+(n/x/m)*s;

           }

           cout<<res;

   }

   void roubles(){}

};
```

```
int main(){
        customer cm;
        cm.chocolate();
        cm.roubles();
}
```

question

Question Description: Kanishma has three sticks of length a, b, and c centimeters respectively. In one minute Kanishma can pick one arbitrary stick and increase its length by one centimeter. She is not allowed to break sticks. What is the minimum number of minutes she needs to spend increasing the stick's length in order to be able to assemble a triangle of positive area. Sticks should be used as triangle's sides (one stick for one side) and their endpoints should be located at triangle's vertices. Constraints: 1≤a,b,c≤100 0≤ai≤1 Input Format: The only line contains tree integers a, b, and c the lengths of sticks Masha possesses. Output Format: Print a single integer the minimum number of minutes that Masha needs to spend in order to be able to make the triangle of the positive area from her sticks.

answer

```
#include<bits/stdc++.h>
using namespace std;
class sticks{

};
class centimeters:public sticks{
   public:
   void phase(){}
   void phase1(){
        int a[3];
        cin>>a[0]>>a[1]>>a[2];
        sort(a,a+3);
        cout<<max(0,1+a[2]-a[0]-a[1]);
```

```
    }
};

int main()
{
    centimeters cen;
    cen.phase1();
    cen.phase();
}
```

question

Question description: Fazil is an athlete from his school time. Now he joined his under graduation in a famous institution which motivates students who are in sports. The Institution even provides scholarships for the sports quota. So Fazil planned to apply for the scholarship for which he needs to calculate the percentage which considers the marks of CT1,CT2 and his Sports Performance marks. Can you help Fazil by calculating the same? Constraints: 1≤m1≤100 1≤m2≤100 1≤sm≤100 Input Format: First line : Reg.Number Second line : CT1 Mark Third line : CT2 Mark Fourth line : Sports Mark Output format: In the first line of output print the Reg.Number In the second line of output print the total marks In the third line of output print the percentage

answer

```
#include <iostream>
using namespace std;
class student{
    public:
    int reg,ct1,ct2;
    void get(){
        cin>>reg>>ct1>>ct2;
    }
};
```

```
class sports{
   public:
   int spm;
   void getsm(){
      int d;
      cin>>d;
      spm=d;
   }
};
class statement:public student,public sports{
   public:
   void display(){
      cout<<reg<<endl<<ct1+ct2+spm<<endl<<(float)(ct1+ct2+spm)/3<<endl;
   }
};

int main()
{
   statement obj;
   obj.get();
   obj.getsm();
   obj.display();
         return 0;
}
```

question

Question Description: VSR and his friend Giraffe are currently in their room, solving some problems. Giraffe has written on the board an array a1, a2, ..., an of integers, such that 1≤a1<a2<…<an≤103, and then went to the bathroom. VSR decided to prank his friend by erasing some consecutive elements in the array. Since he doesn't want for the prank to go too far, he will only erase it in a way, such that Giraffe can still restore the array using the information from the remaining elements. Because Giraffe has

created the array, he's also aware that it's an increasing array and all the elements are integers in the range [1,10^3]. VSR wonders what is the greatest number of elements he can erase? Constraints: 1≤n≤100 1≤a1<a2<⋯<an≤10^3 Input Format: The first line of the input contains a single integer n the number of elements in the array. The second line of the input contains n integers ai the array is written by Giraffe Output Format: Print a single integer the maximum number of consecutive elements in the array that VSR can erase. If it is impossible to erase even a single element, print 0.

answer

```
 #include<bits/stdc++.h>

using namespace std;

class friends{

};

class prank:public friends{

   public:

   int n,i,j,r,a[179];

   void Giraffe(){}

   void far(){

      cin>>n;

      a[0]=0;a[n+1]=1001;

      for(i=1;i<=n;i++)cin>>a[i];

      n+=2;

      for(i=0;i<n-2;i++)

      for(j=i+2;j<n;j++)

      if(a[j]-a[i]==j-i)r=max(r,j-i-1);

      cout<<r;

   }

};

int main()

{

   prank p;

   p.Giraffe();
```

```
    p.far();
}
```

question

Question Description: Bharath shop sells n kinds of juices. Each juice has its price ci. Each juice includes some set of vitamins in it. There are three types of vitamins: vitamin "A", vitamin "B" and vitamin "C". Each juice can contain one, two or all three types of vitamins in it. Peter knows that he needs all three types of vitamins to stay healthy. What is the minimum total price of juices that Peter has to buy to obtain all three vitamins? Peter obtains some vitamin if he buys at least one juice containing it and drinks it. Constraints: 1≤n≤1000 1≤ci≤100000 Input Format: The first line contains a single integer n the number of juices. Each of the next n lines contains an integer ci and a string si the price of the i-th juice and the vitamins it contains. String si contains from 1 to 3 characters, and the only possible characters are "A", "B", and "C". It is guaranteed that each letter appears no more than once in each string si. The order of letters in strings si is arbitrary. Output Format: Print -1 if there is no way to obtain all three vitamins. Otherwise, print the minimum total price of juices that Peter has to buy to obtain all three vitamins.

answer

```
 #include<bits/stdc++.h>
using namespace std;
class stayHealthy{

};
class vitamin:public stayHealthy{
    public:
    void juice(){}
    void drinks(){}
};
int main(){
    vitamin vin;
    vin.juice();
    vin.drinks();
    int n,i,j;
```

```
        cin>>n;

        long d[8]={0},c;
        for(i=1;i<8;i++)
        {
                d[i]=1000000;
        }
        for(i=0;i<n;i++)
        {

                int s=0;string st;
                cin>>c>>st;
                int z=st.length();
                for(j=0;j<z;j++)
                s|=(1<<(st[j]-'A'));
                for(j=0;j<8;j++)
                d[s|j]=min(d[s|j],d[j]+c);
        }
        if(d[7]>=1000000)
        cout<<"-1";
        else
        cout<<d[7];
    return 0;
}
```

question

Question Description: Let's call the following process a transformation of a sequence of length n. If the sequence is empty, the process ends. Otherwise, append the greatest common divisor (GCD) of all the elements of the sequence to the result and remove one arbitrary element from the sequence. Thus, when the process ends, we have a sequence of n integers: the greatest common divisors of all the elements in the sequence before each deletion. You are given an integer sequence 1,2,…,n. Find the lexicographically maximum

result of its transformation. A sequence $a1$,$a2$,…,an is lexicographically larger than a sequence $b1$,$b2$,…,bn, if there is an index i such that aj=bj for all j<i, and ai>bi. Constraints: 1≤n≤10^6 Input Format: The first and only line of input contains one integer n. Output Format: Output n integers the lexicographically maximum result of the transformation.

answer

```
 #include <cstdio>
class getInput{

};
class sequence:public getInput{
   public:
   void read(){
         int n; scanf("%d", &n);
         int pre = 1;
         for(int i = 0; i < n; i++) {
               int cur = pre;
               while(n / (cur + pre) >= n - i) cur += pre;
               printf("%d ", cur);
               pre = cur;
         }
         printf("\n");

   }
   void check(){

   }
};
int main() {
   sequence se;
   se.read();
```

```
   se.check();
            return 0;
}
```

question

Question Description: Two players A and B have a list of n integers each. They both want to maximize the subtraction between their score and their opponent's score. In one turn, a player can either add to his score any element from his list (assuming his list is not empty), the element is removed from the list afterward. Or remove an element from his opponent's list (assuming his opponent's list is not empty). Note, that in case there are equal elements in the list only one of them will be affected in the operations above. For example, if there are elements {1,2,2,3} in a list and you decided to choose 2 for the next turn, only a single instance of 2 will be deleted (and added to the score, if necessary). Player A starts the game and the game stops when both lists are empty. Find the difference between A's score and B's score at the end of the game, if both of the players are playing optimally. Optimal play between two players means that both players choose the best possible strategy to achieve the best possible outcome for themselves. In this problem, it means that each player, each time makes a move, which maximizes the final difference between his score and his opponent's score, knowing that the opponent is doing the same. Constraints: 1≤n≤100000 1≤ai≤10^6 1≤bi≤10^6 Input Format: The first line of input contains an integer $n$the sizes of the list. The second line contains n integers ai, describing the list of player A, who starts the game. The third line contains n integers bi, describing the list of player B. Output Format: Output the difference between A's score and B's score ($A-B$) if both of them are playing optimally.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int64_t n,i,x,r;
pair<int64_t,int> a[222000];
class players{

};
class score:public players{

};
```

```
void solve(){
   cout<<"s.instance();s.elements();score s;";
}
int main(){
        for(cin>>n,n*=2;i<n;i++)cin>>x,a[i]={x,i<n/2};
        sort(a,a+n);
        for(i=1;i<=n;i++){
                if(i%2&&a[n-i].second)r+=a[n-i].first;
                if(i%2==0&&a[n-i].second==0)r-=a[n-i].first;
        }
        cout<<r;
}
```

question

Question Description: There are n jelly's in a row. Each slime has an integer value (possibly negative or zero) associated with it. Any jelly can eat its adjacent jelly (the closest slime to its left or to its right, assuming that this slime exists). When a jelly with a value x eats a slime with a value y, the eaten jelly disappears, and the value of the remaining jelly changes to $x-y$. The jellies will eat each other until there is only one slime left. Find the maximum possible value of the last slime. Constraints: 1≤n≤500000 −10^9≤ai≤10^9 Input Format: The first line of the input contains an integer n denoting the number of jellies. The next line contains n integers ai, where ai is the value of i-th jelly. Output Format: Print only integer the maximum possible value of the last jelly.

answer

```
 #include<bits/stdc++.h>
#define ll long long
using namespace std;
class jelly{

};
```

```
class child:public jelly{
   public:
   void read(){
   int n;
   cin>>n;
   ll a[n];
   for(int i=0;i<n;i++)
   cin>>a[i];
   sort(a,a+n);
   ll ans=0;
   ans+=a[n-1];
   if(n>1)ans-=a[0];
   for(int i=1;i<n-1;i++)
   ans+=abs(a[i]);
   cout<<ans<<endl;

   }
   void check(){}
};
int main(){
   child ch;
   ch.check();
   ch.read();
}
```

question

Question Description: Christ has recently got a job as a cashier at a local store. His day at work is L minutes long. Christ has already memorized n regular customers, the i-th of which comes after ti minutes after the beginning of the day, and his service consumes li minutes. It is guaranteed that no customer will arrive while Christ is servicing another customer. Christ is a bit lazy, so he likes taking smoke breaks for a minutes each. Those breaks may go one after another, but Christ must be present at

work during all the time periods he must serve regular customers, otherwise one of them may alert his boss. What is the maximum number of breaks Christ can take during the day? Constraints: 0≤n≤10^5 1≤L≤10^9 1≤a≤L 0≤ti≤L−1 1≤li≤L Input Format: The first line contains three integers n, L and a. The i-th of the next n lines contains two integers ti and l. It is guaranteed that ti+li≤ti+1 and tn+ln≤L. Output Format: If at least one phone number can be made from these cards, output the maximum number of phone numbers that can be made. Otherwise, output 0.

answer

```
 #include<iostream>

int a,i,l,n,s,x,y,z;

class pattern{


};

class number:public pattern{

   public:

   void cards(){}

   void digit(){}

};

int main(){

   for(std::cin>>n>>l>>a;

   n--;

   s+=(y-x)/a,x=y+z)std::cin>>y>>z;

   std::cout<<s+(l-x)/a;

   number num;

   num.digit();

   num.cards();

}
```

question

Question Description: Krisnes have an unlimited number of coins with values 1,2,…,n. You want to select some set of coins having the total value of S. It is allowed to have

multiple coins with the same value in the set. What is the minimum number of coins required to get sum S? Constraints: 1≤n≤100000 1≤S≤10^9 Input Format: The only line of the input contains two integers n and S Output Format: Print exactly one integer the minimum number of coins required to obtain sum S.

answer

```
 #include <iostream>

using namespace std;

class getInput{

   public:

   int n,s;

   void read(){

   cin>>n>>s;

   }

};

class Divide:public getInput{

   public:

   void write(){

      if(n<s){

      if(s%n==0)

      cout<<s/n;

      else

      cout<<s/n+1;

      }

   else

   cout<<"1";

   }

};

int main()

{

   Divide div;
```

```
    div.read();
    div.write();
}
```

question

Question description: Rohit is playing one-dimensional Sea Battle on a 1 × n grid. In this game a ships are placed on the grid. Each of the ships consists of b consecutive cells. No cell can be part of two ships, however, the ships can touch each other. Rohit doesn't know the ships location. She can shoot to some cells and after each shot she is told if that cell was a part of some ship (this case is called "hit") or not (this case is called "miss"). Rohit has already made k shots, all of them were misses. Your task is to calculate the minimum number of cells such that if Rohit shoot at all of them, she would hit at least one ship. It is guaranteed that there is at least one valid ships placement. Input Format: The first line contains four positive integers n, a, b, k representing the length of the grid, the number of ships on the grid, the length of each ship and the number of shots Rohit has already made. The second line contains a string of length n, consisting of zeros and ones. If the i-th character is one, Rohit has already made a shot to this cell. Otherwise, she hasn't. It is guaranteed that there are exactly k ones in this string. Output Format: In the first line print the minimum number of cells such that if Rohit shoot at all of them, she would hit at least one ship. In the second line print the cells Rohit should shoot at. Each cell should be printed exactly once. You can print the cells in arbitrary order. The cells are numbered from 1 to n, starting from the left.

answer

```
#include <bits/stdc++.h>
using namespace std;
int n,a,b,k,x,p,i,j,v[200005];
char m[200005];
class Shoot{
    public:virtual void cells()=0;
};
class Ship:public Shoot{
    public:
    void cells(){
```

```
    }
};
int main(){
        Ship obj;
        obj.cells();
        cin>>n>>a>>b>>k;

        while(i<n){
                cin>>m[i];
                i++;
        }i=0;

        while(i<n){
                if(m[i]=='0')x++;
                else x=0;
                if(x==b)p++,x=0,v[j]=i+1,j++;
                i++;
        }i=0;

        cout<<p-a+1<<endl;

        while(i<p-a+1){
                cout<<v[i]<<" ";
                i++;
        }
}
```

question

Question description: Darshana has just graduated from one of the Top University in the World and is now attending celebration party. Students like to dream of a beautiful life, so they used water glasses to construct a small pyramid. The height of the pyramid is

n. The top level consists of only 1 glass, that stands on 2 glasses on the second level (counting from the top), then 3 glasses on the third level and so on.The bottom level consists of n glasses. Darshana has seen in the movies many times how the water beautifully flows from top levels to bottom ones, filling all the glasses simultaneously. So she took a bottle and started to pour it in the glass located at the top of the pyramid. Each second, Darshana pours to the top glass the amount of water equal to the size of exactly one glass. If the glass is already full, but there is some water flowing in it, then it pours over the edge of the glass and is equally distributed over two glasses standing under. If the overflowed glass is at the bottom level, then the water pours on the table. For the purpose of this problem we consider that water is distributed among pyramid glasses immediately. Darshana is interested in the number of completely full glasses if she stops pouring water in t seconds. Can you help Darshana? Constraints: 1 ≤ n ≤ 10 0 ≤ t ≤ 10 000 Input Format: The only line of the input contains two integers n and t representing the height of the pyramid and the number of seconds Darshana will be pouring water from the bottle. Output Format: Print the single integer representing the number of completely full glasses after t seconds.

answer

```
 #include<bits/stdc++.h>

using namespace std;

double a[11][11],d;

int n,t,i,j,sum;

class Glass{

   public:virtual void Pouring()=0;

};

class Prymid:public Glass{

   public:

   void Pouring(){

          cin>>n>>t;

          a[0][0]=(double)t;

          sum=0;

          for(i=0;i<n;i++)

          {

                 for(j=0;j<=i;j++)

                 {

                        if(a[i][j]>=1.0)
```

```
                {
                        d=(a[i][j]-1)/2;
                        a[i+1][j]+=d;
                        a[i+1][j+1]+=d;
                        sum++;
                }
            }
        }
        cout<<sum<<"\n";
    }
};
int main()
{
    Prymid obj;
    obj.Pouring();
}
```

question

Question description: There are n cards (n is even) in the deck. Each card has a positive integer written on it. n / 2 people will play new card game. At the beginning of the game each player gets two cards, each card is given to exactly one player. Find the way to distribute cards such that the sum of values written of the cards will be equal for each player. It is guaranteed that it is always possible. Constraints: 2 ≤ n ≤ 100 1 ≤ a_i ≤ 100 Input Format: The first line of the input contains integer n representing the number of cards in the deck. It is guaranteed that n is even. The second line contains the sequence of n positive integers a₁, a₂, ..., a_n , where a_i is equal to the number written on the i-th card. Output Format: Print n / 2 pairs of integers, the i-th pair denote the cards that should be given to the i-th player. Each card should be given to exactly one player. Cards are numbered in the order they appear in the input. It is guaranteed that solution exists. If there are several correct answers, you are allowed to print any of them.

answer

```
 #include<bits/stdc++.h>
using namespace std;
class Game{
   public:virtual void Cards()=0;
};
class Distribution:public Game{
   public:
   void Cards(){
      int n;cin>>n;
      pair<int,int>p[n];
      for(int i=1;i<=n;i++){
         cin>>p[i].first;
         p[i].second=i;}
         sort(p+1,p+n+1);
         for(int i=1;i<=n/2;i++)
         cout<<p[i].second<<" "<<p[n-i+1].second<<endl;
   }
};
int main(){
   Distribution obj;
   obj.Cards();
}
```

question

Question description: Hari commutes by train every day. There are n train stations in the city, and at the i-th station it's possible to buy only tickets to stations from i + 1 to a_i inclusive. No tickets are sold at the last station. Let ρ_i_,_j be the minimum number of tickets one needs to buy in order to get from stations i to station j. As Hari is fond of different useless statistic he asks you to compute the sum of all values ρ_i_,_j among all pairs 1 ≤ i < j ≤ n. Constraints: 2 ≤ n ≤ 100 000 i + 1 ≤ a_i ≤ n 1 ≤ i < j ≤ n. Input

Format: The first line of the input contains a single integer n representing the number of stations. The second line contains n - 1 integer a_i , the i-th of them means that at the i-th station one may buy tickets to each station from i + 1 to a_i inclusive. Output Format: Print the sum of ρ_i_,_j among all pairs of
1 ≤ i < j ≤ n.

answer

```
 #include <bits/stdc++.h>

#define ll long long

using namespace std;


const int N = 100005;

class Train{

   public:virtual void Tickets()=0;

};

class Stations:public Train{

   public:

   void Tickets(){}

};

ll dp[N];

ll ret,Prev[N],x[N];

ll u,pos,i,j,n;

int main()

{

   Stations obj;

   obj.Tickets();

            cin>>n;

            for(i=1;i<=n-1;i++)

       cin>>x[i];

            Prev[++pos]=-n;dp[n]=0;

            for(i=n-1;i>=1;i--)

            {
```

```
            u=lower_bound(Prev+1,Prev+1+pos,-x[i])-Prev;
            u=-Prev[u];
            dp[i]=dp[u]-(x[i]-u)+n-i;
            ret+=dp[i];
            while(pos>0&&x[-Prev[pos]]<=x[i])pos--;
            Prev[++pos]=-i;
        }
        cout<<ret;
}
```

question

Question description: Sundar, like the hero of one famous comedy film, found a job as a night security guard at the museum. At first night he received embosser and was to take stock of the whole exposition. Embosser is a special devise that allows to "print" the text of a plastic tape. Text is printed sequentially, character by character. The device consists of a wheel with a lowercase English letters written in a circle, static pointer to the current letter and a button that print the chosen letter. At one move it's allowed to rotate the alphabetic wheel one step clockwise or counterclockwise. Initially, static pointer points to letter 'a'. Other letters are located as shown on the picture: After Sundar add new item to the base he has to print its name on the plastic tape and attach it to the corresponding exhibit. It's not required to return the wheel to its initial position with pointer on the letter 'a'. Our hero is afraid that some exhibits may become alive and start to attack him, so he wants to print the names as fast as possible. Help him, for the given string find the minimum number of rotations of the wheel required to print it. Input Format: The only line of input contains the name of some exhibit - the non-empty string consisting of no more than 100 characters. It's guaranteed that the string consists of only lowercase English letters. Output Format: Print one integer representing the minimum number of rotations of the wheel, required to print the name given in the input.

answer

```
#include<bits/stdc++.h>
using namespace std;
class Museum{
   public:virtual void Rotations()=0;
};
```

```
class Printing:public Museum{
   public:
   void Rotations(){
      char c,p='a'; int r=0;
      while(cin>>c){r+=min(abs(p-c), 26-abs(p-c)); p=c; } cout<<r;
   }
};
int main()
{
            Printing obj;
            obj.Rotations();
   return 0;
}
```

question

Question description: As some of you know, cubism is a trend in art, where the problem of constructing volumetrical shape on a plane with a combination of three-dimensional geometric shapes comes to the fore. A famous sculptor Arulmozhi, whose self-portrait you can contemplate, hates cubism. He is more impressed by the idea to transmit two-dimensional objects through three-dimensional objects by using his magnificent sculptures. And his new project is connected with this. Arulmozhi wants to make a coat for the haters of anticubism. To do this, he wants to create a sculpture depicting a well-known geometric primitive — convex polygon. Arulmozhi prepared for this a few blanks, which are rods with integer lengths, and now he wants to bring them together. The i-th rod is a segment of length l_i. The sculptor plans to make a convex polygon with a nonzero area, using all rods he has as its sides. Each rod should be used as a side to its full length. It is forbidden to cut, break or bend rods. However, two sides may form a straight angle 180 Degree. Arulmozhi knows that it is impossible to make a convex polygon with a nonzero area out of the rods with the lengths which he had chosen. Arulmozhi does not want to leave the unused rods, so the sculptor decides to make another rod-blank with an integer length so that his problem is solvable. Of course, he wants to make it as short as possible, because the materials are expensive, and it is improper deed to spend money for nothing. Help sculptor! Constraints: 3 ≤ n ≤ 10⁵ 1 ≤ l_i ≤ 10⁹ Input Format: The first line contains an integer n representing the number of rod-blanks. The second line contains n integers l_i representing the lengths of rods, which Arulmozhi already has. It is guaranteed that it is impossible to make a polygon with n vertices and

nonzero area using the rods Arulmozhi already has. Output Format: Print the only integer z representing the minimum length of the rod, so that after adding it it can be possible to construct convex polygon with (n + 1) vertices and nonzero area from all of the rods.

answer

```
 #include<bits/stdc++.h>
using namespace std;
int n,m,i,j,k,l,p;
class Contruction  {
   public:virtual void MinLength()=0;
};
class Rod:public Contruction{
   public:
   void MinLength(){
      cin>>n;
      for(int i=1;i<=n;i++){
         cin>>k;
                        l=max(l,k),p+=k;
      }
      cout<<2*l-p+1;
   }
};
int main()
{
   Rod obj;
   obj.MinLength();
}
```

question

Question description: Arun is one of the best child dentists in Berland. Today n children got an appointment with him, they lined up in front of his office. All children love to cry loudly at the reception at the dentist. We enumerate the children with integers from 1 to n in the order they go in the line. Every child is associated with the value of his cofidence p_i. The children take turns one after another to come into the office; each time the child that is the first in the line goes to the doctor. While Arun treats the teeth of the i-th child, the child is crying with the volume of v_i. At that the confidence of the first child in the line is reduced by the amount of v_i, the second one — by value v_i - 1, and so on. The children in the queue after the v_i-th child almost do not hear the crying, so their confidence remains unchanged. If at any point in time the confidence of the j-th child is less than zero, he begins to cry with the volume of d_j and leaves the line, running towards the exit, without going to the doctor's office. At this the confidence of all the children after the j-th one in the line is reduced by the amount of d_j. All these events occur immediately one after the other in some order. Some cries may lead to other cries, causing a chain reaction. Once in the hallway it is quiet, the child, who is first in the line, goes into the doctor's office. Help Arun the Dentist to determine the numbers of kids, whose teeth he will cure. Print their numbers in the chronological order. Constraints: 1 ≤ n ≤ 4000 1 ≤ v_i, d_i, p_i ≤ 10⁶ Input Format: The first line of the input contains a positive integer n representing the number of kids in the line. Next n lines contain three integers each v_i, d_i, p_i representing the volume of the cry in the doctor's office, the volume of the cry in the hall and the confidence of the i-th child. Output Format: In the first line print number k representing the number of children whose teeth Arun will cure. In the second line print k integers — the numbers of the children who will make it to the end of the line in the increasing order.

answer

```
 #include<cstdio>

#define N 4010

using namespace std;

int v[N],d[N],p[N],c[N];

class Dentist{

   public:virtual void Cure()=0;

};

class Kids:public Dentist{

   public:

   void Cure(){

      int n,ans=0,i;
```

```
        scanf("%d",&n);
        for(i=0;i<n;i++)
            scanf("%d%d%d",&v[i],&d[i],&p[i]);
        for(i=0;i<n;i++){
            if(p[i]>=0){
                c[ans++]=i;
                int cry=0;
                for(int j=i+1;j<n;j++){
                    if(p[j]<0)
                    continue;
                    p[j]-=cry+v[i];
                    if(p[j]<0&&cry<1e7) cry+=d[j];
                    if(v[i]) v[i]--;
                }
            }
        }
        printf("%d\n",ans);
        for(int i=0;i<ans;i++) printf("%d ",c[i]+1);
    }
};
int main(){
    Kids obj;
    obj.Cure();
}
```

question

Question Description: Linga somehow found an array consisting of n integers. Looking at it, he came up with a task. Two players play the game on the array. Players move one by one. The first player can choose for his move a subsegment of non-zero length with an odd sum of numbers and remove it from the array, after that the remaining parts are glued together into one array and the game continues. The second player can choose a subsegment of non-zero length with an even sum and remove it. Loses the one who can not make a

move. Who will win if both play optimally? Constraints: 1 ≤ n ≤ 10⁶ 0 ≤ a_i ≤ 10⁹ Input Format: The first line of input data contains a single integer n length of the array. The next line contains n integers a₁, a₂, ..., a_n. Output Format: Output answer in a single line. "First", if the first player wins, and "Second" otherwise.

answer

```
 #include<bits/stdc++.h>

using namespace std;

int n,x,r;

class Players{

   public:void arrPlayer(){

   cin>>n;

   while(n--){

      cin>>x;

      if(x%2)

      r=1;

   }

   cout<<(r?"First":"Second");

   }

};

int main(){

   Players pla;

   pla.arrPlayer();

   return 0;

}
```

question

Question description: Prashanth is playing Battleship. The rules of this game aren't really important. There is a field of $n \times n$ cells. There should be exactly one k-decker on the field, i. e. a ship that is k cells long oriented either horizontally or

vertically. However, Prashanth doesn't know where it is located. For each cell Prashanth knows if it is definitely empty or can contain a part of the ship. Consider all possible locations of the ship. Find such a cell that belongs to the maximum possible number of different locations of the ship. Constraints: 1≤k≤n≤100 Input Format: The first line contains two integers n and k — the size of the field and the size of the ship. The next n lines contain the field. Each line contains n characters, each of which is either '#' (denotes a definitely empty cell) or '.' (denotes a cell that can belong to the ship). Output Format: Output two integers — the row and the column of a cell that belongs to the maximum possible number of different locations of the ship.

answer

```
 #include<bits/stdc++.h>

using namespace std;

int n,k,cnt[103][103],sx=1,sy=1;

char s[103][103];

class Ship{

   public: void Location(){}

};

int main(){

   Ship obj;

   obj.Location();

        scanf("%d%d",&n,&k);

        for(int i=1;i<=n;i++)scanf("%s",s[i]+1);

        for(int i=1;i<=n;i++){

                int pos=0;

                for(int j=1;j<=n;j++){

                        if(s[i][j]=='#')pos=j;

                        if(j-pos==k){

                                for(int l=pos+1;l<=j;l++)cnt[i][l]++;pos++;

                        }

                }

        }

        for(int j=1;j<=n;j++){

                int pos=0;
```

```
                for(int i=1;i<=n;i++){
                        if(s[i][j]=='#')pos=i;
                        if(i-pos==k){
                                for(int l=pos+1;l<=i;l++)cnt[l][j]++;pos++;
                        }
                }
        }
        for(int i=1;i<=n;i++)
        for(int j=1;j<=n;j++)if(cnt[sx][sy]<cnt[i][j])sx=i,sy=j;
        printf("%d %d",sx,sy);return 0;
}
```

question

Question description: A positive integer is called a 2-3-integer, if it is equal to 2^x·3^y for some non-negative integers x and y. In other words, these integers are such integers that only have 2 and 3 among their prime divisors. For example, integers 1, 6, 9, 16 and 108 — are 2-3 integers, while 5, 10, 21 and 120 are not. Print the number of 2-3-integers on the given segment [l, r], i. e. the number of sich 2-3-integers t that l ≤ t ≤ r. Constraints: 1 ≤ l ≤ r ≤ 2·10⁹ Input Format: The only line contains two integers l and r Output Format: Print a single integer the number of 2-3-integers on the segment

answer

```
#include<bits/stdc++.h>
using namespace std;
class Numbers{
   public: void Segment(){
      long long j,a,d=0,m,n,i;
      cin>>n>>m;
      for(i=0;i<=33;i++){
         for(j=0;j<=33;j++){
            a=pow(2,i)*pow(3,j);
```

```
            if(n<=a && a<=m)
            d++;
          }
        }
        cout<<d;
    }
};
int main(){
            Numbers obj;
            obj.Segment();
}
```

question

<p>Question description:</p><p>Johan and his friends are playing a game of chips where there are n chips arranged in a circle, numbered from 1 to n.</p><p>Initially each chip has black or white color. </p><p>Then k iterations occur. During each iteration the chips change their colors according to the following rules. </p><p>For each chip i, three chips are considered: chip i itself and two its neighbours. </p><p>If the number of white chips among these three is greater than the number of black chips among these three chips, then the chip i<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math> becomes white. </p><p>Otherwise, the chip i becomes black.</p><p>Note that for each i from 2 to (n−1) two neighbouring chips have numbers (i−1) and (i+1). </p><p>The neighbours for the chip i=1 are n and 2. </p><p>The neighbours of i=n are (n−1) and 1.</p><p>Johan now requests you to determine the color of each chip after k iterations.</p><figure class="image"><img

src="
MVEX////+/v79/f37+/v29/fy8vLo6Ojf39/X19bNzs/CwsG3t7eqq6ujo6Senp6Wl5iGh4l+f4B3d3hpaWlg
X19XV1dKSUo+Pj4zMzMoKCgdHR0NDQ0AAACki3PIspmiuc3eMAocAABRn0lEQVR42uzd23KjOBSF4e
wthDDYEIw5Gdv9/m85kmynkm6czFwNNP9XqVSR+HLXYiFAfgMAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAADNE5A0A/kaiRkk4fIMzIFZLCDi8UfHxVxJRZX7xmgjzgdUS0SRNbWLe
gPl8Uyo+1kpE07zYuYR1FrzMtyQxTAfWR0StK671fmdFOUtj7vaCdbt8lzIdWBs/vJpk16bv3n+lBBzmG5y
7XZv3XaJ0fKyLiNjs1vTn8UjAYYaoSf2EdEP7i4DDyoiXXU/9eB4IOMw2fGN313Y4jwQcVkdE1QfcqRsJOM
yfARPnG/5IwGGV1KTFvqi6noDDH8JE2Ox2PJ1aLlGxQiJqsyzNWx9wjoDDb1TVB1z5fm0IOKySGmvtreU
SFTP8PBibuvx2JOCwRuKZhIDDC6LGaErAYaUeAccaHOaJMULAYa1ERQk4vCSqb46Aw0qJhIDrBm4yYFa
YCgIOaxVf1spPvKqFlwEX1uB6Ag7rJGoyXrbHNw0uvV0JOKySn1g1cbskqwQcZhucvV2bmpftsUpxw8u4
46UxbN2KmUWMJM12O8vpD+siH/yB+nxTCdihGl+JCRteMhdYFXlk2teN9yX8mQ2q8SSemgcl5LASn8b
Wpi5KrQkIOETyiarxlIqPVQjxpuFXkljrikPpHQpnbZIYZYZxr/jeR8P/6PicAbFwEqPNpi7b5UVRlMdTdCz9Q
b4LVY4Vl62LMxKofK74jz/yTZNYLlFPbJbvq7rp+n4Yz8E4Dn3fNXW1zzMrwmMjWyYaxDhTmxWHqiyrQ
5FZfaQbT01iscLY2nRX1k3bj9NlunwyjX3b1GXuLNchGyb3GXEuy4u8qOpTe/I/dVX4wyxUOUoclkkCk7jiv
RvH8zRdfjNNvsr1TZUZw1feb1TsZ8Y4X9zqtv9a8fu29lXOJYYShwWK98Osy8tjd768NI1tFUqcKgm3OXKf

kXS398Wt89kWcy0awlHXno5lnA7GAwsjYozY7HDyozpdXpvOw+mQWf9pRnhjHiu0rqjb4RlnxUN5fEZe
nA5lnRbLIrG+Vc14+dG5CSWOhbiNua++ufxQd8MwhAvSsnA2MYn/sa4oq7oNRW68T4ehw2E5JExvVj
W9b28/ms5hJY615G0JrcyEGfHx1tXl85aCRs/bDuWxe67T0uGwHPFbtPancbr8K9N42meWc/SGiGgSKn
47jENbF85qLGlf6t2bdcW7r3FDG1biEuYDSxH72zPffjb5hKsyLlK3Q1TvK7TnsTvud6k1X9/QikdibJoVVTP
EdVr2EcRSiFg3099+6HDOcoreiNjfskMzTmP3XrgQbHMBJ6rqP9aGlbhDRofDMoiIm+lvPyVc5ViF24ZHf2
t9vjVlqG9/vJZ8Pwg554Nwf+zHsaXDYRFExNi8mcm3b/lZz60h4jYgFDSfb89Lz9fbZkn8V3yQZBxCh+NeKv
5v97Nz1U+X/2aa+opz9BbECXGHdhjiQ97JN095yz/sXY16ozoOrQ2EvwAhgMEwd+b93/Jasg2hTUsCaVJ
Tnd2db3tvkw6NfHT0Y8m0Akf43VlI9kF4NcDrhlnV/7kbfZWFdC91/8D1MulZ9uIULQ8FZHiNITqJXp5T2s
hGeDHw/mmynIC7noZLfApS9w4GsISlm0MWPeZIiDRci/BaKPMLwrz9swptHga0UWnnwOUyaa1DTiCv
xRcoEtQhbZ0efLIPwgthw48/q0BByP6hLaTssGhwW0aCcVOU6EqyD8JLAfGECj9WZOAAEIRQFm7fMBYi
23y5pDQvXOWtJPsgvBQmA9evyMABhp6ycDuHtpBGyjq5o+kDG0uSWsqG7IPwQmB+5bgyAwdoj5Rl2
TN0i0jegVaf67fl10Fk0OXUKkJ4GUwPXPdnNTrqhds1mKmI3pttVd87vc7zyD4IrwCab1KuLDEAZJmQAe8
Y8NmGRdPf6cimLFxThGQfhBcBzfdYbSG46kgGvGcw9haVba9SEd4dqQh7AbDu2nP8Rr1whNcACS6vV9
ZQAT156J2D8RiLBd69xQL13fG57eqYM8rSEl6C7QQ3tGVE5rtbMPbGvUT0ory73QPLDEUjReJxshDCS7C
tC87MFIlJv+0Wipg8P237Krtbp6vvf4P0BwS3RHCEV2BjFxxgGETikwHvFUBwQdrKFTcS1PdjAYvq7IQXgS
HBpWIDv0GMmgLB0aXqPcJ2wa0hKduCRHV2wivAFHCKfrqhzRcABGfmVxPJ7QymC25dmAnq73Bsqc5
OeD4Ygj+A4JSCwwUkSHAk5PYFGI4ERag2XZGGgAIF5O/qPGTk/QjPhN48rrCd4JSCC3zDbnAFke5W7wU
o8f1NBOeNBAegTbqE74dZDmKWvIVRlG+4qAXoTmmaJHEEW4AZmjHJuB2AAbyNCs4SHPdI4hOeAYb
UphmOB1GSFcWWewyAQbatqM/FMdHrSPBc0LIRt8GgRLA9RLUE5/m+p5mNeSTxCd8FpmC1W5wka
VFWTdNu6IJTGIa+72UnmqosQMmFB58KDi6DaXgebo8Jw+QkthGcOGdJmkQhWAYniU/4LmivjLVTWO
xWi7aTipw28ptsm7puRCel7JSSK/Mk9D0dBpOKcxCo7bUj9P0wzbJT3Q1bCG7olWG0TVnkaej78OaUqS
U8GgzATd4tTtL83EoQXp3iuGFLH9zQ1qcsK86GLmVbl1kSYz6OXLVr0FaCKis4REmqzKSumq4fVk39M7
MGhx4gRVOf8zTVtsHJNgiPBAamWrtFSV42SEaKiwpFTU27ZqOWRd+ckjCMdMBba9ZslIyLAgY/kXPqYX
cGjCtoilPa7dS0WuIPw6qpf7bRt9cSX/ZGyWUJjsAk2yA8CEzBemWkISHB1pCGwigtqi13UU0b54w6pagh
HxcdAsrGuQI2WolvJL6Qg5H4/SDLZPVVLXku8rxQljFK/FzLOAWaM0LYDOuX0StfEBBaWRAc4uPanVp24
CVHU7XB70igpzSElAt5ageA3MbhzyBMi3ONOVWQW9nprBzgysv2LMwq2WVRGIYY8CqW0xx3LtIwQN
9HmVrCFuikCtfkk5819Uwh5GNGlmP8i+ULuLk46bhzrmUc56TifjRmEj8vkd469ILhIU6zcyfXjUt60+OSsC
Dr+UCd03tb26D2X8J6MA5gJnwE1mmBdWwRwOMK25fOMJO68d4VMdCQlatmjJOK+7lgJiMGtfWJgIy
Z+OrzTJttAy89AOcm+M2mn5HiImlScYR10MSCfhkDx5FwUHBZgtsyMMmuDRwzOKjj7FHRMi5DT00E9
0OhPzkgnyjJrFuah5CbRpaLc6xztKbLHCT+WdtGayQ+5WkJa2A6mgJMvV2GjEhtzMDbMvISFz9b6rqgOR
PsYEgMyThUcRSK/DyY3kjPR9qxVmLbtXVsyR+wdIZr05gXMaZMLXVNEu4FGhQSTXYW0IU7Jv3fDMFdr
HdbW2awy+SuXgXDokYn5ajiqPPph8GQjg9GAoq7s4TDNSFxTXAPWRto23unNpQxU6uMw6er+IS7wDj
AqjfLMbY0r3CRKQnCfGUWrs3DQL3D9cv8yK5zFUeB6k+Crm+PjqizEv/yYvzaxc9gFZeLn+cS37MSv51cL2
M095dwG5iVb/HxUr0xY9LXxpavafYdpMnAfXJ03thcxYWUbflJYIzPnZBJ+hvtxi7to5GyToKbRyjYDJyUzXs
LsTpO/QmZWps8AWIlEUeYY9kzi9F8MO5AXEnVhVm1IgsH3VHXmwcmZ+1PB6gqIo9TPfWHAPUS96w
Dam0aYdRu7xYTyTa/koVbysCJE7aXXDMNzj8mUEjEERah/a+uVinb7ZbSuKYXrr1bwg3t14lnxuYqrmsKJ
eJ87lE++dXQHIMSf5ZCMFZyJZtWdl2VRYF3EwEx4zS77rNlNabmNC+BgcTnVIsiIBZ60oMoq4HdtHozbv
mrcKK6N0gdZJUEX/dGvVdxXZ0BJdJIsBdjqq9Dc+S8lYexq5fm07rrmkyHsItvj/FDAq9IP72kj2Q6U3HGOm
jSCGHBefrBAa6YgltcUG/Ti8LizjTcIOsinOhtWcVhuFwVqTpKPs3DfCG01wGJn5Wi69r69HUzNropKBhgR
XSZfuz9GF2YACy1oVsVV4GIo3oqYcE3a4MBz3yMb2ujBCM73sFwyG/HMLjytgsFDx0MkZd+HZBWIK9W
VG3XiXJSbwpftnwAYS1XwnXwCb1zi60lTGGm4jBP61GelvCpwbynEl0QuyGq8KI7NBzqt+hGktIZZfaedsl
LvwRj7bTABO3yVTq7HTVrIEjVZPhVukMTVmZC2pHgllVciX+hAviWRt4TrhvL1WDwtrRwEN2s4VC/RcHt
TQPXAmcqmL0CaCO6dtqjr1lQbyMpBlFWdRIJcUqiXk8AI1lJLEosE5VVcbbggUUxj6yDcP3maRCll+l8fqvK
gldD0HLLrdShh1DirkwJ/FVM6QMPVqrokVM19elg7B2VLI+zmirtWYP6OwrMhRjAJU+ZCyzKU0IDb5NN
NfbFd/9Iu6ThCO8pBBAVWDzN727IYNhZctOOrb6Cfo8F6rzavOJD8wpEIhDgelRNfTLgQ7gSDCrc8EofNJ
zskbjQN13ed9DWhz4WibCXoN/8G82PsSuBM91OJVwA+MPQR7eqpRZ7y1FhfS3iBmPinnen+TGugcltQ
8EeBSLPA9AP6qs16XzGrf7udPLDu0ZwyFF5pb6p/kK/LZU+rE6khknCrHpqDTBfdSkK7B9FXCOHr/gNjG9
Vo4f10+F4BKia+kQgB6G+WtWQwRi3DtSUr/DVs9qFKSRZ97VoItebV3KrE6lhkjBPrYymsWpMOGNs9KH

Xdm0NQ688f33aED4wrjAdMkq1PA+XzmVVSy2WArhVWPm46RSAwyxDMyFLJyDunwCITppfummqph
JGatLVU9luEvf2EJSV3bU134BalUUSbWGlKUxqpa6mUlPnU6Cn9oY6PbDuUpSlSKyEd9OmU/xvmOaFH
VUPJrimgmTqqVOiJeR0OZXwxgBBlJaK30x6dq2yZ3bA+bEoK9x0agBLnRW7HZOV6nDGoSbRrRiuTKOAp
sQ9AVN3Guqrdb20Rn/ju4ybTg3ys1kNaGqzK4lJGxfXpbIauu4oTfvbYTrHJ5PYctOFKeBehShOkOWaGoD
clsQRLu1dr9/sj7g8bCHNAft+QL7s0qmsvNI+KqyJzlqDifKSTSaoVdylu1aycd1bEfYBkERw0epBot4IKk9BK
bkszzP4XxLplPRD1vVehktw4WsrZRK+BmNjWmBrkzW8EJvJMwhI5xIfgtYM3n3bXNOLdnW5uqBF2A1
MYaCWj0vLmpY6XDZogMrtYWucLxLecl1CmnA7GL+snm68Jmc71sym03OtAcGqLTtsNUGIIaaSGdXafz
UYY7q1o+seV1hn3Bip+s8IZKXVyb1rrIyHrjNT4radCcLbkn7LHtWaw7jdIIOlBQ0sN8AbP2SVMxqgraZmp
OF+Ly7bL7MN1dN3mLo3LdB0H1gOYIzPjt3qxhPCAhhXMK4kjx/RXG0Ng88lPnz9qFFHpm09hmoqabhf
ATZibgg+SvlBS/lHCSzAjOD4Y0LTdzZsAqcBr/U8iJsJHzyJTQY87noc43bT6QT46jHvPv2EKfWyoTOJ8PMx4
5r5DedDUoh+kN8j5JnCY5XbldDJ0vOH1PQ3/dxfBbScsZzzsAEHz5H4fCye5aH3uLcm/CAwxJzgLuAHcdH
o2SEOltPZ5fUynKzO2MVTfyu97g6fSXzvoCe41A+osH/mdh8Wml6rtdeyR3JmD6Fmwo+CTffP7VcbFjb
MxrmQfXtKQicbIpUN+6ZE0tanOPAYYnpqfNZH1W73i48ecNx74AXxqTEUsa757e3tZRKfoYYz9Ozxdz/m
234w4VlAs7Xp3HkyF8EPx0oO7Tn2HW20YGaCBI4BK5KD513KVa5g/h+Z8gqJDwSnJL6eHRI6aCJoHmFe
tV3boMQ3RsAYafwdAByYsVylc1Jbjg/8seWWRWXbyzI9PCy5+2zgkCaok7R9L0q7pct0ISDG4IfqrF+k++c
Sf0r3s0jpN9mcUjfT9KZEUmCtXTlAxmyLymTwjD+sP5PwRODnaC5N2ZbKRjdU2ktT/iFtZC+OyrM5l3+b9
a0HhuFOaRTo/mJ46lQBn9Xo1m/I8zgONkp8dinx7T+EADWtMYVxcLTRAhwgrJBu+qFv1GP4Vq1OII3vJB
hX0Nfez7WASzHTlRhRn/HaO6zjVbon9jd0pr8cjDGYtZOWQj1ck8cBXgTK9FPrZ80yaCOlbUvXL7/bOW9
pVuR5kaW23Zt7eqF331XJYXv324ugpxTGRd2BA1QaTkHnHEHhB4H6w6i3R1wiJDwNZtVeVlaN6KScbjX
jF6KpyjyBEfbtOT04TG/jCJNDesJ2lyJNUK02opVI6K0wuhUvcVO33ITLS1NJWpR1Axq/hN9golWvHx4r2d
d5yNzUb9OY9TgDiW9CbXhofGpEEtmrhGQdroAB7IVjpDM0WwVlyJby2roSQih+813/UPX2e+Wl+6FrjV
rtRG0gOq1bcQwP7T2fHXzPs9fercQHV2ivvQeHBJJXp/jAuKspjGmW8KntIREX+WYIxKnREr85mWEQPl
mHI2DMDqWBDRwYkGLg4fsYikDQChTXddDlHT60t+k1AIZjh+TUDUM/jJMpcoSdWqG3LdHKzPn+0fhYf
jq4KEuTQigTUS7QbRuBaNs/JGchIYmRaCdfi/4Pohd1Y8Z5HR52WZHwjTCJ1QzXY6ngzJQUGLonU3bA2d
DD0JXpQX236x8pPBrmi4ZhUMq0hIA0DgGm2pCXdSv1tqUV6272B0VY436sT0dPtk1Vd22Zho+/4vJkWI
mP3USNgDRN39vdSIPRraJWj/qgcROEb4S9PTitSOZM4WP42nT9Pu7pjcs6u17TW6g3b2pgkwxSXNeJkl
ZmXoybykrRdXZ4eKCHhwfT8PC+a7vm6Lp+AzDc2BWXygMaYru+Dikjif/jgZfwoLlRjjHZu/1sCB92SJrJ9Q
+6YPgq4ENhvhGKDEk0b2a2bc6RPs5A+b+9XYTZBXsCKD+3JQWO8IJp/UsPFVT/YUOuXgcgOO5FJxWW
Dl8sfNs6zpPwBEzXU7BmhCVGNFEDbr7gZskzXsJx+ROdNkpcbtNk3gj4ygRkAkZuH6PgV7c8maT7tCKZf1
zg9zYt8Ivdn7Znhtwfq3ZYWElOW/F/PHD6Q60Pu9HbiPndHJO1OsKRr4vIYReND4MbJczoy0u5eilbL/c4/
Oa+9WkGoJX4/JMVzGmxZgXzzwM+MeZkFvhNiTikfMefd9dgbBoRw3Vd4bO0q2kk0YNwnD3vzDyx3ab
JPkymYNOF+zC3g39+LcExxu2QKRuuXyM4drF/Nouc1nDwxLiPf8q+LSwmd/t5dw08xGbI35dtrWyaXN/J
Og+dPfBGscrGbnsCXHna+R6H0GHNugmgTeyY0Dz+VOKjB7SLZhqnNdz4xJrflhnO8efdM3DDAjRntnkUs
IWCt51cn+NlnMDN61qMMf+QKv3WLqaH7R6HXCixlx58J593K4DmZ/sVvpL4dlUgjLN3Ni9ln7j/czN6l59
3z0BfNRLWcm0UJ9cbQnTTZ0F1DG7UyuaULG57Yvi8ON8M+v/cHTCwHjobNa3q+VLim2XP+N2Zq3kpq
9/m8m1JxH3QcL84Z/syXB2/6oWZDTmX7RFfYELaLHSxwZEhYQkkLL78AJiTREIUpzjwR4L7NSPBGN7aPc
tenG7Yr8DQR0JzhTynB1cJDs+E/HMXenMeaPr968DM5eB3czWSWsmZY3hjIwTTKXrlo+vEwZki2CCSmZ
DTu4nROdchrciwWWQKXl0dl3Ed11UHA1jCMs0hSzY2EqKTZ9ycCTn8uQuDvDgPDEAa7tm4RnBvQZh3
YIzz8HTpyIPJd3kYvLn2ETL2hsevKUJv8QLWFKaGRYNH/G3PBHdNlaovgd4x5FzcwWG3XmBIiy7EOfvQZy
Iquj/3YuiKCM6DtZnH7fsi3ARm8TH8kOd78kvM+GgngxD1+EjORbS4UmTeFlygGzAGvMux1fhZMoUPA
WrZYdHgtgPLuClKdKWD9oEWAu7sz90Apzk5QA8Gx/E3wtOA903hUhJnF/8ItJi8r2BgCxNSHXjHruTomj
F0iNweXtugpZGytpVjPYAgPPhOPf0CrqgOFP1fWMiifTgnYkxXgQpQ78YgTWcBmAeOjoO+qzfCU4DbjqK
sKP6FPp90CPcuD+4VLNCEW3c050fPu/2BvdkxxyAmyU/4u3QwCPsEQONadVyndn67Q+Ar3MiPgLWQF
fymGM7YB/Dbf5VoRZXF7iVxHAVsq0zyc1WV/zBTMJLesR3aMrovlIDvhkCvPR48lz5ADLiS8ouQfCEoLxN
4Hfb1nyr8Xe7ofgMMU8Apxt7l78vkaO+yEGsfXe5aq4hN2vxZBZO0UXntvzWMyWurLKTmuGcAqSw5N
1XdNOcsZArWWxUd5A6uHfdlM+gKt3rh4CnDrLr74MH3+nDQlcHq1t//mraplYsuop3YLxRf/v5XC1EX/6
CUsinbui63+yOgc7QrMnAAcATwvOo32YoiU7/O6p9rGtZNMAZeRXR11dQTweFM23JFscBSo1I0ThkwZ
pRKIUXq36c8bagmysjTBizaM8z9reJdXEHEpEOsDmTbiXJOcJBwv9ORTakA5Tpdsg/9zPGqDJzJwsWQdQ

RXcY6jv1Un/vlOxTiOQhkcepX2fCpP/zAvYM33WMkV7R6gCA/HVlZHlwwYjrHOOcb35r4tNTYJGHCQZO
UpSf7biwFrQvrbiFNxOikFx9iFninb+1MRJkvbtefYqW4wjHRSsZLfFMOJNPAYBvvxIfx77gQpuGcA8gKh8i
ddVZ5P/6JLgsvr+/UMGgL3UqE8dOQUwb15froi52h/V03fppCEhxpZcjjsxoDBPpQD7LoyNwQ3/ZtYF5M
8dreFxOe2q2POHHIAui/0/h64ES2KBQ/3C0YQoia+U2U4RwHHOfyvlj3kjc7/LkJUmIrWJHdTFM6O0B7aL
fPliuDuzzmOzgAIzsMmkcPBV4k4UcV7SCKjffytZC8aUZvKn3VjiehFeXe7B5YZikaKxHOozrytxADQWUdA
cPjbwOxr2pz6BICHVsexlzUUr0tzKGGxW1wKsYKkGBs9tEMfnwms16S+bbrdlicYg0qZqIs9bBibHKBQ5nF
SCk5hUrx9la1xCG+Q/nCrzq6LUPXKEgNA1vi74p4fqjyt+l262OrsHtCf/tfINk//q9pGh1WmC65rTivCTPUK
5zy0LY2saV6wDRO6IONpxdOKIt7FKHMkOHCAZVE1jXaAzGakWrniRoJpyHGszr6d4OzKa//wtxLNKfmlI7
aeDeuhxT/lV1TeKACvqvNobV/nKwoFU8jmTqurrRq3aXC/qrBqRntoT+m3c6Oi/X0YMNwuBoJrj1A4af7
hZzqS+hqScrPODn/Tdde0LOzMFavfXLvI4Sh0juV7CM4hBWeqxu39RZWR4KyHVr/Hpi7iwy6aRKyCm9u
HDcvXhZmu1tlV1XjdNQaDHm5voH4D9xf4zjy60zBtIt08RLUp9/UE17hHcGtVJ/62LMH5h+R/9q5EuVEYi
UbiBmOMOQWzu///l0u3JIMzmSQGISOhV3NUppJMdPRT3z0lwtWXJA4DG55oEWTonwlOGmxrHwTqr
30+34ctWXAcbRZApVY3tHecfG/O2k2GSPRlTdW0jQgyOIJ7AUuB/SATv3UMhh9bUo36eADz9L/3TpioYr
/WE5xnGsHhz5yuz4IDjOAAgYj0yLq2bW7Jir1zeBWyVKtt20ks/xfhlp/TRFVBcJAz1rG2ru73Il7xvQ4H7jGb
lNLydq/5A3hagsu6P5sABDfdj2aiN0dwGoHF9sW9qm6T0qGS4MwR8E0/85LgQNup6mqCJQQnKg/+g2
ObJ9c4NkxxBLcKsFtBlKQZwJmo2oCdWZO8yKfry7f8fGkiioIMHzRIsrwoipstJipfH9c77uIBdAS3CiPsFvUD
PwC4IINGwCzbKIImjct2lydK9FWXJuIHYTQhtqrlJeHtkpJwojPA+YIMKggONTjiAVzPcr0gCPh7Fnh6plItZY
m+OHoFqkPMSWH9BUCh9yfI9+98aSKKggwrngMHJSCLob1nLbZXU6pFqYUv9KdBOidM9FWQJjKCNNk
5s+P4WBLcOdslKSq2J3YS3POouxOWam1P9B0x0Zc4gnsPllt+woaXqtolEVsv8HJJm4vtiaHF9ttLtcwhdLt
xupblahpe2tzfa+a37e2SPkyLsisstjdlwZbjhENntrcsN2xQ4kacq+GlCoJ7Xdt12AsgtucaG6hg6Mypri8h5E
wty5U1vDRlwbbjbIOflYwNNMck34zTDZ1R0rL8TDfk4Fg33o1QumqY3PuhavDzx0lwurGBSobOmGOS2
w/xYg2gi72iwYHeN6xIL3k3CCErZq6jB26FKW881g5+hlth5uDnbZlwMDbQIDo/Abjo1n3fXKLfNd9GevP
DrOn7OjVwopQUvSL+dbNx+LQgLlDMDSP0rVhS++fpOqqfkWNg8+Dnk92QowPiilFe9Ux0qiW/+QIaZm
XfV3lkYKIrIR9gPA0Nzjv6FaMTStETBQbX2a4voa8b50tHgGk3ZFuYwTynjfWQl3HoqzTwIUXx5wvvB4m5
/ihCpvXmLevrLPR/MsulxuoFadWzNo8Dw/SRzYATFs9ZHPwug5XIR7NfUQHxbgh5WFfMMBoqE1aDEILi
O8Dh0J8eXCz2mi7ARBCVof4oIggaxO+n9UqNVegjyQm732CbrQycGHkE9+M3D4J0e2Shb5zGy+VhlRdu
ZKbKhN0ghELJfV9dkzDwvH97psgEGGibXMGkLSIzj5JHBsueNUhY35upBD2OQYz6iGERQTUgEzCk9Oij8
uNtokR6LI0sZiNkZbkWuD2MI/QzAIYYX+q+b29Z5Ht4if+lyxDPj7Jby4MShp4lIQR1EtaiziqX++/uBBO/QU
gls2NE4KuQXilOWGDU//DZgiCM9UfxkNLruXBjX8SmZRWcBIR6cVEz1paXOECJ53iuwRbGWla2jNVFbJj
3+NN6owJ11jQK5HI/PkEuOYhSobEavOIN4KkiQPFNHofgh/tOxQd/ZRjnaNKalyKyzCx41UgdmZFZBacAo
RR1ONYDcaHA05nn5Af4d1zgp12iwMS7K0CkztpV12he32I98h+RCquuhxWfLsCwCBqgkc6aMouEzvuN
vhtlZcN4UMLMobEiSPJiJJWZmVVwChBCpSA3RQrP9AS6JDi4uvA0p9cGaAFSLMy8vLMzMeJUncvlkk+Q
S8458UdG+pOUgHvV8oax9n6JAx/F+G8VX0TY48u9Y6zJzY0nyvUOL+hw42Dyiq0HIVSYYkNfoydOGiKz6
YHet6Yf0LAz9XFeCOQkiWULZnkmTSniPUD4NUdtBD8nDg7Bb+/5IQjhgRY2sOaKO4F79azie+igQyIcGO
hv5g79lzrrC4GGwWCN9QwgRDjTh3Fo79csTeIoDDx0uxEvCKM4SbP83sIVz9E1b/ZRolDycElb5ikuduLw
ByjFRaewZIbBl0N0TXlXk02h09TwvHEV/yuCA31XPJKGazMPnfV3Stw4GK2xngI4cSRIrvUwnVbf1vfikgKR
+T4QX3op7nXbM8Y6NFLMf6rwkQ6TSzlRXFeX1yziphcCTa0oK8q6mxgd0meO8TgTQie8Q4xQh4vSooH
duoHOi5v1lNsL+u4NJ6te08hg/U0sCI3t9jcMBzrBxWSN9RQgVHSOmAgOKK6pyiJLM0RRVg2ntwYNOpP
9bxKESIf4tDCkuCyNIwEYn3e9Nx1E0rBq9RjeY0LeNgcC6zkoxNoH1pZX3KvQ9yjC80PcsyvouzxQ9RYaVoi
Hg6L/keJG1twyQyPGZ4IcqzI0laCzvhOAD5Dyrll8EGVmO0DrQCUODO8BFtuUxRVRlA0sehhHbIAT+kcRV
65Rv8VMRe0R0kUG2CzcqyzyfQ9/RRnsGWwZj0MYHYJ6UvFBifuR33J7pMJiwAVOym7ob5ccDFK4rgi80
NJoJe8xkfYAOhjRE8fpbBLOpkY0kvL6YYQOX4fRR4I4ScFefocrTpRgNQwuBO7V/ZoJXO9yz0CZ8agN2gyu
F9Pav/fEQcP/NzkOHF4Aup7SZhjaSxRFSZqCC6pBca/BWBVhB2rRS8Wn/+E8dymiAlJ8i3YYu0voH0Vgo6
Kq0UlAqH5LFUdFZu3Qt3VVw14tVXy5ZxCBMKuN/Y8qfv6tJw7HzLzhNBxeg5x0OvZV4gFIEKd5gQZbngo
vFLFIf0MQCvBmI4tDGGCBn9z78TiTw6AVY8+6e471F7ppZK4xvV8v0ojnkAZ+jqV+1lyROXEZ7Rl0WQiM
48CEin+6OR1mQl5feI88TyaHcIiEkQlUv2W0L3jIdHaTC6ALPcB89uEgbbdxRmnajuPIOlFvopd2H+3Lqzw
BFR+1Xg7QdmXYwbNNxfeDMEnTidIrDDohpjPAINzl1gxQYq/3JBxW4DEE+paGwGWQy7kA6utmDZj5J
QRNyOwQBGcPeL2TW3+QSZdyaPUwMCbqTfQ+OI8BNLcEbwhqvQJZJDaNWBFhf3r38fnzghg5ru26tpt

+VyV6pEO4H12ZWETqtkJOIu/R4yQrTyWolXPcBeZUVUqFjFKxan8y2o9FcENXVw0b0JnvexpdBnj6MEK
wnW4I7s6Tig80oN9s3h+oxMnFJrOKn+CiffBJDgY2Zj8fiOctJ5GTJcFRrZrCGyAXK5coC5FwrjtrivgYBEeB4J
pbXjQ9pGPwijlPk8pECA6B7vs6eRSzzYCP3pWitzPEI4J3xPeDwJ/+mNXVpOrHLvPdJK3DYzmJfH6zbdbcn
kC+AM51P8rwYklwdZHEaQHZyaIgShOtEEH37S3GmOF5VHz53H8FbAXqCM4ISGE+8VGRB/4i/bfviiA4D
AL5ccGLyApIvPa1GIYiys6qCzfYZ5m30jR9AlfWFuET7qH1hFunu4SO4A4Ocihz7CCAnZBm+9sv8ILgsPML
tgHoRJHQ/tmJMsrOyhRjyk7FnyACc8dJJHL43oV9FIf6QSCL17rLAaaVPxGcR3lhO3ZoS0INWXFOmBFE4D
PpHyGRyMER3GtAgsvr4RBCvSA42YozKyqomdISTyXOHPvObHdSc2zw3gnX3thBIbtAjltR/UKTFQD6Wjx
BGDsNeLfh5n7dPyvuoc1m/vvN9aPgEdkuHMEdGzJLvb9Gbi7Q80zQifYVv9CEeqtAZ4LDcnbehPne9LLEf
U8tjhNc4wjuC7sHyu3dnhwa4n12AW8dhrsKgkOVbu52IV1xdLc+RQS6wZUta1LPJbVKYJwhbRirk70eFgc
lcASnkeBUmKiQprFsaMf6ZtcuI6LXDGvLeJ//wEzAdk+0f4w4u8NvCM4d1Ccfi3KCI9QPo3WI86fKWEIBEE
+tOz5bYresOEwjct6mr+tzneF+eDiC0yTWSJlRll/X4c6eyFYqcXGal62YlrBPS9KddFnT4eTGEGBK663pm3S
X999ULAwzNZ1UIC478VuJbURfA35BNwx1vqQYbpISOVuivefJLvFUR3Autmw0sCipbctEjSDbAulaV0X8P
FpddsNa/JEE9zxoS7ji9suK28VYNx88U9L5rg0AoWnT18pMMUsgfCzKTBApECPrVqK5fW6/ycmMz5aQX
UaUT5EQKa0Haf55GMzpVW6q1pEhK1Edwe3sY5Hfj92ylUiT0P/0k5DZFXedu4wQqkTnfK5EdYL8dYK8I/
5jwxkgmnws8/cLPBy45/mv/QqmP8m/Z0vERSW7jCidvSUrUbvsAFW5R8LDN+nmMhwZqME5gtNJcL6
Hjdxew7+7xnMlbtllRGlWnKxEdb4mF3wxEU6D009w9EORdvUpniq7jCidvaV6H6yBkxwj4N4hTYL9/P0o
WYFvDhHw6DKidPaW6n2wBo7gjIAjOE2m2Z4Exznu0WVE6ewtR3CO4EwGHJNzlWpwru9oogrMXUZa
hbO3HME5gjMZLstJU3rE/kSx6DKicPaW6nQZa+ASrEyAajG2CIqpf3+Cm7PiVM7ecgQ3waXImwrVhphF
UGy8ayA4/mAhyfHZW72YvbVlLo2rVf5wRY7mQrUr3SIoDr/oIbivs+I2OeKcGDvqNxeaxM5EmElwn7qM
yNlbWxbgDDFnvBsLbWJnHhRHyfTtNAH81WXkYx2cK90RnMnQJ3bGwViCE52UiIddRsTsrWiIj1XxLtgErS
fqsA7ukHSJtuadXjT8BY6r85XzXV2t8lFO1GEN3CFZSnDCFUd9CCH9+cNu6bqpt47gDnOiDivgDsligqOKC
E7hLtgE3SfqsALukHSJtu6dFoVbYKKyYZuJqnAXbILuE3VYAXdIukRb607/VXrvggzqofVEHdbBHZIu0da30
wSwbJ5UFfHqBSjeBZug70QdVsMdki7R1rbTX7a/XF2EoHgXbIK2E3VYD3dIukRb005/Waq1ofOl4l2wCZp
O1GEL+CG56Y77i7YecVgW288jaDZ0vVS8CzZBz4k6bIIjOF2irUEclu2SVA0RVLwLNkHDiTpsBR5S7g5pf9
HeXxxkw0ulY6AV74JNcLWoBkDO5+4uodPgvuwmwqqLEtHeu2U5aGqe+pblrpLBEZzJkPO5++L/7J3beq
ssEIbLxqgYdxEQNPd/nX9BSLPpyr+agkHXvE8Pmh70iQLDN8ww82Ii6H4xBi6rRt2zCBV9wzedsYV7wzed
AQP3Rk0O/J7Ay3g/hDb9UQ2cPXyjS+abCto20P5nJpToCpgf34xoy7VkoODSJbAjth9CO+8RXVRk753GavyMCS57KU9liLewJ5yBU4L9ehsBogGHyCuFX67j1dad/CHWYD3JfHP1LbUS/ZL59qtWM7CMHWD6Nw4YuJUSaL7+X0YIfQ2CnsZOpZ58hXKMfnv4Bo6YA5z3TQMGbmUDp7vqRVh5oP6bPBa2bE5CaRs7JQT9rhUqHKV/AeGXjQPDtNLC9i7vrMcXEd1Dh1YbRsBXpclt7PQ3XQKjv4fdACtnE8AwrZTltAQt+nF6ldl3aP1jc5kwmW9g4L4A32fz+GES0DUp7sI2WovmVc9fZJy8gXvSHjBY7DTee9gNgdtKAnFwbeEm2RfQ2PehrW8rwh2u260kr+rmJdqT9kvpvmpIL5Uaua0agnFQ9zTGWeRuAAO3FRAuudbQn/u7ju6j4mU4s48wPeSvUdTKL6XbqiHDqJTsa3P4Fly9gYH7+AAXdesghIp+nMEF+c7wqzFslhP6BP8Mm8R7rRXuqoYEz3x7MHAfpnGNanIa4GbEfjAK7nCEFPnUWfKcBOzQ33d0F23ILCeEyUvgawN3VzXEZ77hgLkh98ER1mt9quAy3/1VvqJVcMkxccAFWdEBCWHgMML4rmoIwaFjp48LeeJ1yDexebzhD3VXGYgERMnWTJ9BL4BvXVTjsYavGvLcFcuOwd/E1rHr5jgEu6sMRMIOVD1MMFArKLjXuPouVgH6flm+akg09Zbcm0gIEAYbAaR28kkAtwYOx6kasok3kRBwtLMR4LD0WTW4JI7Wrw3cdb+siLHT6OGWzQMGbitAuPtJNTjV5O/XtdcGjhbtcNMvK97hW9yEma3jsg+CXeUDYgFnLKnvz5cRasuA/bIeWTHlefuYanBFLyFBPnn88hFg4O4NnEjjCPkyQl3dhuuX9cBql9b2gKsGp+GKY/IghAgMVcr7sz/mH/kgolUN2YiaTQWQBdvBnFMns5jTIK392Rm4eZq0jZ2WkTPfwMB9AbHl7ZOUO5YGaZ1L2sXE5DzPerT3TtdTby6inE7AJRGSirIDn8AO/QP8/pzO9SREykHp8WSrhtCVlTZCH7CYE46yAz8dLPTJOgkIaWKLUya2pIt24K5qyNpeM7hjIAq2zJ3ctoXGDNFz5FMCWe5Nflsmsz9nRcliVg2xQNXnOyALbg/Y69RuAt8ZuH9AyqEHLg1iUrq+hgmlMauGGP4y5AIS/xJllxCY2wCXNITeXsa5nb9mcse+DfQ2EPKPZ5/Pf8CXFn/JXF9Dpo7ImrHTPybN+Hf2xu/zBpDj1uSroVxdTwM/xqchmOEixNaKzS3WIyJkrwYOWTC5xz5z0clJdeUhEQWHEI5cNcTx1CE7ZgThf0zio+/wK+ZU7vrZd4Ifrokz8gnN8qqxVHlGzSze5xy2S9XtxpRm9sc/KyGMT/NYH2gqJyzvHAN/pO7DULcuKtrtDugU/tXjuU94OdSZNS//GQ27YcyCpkxMk6yLsmRV1Zw454LzU1NVjJWFkXLvW15RcFPXqdWS+ebxTrfSw3GcZ8koSeap3/j+L2eSXZmZ368lPsL7DUc5hf8AdlH2WQ/lPp98ZyDjkrVi0kN34nIclZ4sWo2j5Kf2yMzV7nc5SDFA2ECoUat13fbmqc2z9m1dV3lGDqxTxgEJ3oVvk3gDZ0pqmVhHVlR12zRtXRUZdtYtEVc+JAiZ0UdO4tMso07i+xqKk2gLnMYZLfB/Bs6UX55HKUettRody

nyQYujN7e79+CFLLhnN8kWtDmLU8/k8n81NgcHq1mMntOZNvsNl+wIubUZp3rKCVaxqey6Mxu/b6vNj
keeReha+ESfe8KJWCy/xC6tb6SIIUsqTBJ7qcELzetDn2ZozO20/MRN5EFJp7Ypjr1SeJzYuVmr6Gji1Os9ny
zwvulWaP8uEsuDei1nEWV7LSYm+E+NF4i9boejb2lz/35WIM3PEzpOsYHXTdGKR+KJrmpoVGc3YSc2qg
xmSPAhhVybWtNi0Dql1PIwaz4rq2J64XNqbGBG3h6C49ThcX4PzH5kncTxAJ1CDtVz0UIlp0lLqabqW+O
Zvgi8FnNbPQY4EQt64saPZ5LmczpZJcisAqqqTWpsZsostf8eY8wTXo0nJU1O5kIJV5/YwuShZVdsGdbbIIi
FbH1CEXF8DPk7zeeE/9q5DyU0YiEaiF4MxICRx5f//MtqV5HLFNsVnC3iTycyV5E6w2n3bf0CvxDdawnkn
QxcXB1HeCsVwOaqzDIEOfosqD6Tjj4cAPA6EAsANrYHLI8e3Zk8a/c6laPeJvzmorw1C9IrN2swZCy5LNwn
ArhjmNZhp18PueOIwUTobhPYag2NVnoTOn3cy4IEhxa8Y5xwcUl0+5Pu6oKisWiByokaKv4THpU8cwvR
k8YuMwKdlvYsWWyCzEGBqNCrqTtgF6Z7Nip2nyH182Zx3dRk57oZYhc2E0m9X0WuV/0KFIk8BOXIZDh
TfphQoAEwFBuALXINTl7H3Z3OGHwd1YmvUr8oIzJpZwnmXDEJpEOUN0m07Z8yjJwVnP0ARBy+2yaPA
5TdKFII4P0BM5Tak0vpx8MQCtOeDEMw2lxCgxVl0OtV0QXb+6UWGnLdA8X2nDeCR4t+WkY3ivz4I9aK
y0fE1m+n/4qLadLlSCxUToiljh0k5HkadGE3zbfSSN2W05lp1QnW2mUtL8an3jeITs4qagxy9zniCccATR1n
V8tsUH1T+Ng/uhUEI8jfe1UVEf9+wCV8xjh1HDufsfSf6xKJ/uxO9gPOuVsEhf4t3tejRXYeMy08KDpi/+rY
WInG72GkOp0+cY4T2Jnqg+G6fd9EgBDoMhUkeXCkBIQoe8Pai5lD+6uyFR8Y6QL+hhisjhznrJBj+1ope1E
XyK8VHC4hqoROidZrDHU98L8V3/LxLBiitIK257Io4ILfSQaYuqOgkr9PAzcA7IVDNNUi/oYbLVlrthNGoeG
ddT5N++rVyGh07wRWHczYuZU8s3+6GdPm8S4a2VVZh3c6NEmjdNArRTZulG44q3r8NQd/zap0NOToa
tWu5LvL2r1N8TKnG+N07V+NSlr/JIRRfbhzuJQFWN9oZl/Oe92PHQIKNjlws/yFKQSd7lRsbCMn2SeCfK7i
FTVZB/L5NS50/vj0UkKCNjPcMOjRfZYreMNg7Id4GQTp6H5YNPSKp4Vivf999JTpE33LevNIYoXuBBSIFBI
+HQrIiPk80LHnG8RctbhUWRe/0tgG0CtFJE2DuhBgoIb1w8j4sCcTg+yo4EMav7ul1K61EHgcfvsogyDuBJ
4br9zYCcMXteRc+xP3ieUHEUruc9wwFwhY4dGmb7HUGhQ4A3Im45G9D0fMydu8+LAi2IYGSb+7HsIEv
dtuUk06IEsB4hIMKADOgBXg9C1e0hFQckwb36XNCTVKCVw7KB0pIVLYjJATWNW0K7nn4UcF5wMXEsI
SBTUwIdeH/ejnnRGDOeLj7cXJCbOYY3JgQhoL5y5ZoEJrrEnJdPpzjuKaqYIyE9MLZyoJlgBh8WwQnmmEv
xqoJ0Tq2Ls1evRHSC+jtNcdSkygtyjwNF1QaQAy+xKOshNx9UkKokQ/nolKTJKR3tbJgEYB5IVEcx4p0eOeb
UPOu76p40GtBMQBHr8tDpzZ6o8OVVqMicAAY1k2op1v1P97r+lCmC6qOIwYXz8vEaAdJiJUPXrhWKgK
/7dgY7dubk0GbRQBVWZyX+/2+SHTo11qrkg/fVW7F4JVWIt+E3SqQ1yPFVwlwnZul2Eq/NV3bdu3+c8Ei
PTbaOi62+xKYEKPVUVqXDP5ygJmhj0NT13X1aZWSWZoxIllgVaOoXmYl8k3YiNJ48dUC7FEPT//BurrueF
MuZ2PDcTXW102oVwzZ9SicMp0uyYc+czIqAmeicIlbp10I0KJ+vKsrWR32RsGd+MyIcg9ghGHeiTp3SYBN
RGlEDZyFxFonSsFcVPssb1i7j5ei4KAEGuDTCz5TdcNDESZKy7tD8sxFh4OBnk7GRgtIz7LAqaDNQgBXMPq
oOa+Vi5pGJmpETE8Cy3xvuIKjXsZEW8ZOKbhp4qvAMqPgoiTLko+G1UthcFAAneqVUcF5FC7RySSPDJa
Q5NBxWIns0IXXdaHDa+CO6JyLOi4CGPR9hxboMk9xOu3JAWl4mw5WUTg7Qltot8QXNpG/jUffGQvtBS
FOVeELicFBQj1+rxljbV3GqJSMGUuZZNXgcg+UuLIVLPWoOxIyLcUAcDDquARg78E7E4rBlXlsS0IgH5hUjI
1QUoQcLbRDlxur8vMpCu7N5gXVoanixF1XfYaLGFdNqJ/WnPPuUsF5ftZBl+XQS6u+/x+EP9zKs5vV1qNjt
ErBNcOf1YapwHq3pAbpbTo79cBWwfF2P8LNJJ7nnIWGU49qwvnRQhMaqEqRpj0kbtUC/s5tP1plAPM
UXFRyznjFiI4EU5DjWJ59uoKTTbEpuD8Huhrxe1PXVc27SjtVJo7WjXslJn4nu8x3R8F53qQquHMLDS5q6
AOHaz/deQLXTKAffnQCFBx2Z5xXwY1RUm7m2eE3HdemZSFd7N5YBGgQJ0kS5xgWn1PBOcTgJlbB2R1
KejdFmqWhykxz9uk75IRdU0fvvBcdq3eJnpliq+DGuZmu5tlV1nhcG4OBdK97YxFAyhH4NDlgYYNVcBBh
accruNY9BTfJAdEuCEUmWFT7NH0HBue+QOM2heSdiY5xdthF4HSfnlc3PMv+T0vXWPP5PEypgtMYUZ
OwYSIwWBxlWRL6Sc2aclNwo6EVnE91VeGBKXORuD+pmujS5eqw270riq9VtnnH4xWc55qCw985nVR
G9NaPelobJgFfXPxeFZ9J3rLD55pd1FkUHLioRdu1LWMqounK+a8AhwckeRZF6HQH/r/VKrhJZUQK5mm
tZ+TzCwC5GjinzaHmXbnmJMMcCo54CkG822OzfeTiUMcfoNMmYfLedZ/hpuBGwzyt1UxEfQXYLCpjbas
cED3/YoVlIp73b2qSwWRRKSGw6B3GJS1CgjGLGidJAAqObQpuPKyLqgd/eu6HL9wAIVTFWJByfOpOBjv
uckWFvnOUiUir4CBrE8ULGXhpepXrQxJ/1OCiwhVdX5JhDgVnGdxqZj6/BpDDhVlRFonmb3Z6/ppatWYt
9DWS687p71BwHSuLQ6sMng+ly+srE5k1yUAojnwOfHcYgNsg1EfKYfjbSpvt52rVgue5oBALPBnsPOs4w
+YzQtZY6DtDmUgPt8kQinRX7hInF++4CMuVLX9b5bik2Zrtl6fgqK+b7Q+fmDZZZavW9ELf/ljoG0QmJBQ
uRUheHV+CnisceDl9XFJ/xnYXFl8hFEdAZYkdaT+12Z442mw/Q6sWqLRImYuuhaqsbULc3+Aybb3GkeW
TB172csnLfZUB9APfo+bDqeOS/rmWZZ+jkMgkVSihULXQ1o1gn5uT+lc4Uo71Lp2ZY2S5Mwp9GAhSfH2
8lQ68nEPB6Sw7pVGxL3NVNt19LqVS0j0Ao1nX2sCJ7fbSrZjjMBB8ld/WBq5pZPl8Ay+hZiFLUshMfy5o75
prWNvi53nWBrp04mGwYcXVLp2ZbWS5rln4aHgHnb2bfnsSxq13I5SOWib3fExa67uGxb5EYc1rA6emo

U5LZyhO2Gu6rt4Y3BNhLJYELjaEwQHvkyPKS54N4zqNq3TqwZVfbIrhDJMXP4NUuLn4eVol3OXaQA9GP
jNWLmNrh5vQecWG8zaPgpP5vqXe/DBrOXcyozhhse8K1/qifJgo7d1tlacYrYPDHyfIx0lC7PzFIPxg3WFTc
E8E5BWjXc2FTW/f8w9omFWc17vIwUJXJX5jw8jikC1hNNIQ4Ou+EqW9HoFzcHz3tDSDceXN/MU8CYO
0VT7qQhZLOgkrjJLXKdRAUXJb4P0gUSLvaDyKEDgvG1ELJ5k6sWN8ZDLgDRtzFgf3VcgQazT5iA6IZ2NSlN
bGaDGSp+cv7jYG92Rod6LmEl4OvWVwsVYYFITgtaPxKAIKeoQTotyPJPAdCpjPA+zfzSCIsYtAPu6g+NSGP
TIHS1z1fRgVheuFuRN2/iLT8xcTF4nAgkAIhZZ7XheKUnv4gn5PsSn1hguPRVNGLqo3mxms+EAJ7nnlWEZ
wHhAFTCmZjOhNim/6YzAx4WQzGyEj27WgI8jukyUwfxFbtbYs6tMBwxvzhnO2zyLfQyH+jb0RiC3smU5
KOPrWcLhjNjCT2osmW6eg2qiUVli3cu3ooMEoVOleEdGJs44aqtVDteCxTxl3XJj5i9vQyyeDUC8uGyFYlcc
BhdpdjePXNfQugqxiEsYkuSi7p/NG5SAN1wNjdSxePhN0qciuBSc1DiEOd43iK+kJwlh/t5MxWpM5Hl5K1
AtbVWC7eKOs2MYlvQSUHCKHE7wpMe9l+/IN7AfUA0XIpWA7p8PtBDnrABFWwptHLp94CuC9BzFk2ts
qgzjcFYoPJlBR/FZgUsLRed0mSTIwkyq+VBUQiiOfVxi2fT2g4Y3KuuO8LdM49D0Feq7gQHTBNKdFw/ve9
eW2RAE4HL8zlyq54m/UxXjSLNBRtV0rBDsoiu/jNf5O8U2GPc4PTAjgb9RF/nZ2XjmE4ssvjFU/oW2x1ms
AjXSUFjWXvMVIHDoil66HD6aZSyn7nuWhywoOzmuc7bc7IPFer1hSobMSOJyQR8VFvK8U38MAHSpCK
ZoyCXxnXXrLWQckGuRXxrotnXkpgMCC1UL/87DL0iSOwsDDsJter5KkWXFgUvKu4z3H1LfD3JtQQu90Q
3oByRenGetkIKdB6eB1mUEk7icFBxQ/K6HCQhxypzPOoJtQVd9H4nopfoo4Lmnks/NAjgab8FouBGfNoc
xTqIvzfUgspHl5aBgXUrCqUCIM8QbyuJf3+G1EGEiOi3smfwkon3E1nDQXgMMBxW8574DiB+SCm6CKIx
h964Si+E7t0volDOcjxb9Hw/Wa4l8kSxc48tlxoBuK2dROgIpr66rM0gxRVnXLuBBCCXcaJ2Unu0Ma+A/Qc
OSIhy5dgwsZJhAskoJfMdO9+nIDTtkDjuoWUDhoXNYdROKKLI0j2F9EER7sj4qR4oPoKJ7fVYnv9GRQ4nn
3JqJ6oVS+e0MFVggww2CmrTrjnQG3Kq8A9yTMWilZmQQzE3Dj65z+U/3RA4L7WK2e7FsueLvfK8f7SvS
tSKPA7bs6D4iRjrIGo9C1VVlkkeL3+AcKIqq2Q4p/yEum5CMLffo4Dvcw0/ctsdLf1m/KBK6e4jsBgtFj65CC
uGqAQFunFXgezHg1jtuMo/xsMOcr6OzCDGcIk5IJiCilSbZvOn3YkzDrQ3c64bLxNysdmERo4FkJ1jaKyBkU
h6Zl8FlMUSUH3ot9Gj4kiPEnDB9+ClL8TohbFL+ry9ijG39zAUTBphTSrKyatlFoG3BWTdpBgcJM3J43ZTpn
6J0QarLqEPjzg0D9ZdgbobMWHMCPwAkZip9lUQjFL3hFGyZPQZUGr28aw5HpZpytXcA47UmddQYnlQ
dPLNrVUvc9zPrejhzfkiX1wWMYvpn8XLZC8qYCit//Hn3bKYq/JUtdgTGJFHOq6a4sFModMDdb6ks9aHS
CV99W2Vy16rZPApQrRnMAGOeJgPx7cyqZo/Mh+SELdQGMH0TZrti3rEOwdl/ssijwQYHPc8JFAB4FNuv
twCG9pPjgtKpn5oPkQJRWtHk0r+NGEPTxDF+nGJKylT1vCkXxK/RnhOz7o2brpUC/poLo28yxmg0PhY18
obJB2IIRK0yeDwvheix/nYVcEerpHwBKtQBFY9VMAcrVU1RuNjcR6WBUNuCfQpwItScUMVvFalVrGOBX
5r48ToMQYp6VYb0IZLs27aDgB+kBgxjpnKF3om2cofgB/rErj8h89k8BjLnSb5gryeIwDJM0zSHR1onext06
SMLlaYoJ9ll/+oYHA+sTrbto8U+TN0rMyMi0BB+FQwMTnbpXixCj3IwYHV1FyZqTGPmzkDhCbFuaxBGfIJ
t4qpNrfHKO5744SwA+EXxhmFrQwHSDMX9oBiMoClacLo9nKve13E1b3cRS/ERbX4UZt7sZ/tbIXptAos6
qpBOFkym7C7YXxNKY3k1GXAMxoJduAMJ8gw8jk6DGAjjcRCcOKRV2udboCJyC/b2UZ47AHFWTKIyGv2
F3KeDX7Mbsrs8ScEx100uKT+nZMwviohVSl7/OkUtFGbGhBOUfN5riN8orNqEEOosrYfmbbtQSdR75HkI
fNjlR/AQPvbUquIrzO2+7Ui+b85N833Rcc7gpHN04PTGEdK+FcpNwcigXNLHhb7yriwiPdX5pT4Wr83KCZ
eH4rMgZ7DPTn/axCAdUhNJw/sTXpovQsdAY3WLjKvbaUZw5GQSSoNvSemnUM7qgluIjv8cS0Aekvzb8
Ka6k4rGuPU714MvY88aTK+BUpsv1WjLetEtN4or6YuJgKI7xIY+Sv64wXgRuUXzwU8O0gOSjaKeXSePPw
cTQsZzHUvzeJDiaucp5CDX8TWjljCf7uwKmDS8CWxOFc7+KJBodIcO25jDJTbHt9Yb3aR1TmC4NoqQws8
ro9yK+Tbfdj+8K7ksVWQwUX8LYpNHSceJUpsv1Wq9UEk7mikjxkb9xyZtTdylRsJFpBEakH1KgsuE1gMHk
M1EYGSHTwxSzCunbVfRQljKlIQZts2kYn8Vx2vA72yUEdVKKxmS0dCAI0TPmQFv2t6SDTqo8R41tzXaZRt
hdulH8deIkDAK7GkbFQPQYnvwwqaX5Or5ukziN/Bl74zbcAYyanYUDxkjHOcXfXZURG6edSPFNM5oNvG
Ds7c/HQGx4ERATjsV2FjYqBgJsbPpQmq+4vk1CiO6Qb/zt4dD+q03oQK59VISMEPveblP8iWOtbIa9tqmz
TUTWDuM8VHrU61US97ixgrdxvk1C4NjtWUoXNihc13CoMHAS/m5cmpMogA19+GDS0zYJPbg/2jpYN
oAQn4Ti+rz+Rw2GvsT1OrvTVduGg/wBCMKalVGVjIQYFXknxR89Wh608clYQwvi5ohuuKD1JxL33UrPud
pD2tUe9wLpmw14d/X0wuQN9wJ5kQ0MjKD4f7UcCNnbZbhlq4Hc8CUwK44xkO8y/KjlbLeByszzdJ0dr4
tt8sOfghBqUztj4rR/s95R0zeTMNvmN2/4Ih2n8gtD4n7ucXrEet3rsBPmAju7d2LBwobhQOVxIR0/dog8c
UE3sXV2uHJpm9+84dcpYa2NxJHbGTNUcFHZyrfBkG0Z3aXg9G9h6uzQhZ5YcrphOLC8+lI66H/2zkU3cR
6IwrWdewgQyNXZ9n//t/wzY7sJsOWS0CXA+aRWWlVqpe3p+HhmPHP5eZMbGh6keXObxe+6Jk9vWH
gjZI/dJtE0JeY3gx+2Odh5/XYA4KVzmgdrJrSA6Wa6dp/4ly+pbkZJmGS0KcCsSsHkh3+OfZofHedpL1t8RS9
aq5uPQF1t4yu3LQs3/ilmjTQznx2Cl2VokbQjnJVJ5YuzFdR1PSG+9RGuXl+spJqfbmaU1C0vu8Pc/McghD

zN0152cUKQQip9u0R0tY6GQsNF9+b67Prom0bzBkeAV2U8r78lox+dP6d5JHS6a/9Mot2lwdkAJ8S4dtq
azViY/PAorDpcJs7MOZAXXFz/takKIX18XHb4oxHsVWsWnmO3Avg7fxEL3RLYR/0g32jC9cNA8/7PCFiwS
O2MkuFPCp3pD+NbHZvhuBGjCW53VojTx+WgK8ThoYwRWeBcS9zY7pNgvJ9HyggVT8rA2SxcrMR59+Y
FEQdbdymCf3so5r9fjhMGZ10c52hvbyIaWok4S3vJvfGMkiGtAomAMxwmbNvBxZ1mbXvpKT+tJqmX6K
rUV/Rdzrs3XlXNT7B/Z/cSuBpWx3HJx7m4O+doL2VpB/dGIvkujGE4L2CuK7mPXZzzcQc9cOH69h64b+r1
6eAkQRy5N/5DQvZtETh1uKadLHEujvjb2r58UgaOaPOEs7Tn3RsHW9e8Mm0YAHgrhqbJQxcnrHzuUmI
g2t1Jr5P5CUfujf+IPGRWFgGpYGi7Hrm4U4tPv9twtZ+sENqmwPq45N5c+zEWcICrILkeyidLYrueQwrGyj
crJiWQDW2RjQQsGGkG9By7N4HJb4th/HCuHVzcicXn02pqicGVGU6jptuioI6OYKXQHwkclzV8LKHc7sd
0Pu4OAU4XaxfgnHfjEHfi3hT6mhaFycSdphDsukExfqdcTuiBc+iTF8v8kyklLI6SKPOXGYG34vQSUORrtnF
2QRcHpGnPtMYnNH2v8T4QL6C1bofu7X7L5MB9cM21By4udJlaZn4RqqdKvaEMNchEHBy9qK+DCYzTu
Ou84IBT7DZpaIuZnHOJ8qklMkaXiaeGTZy2VX5XVE2rRwls1E4XhxXAqBBUUJ+i8VvWx5kAV/+ZTle7Ovv
YuynFRY5xCQzrIcEEvl0chThu7iQbl5KP8xQxowtuOKFJuIN3y/Kisj8J7m3JjFt5nDZsppYDHH/dC1ZzAtyfxt
TZ5di7WYtfj5uYkH0DU3AujkU1Cj18ViupvGTWBcSd0NLk3dgofodRuLelM7TzOG1wpjbyBW/Fkv3nSaO
0TursxODd6LCtm8G9oTsSzIAtFKeLzeWRU/9FnpGPC6M5XXAEBTgKn5HxbnWrxxdh5FUWjnFxThs2U7t
Jh4r7nC64oc7OOO9m0yVNDfcG7oHokUfpf00ua7fJNtN7nIYA50fJakOhk6PbUMrAyfwEfLu4MEqG0M
MVdwpwc7rgRnV2KcfezYXRCG9bwF3gADc0cJDI6CJZFmU9o4Zq5vomSbrJ9yU5Q/7TyGwzCtzbkyDsb2
sUgHRtUrVRmGznVNlNgJOeH4RhRN6NLf74Igz3Bu6BzfG6FlwXkrSepd4eXRaVC5f70eUGLelPhO2MHK
X/tfFxZPFnVdk5wCk/TNcbGzrbeihloHIK7ohrxFVKuEtlXTczujh7uq5t+4+mpgvvyp7KeE/4dHxrw7xStekG
tvjzquy63K4oZhalS4zkthkFuTdwZ4TzccKWBZI03ZTzBNyHuI7G7iRJHJlTGebtGTlswR1Cku5myqOpqqp2
4XJnW5SQnwW/ha2bcROA56l4ZqMvmbf9KvLddUNglP7T4hpxSRlhmlE+br7F163uOt3UfOFNQ9sTgplI
4NcY0nF3eKq1y5IkDn3b7CsFzNsTw787V3HnskCSbvYz9NHTO8CuLtZpmkQhezdYfPDbCLNFVd7lsb30u
JdJSAwjfwlM9ZtQnpJzG327tq6rXRbat1jIz4J/giDuEeCyUBrpwru9ClYbzPynWnmaksWXUsHig3+LUOo+
Ay+RMH41bMZh9mN7agT3jGUT2HQK/i1UbrjPyHLI9tUQhJw9LqmrUt6jBosPHsDceV9m6QxU+6rMHn
jZ8cBLKSUaesFj+LW1geAFuM/IctSewKOYsdbXyBf+7ZWZu3RG89IZHILgQczbq8UlBjyof2HusjYQAQ48il
mLfXmtLwLcKzNLH1AIeDiURvaTSVm4rt0nvkLy+KUx+iim6IMUUkAh4MEIIaY919LlJkQG7uWZkaVFjhY8
nl6Ak57jdM0m8iHfl0eIj6lZWsrRQiHgsXCv0+2XELp+eLh+vD5CUBaumtALp6t15EMh4MFQr1OY3doK0
O6zEC9v3gGhPD+ecEnV1Tb2PVRQwYMxlbLspn71TpcZ6mPvgXmxnDe3WvwmT9EiAhYAjY7wo+yWuV
96n0U+GtTfAyGEF6Q3JTE4hZEGHi6oYAE4D9deZ+I63cK/vRVCqnBzQ4Tj+LYJkcEAy0AI6fnRandVKrnT1
W4V+R7829tAhYZwdXWE4/i2ClFgAEtBSLNNqbw8gb9raSE5/Ns7lXrIw127n0E3vX/DjECwHDgPF8Rk4i6
ezVkUIP/2XogeY/H/XIGx+IhvYEGQI+Ot91WruzPZt3q/iZSEf3szhHDquMLiV73FR48kWBbk4cjEZT9n4jp
eohX6GPD1fgghL6jD0e7XMSw+WByCt8eYveZ107ajlb9dp1veXp+noY/Np2+JEFYdZXPW4jdFFvnYnAW
WBy8T9PwgTpLVJt+Xddt9r34r9/lmlSR8NGMHw1tCHs47m6fl7Ns6CX1PIYUBFghFOApyyo84xlV1XdX9
B0e3yKc9mVgx87awOjgTV9R102o9cnL9P1raXs/ZN/g3sFDMRmhF6ZYwjJPUksRhSN4N4/XfGqMOP4h
6Yax3RVHpITlbFDveXt/LROEMBIvF7DX/EHwh8Xzf6z95HNcEfQ1Xj7eGFcDCCNNsvS2rmqnK7TpLQ9+j2
ikq7GDJCEaqMVhKDgxGGlKxxY8Gix+xxaevQCNg4Qg+hTmiDfFOYns9IIxDGyy+zy7fWPwPZGjBcyAcFOj
g3cCIHy0+VAKeCxYySv7gCBfgvnOyQvVgez14OuDdwE/CMOUGJXogE/CcSC8IfE9+AHA6BjOM4thHbAN
Pi1RBksaBh91I4HQ4Q/j1ud9GUkAd4BnpZesFyXb7H6b3gr+9vU8+q3ofozEEPCUU1IKvz6ra/hcgwIHTIf
dfIW4Q4MCTQn1O0Weh2xwBDhxBqya/do1u9zGqp+AZ4V1KX0XbIcCBU20oP9lVTdMgwIGnhLoAguSz
bHWDAAcOER/Cj7/2VVnV+8RDgANPB7dyRp/F/+3dXXKCMBQG0EFRxHYKhZCAnej+d1lw+tCOdgFxzln
DnY/8kHvHMCdncDxcPzV5+OzzEMPZCo7yrCV7aK5h6vsYreB43KCO0xZwaboZ70x51pI9XvM0vr3HJOB
4uEHNaQ5DiJd5Xd8LOAqzncCdckjTR5eWqWvrvRrmV8C1Oc5TiPPXMgg4SlNtAdfmtCwpXb6W6dbYhv
D3H7ghjGNcBBwFqlbbFUOMUcDx7IVL+951fUiXZItKiarqsDVsvWZbVB7s60PTNOc83i8Z1Ablqfb3hq3X
4KkWzydsHc8fn/3JCHuKdO9pWJ+6/tZ6bM+z4jhu7ZJUBqVac027JP4JuFVd10agUqhqpeElT1Wb3c6cXI
pV/fCJBl5TtbMH4T+OLijcTsAh4HhZfgIAAAAAAAAAAAAAAAAAAAryDc7kKf9aU/6hAAAAAElFTkSuQmC
C"></figure><p>Explanation:</p><p>The above picture describes one iteration with n=6. </p><p>The chips 1, 3 and 4 are initially black, and the chips 2, 5 and 6 are white. </p><p>After the iteration 2, 3 and 4 become black, and 1, 5 and 6 become white.</p><p>Input Format:</p><p>The first line contains two integers n and k representing the

number of chips and the number of iterations, respectively.
The second line contains a string consisting of n characters "W" and "B". If the i-th character is "W", then the i-th chip is white initially. If the i-th character is "B", then the i-th chip is black initially. Output Format: Print a string consisting of n characters "W" and "B". If after k iterations the i-th chip is white, then the i-th character should be "W". Otherwise the i-th character should be "B".

answer

```
 #include<bits/stdc++.h>

using namespace std;

void solve(){

   cout<<"template <typename Circle> Circle color(Circle c) ";

}

int const N=2333333;

int n,k,f[N];

string s;

int main(){

        cin>>n>>k;

        cin>>s;

        for(int i=0;i<n;i++)

                f[i]=s[i]==s[(i+1)%n]||s[i]==s[(i+n-1)%n]?0:2e9;

        for(int i=0;i<n;i++)f[i]=min(f[i],f[(i+n-1)%n]+1);

        for(int i=0;i<n;i++)f[i]=min(f[i],f[(i+n-1)%n]+1);

        for(int i=n-1;i>=0;i--)f[i]=min(f[i],f[(i+1)%n]+1);

        for(int i=n-1;i>=0;i--)f[i]=min(f[i],f[(i+1)%n]+1);

        for(int i=0;i<n;i++)

                cout<<(char)(s[i]^(min(f[i],k)%2?21:0));

}
```

question

Question description: The city of Hampi can be imagined as a grid of 4 rows and an odd number of columns. It has two main villages; the first is located at the top-left cell (1,1), people who

stay there love fishing at the Tuna pond at the bottom-right cell (4,n). The second village is located at (4,1) and its people love the Salmon pond at (1,n). The mayor of Hampi wants to place k hotels in the city, each one occupying one cell. To allow people to enter the city from anywhere, hotels should not be placed on the border cells. A person can move from one cell to another if those cells are not occupied by hotels and share a side. Can you help the mayor place the hotels in a way such that there are equal number of shortest paths from each village to its preferred pond? Constraints: 3≤n≤99 0≤k≤2×(n−2) Input Format: The first line of input contain two integers, n and k,n is odd, the width of the city, and the number of hotels to be placed, respectively. Output Format: Print "YES", if it is possible to place all the hotels in a way that satisfies the problem statement, otherwise print "NO". If it is possible, print an extra 4 lines that describe the city, each line should have n characters, each of which is "#" if that cell has a hotel on it, or "." if not.

answer

```
 #include<bits/stdc++.h>

using namespace std;

int n,k,i,j;

char c[4][200];

class City{

   public: void hotels(){

      cin>>n>>k;

      for(i=0;i<4;i++)for(j=0;j<n;j++)c[i][j]='.';

      for(j=1;j<3;j++)for(i=1;i<n/2&&k>1;i++,k-=2)c[j][i]=c[j][n-1-i]='#';

      if(k>0)c[1][n/2]='#';

      if(k>1)c[2][n/2]='#';

      cout<<"YES"<<endl;

      for(i=0;i<4;i++,cout<<endl)for(j=0;j<n;j++)cout<<c[i][j];

   }

};

int main()

{

   City obj;

   obj.hotels();

}
```

question

Question description: After the big birthday party, Abilash still wanted Shalini to have some more fun. Later, she came up with a game called treasure hunt. Of course, he invited her best friends Jai and Shalini to play with her. The three friends are very smart so they passed all the challenges very quickly and finally reached the destination. But the treasure can only belong to one cat so they started to think of something which can determine who is worthy of the treasure. Instantly, Abilash came up with some ribbons. A random colorful ribbon is given to each of the cats. Each color of the ribbon can be represented as an uppercase or lowercase Latin letter. Let's call a consecutive subsequence of colors that appears in the ribbon a subribbon. The beauty of a ribbon is defined as the maximum number of times one of its subribbon appears in the ribbon. The more the subribbon appears, the more beautiful is the ribbon. For example, the ribbon aaaaaaa has the beauty of 7 because its subribbon a appears 7 times, and the ribbon abcdabc has the beauty of 2 because its subribbon abc appears twice. The rules are simple. The game will have n turns. Every turn, each of the cats must change strictly one color (at one position) in his/her ribbon to an arbitrary color which is different from the unchanged one. For example, a ribbon aaab can be changed into acab in one turn. The one having the most beautiful ribbon after n<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>n</mi></math> turns wins the treasure. Could you find out who is going to be the winner if they all play optimally? Constraints: 0≤n≤10^9 Input Format: The first line contains an integer n — the number of turns. Next 3 lines contain 3 ribbons of Jai, Shalini and Abilash one per line, respectively. Each ribbon is a string which contains no more than 10^5 uppercase and lowercase Latin letters and is not empty. It is guaranteed that the length of all ribbons are equal for the purpose of fairness. Note that uppercase and lowercase letters are considered different colors. Output Format: Print the name of the winner ("Jai", "Shalini" or "Abilash"). If there are at least two cats that share the maximum beauty, print "Draw".

answer

```
#include <bits/stdc++.h>

using namespace std;

int m[4], r, n;

void solve(){}

class Friends{

   public: void Hunt(){

          scanf("%d", &n);

          for(int i = 0; i < 3; i++) {

                 int a[256]{}, l = 0, c;
```

```
                while(!isalpha(c = getchar()));
                for(; isalpha(c); c = getchar(), l++)
                        m[i] = max(m[i], ++a[c]);
                if(m[i] == l) m[i] -= n == 1;
                else m[i] = min(m[i] + n, l);
                if(m[i] > m[r]) r = i;
        }
        for(int i = 0; i < 3; i++)
                if(i != r && m[i] == m[r]) {
                        puts("Draw");
                        return;
                }
        puts(r == 0 ? "Jai" : r == 1 ? "Shalini" : "Abilash");
    }
};
int main() {
    Friends obj;
    obj.Hunt();
}
```

question

Question Description: Abdul is taking a geometry exam. Here is the last problem of the exam. You are given three points a, b, c. Find a point and an angle such that if we rotate the page around the point by the angle, the new position of a is the same as the old position of b, and the new position of b is the same as the old position of c. Abdul is doubting if the problem has a solution or not (i.e. if there exists a point and an angle satisfying the condition). Help Abdul determines if the question has a solution or not. Constraints: |a_x|, |a_y|, |b_x|, |b_y|, |c_x|, |c_y| ≤ 10⁹ Input Format: The only line contains six integers a_x, a_y, b_x, b_y, c_x, c_y . It's guaranteed that the points are distinct. Output Format: Print "Yes" if the problem has a solution, "No" otherwise. You can print each letter in any case (upper or lower).

answer

```
 #include <bits/stdc++.h>
using namespace std;
int64_t ax,ay,bx,by,cx,cy;
class Geometry{
   public:void Angle(){
      cin>>ax>>ay>>bx>>by>>cx>>cy;
      ax-=bx,ay-=by;cx-=bx;cy-=by;
      cout<<(cy*ax!=ay*cx&&ax*ax+ay*ay==cx*cx+cy*cy?"Yes":"No");
   }
};
int main(){
   Geometry Geo;
   Geo.Angle();
}
```

question

Question Description: n evenly spaced points have been marked around the edge of a circle. There is a number written at each point. You choose a positive real number k. Then you may repeatedly select a set of 2 or more points that are evenly spaced, and either increase all numbers at points in the set by k or decrease all numbers at points in the set by k. You would like to eventually end up with all numbers equal to 0. Is it possible? A set of 2 points is considered evenly spaced if they are diametrically opposed, and a set of 3 or more points is considered evenly spaced if they form a regular polygon. Constraints: 3 ≤ n ≤ 100000 Input Format: The first line of input contains an integer n the number of points along the circle. The following line contains a string s with exactly n digits, indicating the numbers initially present at each of the points, in clockwise order. Output Format: Print "YES" (without quotes) if there is some sequence of operations that results in all numbers being 0, otherwise "NO" (without quotes). You can print each letter in any case (upper or lower).

answer

```
 #include<bits/stdc++.h>
```

```
using namespace std;
const int N = 100005;
double a[N], b[N];
char s[N];
int n, m;
class diameter{
   public:void circle(){
   }
};
int main() {
   diameter dm;
   dm.circle();
         cin>>n>>s; m = n;
         for (int i=0; i<n; i++)
                  a[i] = s[i] - '0';
         for (int i=2; i<=n; i++)
                  if (m % i == 0) {
                           while (i >= 2 && m % i == 0) m /= i;
                           for (int j=0; j<n; j++)
                                    b[j] = a[j];
                           for (int j=0; j<n; j++)
                                    a[j] = b[(j+n/i)%n] - b[j];
                  }
         for (int i=0; i<n; i++)
                  if (fabs(a[i]) > 1e-8)
                           return puts("NO"), 0;
         puts("YES");
         return 0;
}
```

question

Question Description: Winnie-the-Pooh likes honey very much! That is why he decided to visit his friends. Winnie has got three best friends: Rabbit, Owl, and Eeyore, each of their lives in his own house. There are winding paths between each pair of houses. The length of a path between Rabbit's and Owl's houses is a meter, between Rabbit's and Eeyore's house is b meters, between Owl's and Eeyore's house is c meters. For enjoying his life and singing merry songs Winnie-the-Pooh should have a meal n time a day. Now he is in the Rabbit's house and has a meal for the first time. Each time when in the friend's house where Winnie is now the supply of honey is about to end, Winnie leaves that house. If Winnie has not had a meal the required amount of times, he comes out from the house and goes to someone else of his two friends. For this, he chooses one of two adjacent paths, arrives at the house on the other end, and visits his friend. You may assume that when Winnie is eating in one of his friend's houses, the supply of honey in other friend's houses recover (most probably, they go to the supply store). Winnie-the-Pooh does not like physical activity. He wants to have a meal n time, traveling the minimum possible distance. Help him to find this distance. Constraints: 1 ≤ n ≤ 100 1 ≤ a ≤ 100 1 ≤ b ≤ 100 1 ≤ c ≤ 100 Input Format: First-line contains an integer n number of visits. Second-line contains an integer a distance between Rabbit's and Owl's houses. The third line contains an integer b the distance between Rabbit's and Eeyore's houses. The fourth line contains an integer c the distance between Owl's and Eeyore's houses. Output Format: Output one number minimum distance in meters Winnie must go through to have a meal n time.

answer

```
#include<bits/stdc++.h>

using namespace std;

int n,a,b,c;

class Honey{

   public:void Path(){

      cin>>n>>a>>b>>c;

      cout<<max(min(min(a,b)*(n-1),min(a,b)+c*(n-2)),0);

   }

};

int main()

{

   Honey Ho;

   Ho.Path();

}
```

question

Question description: You have a team of N people. For a particular task, you can pick any non-empty subset of people. The cost of having x people for the task is x^k. Output the sum of costs over all non-empty subsets of people. Constraints: 1 ≤ N ≤ 10⁹ Input Format: Only line of input contains two integers N representing total number of people and k. Output Format: Output the sum of costs for all non empty subsets modulo 10⁹ + 7.

answer

```
#include <bits/stdc++.h>

const int mod = 1e9+7;

#define ll long long

int n,k,d;

ll f[2][5010];

ll Pow(ll a,ll b) {

   ll c=1;

   for(;b;b>>=1,a=a*a%mod)if(b&1)c=c*a%mod;

   return c;

}

class Team{

   public: void Work(){

   scanf("%d%d", &n, &k);

   for(int i=0;i<=std::min(n,k);++i) f[d][i]=Pow(2,n-i);

   for(int i=1;i<=k;++i,d^=1)for(int j=0;j<=std::min(n,k);++j)f[d^1][j]=(j*f[d][j]+(n-j)*f[d][j+1])%mod;

   printf("%lld\n",f[d][0]);

   }

};

int main() {

   Team obj;

   obj.Work();
```

```
    return 0;
}
```

question

Question Description: It seems that Balaji is seriously sick. He is going to visit n doctors to find out the exact diagnosis. Each of the doctors needs the information about all previous visits, so Balaji has to visit them in the prescribed order (i.e. Balaji should first visit doctor 1, then doctor 2, then doctor 3, and so on). Balaji will get the information about his health from the last doctor. Doctors have a strange working schedule. The doctor i goes to work on the s_i-th day and works every d_i day. So, he works on days s_i, s_i + d_i, s_i + 2d_i, ... The doctor's appointment takes quite a long time, so Balaji can not see more than one doctor per day. What is the minimum time he needs to visit all doctors? Constraints: 1 ≤ n ≤ 1000 1 ≤ s_i, d_i ≤ 1000 Input Format: First-line contains an integer n — number of doctors. Next, n lines contain two numbers s_i and d_i. Output Format: Output a single integer the minimum day at which Balaji can visit the last doctor.

answer

```
#include <iostream>
using namespace std;
int s,d,k;
class Doctors{
    public: void Diagnosis(){
        for(std::cin>>s;std::cin>>s>>d;k=s)
        while(s<=k)
        s+=d;
        cout<<k;
    }
};
int main(){
    Doctors Dr;
    Dr.Diagnosis();
}
```

question

<p>Question description:</p><p>A prisoner wants to escape from a prison. </p><p>The prison is represented by the interior of the convex polygon with vertices $P1,P2,P3,…,Pn+1,Pn+2,Pn+3$. </p><p>It holds $P1=(0,0)$, $Pn+1=(0,h)$, $Pn+2=(−10^{18,}h)$ and $Pn+3=(−10$^18,0).</p><p>The prison walls $Pn+1Pn+2$, $Pn+2Pn+3$ and $Pn+3P1$ are very high and the prisoner is not able to climb them. </p><p>Hence his only chance is to reach a point on one of the walls $P1P2,P2P3,…,PnPn+1$ and escape from there. </p><p>On the perimeter of the prison, there are two guards. </p><p>The prisoner moves at speed 1 while the guards move, remaining always on the perimeter of the prison, with speed v.</p><p>If the prisoner reaches a point of the perimeter where there is a guard, the guard kills the prisoner. If the prisoner reaches a point of the part of the perimeter he is able to climb and there is no guard there, he escapes immediately. Initially the prisoner is at the point (−10^17,h/2) and the guards are at $P1$.</p><p>Find the minimum speed v such that the guards can guarantee that the prisoner will not escape (assuming that both the prisoner and the guards move optimally).</p><figure class="image"><img

src="
MVEX////+/v77+/v39/jz8/Pt7e3n5+fg4ODa2trU1NXPz8/KysrExMTAwMC9vb26urq3t7eysrOurq6mpq
aenp6Xl5eQkJCJiYl2dnZsbGxiYmJYWFhRUVFJSUlDQ0M7OzszMzMlJSYZGRkQEBAICAgBAQE/KMcyMv
9KSv9jYv94d/+Li/+jo/+7u//T0//p6f//8/P/6Oj/2tv/y8v/ubn/qKj/kpL/fn7/YWL/Skr/Li7/FRX/AgHgEjKXK
ZKZddDzEwwIAABQFElEQVR42uydiXKrOraG0QzYeGASM0jg6qpb1bcfod//rdoDwnggdojj7H2yvqRiIgsJ
G60fDUuSBQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAADAAzDFqD9EhDJKkAUAAAC8EoQJFQtOeqllwrEdwT8htyDMAAD8w0
Gv0DnCxCpREbMOYBbGQRAlUmDrSdBLrgIA/mK+YANgP38DiAjxgvtEZZbknTyJLQ9jQQl1s5BZT4IFZy+4
DAD4W6FcEGsmiDgcg/08wrTBzxDyxq8NUSfxsfVliC14YsR2US/p/oWGhZjSVkIpY3gU4EmHvuA6AOALI
ELoAHmnJSLqyIBaT4FuzAfRJBQwRvIMmHJxhjP6vnuM3Sxi1gQft3LQkdH9lr3YIl9t2f4Nuintyae47W1D
Ni7lYbagFgD8JJj9lCUSJ5EUXZrWFIhyx/MDMg4TieRQWXkCvCnKRqm6Kg8UmVyKN91kJGTCnrtHiIjxsx
NTYTuOLSi+Ets9m6Zcc4K5TKd03MlKpRQfJ89k5sxuRgHAKyBhUZ0tMU9j902tc8KThGOj+PbBtKaHl/Eyr
5Rq6EWYm/vcAh6CHD+suq4Igz1hnNWNdN5Sy8M0KtwnFA4TyrgrSzEoK+aeTPbINSc3Yity3cQLe51tpt
Lmfqi6mo+DEM8lf0vBBoAJ8CKImq5NT5Yo86aJ7LfUAHhQethUarYy2RMt2FTWth/rtriUCOrXG+hIeAqa
tG0vTYj6qs1eMWr1ELqqJbYeQ7lYybJtBrHFLKwTFxM7biKOrsUWu2Xb1lm2pdYUCFdtxqwxZFsG76lGA
MAkvOiU0+sej7WWbxi4RXhdx2yo41aRjambVls2eQJTWpLLIJpn73kw/PWwqtPmrmKS7ZX3DVVbJPJK
WE+wKKoyz0diS7Z1zghCmCXKJ+hKbBHzcq3bwuN4ekBN6Yhe9Zfl1fzxWAB4CUJ1DTGWyOpWv6FzC/
OkMbbCZSMpRog4RbOcqnzQRauW5LpzoY6YBTyE2BfNAtnu0jd8b2Sjwqc0XWy3jpBnscW8atbkmITX1
Da5ElvmxkmUK10H3JqAbrVakSvtDxsoLcDPQhZdm6Lhv7Tt4u8vk2RVSdofuqp2TiYS6XSqtsLitrLxdW03
rQWMkT2GBm0rL8Q2f0NvN08aG1lPws5ii+i2LU/HiJU6ZFfeCItCCszDpm3W1n0Qk/q2ZNhVYX9/mw0A
pmGy08F51nnadcm3iy2iaeViI6O7PkPs6GZJJiw36zKGrpMJtf+G9vBfD0+6c0UPoaTbZd9+iy3LqbNJSZ8
W2z08NfcaobzNxFlsTwc1xwhRJ9OpNYHI2/ymjUSl9uDRDPwkotgpGw0qWHad/HYBw1wN0mlXu4j0m
Zuetlvcur0dbEHLOhcW8Ai77GoHD9992bXh9/cUoaiZuJmPxNatu8RIZdLWzoXYYpEn5OQ6WFdTWTuNl
hYmlIwHUInXvqHNBgDTuGqXc3QeWej0+rstEVFfS2NOG73b9FZGy3aiyoW2SgWIEHo560LkzcoCHrFSX
c6Q+e43qq2cb29PI1w0HrGmmRbbteqSc5eHXvaTWHqxtfOEokMIyUrrPthrlU8Y55yN543ZCh7NwI8Stj
s5NDFZ2Lb593sj8FT55jhud56R+qJt2H3LjdpmeTIfgsYmquNvv9i/HhS0XYKMEIlUq+Db2y77fJr6qpP9Wb
ENdBuPSsf6KK3sILbocJDmR68CInI59SyPdo27CGWaJRuKLIMolWsBwE+BcNK128ES3bKtF99dsbUsp2o
8c5y0ejF0GLctR3dHPNKudBdRkmVyOVJb6uuUgtpOY1rgOuzdvpiIVB2z+X2XCNM97ObP9dRD4qlq4tY8
FNu2jUZi62MLM7HM2mx5mFGG1lVkc8ZFnNjWfXi2y4N443CvaBZ0/IgPLQD4KTDLO7U0lmgnqgq+oF6
Y3rfEa9teNdVgKGmnFmdXiPZubQjbZZdG8crmfl3a5Bzu6gL8ER6Aeb5TG3qAcTdpqvVX6rWICfse4uom

01Dl1gQPxDbqzmIbtbuIWiQ6zDhuyjwi+5TdLAs3YZZMVpztZpcfpiciIsejvUwqaQHAT0EWdVe5vSUu8yZ
3v1CvRVTYd7m27kAX3BwXnRrXbD1y9ypVl0UM7XPIdchGlq+rN3gF/91gp9mpKPD3P3Fa5PHX1u/BXq
XuUV01iFisUmuCB2Ib7y5qthE1C5edeuwRZosg9O3pRZN83RXi8C6Ou4qf9T/aXxIA/BTUV10V7u0wCGR
aZMGXVkbATq3uceMQGer0Rmz3TIttpNuUHw0tb8cO+VQ1SxDbjyEH7ZFH4mBtU4Ssh5jFwmzOrgoEE
n54D//KQ5olKrlYtu0OGD0QW1OzNYxTw9YkSatWpB9VK/lwFg1UbgHAT8Fk22YHM5QyCrzndxrBhPOb
8o5YEN4jENYlcZewiZrtityrMedtc6zBYla0+1MHSNV8u+/E387hWw2YUbePIGT8nfM4L8tkwcmsPWZYo
uLzGYSJWzgjT9ZsPwUiZXuatIF51o0ezTRQlQUAPwUrOr2ixhLRB+KK0bhmwbifz2/Ax13MPlGzxUzphPb
OlReuuLRU26/0QP4CEK07bWN0xPoQzx0plpcGgm9yJectWnMhtljI4pY8sZ/qs/1kMSNO05cR7NRtQMZ
iW1sA8EMgrjrFnjBE6gkysh0/SpTyXiS2+lGfLV22/QwoulXaIyC2n4A4uquY9QxZNBqSTAKGEfEaNW+S3q
XYcpneQYoJb4QuvvRG+BR0q/rSQnytbQxiC/wREK9r82c0k6cBHYltsN2+SmzTrl2OvRGM6F8tjODg3rG2
YRjE9hPQsG0Tet3piY5/9y9jRo6rZFNLhg6a2aVsHAsRyu5BCboWWzmxM88AwffEFvn67BYs225lPY8pI8
ekEM10ziyEoBsB+HkQk52O8bUholtD5FnMzlEQIqK5FVuEKbsL+UBsk3bnod6O81azu36T7Wl2L+KlTvE+
/5HYbmZL/u+AJ60O6PV9Iujwd/8yJbah0ofnG426UphIplVzD1/ga8lLxlne5b43gtd0CT6PdS2sz8HztkCnS
1U6oueSDANkwE8iik65l1aCjxUOYgzxSmwN+J7YYh5EdwhD+4MBslDvVmiYrluziYURUN8kVms8ehDQq
plfv/4d2GXXXLllUTfwHc4Xfrjh9L7YYidNjjVb2RUcXW2acY/cvRLbWM91/bJLU5tGKGsrx/oUyG60PKUZt
LVLqG2TWd5oAPBSFmpXXtgSonYQOoy7QbCxMfqc2CKnqO5RLD9w/VqqXUj6vBud0Tvms1E6OB6w+
GD61OH47Pr1hglvfzPIU93VxA8skrDIFqH0o1ryW7E1DX90XAC8TehlAXFW93CIiXWuRuJ5M8hY0uanY0
yLNuHWx2B6sXMeXnRqPWoQcbkiMKkB+HFQqHeXtoRFHJZHQwyqlOFPii1xvNUtnnszqUHlfEi53Jn9TrJ
WATla+rgLEJGwaxbWAbtuY4JE4uLzpAYbZpB9BPK7Lr1UPbYJRNpmIcU8a9bXYms4dSZtVOOhywSnsC4
gS1XReWsjEE/3c8aJW6stefAB6TJaj0oqi3a1OL6xaHSMLDdnaJiuG1gA8CMglHRdiC494EOWqjxiCKd6SZ
4W23N/7y34xtPBa2p7yDHqctEPI/fjYJiFoXMWW5Z0Be+nBqk1srzMKDFxdQFb7E6DCeVy10pK8IWwO
XbWxgxZPG/8Y9XwRJGwE0aciZOpcN6yhFg09ZO+gZgQuld/5TJymiLGDrkeD6ImZ4/ur1Poej0uqWYfirV
SK8yC0LwnqsaxAOD9IEyoyDvtXyz6iZMlLZRkGJFMH8aeKDti51KYUefelr7gjeCUjTda90B5GB2STJUk6DS
tTZ2NDIvSLLEbqcohTG7P48s6/dLk03843HFXVaelY7Ox2EbEqevDsJdT1YuDpC69I3XmHVn2dUHC4yqa
+/3ionlypzNqO+66bFW0cJxjztgtKpdRytyyWjzM3mtanZxzspuul9dF1bhsIU3TBznqHRtUAMANmDtu0H
QqdGw6CpWcq0P9EvPiOBzteEe2ZbrxDiz4C8SWpSpAo11yM3EwLb8pbLMZ7GjHMeyotv/Hb3KHbyU7
O1OoyAImCYuy0Vo3Zb68nAIWqJTtXzdH+aEyy4+oJj+SnfQJibjw52qthWTz5FrdTl6W9f4yVVWWy2POe
JXngW37ebHB1iNEoupRV/+yaVzzUGlkmKzQ2c0xnl1iAeALEHm0RFUXqT22Eeq1KUUWWdbNoQTL/Eix
j5cfSF38dbGlgZYEDTkGeyW3nbDIHNyPeFdNHlIjtuumZsaNqYqjxEGwePhzMNt2DlwvBYQSFVIL0biNG
MGYcS4OP2Uijq+nWduYh8WSYsJmLgPnNk9WJKnYX6W5TtTvbh8kWZaG9hN5Y26vU3pOLTSzMDBbR
r6gI2e0JXQ5AT8B5nZfwgW58tmRRwdLJTEhFjvZoZtL52SI9AU1Wyx0JvBogDtKsyzZCmIcbm3XP4stC01
OhG2iDSfDicsGGoYfgabG76vD+j1Y5MpmHkUWQrcDZIRtE5cS6sQzF5NnSTN/p3lEqOmyeoKF6Ua4nKq
BMBml4JSwUQPwQ0wUZJErf/8mL5oFcTm6M0D2dbG1aFotydi0rucgscCnQ16X5jM6K1T+/Gv4vRBHV
Rxb2GsyZqcM3fNGQCxIV5xzEciZYou2eubg2mdB65haDwmbN2xQAQCfYFHX7nGn24xz6Xwktqv5ZZd4
tfzobMQT18joR5FqqNjOgPoqw8jCfhOLbUzuTmpgoaqrco+a29GJRFa+Ze9wZJo+H0F4VoHjCvBn4auM
H/fMi/k2oven62ImPKUDweYOnyCW1px88L5bPE6aLKtwvt7/YmhUh8eRxyLeRja+K7ZuoXq2cyWKeHV
E5578PIi4sXiYDQvKYGZhBYBvIlAHBcOilqvYxvfFlnlB0jRpuJpdV0BeFfPJwo9ZFDxMmfAsgz1x5kAWAT8+
MxeBL9D9GWRitTmyXs9vPBCZv2F+H2aJjx5GEoV8T58GADyNEx5FEC98X4xnkEVjsV1u1ps93myxtWhY
eRhNvbmR/JH9IBbUW6irzAFhgvvX/cGZTI6jGGbf4sN0lIR/u9qS7eP2DeJRBlMNgT+NfplwdLmwP0+j0YC
vMUU8fxMdTJPpHftI7Dy0DOoW0InwWuTGei3YyYJvr09im+BHUaifwfZJwF8Cjb7gezBhI0nI8JSRPswM
CTl3mByYgL34Fh8XD/P/gJuEl9L5Ay4DAJ4Bs5c3wpCTzPfFQUTGM53tgSnQy79QhJnz8oLzeZDgc4dyAe
DdvN4OLYvQL+ybThagtX8D31FwPg2a390FAP8MvmAD6I+wYgAAAAAAAAAAAAAAAOClQJ8PAHwzYG
PAHvQFh2wAAAwwCgx8BCLc9pb2908lA4DfCmLCXTviDUuRAH8wCPOw1roOOcHAY9D5i+tByIQgE/Kp
SAg9kfa9OPMS+iiSNXWRjz8IQhiYglA3061KbfCH/dUgJnW323VaCkaBhxBjLngIMSKFyRA0tBpuTjuH4D7
O+TSMbk7DQ4jhnL+BTOT/KDcT6Zn8MT1Hups/psAkfNN0u/1PBXuR/Gqop7rdAZ1I4CFxYISMy56A9xY
kwlgeiRZGtFayZ9OLJHIikxDvRYuH8sSwih/Zxn3IhvV3admHyMiYq4hMbuY0OpzmWT2e7PH7hJDJLY5sE
ykw+ZtJTMQzF2lOwyIy+Ys+Eh8+rYuPH14Ck9QnG+sSWMLgN8NkuzvRdV0LvxO/e06HJbVOuF1PZTRqq
fqIOu6VjSZdT8Z6sfVVn1btmmWfG5P8pk+IZbpPKOPmLvUhnTJivzC5qbWRv7wPaaXVM+RfGGl3lEnIN5

Fqk39M+9wS3YeU3FzkkP/mFAktGvNpw0P+KzXna/0lvztDBWvb/WZ43u2Ap+kqasTOBNVnse2/SiN2iK
UmUsZ7sQ10n1CzGMTWpL01Ypu3fUK5uH4kampd5aY3g9iahJJBbHc9pRjE1iR0FltzWsyM2Lbm0wpT1
x3y39I+/8bkHx3yX+sd8BDlgtj+YuhgWK0GHpMbsXNMSOngXn6bPkRFQ81W96SmZrsxkSrHrCJS6R5TR
aWp6hNKTc02VvpEQ4bdkU2Id64Qn1DS6pHDZZuarT2ctjGRyuGyaZ+/NPmb08hiyN9cpFuZ0wJy1F4NT
NF2Rmz/+///+Q/4gP1WiK/7Pts8ASZJk57IiB03IfHQZytNkGf6bH0TEgx9tsNpRv7OpxnVJsH5tD5knRiwd
Z3b0GcbmpBBRzcmJKSmz/acm4kUmxCj2mRrQiJqNNqESOfm0y4OIbZMgCmyvhXQ1v/61//9e6+3BAT
3V8IyfSwHGcPAE5wdpgxoIuRjHy78VAh6lVcXmgjZM5nbOKVHIeD59QHUP/b5dHXw7/+xd2frbSJbAIXZ
VQwaPHSiASFZlhjUoAke4bz/Wx3Q0G27u63EFiDB+m/i5AvR3fpIqXbVJomTzS7NWE5oI+V4Yd5a7xYOt
gWaKK/t3Pf9lydLZWne2yTZ7tNMMyXfOlJ81RF8/bo6AJ8T7QyGf9hKivLqdJvE8Xq3Zzmhffp FbL98KQeA
i6Rw/tnI0t067+025duyliG2QKVE6ez0fptm37ggG/eG2AJVK3q72yRxsXzLckJrEFugBiIq3W/iONnsa/m2T
E64w6w6xBaoh+gsK5Zvk21a8fqtmNZbJosZlSC2QG2Uyvbbdf56W+32BOkMhqPxeDQ8GPx8dKzKPrvFi
C1Qq7y356/LVEX/qVePk6kfRf7Uy01ny+XsD842Lx+xBeolSmfH6bKqph1Em7YbRiPbOuhOfH9kGSgZsQ
VqJ6rYfZvE66q2J4g5i8Lz4Kh2XkL/gbuxykZsgVsgotNdcvi6LNOlrycoy48WlnHmhiuXV9uyEVvgJshx2mF9
mOYte9rBfApDzzTOxuFqZhsoF7EFbofo9LQ9odzDby03DAamcaLciNiWj9gCN0V0Vkw7xJvDbrCSlhPsW
bTs678+cRqtJlSgbMQWuC1ymOZdH6Z5y5p26C+iuaOMI91dRMFTuesWILbADTrsBst7m6y3pUw7yI8g
8kw5/cYaB8HEvvZngNgC90HOh4MVub1qb0WPonB8fo02u4tg7vBiWzpiC9wmMYr32+3pLh19xdqK5a
2O19mLMp2fc3/SuWrMQWyBOyPnu3Q215x2UM7ryn92cp2HwWQ+G/JeWwViC9w2pc6Hg11re4LqB
St/+VpYvE6eOYWmGsQWuHEiSp/u0rnOtINyw8jtdQqOeeX1YBBb4I6J0sfDGNdXmHYQ/RKFD1opRWe
rRGyBe5BXUWWnux3S721PUFYQ+Wz1qhyxBe6GKu7SOW5P+MZZjOZzEM44d6ZyxBa4I4ftCes4Sbb7L
9/Na43DYGx+WFko/i2l9TV3mIHYAvdL8t7q81XoX5sus6eR/6yNN5TpdG1tWk6nY38t4CgQW6BZRNTp
MMZtmv1+HPuL6LX79jExn9yp2+mPRyPPtX6/3jghtkDjiMoO02VfmHaQ5yCame+e6HiO63vTvm2PA65
sKAuxBe6U6OztXTq/FFxRyhyuosn7Pg9GMgkWeWbVMByTg5IQW+Buiah0d9wO9oubb7VpdaarvKjvrt
N1ezIr7nwUPY4GvNmWhNgC90tEdHac5v21aYe+O5n5YfDijjvvEmwvlz1tiD0NO7+/CIwTYgs0mqjT8u0
vbE/4MZl4hYnbM94wH5dzUwzVfX21vjMvAWILNJcYcuhtfNgOpt/ezCvy3w+9ZY78Sf6LHvmuaVm825a
C2AINIKrYnrA53aWj5fBnOg+nmf98me35g/wBc7F8sH78YNW2FMQWaAqVpW+2JyizN3TdYc9UxkWd
10U3f94JXhxn1OPNthTEFmgOkby3yfHwW+tpEa5W4aJvGhd1g6ldxNaf2sMhI7vlILZAk4jSp7vL/reIVrlo
0bv8kDP/qYtnB547YoSsJMQWaBql090m+TNaHYQjdTGfumPKYW7X6XzxdBvkiC3QPjqdrk48k3fVW0Bs
gSYSyyO2t4XYAo2kR+dlBJcphZtAbIFGkv4iXOUi/5nW3gRiCzSTOVgEURT4f+7Yy3UTiC3QTMp8dKdT92e
c7KntLSC2QGNp09R6Hycptb0BxBZoLCnoXbxO+Y6sfsQWaLhsF2+obf2ILdB0ehtvvnAxJK6L2AJNp7JNXl
tebWtGbIHmy2u7o7Y1I7ZA44nK1vGelYR6EVugBVS6jneZgRoRW6ANdJoke2pbJ2ILtILeM0pWL2ILtIPK
a5tyl2N9iC3QDqJ31LZOxBZoC0bJakVsgdbItvGaDWB1IbZAa0i2ibfUtibEFmgPOY6SsZJQB2ILtIgUww0s2
9aC2AJtotKEYxLqQWyBdtknyZ7a1oDYAu0i+zhhA1gNiC3QLqL38ZrhhuoRW6BlRO+obQ2ILdA6h9qyAa
xixBZoHcm2DDdUjtgC7aMYJasesQVaSGVrtttWjNgCLSSSUtuKEVuglVSacHNDpYgt0E7F4G5KbatDbIGW
UsXgLqNklSG2QFtl+3jN4G5liC3QWsVwAysJVSG2QGuJ3ua1ZbttNYgt0F7FKNmGmxuqQWyBNmOUrD
LEFmgzVdSW4YYqEFugzUTSJN5nl/6W3en2ct2jjmNxQuO/ILYAjG+Nkqnn2dwPAn9+9OINuiZrDx8RWw
CXa2t8Rqxu3wujyUM313sYvgYvD2TjI2IL4DOiilvJ9OdBVtY0Ch9NnVPa6i/CeZdX24+ILYBPqd3FUTKxlpF
vyTnPkzAa0o0PiC2AC3Re2883gOleGM7+LsU4XE1tA+8RWwCfE729UFtzGAbjv0vhhqsZsf0HYgvgguJW
ss9GyWwv9B/1uc3Ki1aeZeA9YgvgEsn+z96d7aiqhAEUpgYKlfbsth0AtZXJ4yzv/3ZHxN4teNyhd1Jcrc8LD
SFerlQq8Ndhe1CvY9tf5au+dCrSrDb5mEdtm4gtgEd/cyrZrySfu8Kp6HGSLXq8dtZEbAE0vD6VTEgpnCdy
kmXB/brQvUUa/yIbT4gtgBZuL+4qpbV6yq3Q4SYdixupvDBbvzN0/AmxBdBKcdntRpMomvpuo6TSLDfJu
yn13saLdTTQtPYZsQXQijr/G6dZnq7GjZbKfpInUVAK5/Ng5CmnJalYAhNbAHXSW25KedJXjS3bNJ+PP0p
Dz1Ut8ymElK5hYg2xBVAnx+nmJgtc55Ga5/mHq0uq9VJVKNfzZ+sRfSG2AGpkmG1u8oWpXddJnnrS+Rn
hTsNglTJDgdgCqFMvYqv8NP90nR8SetDrLTJiS2wB1MnJ1zZC6DYGI2ThzzMhpTJzYktsATQIb5ndWpv4qj
kY4UM39mNdLZR2javln3YSiC2xBfBEDldpnmfxRInGYIT1oJZf5U2CoWv86Wzqa/H6/4gtsQXwTCh/FkbB
W6OfwyRfGuk88IJ/ovU0mPr9cN1XrGwrxBZAS0JqY5oP0spxnoeydm0U6CBdj41Sk3SqWdlWiC2A1poz
bZXWJtxkM62k8232S0RpZIQjp/kttlLXKcHKltgCeEmoxrK2N54EyaZ8f8x7uMt39SoeKke4YeaXFdZBGD1
8wg/NypbYAnhJHk/12k6Wn6s4jtfLxdvjbXoQr4x0ZG8Zm3IxrGZBzZtiZUtsAbykdvuiFlvtmoqrapfHyVxco
ztKIldp4Qi2Ea6ILYCW1PYgm9Nkbp8r54EJk2m56bCIR9ofquvPBp5GILYAXpLF9iicFrzP2L/ebtLPnpn48tX

fmXk2dZmxSGwB1MnL9ui00Y+X5np7L5mb4dT8f2ylNv4yC3quIrfEFsCj4rw9OW3005kuZ81EyyDoixdv
NMzmn0kaL6J3DuIltgAeFaft2WnDHd6Ws9IMBq76wwsSnmeMq5gfTmwBPCqO24vTihT37+rHa2whE
FsATcV+d3FgBbEF8Js8HQoHVhBbAL8JKdlftYTYAmCHtQPEFgA6QGwBoAPEFgA6QGwBoAPEFgA6QGw
BfBFS8jiCLcQWwJ0ozmdmxthCbAHciWJ/JLa2EFsAd+KyI7bWEFsAFSEu2xOxtYXYAriTpx17ttYQWwDfsb
0wh8YWYgvgTh6IrT3EFsCd3O0LnrO1hdgC+I4tB+FaQ2wBVITYH2mtNcQWwJ0oOBPHHmILAB0gtgDQ
AWILAB0gtgDQAWILAB0gtgDQAWILoCIV5zRYRGwBVIpzwWQEe4gtgK/R4Wdiaw+xBXAjzltiaxGxBVA5
bZn5ZRGxBXAjT1tGI1hEbAGUhDoSW5uILYCSKA47YmsRsQVQxXa/J7YWEVsA99geiK1FxBZAFdvdiXPM
LSK2AEpCXXiBzCZiC6AiBI/ZWkRsAaADxBYAOkBsAaADxPY/9u60N1UlAMDwrCjudQG0xwWkQavi//93
dxCvx3vs8aY2Y5rmfZpoUOGDMW+mwwYAT0BsAeAJiC0AR0pu0+AXsQXgqGPJ+WNeEVsAjj5sDwIeEV
sAznFfEFuviC0A57grSgGPiC0Ah6vZ+kZsATglV7P1jNgCcMoNV7P1i9gCcMpiR2y9IrYAnLLg0uF+EVsAQqi
yKLl0uFfEFoBzLBnY+kVsAThSc2kEv4gtADwBsQWAJyC2APAExBYAnoDYAsATEFsAeAJiC0CI45G74nhGb
AEIsdtyAplnxBaAEO8bYusZsQUgxHZzZBrBL2ILQMjNhtZ6RmwByGOxFfCL2AJQZfEu4BexBaAOxU7AL2I
LQO+LvYBfxBaA2m8OAn4R2/8hlXY4uQY/mir33O3RN2J7n1S20ek0DMd74yeTivGEd8T2LhUMZmmar
pLQ8lsE8AXE9h4ZRO77cfLliNoC+AJie4dUUfZWy1d97j0K4HHE9g4dum/nLP/F0BbA44jtR6QUFTPO3i7
WHV2/R3MBfB6x/YBUSlSC6foqthMrKooDEwB8HrG9oewgGoWmiu3sKrZ5YoWQpj2JOkwo4GdR+sgY
wjtie8NO0nX22jXyj5FtbIW0/UW2Xg0e3lcmjQ00qcY3o/c7zmnwjtje6Jz2ii1D5bL73zlbqQZp9d7Migep0
eI1YAyBb+b4vikFPCO2N+qgrpddo1vL/NLauVXmpV7+FYgH2dU6m/B145s5breMbL0jtjfCRV7X1dV2kp
5rm69ejB3Urc2TR0e20i6zbMTXjW+m3LwTW++I7Q3TORd22VJ2ktV9TUda9+vX81X74YkA1ZvG9uG1A
T/Kgjlb/4jtDfnvbMF60bV2MF2sVvOkay+vLnvm4V1c0ljLFT/w3RyK/cM7fXGN2H6KlGa4Ote212x2huPx
sN1svtSv5YsuhxPgZ5H74kBsvSO2HzHd80zCOrtYv1XyZVffybRzfrp+qX6+XnRkTcjL2n+sd7Mt53bj1RMn
teFrdsWByS3viO1HLscd5L/Vy3fmEKQyxlgtpbbW6HNctTFuQWj3qKrtOlrV79igaa3WVokTbYKwYbUU1
+tV26rWPDstNwOjf39GSeUeyS2+4r3gpAb/iO2HlBm6se2f7s8h6NZ4kiQ923iJk3gUniKqu+M4SQlTTpI4
VCqcREky0FUog0HsRO1+ZE+htu14Opslg0Cf1uuNoyRuaduLpvG4aeQ550Ennrq/jq063Ju4bRvTdttu8k8
gvmC/Y27MP2L7F6Z3W9t82bvzmzTjRbrOo14Sv/SjxWIUSCF0vEjztDVIxnE6D/TIfWIdWyFUkEy7gQ36r
6vXRhX3RjSPW43mZD5tVWW1SbXesJlE7TBeLbpKCkc34lUSBq3ZMgqkNPEyy7NgOB1Os1lDAA/jgh/PQ
Gz/Qtl6j9jF/TkER+pgmL7Nk9A43X/Yu9OmxJE4juPpKweHytkdzqQTB2cc5PBCcGre/7vaHMAOMugICB
p+n32wJEPcrRrrW3/b0Gn70qQG4VYl1BfSsRvay7H4HWEcWyHbecYoY04zji11Gt2yYIzxQrMdB51yqxp
oKQucMUvppkWNuNANrx6/y3J1VUTvses6KNUsqxV0cgbA1rAMdQiI7WsrCZcrusWota8hVvvSy1MSvR
Jl7Vfit1PL16rCebmrBI2OWklsnU7Lmg/QDcsgptLSTI+Li7LmdNBJvgK/CPxke0choz8j0Qt2rj0nrnveC9U5
Y1VPbv2ZNgA4DMR2M9O9XKEYeeuK1qUSaXd5O/CSappe0DEpYSIOJzGbSWzz3Y5DKYnO2Mo0SNnXF
k2vE21dT/47jh92k5Msr3WyznvuB1U2/5CbrsdBdrphPBhzgQdSAnx2iO1mQl2ukG+1lkSxlfPYsqoOqjyN
bZORdI/cZWydrnZzSSLFOSesFfWYLP6j2jOj8ySKbZOS5BqtS8wgVAX+GUmj3ggahCSxjeNOKH4MBPjsEN
t3xNZIvT3ZRlg+iqWZxFYraqSWsRVSa79RL9um4JTaftAQy1+zpXOs4fiBSq/hSWxpdKl3ZiZsN2iJJLYBtrUB
+BoQ243IO2ObTrbuvJrU7oZdJ41tbS221GzoMAi0555xys504C5jW9FJQInja2kk5rF1OqGvZKrpufPYlvCX
B7vCD0YHgdj++2SrjLesxLYdevk0tpW12BJqVlxPB2HgVTi/0IFaxrasAxkfvIwtc6J6l5xELpe3aRrbIu6whR0
Riv06DgGxNT5oso3b6GyKLROUm3ZRNn3dcfj5i8m2/tfJ1u6EnRxbiL4kYgt7wX7iw7qHgNga+12zdVfWb
DfF1qoxEsWSm+VuWOW2HzSXsa1rnSwNrMVWNEOvYMxRShBb2I/B1TW+iw4Asf2gyTa9G2FTbPNtm
0ZHhPJSqARrBV1zUXg38C36l8mWUBnoc7L4f3OwjAD7QQY9xPYQENt9r9mm1aSiqbsm3RTbQrcm0h
M5rQSt+EGepdeZXa0YWZ9s04ejSU6Sq2hBMcQW9uOm9xPLCAeA2O55sm3ZlESvePIJsjiLcWwZXWzm
ZcWxpVFsW05yjhWiPyWmG7hWcp2o6KZN45k35weSRucoNbUu8+iVqOlOjsXvYqZbIYSwnBeccdxkC7
u56fXxLXQAiK2x3zVbX5k0qmGx69cFiV6IXBAosdgUUTjd0LU4KXhaChLnWXWceERt+VLE15W97nmyN
4IoBGEjuo5wcR4E0oxeEaH8lh19JWpJZcZf7UKH9ehPMJbADmi/1zfg4yG2e55sG1IWnFyp2amalBisojpa
+66qGxFajI4Cv6FYvuuqWt5xCrJxxqMLueO2ZCFfqDYb+WQ/hKrqBvF1eSrd5CtIixrMqrcbpXz+XEqTEV5
RXa09V1Xx9wc7YD97eI75ISC2b062oe+H77gbwT6r1mRU0mS7AlaqS6mklJUktnmZqDNb2nZF1uv1isNI
ulPtRfLGip0c83J6XS1HazJRs2jc/0JdKVk/TzYUK8lEHZ9rgF0MrhHbg0Bs35xs/cqFfsfdCCZlwuTzVVpCF5Kj
6JCx6IAwHr3mpskpWVwbXybi45XriEEX1/z/LkbJH18Na7awk8H3K8T2EBDbNydb7yyv33WfLUnS+Kq4j
2S1k29flqL4uA/sFbvGc8wPArF9c7L1Cjn//ffZAnwNhA7wALKDQGw3+yO27/8EGcBXgZ+UDgOx3dtkS5
goeJdNR+DZefDFmeVqtVarVVPloo3d6fcAsd3bZEsLsukHnlu3EVv42mzpJjcpurFGq92oCHxScWeI7d4m

W1aUUkX/SAdTAHxtlPOSvmxaImEpHSg85W5niO3eJlvCRALLCPDVEUprwWWd0xhjpgx0Hd/Wu0Js9zb
ZAmQGoSrUpUVeadEP2xZ+YNsRYru3yRbgKyKEGGuo2Qq9wh+ZiI6warsjxBaTLZy0Qf9mvbYs3w1blrFQ
9BHb3SG2b062XcdBbCGr6M3367XYLh+Fl6KlIOzg9767Qmw349UwSmzYFFY7/reuGABZQ/tXP9czylUQ
lBddIFxehi7HL8h2hNhuRq2mDgOvRHnFC0LtWgZA1rD+VX89o6ITaosuYpvrhl4Brd0VYvsK7tRclRMGEU
XlViysWUHGEMrN694NexlSauqww9OzhJmu9kv4GPrOENtXMc5p+j3JcZshZA4V56rdqdsvv7lZOQwUpz
HGHel3KlhE2B1i+yqy2M8Q+xpCBgnpB2EYtPL0xXkVBlXLjNgXstVVeeyNsAeILcCp4uc6vIyErmmssLqh9t
qtSLvdqOdMFGIfEFuAU2U2w8uEf2asOPPD9kWxELNEuoJAWQI/4W0PsQU4VTlvHttArja0qi+VYInFA5l
EUSolzwXWE7aG2AKcqmVs9UpsCYuXbFeqSsyackyz1nVN3JSzLcQW4FSZjb8uI1CzGaZnlmipU+eEWs2g
jlpsC7EFOFWsOP8FmRIrp3PdsOOsxlbpjkhG3iY2tt0WYgtwqqio+0EQ+A2HrcS2pF/en0DPmoobhLtBA5
9u2BZiC3CyBtffVENVbLa2MUJ1NQqEm4ISZrd0GbXYFmILcKroTe/HQHBGydrGCA5d3/WWipqncDvC1
hBbgFM1uO711z8bSS0ddsXLs4QVK8pTNm5G2BpiC3CiSL/3ffCytFyY1SBsmi83AyG8UpNN9wIf3N0aYg
twmgj90eszY5VVkw0/DH0lC2sLCZTnWn4JtdgWYgtwmmj/6seAGKtsqZTbaLhKlejKm+Mjyuu6mzdgO4
gtwEki7MdVnxovMCFMkeDkz/fm8slx2Q+qBmwHsQU4SfGtCIwYa0hqNcF53yuy6EXBD2sGbAexBThFZ
BCt2KZNfRu78L2zOLZljUfxbQ2xBcg6Enl5ivZ7PwbGP2K2p2xOqZC67RiwHcQWINsIocPh8GVuo8F2QI1/
RauNoinMs453hlpsC7EFyDQ6un+cTCYP4yFZvRXhekCMf0VEvlavSbeOB+RsD7EFyLLheDKdRaZPt3/Wd
vD96uY92STMtJ2cY+K5p9tDbAEyjIyfnuemt0NjgfWTWxHeB53dDWILkF1k+Dh7nps9jZenB997fbTzwBB
bgOyi4+nz0uyOzltLf/auBwYcFmILkF3D+9nz/x6H/w+2Awy2h4bYAmRXsoqwNBkZiXiwxW+6Dg6xBcgu
+rfYksHV1Q3u4Do4xBYgu4Z3f8b2YWjEWDTYUgy2B4fYAmQXHT+t/YKM3ESDLVp7eIgtQHaR4cNs2drJi
MwH2z4ebnMEiC1AhpHxYtV29nQ7XKzY4laEY0BsAbKMju6nsyi108cxJUlsr3s/MdgeA2ILkEmEDhOj8e3
D4+PD3Xg0TNxcfcetCEeB2AJkUJTau4e5x9ji9cPv3k+CVYRjQGwBMojGm309p2az58XL6MWvX7eI7VEg
tgDZQ8aT2fMGs6cxlhGOAbEFyJ54T4SNZo9DAw4PsQXIntHk+RXTkQGHh9gCZM94+vyKGWJ7DIgtQPZ
EsX0NYnsMiC1A5pA0trPZy5XbX4jt8SC2ANmTxvbp7vFFa79NENujQWwBMoeksZ2MH2ersf3xOENsjw
WxBcieDbGd3iC2x4PYAmQO2RRbhtgeD2ILkD2bYksQ2+NBbAEyhyC2nxBiC5A9iO0nhNgCZA4m288IsQ
XIHsT2E0JsATIHk+1nhNgCZM/G2D4gtkeD2AJkDkm2WJw9rMf2footFo8FsQXInuHd02w6GY3msZ0tYzt
6mM6m99g8/BgQW4DsIcPxeDwaLmJ7//vXPLZ0NL4dD+na20cjPCrnoyG2AJlECDGGaWxnd98Xsf3b+y
gd3U0eMO1+NMQWIKtIEtvI728bY0uGw/H95OkRD4H8cIgtQGaN5pPtt82TLb2bTKeTu9EQjzf/aIgtQFY
tJ9tvv6ebYjuM7k+YPtwOKWL70RBbgMxaTLa/cxtjS0aT6dN0+jjGaPvREFuAzFpMtqXx5tjS++l9tGj79HiL
pYSPhdgCZNZisr3dGNvI+Ok/9u5mt3EbCsMwD73pptehxImtc0iKv27v/7JK0vYMArSFB1YEi/M9KwWSD
Hjz4oChZC+yxJyCE6wmfCPEFmBY/EhsJUSjWazPOTlBbb8NYgswrIcmW3bRKNVzm3Jc8L+y74LYAgyLH4
qt8aIafc2tN8jtt0BsAYb1wGRbadaqoXokDlsTvgtiCzCsRybbiujHkWbjYkoBTzmsD7EFGNZDsf2KiMWFnC
K2JqwNsQUYFbH/Gtv4SD2JNJvQtyYQarsixBZgWIc//2qtTSy9unkh9ZC+NSHmiMd414TYAoyK5uPff5US
rdYmlpK9qIdptv1BByuM+XYdiC3AqOg0Hf9wTjS1drpf22NApLE1YVWILcCo5uN0PnBfCSDN9UD9EtIsLq
QcF2xNWANiCzAofZo+5qcq2XJre26xNeF5iC3AoA7vb+dnf3+BiNiEXNJiUNsnIbYAYzqcpuPh+UCSZjFLz
MkbBc9AbAGGRPP7dH6+td31d8qcgmcgtgBD0qfpc1YrIaXFioJnILYAI2qD7fzsiu3X1YQVP+23hNgCDIjoc
/pYbbCFNSC2AAOi+e35rQiwKsQWYECHOthir9ZrQWwBBnSug62Cl4LYAoxHf06fBwUvBbEFGE5bsV1rjy
2sBbEFGA3pjxX32MJKEFuA0VBdsV1zjy2sArEFGM1cB1u88fvIILYAozlP7ztbRNAsxsjgwzhiCzCYw+d02tl
WBBNiTIHV0BBbgMGc396fe2f49kxIpcTB33SD2AIMhQ7H3Q22isWmC2ILADui62C7wjvDN0YcEFsA2JP
5YzrtrrWILQDsDJ2n4862InSILQDsShtsd7iFCpMtAOzLaZ+DLSZbANgTmrfbikCaKyIt1oomzY2q+pGmH1c
ZYSbV0PUqRfrLBRX9jC0xi6nnebT38SK2AOPQp622IrTG+hgWIy6EGETbxYewUD3l2pHVqmFZ2ulFem61
9It0u8cx9bKK88EvRnpsb3dU3ni3w+UQxBbg9zAft3q1ojYxebfEHII1MTsOKV9KaH2MOZfiuV9l62kxPiXH
/c+US41qCC7npV3LLiXvnI/uFlttY71DzJKKH+yJMsQWYBh0mj62eXiMTMoLs9iUg4jzotks5RpbNv4WW+
lYDFdLzk73cdilS/KLuHot1/M+Ryv1AhtSje31jvq3ZnYZsQWAFzW/b7UVgX1JprXRlyyaWbc1gnyNLbHJ19
hqW64DrMSW0ooklByEJSanqUXY9iprm6+xNSUwXS9cEFsAeE2b/Xw5SR9Ee04vNYqkqltsK32LLS+lpH7
gLtmoRnzJ7QZhJjLtDlLVPcfa3WPLC2ILAC+J6mC71c+Xm/wztq2r/x5bbVLqky2ZS3H3kTg7VqQ19WNLq
qF7bE3J3rQQa2s2+i5bQWwBBnE4TR/bbEVQJLfY0v/EttLGCLOI2EtZ7pNtsvp2Q7okQ7fjeF+zLSUH76Te
t8132QxiCzCI+f1tq8FWSbikPtm6Uoz+j9hWxMb5UPXY3hZ77zfYXD/k62Sr2KbL5R/27na7bRsGA7CA7N
cuZG26pgYBfoCU2t3/XU2kZMdK7CSL41TW8PScuqYtufnzHgSGxKHvc9paYWtha8xG3I3bl39SYVszsZQ
AAKQlEZwNW+CUc/K8rGzdfIAvgz4NWyCOWvp+6LNsLG0tbI3Zhh+fuX05OC1ZnAtFGbuzPVuXS/GOEL

mGLcCY0kdhK8/DFpDIsYQt3kzcwtaYTcCrjiIAwHKBNSRNSQMDPA1boBq285yXJ4AOWtgiLSpbcM/bCM
ChpTRxHA/tNsXC1pgtgN39NbcvR0RYPPfKjpkd1fUnYYv7OVtfhuSgvb2GrYvLytalofAybFGUW7aTSyV2
m2Jha8wGAH5oxxYAEWnkaqKKSAjLDiqEzFQdVbwuz2FLoZ8q2/rYwpji0HPHuuzZErfXj8OWpIRpgYKFrT
FmdWD39f6SwhYarKhi9t6HGFNSzbmUkiMtK9s+CVdEOAeu014Rpu5BH2sM17CFtqJ9ERDtoIXt4YhUn
0xXkOXaUhgfDxc1RGsjGGNWZzcWthfckhD2ZawPcY7XJmfVFIOIsFtWtpz7+S0a3JSdFEqp7yKf6t1mCGH
+Bgwoau4jxQTEOmSpAT3fzabEWufS+I+hCCH51uVt360l+4LMGLM2/3EU4bhLIOJ9mErYkhvNmlIMXnj
knKNqGeXIqRyoYFtzqSRhCUlK32tipFCmlcixlKjCSdtLiWH6b3DKUVh88jr0OQlJSRqE2Sdlu1zXGLMydcZ2
91qXAFvENo7FhzlhaxnbIrYmbAzByxywiB3Mh3ZPccophNBOUuYuLKALqaY0k9aCmBFIotaTOnJhutliqu
prXdPWY4zioqpGIU7Csa6ErWWtha0xt69uX76D17sETnwN2Fz2phJW5LiCJRxBdf58pCU4Qqqc5D5QVy
G5ESHMDx20ldbWpf3fyzoZiNp7cT4EiaZjNrdRg4WtMSsCgC9mjBM8maV/LzfDAcA5XqcmQYhzJzZXqi1
ivedFmwAX8fpi0oHk7PAQl7FP7vDB7QeoJzta2U/qwqkzt0MOPznUA+vK1qLWwtaY9QAkFuHzNR34TC
cL2y/3O6zoqEsQH7sEZd8lCMG3LgGN8P2lI0nRx6PBlxcu9nr3h2yOha0xKwHk69dHRZnOvSEpnd6+/Du
Ra8MEc5dg1mYJPLMb0QRHgABwQe2InAsfKluMm9vB5iosbI1ZCRdLP4x/+hwITmZtyIH2XYJq7hLEX79
07hLsmwSxlbDsKqo+9DdzwJhVWgUO5ELO7oIR3/8NC1tjVqEWiMOkL4InC9+SuEbsfpZA5y7Bz5+PowS
HWQKq4fpx+fqE8212gIh9ykmssJ1Y2Bqzfihl2OuVToSxLyU+6xKkEP786x+eZ7WaC3sEb4LkJGkeaeTtDQ
5chYWtMavgamG71ws+j7fcl6x6NA47twnuvz788aFdgjeYh7qmYQbrIcwsbI25AU6PwzbQs3RzoZTkhRf
DBFBHES66K8K7jR9/9QJ6UyxsjVkFzsORSM/DjVhzIJh0e7tvXx4s8m6Bha0xq/ByZVshccqeniw+XPOe4e
YDWdgaswZA6cWebQXAqm65tvv2aduXm8tY2BqzCiRlOFAHZ64mSMuwhYcv36ywvQ0WtsasAtDjOEL
xdP6OMovnd+Mowl1nboGFrTHrADynbV8CQfcmd2Nh+3Gb4ZirsrA1ZgVg1LmQy0iF3nhZQt3l8Udnbo
OFrTG/GSC5ar4M14ub0Gv1bR1F+G4d21thYWvM74UkqZySo/uXvXvRTlNpwwA83xwAD2kUURBU5G
CTtI0mabvbJN3/vv+7+gEHm5qwNUfI9n1W10odwSGJvH4ZYKiOW+JSGcYZTkV4PxC2APUqJvt60O31V1
4Z0ebA872POMf2/UDYAtSJ+PebX1X0zb0ewK0gSpIk8hSDdwJhC1Anurz+Ve32L84eZPhJulgs0qiLE7/eC
4QtQJ3yaxmq3d5UhG0/Wqz5KG3fC4QtQJ0uvt3++hdVYesmi7W5yeB9QNgC1Oni+7+G7S3C9j8DYQtQ
Jx221z9vtkrarKGqsiXiThm2vrFp5ULIjCjh/gnNgrAFqBMvwvb2x9et42Q/Lq8rK1tafTnJdtxc/EEwjXens1k
YhrPCZDy0cCFvoyBsAep0sQ7bv662w/biZ0Vly1fnn07O/gniJE3mI4Nv2k1nGCRxOHQyQ9efhSMLxW2D
IGwB6sSrw/bBypb46vPpyeflquWMfben+J1npDFKE1fJnLKGUTw2cH1ZcyBsAer0yMqWxPnpydlyxbmQ
ylB/jMsSGX4atwVx4hnDS6IB9u3mQNgC1OlRlS2J5efT0y/LFVGerWXSlsiap1OzbFROkuAs3AZB2ALU6R
GVLfHVl9OTT+eCswp2tPBUGbZyEKchwrY5ELYAdXpEZSuWZyen5ytOxCq4SepsVpDDJJ0hbJsDYQtQp30
rW+L5CEJ2YIyzKiT9NOoR04xxkmCemgZB2ALUab/Klmh1fnZylpe1rBK3pmnYZhp1wnTew77dHAhbgDr
tVdkSX346OfmyEkSsmujOU9/YPBrmhS1O/WoOhC1AnS6+3eRh+/2BsL0twzY/MHb6aanL2krSiZORYgU
u7TAet3BRQ4MgbAHqxPOUvb25uh+232/yy3g58eLAWF7Wsn+n/CTpKZGR0hyG85GJuW6bBGELUKuL
rz9vfn7jl9d6AtsbHbZ08f36+scl5RMhZFeM5SMIO6h5mvS7OdubhuMjibq2URC2ALWii8urywvSYXvz9d
utDlt+mT+x+nKWnVqbl7W78HaSzj234NgdQ3IiLpWSGLdtBoQtQP2I6bC9vvqqwzZvJbHKD4wtBbHd5D
BNPMs0s3+mIfNVuPzgBoF7pFDiNgHCFqAJHgzb/IqxMx21O+UTIxwLKrAcqZHXa9v+HPPRNALCFqAJ7o
ctifzU2vJ8r91aYRpanP3GHddUUll+7OLahgZA2AI0wb2w5frAGGd7sqPUN4htkJqPs/Al4cRhl0HtELYATX
AvbM+LiRCI7W2UpK64szy34niU79u9KLYZ1A5hC9AE22H7v+zU2uUjbiNGcpzGNt1tUX7wQRS3PY8Qtg
2AsAVognthq68Y2xc3p+m8+0f8ctNUPPvixHMMIzQAwhagCe4NI6w4ewzRidKJye4j5SeewaB2CFuAJrg
XtpzYY4hhnLjqoScG88DCdbsNgLAFaIJN2HbLU7/2x6VptadJ6rZNuR3Rsh36bYmLGhoAYQvQBDps//7n4
9+PDltx5Aez/MbmE7/Dt+7F2w28tiCEbQMgbAGaoAzbj/88obI12p3u0Yejbqet+J/PtPyRKTmmR2gChC1
AA9Dlz3XYftFjtn+x/ZGQSqrc1kRfwvQcU5AYdBjUDmEL0ASXP37lri+vbvKvt9/Z83HLdUwhhBrj1K8GQN
gCNMGq+3eRsbxI3dvrK/ZsZPhh4GeCaZtB7RC2ALUjkd0693/X19ffL4lf/bi+/nF1wZ6L1DiKo0JgMqgdwh
agZlRMpfjp/Orq6iKfhfbi6uryBY5okXLctVEPs341AMIWoF7El8WcMzxDxWPOX2S2b5Il8QLZDc+FsAWo
FV9mt879vFxx9hIBu4WIcI5tUyBsAepDvJgh/Hz1IqUsNBrCFqBGy8/ZPcZWAll7ABC2AHXhxa1zUdYeCIQ
tQC2IxPLTyWlW1iJqDwPCFqAWxT3GPi0xgnAwELYANaD1gTGUtQcEYQvw5kis8gNjS0TtIUHYAry51fnp
yRlGEA4Mwhbgba1HEL4gaw8NwhbgLRHPJ0L4vFzhCtpDg7AFeEvivJgIARfRHh6ELcDb4cWptShrDxLCF
uCNEF8fGMMVY4cJYQvwyiiXfVmXtahqDxQ/ysPWwK8f4JWQsAaObQi++pwdGENZe6i45SeLReK3kLYA
r4LUIIyiaNbHFWMHjZQfLzKJrwSHSuX0+eWjjG7RWG7HQsQ0vt1EWwu9ZG93WuhJq/2xkdv9w27KidN
sH0ujj1lZi6r2YAk7TheZNA6nUGUyzMe0zbF+GHSJFeRoqo2kbrIn5UItHaymN9VaZdaWLRNXsoLoBWVT

3pIzpprfEaygxuUyI6b1pppb9t8JNhtZfpxu+nfK3uxJ2b+hW44m5WtbXPfvlS0D3b90p5rHtONgCjvNonR
RmKCsPWTKSxaFNE2gSuzld8trzeKkENm8vHdpoo0VscIw0i2RDjvemiZah62JSaxfd6x0jjlR2VXekjNj3TI/F
mX8blZjmpNovkE6fjf927p/Y5JobtlbuZGxb+oWe7NaGe1W2VtUfiIov9ykCdMGUQK7pYu1EIN1h8zw0w
Xssg7Fdpjqh4Oy+vMX2iZsR+WHV3Skk7U1W2hdtian+oWyjCzjL17opjJsrVS3RD2p47fsP/WZNlxoQXmE
0471MvGgDNvpQjd5Zdi6se4ssHTLoOw/3oRt2Vviyu33ypRpTryA/c0RtodMucliDZVthboq2yT3OpXtCJX
tW0o3H1ImwvaAya4eT4qGNlTptUUeNt2+XTg2yzHbtq21hW6yerrlg1GOmXZtLW/J8fKF+u1yFNXslV0J
tqbszQvpaJe/V2OaZWsdqfs3j+1yI3X/YtN/S+jeyo3sd8r+jU3/SueBPOrrlpbY/m67TNts9sCGSs4kXeRSV2
HE9oCRdPO0TeORIaCSPhuhfKTzqGwpF8j90bK1EFWuRg8s9HsZurca0+h+/3tsJHF+v//qzebV/XMBu6nu
LN/H0kkHWXvQuNGfxvFkYHCCaiz3x8MdLdtNZcPuF3reavTbW/SW/YMdGJctL4rnnikRtgeNuLJaLUsJBg
CvgqSZ7WMGTvsCAHh9mFIRAAAAAAAAAAAAAOC/hu6ejykzmN0IoNIT9w4cZDt0xIWQxyODdOy2R64
3auGkQoCHCfnEa8mEIXDF73tFL/Gr40bX9uJpi7MMyYHXMUw7GO6Zthyn+MKBsdze0971augYDN4lbj
kvUIDy9nDgxLN12Iq+a0op1dFkz/eTclpPe+MBvC4SUv0m5QtdlSBaY0c9rcoRhjvCLayejri46w3HOokb3u
AlwlYaRjeZrTNTjocqf0nTG0m2D9kdI22hiXg/mEwzk4LvDQz5AoOm3PA8k+7GuaweG6A/lyDRHr9EfXSg
SHaOexvH3ZZib4WMsfsiH5NEohPrsFVB0KLsqzEeKbYHIuV6Jt4+0Dy847hhmk7dUc6bRqHzxLHWu6Qz
a4ty5+95vp+neNXLcjUoluipMp374RFqk6cy3PEkiiJ/nPMnU3/wxD8xHs1w/dZ+fxmR7A7knYfcHLieOzA
F00R7E7ZeNLEV5/akK9iDhGmvVy4PqJne+K2+ZYD9kVSWnya2aSgj0xpFkfvsWoh606GissYNBi3DOvZ9
Hb/buDn2e5Zh2RPP0rWt6QUtBk8jTcuOFrOjVqHjRpH37F/oXkRv1t3dE3EhlTEMx8bd6A3G/dbxOOhKv
h224mgWzz3rw2Sgn9smO9nK1rE36UrSPXTDoWQAzaPCNC4rAWFMk+i5N1vgKpjqIoWUOx8ZgnPVn/
smPfi3pz+3DU5COXNPJ7RsR+4LDR4fIKIsbD3Fi/+SMUxj9y1+lmQGvrG7I272HS+I8zu4lER/PrYkl6Y3t+V
22JLoz9IkCm1V0a09C1qShOXNHa7bsizHVJ/QQGQl6URuHnlJ6j6zLBD9+YiTrnGjWYtYxhjHDn+o+2G0T
mEyg/iI64j2p89N/ANGoyQZcp02ohMlM+sNfpjcnfUl20kcee6gO78btt1Z2JWcuGjPwjbfDlve8uZREk8+c
GL35et0JGdcWuG0LXSjFT73PQzwCoSTJK5gpSxs/Wf+2an+396ZaLepA2EY7YDxBhjEYtCW93/FGztgsE0
SilNft52v5/RQFRHFFr9GoxmprINOa6m03fHGdGvrCcMH88qk7OPegytY16a4zcBruxBEC6f3Xk/QOr35/
R8mFqqaszqGmS84bQexRUQa+fEPntmc3ogtDqRc7Spj2i2Z6mu5LgjuzoiQ9LJS14Zg2gIvByucGdYeUOlc
8ZjYYl/3/gAkWrej3cto9RpPGDqqP4ATb5wOu0cEdfsD63T/JlgcXRsMYqucjn+/nUcTnSFvLo0t+dBc27UP
r99a/1psES2lwEQcWlsJ7x7RmrTrSVut+9Gcxq2E7gO8HH7jhmkmoq1z6WOvJklt3D0Px8YF3TVXJr8XW
5obw7vilTFp3wxpVk+Y+v6VkJV2wyQdheZNRb/fshWlWnlj5oot2Wnbzf9RYHRyLbZk3e4oQoRFlQm8O/
Da6k13GSq7v/gRGhisgddjp11xmd+TjXbtgx4+Xqmoex6W1vXP5q2tyYTLwRra3bHSVvZL1pkBr9tCSGbs
gQ1Dn3P1j0S/fs2qvbE8Z4utNLbvI7428lps6aGNPtz4sQ29O2j2poN+QtW6/BI9WGqIHgReDXQwLiPo4u
wy+tFA21A1QXeJa6t5L6tHqyfEtrWKDhPeqv/Rsa4gaXcZtLJ2yKES9ZuOH/tC54BiexsyOFNsaWVNX9NXt
u6s815s03aNTiVopYMpH9ib8gfvydBncp095gwDgJ8GEelMfEncSbX+NAkIfYJ3Q6IvNg45WnUR29rpexl
nyh7psJRz7GU/Uu3KA5bAtFPskr6XG3t4wrBFc5dQbzZjsW1s2+uiaFx77lAkMm1I0PuFUBk9lbCs5FOzqL
ej6MW2do0YOmEpPAB4JZConYq6bUN9qdo9x+gTWWaM38PuDlLNtbxMEbUbi625i+dCzLhebFHQuqZ
XetGqrQcsgKyNLWn3jYpUqVw8MKHGhE5C8K07yAU3RXPFVrmx2CqOEFvF0ug8Dun7dVqlgRDBoYgm
xFw0g9jy2qnQ64h0A3kxwGtBQuVUsn5nk+RVlUecfCbLm2M7Rc29a6TJ2Fyx9Zh1Y8u2Ed0dvNaJByyAZ
tZmjLzDgqRUZfxQ6ipN8kkScpOaUhvfm81YbJlxDR2JrcBYpFku8zyLOfYwW+cn0nDKCBD6reaD2A5rdL5
SoQcArwRJjGsKeSZPQobRZzt44zCXU2S3Yluaw0VszY3Y3ik5H8QWXYltYVIPWACvnNVtc6Kt8pVgD+37x
QulJ1CSelf4SgvvmsViizCl7B1KMDqZ54y/wyZTCv3PxFbUGtxQwGtBC2vTj40RPnp3N/8kU26EafCd2CZ0i
dh612LrMg9YQNg6lWzWJ0JfnDRqFohQSu5vpjwIJwg4uVW9VowfNgFGM8X2fjeFrmXzxRbxyq09AHgh
EG2d8Qk+0VmzmLIwrZf3VFSamC5xI1z5bBmI7UL22pW8U7cvhXa8tokw3+eHbMcwWna4ka8bPkoL3E
+w4egzsWV3YjvNfDcCr942HgC8EDg4dXU0LhK5lOoRh+lYbBunx2LL8cTSeTsK/TpSBJbtQ6CDczlBZ7wvI
QwP12EZC75vjktTSQI1EluaK31Pm5HJBTLt2qVii4R6O4opyxbEFng1aGrtjfuNRSJqHhDbK8u2dpZ3xaxx
mqCpLcf6u0PlaoTAsn0ERIs3kyBvBn7Bh+uqXFHKD6ZcGGTtj8WWhOkESYCnQ7+cvoht69p5fo8hiliNQr
9aAZYt8KqwwpqY3Bg8dDUltojv9lNs6f0CGfUmMsiUPc7IIOvghYUFsgUgcXQzl4bCxvd6tkrFp8OMnPbR
vNCv69vQlc8WkckqBE0nNTg7JDW42vsVePGmL0kNjas4+GyBVyVoXBuSu9LjhNjifTtJI7xriiH0C8fGRf2Sl
7Zyam8Ea/yuODImGywTs9y4/ofBoXa18OYQtoPYBtVx/V45uhNbllWTZF9GIyCE8P2fT+JspR2l61rpzWI6
XXfYJp3XBqIRgJdiq10l8CyxRWwTT7BfUe+a3MiL2PqqTyxCwprNhNjutOmy2NF2ZJOJRu884JchmXEZuy
kLIk4wDaOIYzQttlQE7CRd3TYZAzwtJrlNFxPHpaFfZG9s3wMC+4tDLImsiS/DjB021BVKQ5wt8Eqg1DpJ0

TyxJWwqhew+9CvV5TCbK3prgyauFuh0sY5H8o792h5oZ+S+lXxQgjbygF+GVNZG5DroYC1lGQp5SArJ8Y
3YDndhQlat2pCb5zE/mMC//dZ5aX3kzQWNxBYHjU3px2Xs2uC7qphSjAb3hTLZJWJcDx0rVC1kkAEvA0K
YyjebYoQ+FdslcUBrXfsXgd7rNkAfTjST0tPVtlEFG0n4wfQnNRzNmvble1VDbvuvg5hxmqPrIhnG+lisONu
3B3IttgNIRFlz3DB0q22T3PYZRKXbYW8mCI/EFjFpC96vin67mw32i3I0mjBpKvwhtqWWQ+v3uoL+A7w
MlIuoeTMHwcmd2D4QjeCJZgh3RFzqnKKTpurqw/SRxul4vLRS6gSfX1ctB5lItXzC9il/F4gQtjfuyK/yB8g29
ROjdozQtSroZ2KLV8mhrLOlh4CTxGXUmwHChFBfuaNPCMHngqhpN+dcsa1qQ/LdD5La1CP/R9ios/aSS
B9XQ+X8B84tBYAfAiVl1WhrmlL6Pyq2LDd77HVQP6+zkIm0Ljrn7rYxTTbW/KCoU5/5WSMDOjoqZQf72f
4idJfmR2tVlq7wqDQNeW4Khj0Sa0k9RCg7s25Ddoais23KeXAaGZepLYpm7omJeZoeSuNMkaZJZ9DuVc
kpoX6lEvZtfWncyJ2PSKxKQQkJSp1Q1Jei4hmnUwDAPLAfbXa7/W67XvGfFFtEovGRkUTsZFlVRRJ0kT88
3B3y8f3Uj093yL2P0XAymX7Chtd/GX5R1cemOdbl2FuAGRGljhHyaKFi6pGoas4oq5ozHwKLMCa+1ukys
X3vM/N2fsFhdW5lU1dVEXQt3JRlukvKckfxt/UDWRfJ2EsSVWW6SYpqy9ClkB+VgP4DvAx4SFr/SbH1kG
gqjkYLGowLzilGfQFJU69nuIPRoR0kauVCE+sfhnDxji8EZ3j88aKwbYOTLKnT+bOYrzZnDir5uAgw5hHDZ4
0yx4UaxXKdejNAlJ8a6fvvf3PatZD5630cb3yG0Yz6fhZ7A5gF58oBI14P2bQFnIUP/Ak8JrYey1SCxqbuzf6n
SO5uZf8mTJ7mxyccm/VXgiaKNqbgJ+etKThjqM9TiNqwT1Cgss3pR46LWii2dKXL5eMjpowxOvNHk3zrjU
BdZTRW/jZa3BYAeB4oOC6eTZ4gwbEmn9fH/LtN9DFXErwIPwY5nM/w5IVOyTomqCNsfXTmlOlqK36yf
Wu71LJFpGhWywfIj4bMAtEy+KYybyR/oAMDwLMg69Zky60UD+G0O+R/Epp+twzC8na78J0H7mHF+fh
av1Yrmq/RfTQCYrJZ0/Mh9DpZ+sHjVf2cMzrpPmff3JGq7XLdB4BnwbOi1qYp5PJ0RyRk+WkMEQ7kN+e
04HWTgmHyc/CmPX0bompW+5x7E2KL13UqGBNS50tjvzzEkjZ4gsQhvxD06zuCIxi2wJ8AFX64isLAfyBO
kfhVSj97/uGboBzMpeRPeGv/GURxNgXpupAHTqbTdYMsz7KiTB9Yw0d+Wfz+RAJE0wNFX98hax/6D/A
ngD54yDSgmyrGn/xXSPDXdXP5DAvp34Hys8RiygUjaFJsMWV+EIY+e2QNnwblI96neSC2/cadT5Jy9civA
QB/FJit5Zp8kkH0dXwPTfLgt7+y/xSXgRNfffJBJW6PxVlu1p5AZFXsfvcwiSj9uv+QoFgv3JQXAP5ImB8ue+/
oij9lneWfh61++nNGdPX/J8nyw+Z/bwMAPBW0cCqHHjoNFpgNpo8ZspOP/P+d7YT9/O8FAC/NUq/vw
mrAK/ACIyUCqQUAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAC8/
wD/WTQII6qwKQAAAABJRU5ErkJggg=="></figure><p> </p><p>Functional Description:</p>
At any moment, the guards and the prisoner can see each other.
The "climbing part" of the escape takes no time.
You may assume that both the prisoner and the guards can change direction and velocity instantly and that they both have perfect reflexes (so they can react instantly to whatever the other one is doing).
The two guards can plan ahead how to react to the prisoner movements.
<p>Constraints:</p><p>1≤n≤50</p><p>0≤xi,yi≤1,000</p><p>Input Format:</p><p>The first line of the input contains n</p><p>The following n+1 lines describe $P1$,$P2$,…,Pn+1. The i<math xmlns="http://www.w3.org/1998/Math/MathML"><mi>i</mi></math>-th of such lines contain two integers xi, yi representing the coordinates of Pi=(xi,yi).</p><p>It is guaranteed that $P1$=(0,0) and xn+1=0. </p><p>The polygon with vertices $P1$,$P2$,…,Pn+1,Pn+2,Pn+3 (where Pn+2,Pn+3 shall be constructed as described in the statement) is guaranteed to be convex and such that there is no line containing three of its vertices.</p><p>Output Format:</p><p>Print a single real number, the minimum speed v that allows the guards to guarantee that the prisoner will not escape.</p><p> Your answer will be considered correct if its relative or absolute error does not exceed 10^−6.</p>

answer

```
 #include <bits/stdc++.h>

using namespace std;

typedef long long LL;
```

```
typedef unsigned long long ULL;
#define SZ(x) ((int)((x).size()))


vector<int> SortIndex(int size, std::function<bool(int, int)> compare) {
   vector<int> ord(size);
   for (int i = 0; i < size; i++) ord[i] = i;
   sort(ord.begin(), ord.end(), compare);
   return ord;
}

template <typename T>
bool MinPlace(T& a, const T& b) {
   if (a > b) {
      a = b;
      return true;
   }
   return false;
}

template <typename T>
bool MaxPlace(T& a, const T& b) {
   if (a < b) {
      a = b;
      return true;
   }
   return false;
}

template <typename S, typename T>
ostream& operator <<(ostream& out, const pair<S, T>& p) {
```

```
    out << "{" << p.first << ", " << p.second << "}";
    return out;
}

template <typename T>
ostream& operator <<(ostream& out, const vector<T>& v) {
    out << "[";
    for (int i = 0; i < (int)v.size(); i++) {
        out << v[i];
        if (i != (int)v.size()-1) out << ", ";
    }
    out << "]";
    return out;
}




struct pt {
    double x, y;
    pt(): x(0), y(0) {}
    pt(double x, double y): x(x), y(y) {}
};
pt operator -(pt A, pt B) { return {A.x-B.x, A.y-B.y}; }
double operator *(pt A, pt B) { return A.x*B.x + A.y*B.y; }
double norm(pt A) { return sqrt(A*A); }
pt operator /(pt A, double lambda) { return {A.x/lambda, A.y/lambda}; }
ostream& operator<<(ostream& out, pt P) {
    out << "(" << P.x << ", " << P.y << ")";
    return out;
}
```

```
// P(s) = as² + bs +c
// {l < s < r: P(s) < 0}
void neg_interval(double a, double b, double c, double l, double r,
            vector<pair<double,double>>& ans) {
  assert(abs(a-1) < 1e-5 or a < 0);
  if (a > 0) {
    b /= a, c /= a, a = 1;
    if (b*b < 4*a*c) return;
    double delta = sqrt(b*b-4*a*c);
    double l0 = (-b - delta)/2;
    double r0 = (-b + delta)/2;
    l0 = max(l0, l);
    r0 = min(r0, r);
    if (l0 < r0) ans.emplace_back(l0, r0);
  }
  if (a < 0) {
    double s = sqrt(a*a + b*b + c*c);
    a /= s, b /= s, c /= s;
    if (b*b < 4*a*c) {
      ans.emplace_back(l, r);
      return;
    }
    double delta = sqrt(b*b-4*a*c);
    double l0 = (-b + delta)/(2*a);
    double r0 = (-b - delta)/(2*a);
    if (-0.01 < a) {
      if (b > 0) l0 = (-2*c)/(b+delta);
      else r0 = (2*c)/(-b + delta);
    }

    if (l < l0) ans.emplace_back(l, min(l0, r));
```

```
        if (r0 < r) ans.emplace_back(max(l, r0), r);
    }
}
void neg_interval(double a, double b, double c, double d,
                double l0, double r0, double l1, double r1,
                vector<pair<double,double>>& ans) {
    assert(a >= 1.99);
    double lmin = (-2 * r1 - c)/a;
    double rmin = (-2 * l1 - c)/a;
    lmin = max(lmin, l0);
    rmin = min(rmin, r0);
    if (lmin < rmin) neg_interval(1-a*a/4, b-a*c/2, d-c*c/4, lmin, rmin, ans);
    neg_interval(1, a*l1+b, l1*l1+c*l1+d, l0, r0, ans);
    neg_interval(1, a*r1+b, r1*r1+c*r1+d, l0, r0, ans);
}

void good_interval(pt A, pt B, pt C, pt D, double v, double l, vector<pair<double,double>>& ans) {
    double lA = 0;
    double rA = norm(B-A);
    pt dA = (B-A)/rA;
    double lC = 0;
    double rC = norm(D-C);
    pt dC = (D-C)/rC;
    if (v >= sqrt(2/(1+dA*dC))) return;
    double c1 = v*v - 1;
    double c2 = 2*(1- (v*v)*(dA*dC));
    double c3 = 2*v*v*((A-C)*dA) + 2*l;
    double c4 = -2*v*v*((A-C)*dC) - 2*l;
    double c5 = v*v * ((A-C)*(A-C)) - l*l;
    c2 /= c1, c3 /= c1, c4 /= c1, c5 /= c1;
```

```
    neg_interval(c2, c3, c4, c5, lA, rA, lC, rC, ans);
}

bool nonempty_intersection(pair<double,double> I1, pair<double,double> I2) {
    return max(I1.first, I2.first) < min(I1.second, I2.second);
}

const int MAXN = 51;
pt P[MAXN];
double len[MAXN];

int main() {
    int N;
    cin>>N;
    for (int i = 0; i <= N; i++) cin >> P[i].x >> P[i].y;
    for (int i = 1; i < N; i++) len[i] = len[i-1] + norm(P[i]-P[i-1]);

    if (N <= 2) {
        cout << 1 << "\n";
        return 0;
    }

    double l = 1;
    double r = 20;
    for (int it = 0; it < 50; it++) {
        bool good = false;
        double v = (l+r)/2;
        for (int i = 1; i < N-1; i++) {
            vector<pair<double,double>> bef, aft;
            for (int j = 0; j < i; j++)
                good_interval(P[i], P[i+1], P[j], P[j+1], v, len[j]-len[i], bef);
```

```
        for (int j = i+1; j < N; j++)
            good_interval(P[i], P[i+1], P[j], P[j+1], v, len[j]-len[i], aft);
        for (auto& I: bef) for (auto& J: aft) good |= nonempty_intersection(I, J);
      }
      if (good) l = v;
      else r = v;
    }

    cout.precision(10);
    cout << l << "\n";
}
```

question

Question description: Kanthamaran is the organizer of the Famous Boat Competition in God's Own Country. There are n people who want to participate in a boat competition. The weight of the i-th participant is wi. Only teams consisting of two people can participate in this competition. As an organizer, Kanthamaran think that it's fair to allow only teams with the same total weight. So, if there are k teams ($a1$,$b1$), ($a2$,$b2$), …, (ak,bk), where ai is the weight of the first participant of the i-th team and bi is the weight of the second participant of the i-th team, then the condition $a1$+$b1$=$a2$+$b2$=⋯=ak+bk=s, where s is the total weight of each team, should be satisfied. Kanthamaran request you to choose such s that the number of teams people can create is the maximum possible. Note that each participant can be in no more than one team. Constraints: 1≤t≤1000 1≤n≤50 Input Format: The first line of the input contains one integer t the number of test cases. Then t test cases follow.
The first line of the test case contains one integer n representing the number of participants. The second line of the test case contains n integers $w1$,$w2$,…,wn where wi is the weight of the i-th participant. Output Format: For each test case, print one integer k: the maximum number of teams people can compose with the total weight s, if you choose s optimally.

answer

```
#include <bits/stdc++.h>
using namespace std;
int a[200];
template <typename Competition>
```

```
Competition Boat(Competition t){
    while(t--) {
        memset(a,0,sizeof(a));
        int n;
        cin>>n;
        for(int i=1,x;i<=n;i++){
            cin>>x;
            a[x]++;
        }
        int ans=0;
        for(int i=1;i<=100;i++){
            int sum=0;
            for(int j=1;j<=i;j++)
                sum+=min(a[j],a[i-j]);
            sum/=2;
            ans=max(ans,sum);
        }
        cout<<ans<<"\n";
    }
    return 1;
}
int main()
{
    int t;
    cin>>t;
    Boat(t);
    return 0;
}
```

question

Problem Description: Caleb and Irfan are purchasing apples which were priced according to their size.But their budget is minimum. So they plan to choose one small, one medium and one large apple so that it will fit in their budget.
So can you help them choose the right apple by creating a logic by naming three apples they choose as apple1,apple2,apple3. Then check the condition if apple2 is greater than apple1 and apple3 is greater than apple2. Constraints: 1≤ apple1 ≤600 1≤ apple2 ≤600 1≤ apple3 ≤600 Input format:

First Line: Single number of type integer representing the size of apple1 Second Line: Single number of type integer representing the size of apple2 Third Line: Single number of type integer representing the size of apple3

Output Format:
Print as “Fit into Budget” or “Dosen't fit into Budget” based on the condition. If the input is insufficient for deriving the result throw an exception message “Incomplete information”

answer

```
#include <iostream>
using namespace std;
int main()
{
   int apple1,apple2,apple3;
   try{
      cin>>apple1;
      cin>>apple2;
      cin>>apple3;
      if(cin){
         if(apple2>apple1&&apple3>apple2)
         cout<<"Fit into Budget";
         else
         cout<<"Dosen't fit into Budget";
      }
      else throw 0;
   }
   catch(int budget){
      cout<<"Incomplete information";
   }
         return 0;
```

}

question

Problem Description: You are playing a Billiards-like game on an N×N table, which has its four corners at the points {(0,0),(0,N),(N,0), and (N,N)}. You start from a coordinate (x,y), (0<x<N,0<y<N) and shoot the ball at an angle 45∘ with the horizontal. On hitting the sides, the ball continues to move with the same velocity and ensuring that the angle of incidence is equal to the angle of reflection with the normal, i.e, it is reflected with zero frictional loss. On hitting either of the four corners, the ball stops there and doesn't move any further. Find the coordinates of the point of collision, when the ball hits the sides for the Kth time. If the ball stops before hitting the sides K times, find the coordinates of the corner point where the ball stopped instead. Constraints: 2≤N≤25 1≤K≤25 Input Format: Each testcase contains a single line of input, which has four space separated integers - N, K, x, y, denoting the size of the board, the number of collisions to report the answer for, and the starting coordinates. Output Format: In the only line of output print the coordinates of the ball when it hits the sides for the Kth time, or the coordinates of the corner point if it stopped earlier. Explanation: Assume the N=5 and K=5 x=4 and y=4 then We shoot the ball from coordinates (4,4), and we need to find its coordinates after it has collided with sides 5 times. However, after shooting, the ball goes directly to the corner (5,5), and stops there. So we report the coordinates (5,5). <figure class="image"><img
src="
VR4nO3df5Tdd13n8Ut/N2naJE2KBVx+hB+ljdUiBkF+2XaZgxbcUgFtgK0sAaYiYqhUBFqwZWFycMWWF
cRtXbvEcRexblC7uq47nCntrpgVT5Rj6c5a9hy3Uyhyk9Lmd977R3KnM5O57+8nMMnne+88nue8zqFkm
g5P5r6fuTN3pp0YMiYnJ2NkZGTmr8fHx2N0dDT9ezqdzvF+twaeiYmJ2u9Cq+GnGY5y+GmGo/4MZcU6
nU5MTk5GRMTIyEiMj483xx+muGoP0NZscnJyeh0OtHpdBqflUeIeQkeRDn8NMNRDj/
NcNQfFQsxL8GDKIefZjjK4acZjvqjYiHmJXgQ5fDTDEc5/DTDUX9ULMS8BA+iHH6a4SiHn2Y46o+KhZiX4E
GUw08zHOXw0wxH/VGxEPMSPIhy+GmGoxx+muGoPyoWYl6CB1EOP81wlMNPMxz1R8VCzEvwIMrhp
xmOcvhphqP+qFiIeQkeRDn8NMNRDj/NcNQfFQsxL8GDKIefZjjK4acZjvqjYiHmJXgQ5fDTDEc5/DTDUX9
ULMS8BA+iHH6a4SiHn2Y46o+KhZiX4EGUw08zHOXw0wxH/VGxEPMSPIhy+GmGoxx+muGoPyoWYl6C
B1EOP81wlMNPMxz1R8VCzEvwIMrhpxmOcvhphqP+qFiIeQkeRDn8NMNRDj/NcNQfFQsxL8GDKIefZjj
K4acZjvqjYiHmJXgQ5fDTDEc5/DTDUX9ULMS8BA+iHH6a4SiHn2Y46o+KhZiX4EGUw08zHOXw0wxH/V
GxEPMSPIhy+GmGoxx+muGoPyoWYl6CB1EOP81wlMNPMxz1R8VCzEvwIMrhpxmOcvhphqP+qFiIeQk
eRDn8NMNRDj/NcNQfFQsxL8GDKIefZjjK4acZjvqjYiHmJXgQ5fDTDEc5/DTDUX9ULMS8BA+iHH6a4SiHn
2Y46o+KhZiX4EGUw08zHOXw0wxH/VGxEPMSPIhy+GmGoxx+muGoPyoWYl6CB1EOP81wlMNPMxz1
R8VCzEvwIMrhpxmOcvhphqP+qFiIeQkeRDn8NMNRDj/NcNQfFQsxL8GDKIefZjjK4acZjvqjYiHmJXgQ5f
DTDEc5/DTDUX9ULMS8BA+iHH6a4SiHn2Y46o+KhZiX4EGUw08zHOXw0wxH/VGxEPMSPIhy+GmGoxx
+muGoPyoWYl6CB1EOP81wlMNPMxz1R8VCzEvwIMrhpxmOcvhphqP+qFiIeQkeRDn8NMNRDj/NcNQf
FQsxL8GDKIefZjjK4acZjvqjYiHmJXgQ5fDTDEc5/DTDUX9ULMS8BA+iHH6a4SiHn2Y46o+KhZiX4EGUw08
zHOXw0wxH/VGxEPMSPIhy+GmGoxx+muGoPyoWYl6CB1EOP81wlMNPMxz1R8VCzEvwIMrhpxmOcv
hphqP+qFiIeQkeRDn8NMNRDj/NZI7WbpqIzXfcFxERW+9+MNZumpiz9dfdc0z/rMtv3n7U77Fl2wMzv3
asv9/xRsVCzEtwaHL4aYajHH6a6edo4607YuOtO2b+esu2B77r2K6/7p6ZePf79d4fHtqAioWYl+DQ5PDT
DEc5/DSzkKN7v9qNtZsm4t6vdmf+uy3bHojLb97+Xf2z1l93T2y9+8G+v74Yf2BYTFQsxLwEhyaHn2Y4yu
GnmYUcLRTuzXfc913HfO2miTTmpW9zolCxEPMSHJocfprhKIefZhZytPHWHUd9unvjrTu+q6+X957tL/T1
8tlcfvP21nyqXcVCzEtwaHL4aYajHH6aWcjR5TdvT7+23XubtZu+c7+9F9Qt9IeG2V+rr4mKhZiX4NDk8N

MMRzn8NLOQo6avbffo9+y6lM133HfUM/zF+HT+YqFiIeYlODQ5/DTDUQ4/zXw3MW96dXoTC73gTcxb
hpg349Dk8NMMRzn8NPOdfpq99zXw7+bFahtv3XFUuH2avWWIeTMOTQ4/zXCUw08zpS+Amx/d9dfd
M+e/a/q2si3bHpjzB4Qt2x5Y8A8DXgDXMsS8GYcmh59mOMrhp5nSb02b/9Pb5j97XugPALNZ6CfIzf4+9
h6+Na1liHkzDk0OP81wlMNPM6U/NKaJ0q+zZ/SerbcFFQsxL8GhyeGnGY5y+Gmm9Me5Ztz71e6i/OQ2
P861hYh5Mw5NDj/NcJTDTzOl/6KV441/0UpLEfNmHJocfprhKIefZjjqz5Ku2IEDB+Iv//Ivo9PpxJe//OXa70
6r8SDK4acZjnL46c/evXvj3nvvjfe+972xY0c7vhWsbSzZmH9ky6/GSSedFKufckGcvPy8WP2kdbHinFVxxx
131H7XWolDk8NPMxzl8LMwN/7Kh6PTeUKsfMpz47yLLouV3/O0WH3e+fHZz3629rvWKoYy5qOjo9Hp
dGY2NTU159evfsO/jPOf88Ox6qrb53zrwcorPh7nPPmCeN/73lfpPW8vDk0OP81wlMPP0fzYq66Mc5/zk
lj9+q1zbvU5r9wSq85/Rnz0ox+t/S62hqGN+eTk5IK/tnXr1lj15Gcf9T2EvZ278XNx+vJz4gtf+MIJfq/bjUOT
w08zHOXwM5dPfepTcd4zLul7q1e/7jNx8imnxvbt7fhxqrUZypiPjIz0jfmGF700lm94W98PkLWbJuKM57
wyfuK1b4zp7l47sjvvmqj+PrR5/HDEz+Ju3QUXx4qXvie91ctfMBrXvuPnT3Bh2snQxrz3KfaxsbE5v3bm8h
Vx7hv/c/oBsvp1/yHOPO9ZcfnN2+3IXnD9RPX3oc3jhyN+Fm+Xfeh/RqfTibX/6s/TW73qX3wqfuCHfqRSa
drFUMa8x9TUVHQ6nZln6QcPHownnHRSrHnzn+Uxf+3vxKnnPCme/54JMzM7wfuBd34+Tjr1zPROr900
ESt/4jfiyU97dkxMTCz5DXXMIw5//Xx8fHzmr5990SWx8tWfSD9Azv7R98cPvvSK+I/3TNuRfej2iervQ5vH
D0f8LO5Wn//0WPWTv53e6rN+5F3x02/8mYqFaQ9LLuY33XRTrH3uy9MPkHOeclF8/N/+VvWvGbVpvp
7HD0f8nMi9+e3vjHMueEX/W/2W/xbnPPHpvkXtCEMd896n2ed/a9oLXvTiWPb9P33UB8eaa/4kVlx0R
Vz241dV/0Bu2xwafjji50Rt+477Y8vHfi3OfeJTYtnzrjn6u47e+Iex4oJXxGs3vrlSXdrHUMZ89veYL/Sq9m98
4xvxmtddHWufemGcedGVcfqzXhHLL3xVnLnq/Lj6Z66NB7+1p/oHc9vm0PDDET/He397//+N239na7z/
hhvj/TfcGO981y/EhT+wIU49Y3mcuf6qOONZr4jlz/2xOP2s1fEL172nQl3ay1DGvJQvfOEL8cLXXR+nP+2lc
fk1N8Yf//lk9Q/mts6h4Ycjfo7XvvbQzvjsH/5x3PThj8yEvLftO+6Pz33+T+PiV2+ONRdfEa+59sPxpS99qXY+
WseSjnlExNt+6yux6jX/Lq77zFerf0C3eUv50PDDET/Hb71Pqc+P+PtvuDE++4d/PPN2P/lrfxMbrp+ILdv+o
XY2WomYi3nRluqh4Ycjfo7P5n9Kff5u+vBH4msP7Zx5ezHPEXMxL9pSOzT8cMTP8Vn2KfXZ+9O/uHvO3
yfmOWIu5kVbKoeGH474Ob7bvuP+NOLvv+HG+OSnbz/q7xPzHDEX86ItlUPDD0f8HP81BX37jvuP+nvEP
EfMxbxoS+nQ8MMRP8d3X3toZ9zyiU/G298+Gr/4nl/q+6K32RPzHDEX86ItpUPDD0f8HL997aGdMy98u
/6X3hu/sPndfV/0NntiniPmYl60pXJo+OGIn+O32SF//w03xsdv/Y2493/9bd8Xvc2emOeIuZgXbSkcGn444
uf4baGQ956Fb99x/4Iveps9Mc8RczEv2rAfGn44qr1h9pOFvHRiniPmYl60YT40/HDUhg2rn8UI+XRXzJsQ
czEv2rAeGn44asuG0c9ihXy6K+ZNiLmYF20YDw0/HLVpw+ZnMUM+3RXzJsRczIs2bIeGH47atmHys9ghn
+6KeRNiLuZFG6ZDww9Hbdyw+DkeIZ/uinkTYi7mRRuWQ8MPR23dMPg5XiGf7op5E2Iu5kUbhkPDD0dt
3qD7OZ4hn+6KeRNiLuZFG/RDw0/9cTS8fo53yKe7Yt6EmIt50Qb50PDTjnE0nH5ORMinu2LehJiLedEG9
dDw055xNHx+TlTIp7ti3oSYi3nRBvHQ8NOucTRcfk5kyKe7Yt6EmIt50Qbt0PDTvnE0PH5OdMinu2LehJiL
edEG6dDw085xNBx+aoR8uivmTYi5mBdtUA4NP+0dR4Pvp1bIp7ti3oSYi3nRBuHQ8NPucTTYfmqGfLor5
k2IuZgXre2Hpvb44WiY/dQO+XRXzJsQczEvWpsPTRvGD0fD6qcNIZ/uinkTYi7mRWvroWnL+OFoGP20Je
TTXTFvQszFvGhtPDRtGj8cDZufNoV8uivmTYi5mBetbYembeOHo2Hy07aQT3fFvAkxF/OitenQtHH8cDQ
sftoY8umumDch5mJetLYcmraOH46GwU9bQz7dFfMmxFzMi9aGQ9Pm8cPRoPtpc8inu2LehJiLedFqH5
q2jx+OBtlP20M+3RXzJsRczIvmEPPD0XD6GYSQT3fFvAkxF/OiOcT8cDR8fgYl5NNdMW9CzMW8aA4xPx
wNl59BCvl0V8ybEHMxL5pDzA9Hw+Nn0EI+3RXzJsRczIvmEPPD0XD4GcSQT3fFvAkxF/OiOcT8cDT4fgY1
5NNdMW9CzMW8aA4xPxwNtp9BDvl0V8ybEHMxL5pDzA9Hg+tn0EM+3RXzJsRczIvmEPPD0WD6GYa
QT3fFvAkxF/OiOcT8cDR4foYl5NNdMW9CzMW8aA4xPxwNlp9hCvl0V8ybEHMxL5pDzA9Hg+Nn2EI+3R
XzJsRczIvmEPPD0WD4GcaQT3fFvAkxF/OiOcT8cNR+P8Ma8umumDch5mJeNIeYH47a7WeYQz7dFfM
mxFzMi+YQ88NRe/0Me8inu2LehJiLedEcYn44aqefpRDy6a6YNyHmYl40h5gfjtrnZ6mEfLor5k2IuZgXzSH
mh6N2+VlKIZ/uinkTYi7mRXOI+eGoPX6WWsinu2LehJiLedEcYn44aoefpRjy6a6YNyHmYl40h5gfjur7Wa
ohn+6KeRNiLuZFc4j54aiun6Uc8umumDch5mJeNIeYH47q+VnqIZ/uinkTYi7mRXOI+eGojh8hPzwxzxFz
MS+aQ8wPRyfej5A/PjHPEXMxL5pDzA9HJ9aPkM+dmOeIuZgXzSHmh6PjszvG/yDecu274tnrnxcjr7oqbv
iVj8SX//4fhHzexDxnqGM+NjYWIyMj6duIedkcYn44WvxtvOatsfKJT43VL3xrnPPKLbHi5b8cp6x5TixbsSq
uvOoqIZ81Mc8Z6ph3Oh0xX6Q5xPxwtLi75q3viGVrnhZr3vxnsXbTxJyd82Mfi5NPOTU2/+L1Qn5kYp4zt
DEfGxvzzHwR5xDzw9Hi7fe3/WmsOPdJseaaPzkq5L2d9eLNsf6SDdXf17ZMzHOGMuZTU1MxMjISk5OT
Yr5Ic4j54Wjx9ksf+FCct2Fj35D3dtoZy+KB6W7197cNE/OcoYz5unXrYmpq6phifvWWibjzLjOz47/v3/DSO
PvSGxpjvvrpz4stH/9k9fe3DbvshonYcP1EXHvLRExM2PwNXczHxsZifHw8IsIz80XcnXd5VsUPR4u1azb9b

Cx/wdsbY37WuU+Ke7b/XfX3tw3zzDxn6GLe6XSO2rp16/q+vZiXzSHmh6PF2yd+8/Y47ezvSUO+6qrb48
yzVlZ/X9syMc8ZupjPxjPzxZtDzA9Hi7tnXXRJrHjZ9X1jvvIZG+J9Hxqr/n62ZWKeI+ZiXjSHmB+OFnfb/st/j+
Vnr4pll7wxzn3TtpmIn335TbH66T8YL3/FFdXfxzZNzHOGOuYliHnZHGJ+hsnRQzv3xiO7D8Se/QfjG7v2zf
m1h3fti937DsYjuw98x7//rt0HYs++g/HwI4d/78f2Hozds/66t7/5+6/Fpa98TZx25vJYdu5T4vRlK2Lt+U+N
G//1v6nuqG0T8xwxF/OitekQt3H8DI6jr+/cFwcPPX4DHt3zeLR3PrZ/zn3Yu//gMf/+e/cfmvN7fHv3gZn/v
O/Aob5/3y2/+e/j7/73P1b309aJeY6Yi3nR2nKI2zp+BsfRo3sPRkTE2k0TsfmO++LgoYjp7t54qLs3Dh46FP
d+tRtrN03Exlt3xKGIY/q9HzkS7o237oi1mybi3q9248DBQ3H5zdtj/XX3RETEzsf2t9pPWyfmOWIu5kVzaP
gZBkfffGRfRByO7cZbd0RExK7HDj8zf3TP3BAfa8wf3nX49+79YaAX830HDj9TX3/dPUf+8HAoHtrZTj9tnpj
niLmYF82h4WcYHO3Zf3Amtr3QTnf3xj99+/Cn17dseyDWX3dPXH7z9mOO+Z79h5/xr7/unjnPzL+950Ac
mvV7R0R8e8/RX49vg582T8xzxFzMi+bQ8DMMjg4cPBRbtj0Ql9+8PSIiHtl9IL6+8/B/HxEzAT7WmPc+vb
75jvti4607YuvdD878Xo/sPhB79z/+qf2tdz8Ye/Yd/bX4Nvhp88Q8R8zFvGgODT+D7uihnXsj4vCn0TffcV9
ERHx95+FXrkdEXH7z9pn//lhi/q0jz+p7AZ/9n3sx771N759x4ODRL4Sr7aftE/McMRfzojk0/Ay6o97Xyy+/
eXts2fZAHDx0+NPdEYcD33u23nubkpg/vOvwK+N7n7rfeveDETE35t/efSC+sevor9U/1DI/bZ+Y54i5mBf
NoeFn0B31vu1s/XX3xNa7H4z9R16YtvmO+2a+lt2jF/OIiF3J95v3Xty2dtNEbNn2wMzfPzvmBw4eiq/v3B
eHDh3+Z/X+0DD/e85r+2n7xDxHzMW8aA4NP4PuqPep7l7MDx35dvCmf9lJRET30f3xT9+e+y1lXz/yafv
Nd9yX/v2b77hvJvqzY/71XWJ+LBPzHDEX86I5NPwMuqPep7p7n2aPiJmgz2f2M/MDBw/NPIuf/1PhHjvy
Pevzmf3MPCJm3q73afbe97a3yU/bJ+Y5Yi7mRXNo+BkGR4cOzX0B3M7H9seBg4dmdvBI3WfH/OChud
9W9q15z9Af23twzu9xKObGfM/+gzOf4u+9AG7ffi+AO9aJeY6Yi3nRHBp+hsHRvgNzvzXtsb1zv0Vs9qvO
ey+Ai5j7B4D5P8t9/vYfOHTUq9lnf219690PHvXPbYufNk/Mc8RczIvm0PAzDI52PXZgzg+NiYj45rwXos3+
WeqP7nn8B8FsvfvB2LvAM+r5e3jXvpnvW39s7+PPyrdse2Dma/Dzv/7eFj9tnpjniLmYF82h4WdYHO07c
GjOt4jtXuAHuOx67PAL3nbvO/wT43qfYs9e2T57D+/aN/P19d7X5Xs/znWhZ+Vt8tPWiXmOmIt50RwafO
bFUe9Ht/ZeaZ4Fuvvo4/8WtUPR/Cn2hXbw0OP/opVDhw7/oJo2+2nrxDxHzMW8aA4NP8Pk6NFZr0Kf/
/3e87frsQPx6J4D8fB3EPLp7uF/t3nE4VfOz3/xXFv9tHFiniPmYl40h4afYXP00M69C/7by47XP2vQ/LRtYp
4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5k
VzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH474
qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/Mc
MRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+a
Q8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET
82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniL
mYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3nRHBp
+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4Ycjfmp
OzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8z
FvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tD
wwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET82Jec5QxnxkZCQ6nU50Op0YHx9P31bMy+bQ8M
MRPzUn5jlDGfNewCcnJ6PTyf8ninnZHBp+OOKn5sQ8ZyhjPhsxX5w5NPxwxE/NiXnOUMd8fHw8RkZG0r
cR87I5NPxwxE/NiXnOUMZ8bGwsOp1OY8gjHo/51Vsm4s67zMysjbvshonYcP1EXHvLRExM2PwNZcx79
L5mPjU11fdtPDMv2513edbAD0f81Jtn5jlDHfOIw69sn5yc7PvrYl42h4YfjvipOTHPGbqYzw731NSUZ+aL
NIeGH474qTkxzxnKmPe+x7zT6aTPyiPEvHQODT8c8VNzYp4zdDE/VsS8bA4NPxzxU3NiniPmYl40h4Yfjvi
pOTHPEXMxL5pDww9H/NScmOeIuZgXzaHhhyN+ak7Mc8RczIvm0PDDET81J+Y5Yi7mRXNo+OGIn5oT8
xwxF/OiOTT8cMRPzYl5jpiLedEcGn444qfmxDxHzMW8aA4NPxzxU3NiniPmYl40h4YfjvipOTHPEXMxL5p
Dww9H/NScmOeIuZgXzaHhhyN+ak7Mc8RczIvm0PDDET81J+Y5Yi7mRXNo+OGIn5oT8xwxF/OiOTT8c
MRPzYl5jpiLedEcGn444qfmxDxHzMW8aA4NPxzxU3NiniPmYl40h4YfjvipOTHPEXMxL5pDww9H/NScm
OeIuZgXzaHhhyN+ak7Mc8RczIvm0PDDET81J+Y5Yi7mRXNo+OGIn5oT8xwxF/OiOTT8cMRPzYl5jpiLedE
cGn444qfmxDxHzMW8aA4NPxzxU3NiniPmYl40h4YfjvipOTHPEXMxL5pDww9H/NScmOeIuZgXzaHhhy
N+ak7Mc8RczIvm0PDDET81J+Y5Yi7mRXNo+OGIn5oT8xwxF/OiOTT8cMRPzYl5jpiLedEcGn444qfmxDx
HzMW8aA4NPxzxU3NiniPmYl40h4YfjvipOTHPEXMxL5pDww9H/NScmOeIuZgXzaHhhyN+ak7Mc8RczI
vm0PDDET81J+Y5Yi7mRXNo+OGIn5oT8xwxF/OiOTT8cMRPzYl5jpiLedEcGn444qfmxDxHzMW8aA4NP
xzxU3NiniPmYl40h4YfjvipOTHPEXMxL5pDww9H/NScmOeIuZgXzaHhhyN+ak7Mc8RczIvm0PDDET81J+

Y5Yi7mRXNo+OGIn5oT8xwxF/OiOTT8cMRPzYl5jpiLedEcGn444qfmxDxHzMW8aA4NPxzxU3NiniPmYl4
0h4YfjvipOTHPEXMxL5pDww9H/NScmOeIuZgXzaHhhyN+ak7Mc8RczIvm0PDDET81J+Y5Yi7mRXNo+O
GIn5oT8xwxF/OiOTT8cMRPzYl5jpiLedEcGn444qfmxDxHzMW8aA4NPxzxU3NiniPmYl40h4YfjvipOTHP
EXMxL5pDww9H/NScmOeIuZgXzaHhhyN+ak7Mc8RczIvm0PDDET81J+Y5Yi7mRXNo+OGIn5oT8xwxF/
OiOTT8cMRPzYl5zlDGfGRkJDqdTnQ6nRgbG0vfVszL5tDwwxE/NSfmOUMX86mpqRgfH5/5606nE1NTU
33fXszL5tDwwxE/NSfmOUMX8/mMjIzE5ORk318X87I5NPxwxE/NiXnO0MfcM/PFmUPDD0f81JyY5wx1
zEdHR4u/Zn71lom48y4zM2vjLrthIjZcPxHX3jIRExM2f0Mb85GRkcaQR3hmXro77/KsgR+O+Kk3z8xzhjL
mTV8nn42Yl82h4YcjfmpOzHOGLubj4+MxOjpa/PZiXjaHhh+O+Kk5Mc8ZupiPjY3NfI95b1ncxbxsDg0/HP
FTc2KeM3QxP1bEvGwODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4Ycjfm
pOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8
zFvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tD
wwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VN
zYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWI
u5kVzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH
474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/
McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VNzYsS1NfEAAAtWSURBVJ4j5mJeNI
eGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+
aE/McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFz
MS+aQ8MPR/zUnJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/
HDET82JeY6Yi3nRHBp+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zU
nJjniLmYF82h4YcjfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3n
RHBp+OOKn5sQ8R8zFvGgODT8c8VNzYp4j5mJeNIeGH474qTkxzxFzMS+aQ8MPR/zUnJjniLmYF82h4Yc
jfmpOzHPEXMyL5tDwwxE/NSfmOWIu5kVzaPjhiJ+aE/McMRfzojk0/HDET82JeY6Yi3nRHBp+OOKn5sQ
8Z2hjPj4+Hp1O8/88MS+bQ8MPR/zUnJjnDGXM161bF6Ojo2K+iHNo+OGIn5oT85yhjHkPMV+8OTT8c
MRPzYl5jpiLedEcGn444qfmxDxHzI/E/OotE3HnXWZm1sZddsNEbLh+Iq69ZSImJmz+xNwz86LdeZdnDfx
wxE+9eWaeI+ZiXjSHhh+O+Kk5Mc8RczEvmkPDD0f81JyY54i5mBfNoeGHI35qTsxzhjrmJYh52Rwafjjip+
bEPEfMxbxoDg0/HPFTc2KeI+ZiXjSHhh+O+Kk5Mc8RczEvmkPDD0f81JyY54i5mBfNoeGHI35qTsxzxFzMi
+bQ8MMRPzUn5jliLuZFc2j44YifmhPzHDEX86I5NPxwxE/NiXmOmIt50Rwafjjip+bEPEfMxbxoDg0/HPFT
c2KeI+ZiXjSHhh+O+Kk5Mc8RczEvmkPDD0f81JyY54i5mBfNoeGHI35qTsxzxFzMi+bQ8MMRPzUn5jliLuZ
Fc2j44YifmhPzHDEX86I5NPxwxE/NiXmOmIt50Rwafjjip+bEPEfMxbxoDg0/HPFTc2KeI+ZiXjSHhh+O+Kk
5Mc8RczEvmkPDD0f81JyY54i5mBfNoeGHI35qTsxzxFzMi+bQ8MMRPzUn5jliLuZFc2j44YifmhPzHDEX8
6I5NPxwxE/NiXmOmIt50Rwafjjip+bEPEfMxbxoDg0/HPFTc2KeI+ZiXjSHhh+O+Kk5Mc8RczEvmkPDD0f8
1JyY54i5mBfNoeGHI35qTsxzxFzMi+bQ8MMRPzUn5jliLuZFc2j44YifmhPzHDEX86I5NPxwxE/NiXmOmIt
50Rwafjjip+bEPEfMxbxoDg0/HPFTc2KeI+ZiXjSHhh+O+Kk5Mc8RczEvmkPDD0f81JyY54i5mBfNoeGHI3
5qTsxzxFzMi+bQ8MMRPzUn5jliLuZFc2j44YifmhPzHDEX86I5NPxwxE/NiXmOmIt50Rwafjjip+bEPEfMxb
xoDg0/HPFTc2KeI+ZiXjSHhh+O+Kk5Mc8RczEvmkPDD0f81JyY54i5mBfNoeGHI35qTsxzxFzMi+bQ8MM
RPzUn5jliLuZFc2j44YifmhPzHDEX86I5NPxwxE/NiXmOmIt50Rwafjjip+bEPEfMxbxoDg0/HPFTc2KeI+ZiX
jSHhh+O+Kk5Mc8RczEvmkPDD0f81JyY54i5mBfNoeGHI35qTsxzxFzMi+bQ8MMRPzUn5jliLuZFc2j44Yif
mhPzHDEX86I5NPxwxE/NiXmOmIt50Rwafjjip+bEPEfMxbxoDg0/HPFTc2KeI+ZiXjSHhh+O+Kk5Mc8Rcz
EvmkPDD0f81JyY54i5mBfNoeGHI35qTsxzxFzMi+bQ8MMRPzUn5jlLOubf/OY348c33xbLvv+n46fevzX+
z//7p+ofsG2dQ8MPR/zU2t9N/WO85G2fjKde/s64duz3Y/fu3bXz0TqWbMzfd8OHotPpxLKnXBKnrF4Xy
5+8Pk459bT46Md+vfoHbhvn0PDDET81Nvrz74kzlq2IZd/7g7HswlfHWU+6MJaddXbcfvvttTPSKpZkzF99
5Wtj7XN/NFb/1O/F2k0TM1t15afjnKdeEqPveFf1D+C2zaHhhyN+TvReculIrF0/Eue+4Q/m3OqVr741zn7
yRfHBD36wdk5aw5KL+W233RZrnnbxnA+M2Vvzpj+KM885L37vc39U/QO5TXNo+OGInxO5mz7yq7Hm
mRv63upzr/5PcdqZZ8UXv/jF2llpBUsu5s9/wYvjrBf+XN8PkLWbJmLFy66PS694ffyP+7t2ZJ/67ET196HN4
4cjfhZ3T37GhXH2pR9Ib/VZL3xHvG30HbWz0gqWXMxPP3N5rHnT59MPkNWv/904Y80z4mUf+pId2fN/
caL6+9Dm8cMRP4u3l3zgi/GEJ5wUa9/yF+mtXvWa34qLn/fDtbPSCpZczE8+5dRYc81d+QfIT/52nLzi/Fh1
5afNzOwEb+UVvx5POPm09E4f/tr5J+IJp6+ITqdjteN6orng4ufHylfdkn+a/eXvjRe/4sq4+++/ZWZmFfbE7

31mrHrNbfmn2V/0znjDNW+pnZVWsORi/rGPfSxWP/OF6QfI2d/zzBgfH6/9rgLAkuX6X/5ArLzg0v4vVr7
mrli++knx+c9/vva72gqWXMwjIl526T+Pcy6+Mta8+b/OfXXkxs/FWc++LK58/Rtrv4sAsOT5vks2xJnf97qj
X9f0U78bay54Wbz9Z3+u9rvYGpZkzPfs2RNveevbY+UTnxorLhiJ0/7Zi2LVBZfFmWetjOuu/+Xa7x4AICK6
3W684U3XxMmnnh4rnvvKWP5Db4m1F14ap5x2Rnzgxptqv3utYknGvMdf//Vfx2233Rbvfve74zOf+Ux8
5Stfqf0uAQDm8Vd/9Vfx6U9/Ot75rnfH+Ph43HfffbXfpdaxpGMOAMAwMJQxHx8fn3m5/tjYWO13ZyjO
OcXRjIyM+HhrYLYjLy7tz9jYWIyMjNR+N1rJ6OjonG/Fmpqaqv0utYqhvM6z/4/udDoxOTlZ+T0abNatWzf
zQMJcpqam5sTJkVmYnqPJyUkfRwmdTkfM+zA6OuqWJwzdo2p8fDxGR0dn/npsbMyzpUXCEW5mZGT
EwWnAx9HC9G6VmC+Mx1bO0D2q5sd7ftzxneMIN+OZec74+LhYLcDU1NRMrPhZGF/O6s/k5OTwxXx0
dHTOpz3FfPEQ85zR0VFHpg9jY2M+hZywbt26mJqaEvMCpqamfPl0HuPj48MXc8/Mjx9i3p+RkREhL6D3
NXOfvXicsbGxOa8pEPNm5j9pW+oMZcwX+pq5/9MXBzFfGF/LOzb4mstC/9KMdevW1X63Wo2Yz2UoP
83e+xRMD88CFg8xPxqf+Wlmdrh7j0+PyYXxzLwZH0MLM5TXefb3mfvT2+Ih5kfT+1rw7In7XHqfWu/Ns/
L+iHl/fAzluM4AAAw4Yg4AwIAj5gAADDhiDgDAgCPmAAAMOGIOAMCAI+YAAAw4Yg4AwIAj5gAADDhi
DgDAgCPmAAAMOGIOAMCAI+YAAAw4Yg4AwIAj5gAADDhiDgDAgCPmAAAMOGIOAMCAI+YAAAw4Y
g4AwIAj5gAADDhiDgDAgCPmAAAMOGIOAMCAI+YAAAw4Yg4AwIAj5gAADDhiDgDAgCPmAAAMOGIO
AMCAI+YAAAw4Yg4AwIAj5gAADDhiDgDAgCPmAAAMOGIOAMCAI+YAAAw4Yg4AwIAj5gAADDhiDgD
AgCPmAAAMOGIOAMCAI+YAAAw4Yg4AwIAj5gAADDj/H6so1FCbPSUjAAAAAElFTkSuQmCC"></figure
> Similarly assume N=5 and K=2 x=3 and y=1 then We shoot the ball from the coordinates
(3,1), and we need to find its coordinates after it has collided with the sides twice. After shooting, it
first hits the right side at (5,3), and then the top side at (3,5). So, we report (3,5). <figure
class="image"><img
src="
R4nO29e5iddX2vvThDQsiBBEXp9hBLQSKUiuDZLaROVTzgsRK11JrXDh6JqWhFsIK7Ti53pWbL5rWFXmY
Tx60b6Qut2Qe3e9mJiYJ5cZtid0LHUtvXTJDDCjlOZibf94/JM3nWmrV+32eSteb3eZ7ffV/X5/LCZCZrbp78
1s1aa9bUrALUarXYN0Geer0e+yZIgx8fHIXBjw+OwuDHR91RJWqEqPJRvxBjgx8fHIXBjw+OwuDHR91RJ
WqEqPJRvxBjgx8fHIXBjw+OwuDHR91RJWqEqPJRvxBjgx8fHIXBjw+OwuDHR91RJWqEqPJRvxBjgx8fHIX
Bjw+OwuDHR91RJWqEqPJRvxBjgx8fHIXBjw+OwuDHR91RJWqEqPJRvxBjgx8fHIXBjw+OwuDHR91RJW
qEqPJRvxBjgx8fHIXBjw+OwuDHR91RJWqEqPJRvxBjgx8fHIXBjw+OwuDHR91RJWqEqPJRvxBjgx8fHIXBj
w+OwuDHR91RJWqEqPJRvxBjgx8fHIXBjw+OwuDHR91RJWqEqPJRvxBjgx8fHIXBjw+OwuDHR92RbI30
9/dbrVazWq1mQ0NDwd9LVPmoX4ixwY8PjsLgxwdHYfDjo+5Iskb6+/ttYGCg8O8nqnzUL8TY4McHR2H
w44OjMPjxUXckWSNLly6d0e8nqnzUL8TY4McHR2Hw44OjMPjxUXckVyNDQ0PW399vS5cutVqtZn19fe
7HEFU+6hdibPDjg6Mw+PHBURj8+Kg7kquRwcFBq9VqNjw8bGZmfX197lOBtVrN6vU6Y4wxxli0SUZVf3
9/0z97j1bxSJVPva5d97HBjw+OwuDHB0dh8OOj7kiuRoaGhpoiqjWy2kFU+ahfiLHBjw+OwuDHB0dh8O
Oj7kiyRvJvo9DX12eDg4Pu74cw6hdibPDjg6Mw+PHBURj8+Kg7kqyRoaGhqfeo8h6lMiOqiqB+IcYGPz44
CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+Puq
NK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGP
z44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyF
wY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5a
N+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfH
xyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1Ah
R5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44Co
MfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK
1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz4
4CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+Pu
qNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYG
Pz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY+PuqNK1AhR5aN+IcYGPz44CoMfHxyFwY
+PuqNK1AhR5aN+IcYGPz5DQ0Oxb4I0XEM+OAqDHx91R5WoEaLKR/1CjA1+fBqNRuybIA3XkA+OwuDH

R91RJWqEqPJRvxBjUxU/q9Zts+W3bOnJ515z7yO2ZOURT6vWbbNlqzc1/XP+10Msv2VLz25nLO6///7Y
N0Geqvw96xX48VF3VIkaIap81C/E2OT9ZHGQ3+bt5XiUJhRVy1ZvOqavK9Wo2ry9YUtW1m3NvY+Ymd
mKtVubHK5at83MZvZIXuu/h/y/iyUr67Zi7dbufyECcA6FwY+PuqNK1AhR5aN+Icbmpz/9qZlN3mHmQ8F
sMhbWb9wR42bNmHZRlcVOFgX5/38mX5cXVVVl+S1bpsLJzJqCJwuu/K97ZB/TKWizXy/LNTcTOIfC4Md
H3VElaoSo8lG/EGPz0EMPmZmV/s6sNarWb9zRtUfaUoyqzF+IFWu3zuiRpSyauvk5ywLnUBj8+Kg7qkSN
EFU+6hdibP7hH/7BzKztIzrtyD+V1i4ssgDJ1u7RjU5Pw+Ufqcj/Oe0eDcn/evaISj6qWh9lCdF6m1u/Li+qVq
zd2vRnt36d+V9btnrTtNvV+rRaa1Rkn7/1dvaSVeu2uXGz/JYttmLtVtu7d2+hz1kk1Ir8njLCORQGPz7qjipR
I0SVj/qFGJsf/vCHZnYkHEKv92kNr9aYyJ5uy4dSFhDZnWX+47M/M3uELB8j2efIPi7/KNqy1Zua7vCz35O/
LUUfeWt3m7PIab2d+Y8JRVXr15mPqNaoWn7LlmkRt2z1pqbPl92e/Ndc5LVZy2/Z0vY1TN6/5+xjQ5GdO
dm8vWG//OUvg5+r9WO817aV/VHTdnAOhcGPj7qjStQIUeWjfiHGptVPpzvddo9c5F8j470eJntUo5X80z
2tL4zOf2wWIq2Bk/882W32XrvT+vW2i4dlqzdN/f8ziSrvz85HVaenKFs/R7vXu/X6EZ1lqzdN+3eZ/Zmt18
euXbuO6s/Igrb1z8m7rwqcQ2Hw46PuqBI1QlT5qF+Isenkp/UpptCjHus37ugYO/nP1y648oHSKczyQdb6
qFD+87SGjfdoRyhM8n/mTB+pyly1u535qAp9x2Lrd921/r5uvmas058f+tzZn3+sr39qF9szeeq2LHAOhcG
Pj7qjStQIUeWjfiHGJuQnf4fX6ZGmjDX3PhJ88XY3o6rd7WgNlHavXWqlV1GVvw2t4TGbUXUsT/8VCbZu
PFrW7t8nUZUe+PFRd1SJGiGqfNQvxNg8+OCDHX8tf4fXKRwyvKe9Ot1Rtnv6LxRVnb7zrvU1RkXfO6pT7
B3t03+ttMZPu6f/Wmn39N9sP1LV7um/VroRVe3il6f/0gM/PuqOKlEjRJWP+oUYm3/5l3+xzdsb0+60W1
/f1Ol9ido97ZWnyAvVszAoElXZ78k/upEFVOvXkN2e1vDIv09Vpxeq56NpJlG1fuOOJketH1v0her5ry9GVL
W+UH3NvY9M+/fS7t9DKLxXrN0a/IaA/OflheppgR8fdUeVqBGiykf9QoxNFlWdXiuVp93va/09rW8PkL9jz
r/Qud3bAhSJqna3Y9W6bR3v0Nt9x1m7NzkNPTV2NC9U7/Qdbu0emWl9mq7dWy7MdlS1fmNCu3932
e3MXqjufcdgq+d2jzjylgppgh8fdUeVqBGiykf9QowNP7fNZ//+/bFvwqwzk7j513/9VzMr9josD+9p5rLCO
RQGPz7qjipRI0SVj/qFGBv8+KQankVfML5r1y5bv3HHMccQP6YmXfDjo+6oEjVCVPmoX4ixwY9Pqo46v
W9YKz//+c+78ud14y0aVEn1GioKfnzUHVWiRogqH/ULMTb48cFRGPz44CgMfnzUHVWiRogqH/ULMTb
48cFRGPz44CgMfnzUHVWiRogqH/ULMTb48cFRGPz44CgMfnzUHVWiRogqH/ULMTb48cFRGPz44CgM
fnzUHVWiRogqH/ULMTb48cFRGPz44CgMfnzUHVWiRogqH/ULMTb48cFRGPz44CgMfnzUHVWiRogqH
/ULMTb48cFRGPz44CgMfnzUHVWiRogqH/ULMTb48cFRGPz44CgMfnzUHVWiRogqH/ULMTb48cFRGP
z44CgMfnzUHVWiRogqH/ULMTb48cFRGPz44CgMfnzUHVWiRogqH/ULMTb48cFRGPz44CgMfnzUHZ
W6RsbHx+3++++3Wq1mP/nJT2LfHGnUL8TY4KczjUbDvve979lnP/tZ+8d//MfYN0cWrqHOjI6O2ubNm+
1Tn/qUbd1azZ9r2A24hnzUHZU2qv50zb+3448/3hadc56dMPcsW/SMpTZv/kJbt25d7JsmifqFGBv8TGdi
YsL+8NoP2UmnnGqLnnOxPe28l9u8RWfbsoteaBs3box98+TgGmrPTZ/7vNVqx9mCc863sy64whY8/dm
26Kyz7Vvf+lbsmyYH15CPuiPJqOrv77darTa14eHhpl+/+t2/Z2f/xott4VvvtCUr61NbcOWtNv+Z59mnP/3
pSLdcF/ULMTb4mc655y+zxb/5Jlt8zYamv2fzXnW9nTr3DLv77rtj30QpuIam87o3XGVn/sYrbNE71zddQ/
Nfu8YWnv1c+8IXvhD7JkrBNeSj7kg2qoaGhtr+2vr1623hM89t+gua35kr7rZT5s6373//+7N8q7VRvxBjg5
9mPvTR6+zpF7+h49+z+X1fsLPOPseeeuqp2DdVBq6hZm6//XY767kXd7yGFr3jLjvhxJNsy5YtsW+qDFxD
PuqOJKOqr6+vY1Rd+tJX2txLP9DxL+qSlXU79Tdea296+3tspDHKDu+eDfXot0F5+GneCSeeZGde/a3g37
N557/W/mzt/x39tqqMa6h5S8+70Oa98hPBa2juZf127Yc+Osv3MLqoB4MC6o5koyp76m9gYKDp106b
O8/OfM//E/yLuugd/8lOO+vXbfktW9jhXXZ9PfptUB5+ci4++HU7+cznBv+OLVlZt3mv/CN7xmXvjH57VcY
1dGRX/MmPrFar2ZI/+G7wGlr45tvtN1/0skj3NHqoB4MC6o4koypjeHjYarXa1KNWExMTdtzxx9vi9/33cF
S9/Wt20vxn2CWfqDPGZrjzrv6ynfy05xeKqkUX9EW/vUxvv/mR++z4k05zr6EFb7rNnvnsc61erzNWiUlHl
dnk66sGBwen/vncCy62BW/8D8G/qGe8+gZ74SuvtP+8aYQd3p/cWY9+G5SHnyNb9z9/bsefcKJ7h3j6sjf
bNR+7OfrtVRnXUPMWnf0cW/i2vwpfQy/7mL3rPb8f8R5Gi3pd+1EYBdQdlS6qbr75Zlty/r8N/kWdf84Fd
utX/iL6awqUxus98DOTvbrvDTb3kvcFHw0+7vjj7X/9YEv026oyrqHmrfzgKlt4we90Pqvf/z9t/tOew1sr5F
APBgXUHUlHVfb0X+tbKlz20pfbnIveNe0v6eJrvmPzLrjSrnj9W6MfKGrjwMfPTPbAT7fbqXPm2ryXr5r+O
pi33mFzzrnIXr38d2z9N+62h3/xaPTbqzCuoelbet6FNue3rpl2DZ35nr+2eee9xt6+4n2R7l00UQ8GBdQdS
UZV/j2q2n0X4K9+9St7yzuutiXPer6ddsFVdsqvv8bmPv8NdtrCs+3q37/Wdjx5IPphojYOfPzMdP/1ez+wF
/zWZbb4119scy662uZe1m+nn7vcjjv+eLui7/V2w403Te3eDd+Nfntjj2to+h782c/tVX1vspNOnWunLXur
nfrrr7G557/OTjl9kV23+hMR7l20UQ8GBdQdSUZVUb7//e/bS95xvZ3y7Ffa8mtusr/97lD0Q0R1HPj4Odr
d9c2/tk/88U12xevfal/+j3fYj37yf2z9N+5uiqobbrzJbl17m9V/8ED02xtrXEOdd/d9/80ufOMqW3zhlfaWa
z9vDzzwQOy7D0nUg0EBdUeljiozsw/8xc9s4Vv+0lbftT36waE8Dnz8dNPRP+/cZWu++KVpYXXDjTcl+5Qg

11B4b/vS/7ZLr6/bmnv/KfbdhizqwaCAuiOiKpFx4OOn2442/Xhr26hK9SlBrqHwiCof9WBQQN0RUZXIOP
Dx0wtHt669LRhWKT0lyDUUHlHlox4MCqg7IqoSGQc+fnrh6OFfPBqMqiysYt/2WH7YkRFVPurBoIC6I6Iq
kXHg46dXjtq9aD0fVP+8c1f02x7TD5scUeWjHgwKqDsiqhIZBz5+euWo3YvWP/rRj9m1H/yg/Y/6pui3O7
YfNjmiykc9GBRQd0RUJTIOfPz00lHri9Y/+rHr7NOfudFuuPEmXlPFbKRBVBVBPRgUUHdEVCUyDnz89Np
R9qL1W9feZj/c8lBTZKUQVlxD4RFVPurBoIC6I6IqkXHg46fXjh7+xaNNr6HasvXhpMKKayg8ospHPRgUU
HdEVCUyDnz8xHCUUlhxDYVHVPmoB4MC6o6IqkTGgY+fWI5SCSuuofCIKh/1YFBA3RFRlcg48PET01EKY
cU1FB5R5aMeDAqoOyKqEhkHPn5iO6p6WHENhUdU+agHgwLqjoiqRMaBjx8FR1UOK66h8IgqH/VgUE
DdEVGVyDjw8aPiqKphxTUUHlHlox4MCqg7IqoSGQc+fpQcVTGsuIbCI6p81INBAXVHRFUi48DHj5qjqoU
V11B4RJWPejAooO6IqEpkHPj4UXRUpbDiGgqPqPJRDwYF1B0RVYmMAx8/qo6qElZcQ+ERVT7qwaCAui
OiKpFx4ONH2VEVwoprKDyiykc9GBRQd0RUJTIOfPyoOyp7WHENhUdU+agHgwLqjoiqRMaBj58yOCpz
WHENhUdU+agHgwLqjoiqRMaBj5+yOCprWHENhUdU+agHgwLqjoiqRMaBj58yOSpjWHENhUdU+agH
gwLqjoiqRMaBj5+yOSpbWHENhUdU+agHgwLqjoiqRMaBj58yOipTWHENhUdU+agHgwLqjoiqRMaBj5
+yOipLWHENhUdU+agHgwLqjoiqRMaBj58yOypDWHENhUdU+agHgwLqjoiqRMaBj5+yO1IPq9h+1EdU
+agHgwLqjoiqRMaBj58qOFIOKwU/yiOqfNSDQQF1R0RVIuPAx09VHKmGlYof1RFVPurBoIC6I6IqkXHg4
6dKjhTDSsmP4ogqH/VgUEDdEVGVyDjw8VM1R2phpeZHbUSVj3owKKDuiKhKZBz4+KmiI6WwUvSjNKL
KRz0YFFB3RFQlMg58/FTVkUpYqfpRGVHlox4MCqg7IqoSGQc+fqrsSCGslP0ojKjyUQ8GBdQdEVWJjAMf
P1V3FDus1P3EHlHlox4MCqg7IqoSGQc+flJwFDOsyuAn5ogqH/VgUEDdEVGVyDjw8ZOKo1hhVRY/sUZU
+agHgwLqjoiqRMaBj5+UHMUIqzL5iTGiykc9GBRQd0RUJTIOfPyk5mi2w6psfmZ7RJWPejAooO6IqEpkH
Pj4SdHRbIZVGf3M5ogqH/VgUEDdEVGVyDjw8ZOqo9kKq7L6ma0RVT7qwaCAuiOiKpFx4OMnZUezEVZl
9jMbI6p81INBAXVHRFUi48DHT+qOeh1WZffT6xFVPurBoIC6I6IqkXHg4wdHvQ2rKvjp5YgqH/VgUEDdE
VGVyDjw8YOjyfUqrKrip1cjqnzUg0EBdUdEVSLjwMcPjo6sF2FVJT+9GFHlox4MCqg7IqoSGQc+fnDUvG6
HVdX8dHtElY96MCig7oioSmQc+PjB0fR1M6yq6KebI6p81INBAXVHRFUi48DHD47ar1thVVU/3RpR5aM
eDAqoOyKqEhkHPn5w1HndCKsq++nGiCof9WBQQN0RUZXIOPDxg6PwjjWsqu7nWEdU+agHgwLqjoiq
RMaBjx8c+TuWsErBz7GMqPJRDwYF1B0RVYmMAx8/OCq2ow2rVPwc7YgqH/VgUEDdEVGVyDjw8YOj4
juasErJz9GMqPJRDwYF1B0RVYmMAx8/OJrZZhpWqfmZ6YgqH/VgUEDdEVGVyDjw8YOjmW8mYZWin5
mMqPJRDwYF1B0RVYmMAx8/ODq6FQ2rVP0UHVHlox4MCqg7IqoSGQc+fnB09CsSVin7KTKiykc9GBR
Qd0RUJTIOfPzg6NjmhVXqfrwRVT7qwaCAuiOiKpFx4OMHR8e+UFjhJzyiykc9GBRQdyQdVQMDA9bX1xf
8PURVsXHg4wdH3VmnsMJPeESVj3owKKDuSDqqarUaUdWlceDjB0fdW7uwwk94RJWPejAooO5INqo
GBgZ4pKqL48DHD466u9awuv2Ou6LfJuURVT7qwaCAuiPJqBoeHra+vj4bGhoiqro07hDxg6PuLx9W161
aPeMfwpzSiCof9WBQQN2RZFQtXbrUhoeHZxRVV6+p2z0bGGNsdvdXX7/brlu1emq333FX9NukuCturN
ul19ft2i/XrV5nrJqTi6qBgQEbHBw0M+ORqi7ung08yoAfHPVqW7Y+bNetWj3jH8Kc0nikyqde134URgF1
R3JRVavVpm3p0qUdfz9RVWzcIeIHR73dX3397hn/EOaURlT5qAeDAuqO5KIqD49UdW/cIeIHR733M9M
fwpzSiCof9WBQQN0RUZXIuEPED45mxw9h1X5ElY96MCig7kg6qopAVBUbd4j4wdHs+SGspo+o8lEPBg
XUHRFViYw7RPzgaHb9EFbNI6p81INBAXVHRFUi4w4RPziafT+E1ZERVT7qwaCAuiOiKpFxh4gfHMXxQ1h
NjqjyUQ8GBdQdEVWJjDtE/OAonh/CiqgqgnowKKDuiKhKZNwh4gdHcf2kHlZElY96MCig7oioSmTcIeIHR
/H9pBxWRJWPejAooO6IqEpk3CHiB0caflINK6LKRz0YFFB3RFQlMu4Q8YMjHT8phhVR5aMeDAqoOyKq
Ehl3iPjBkZaf1MKKqPJRDwYF1B0RVYmMO0T84EjPT0phRVT5qAeDAuqOiKpExh0ifnCk6SeVsCKqfNSDQ
QF1R0RVIuMOET840vWTQlgRVT7qwaCAuiOiKpFxh4gfHGn7qXpYEVU+6sGggLojoiqRcYeIHxzp+6lyWB
FVPurBoIC6I6IqkXGHiB8clcNPVcOKqPJRDwYF1B0RVYmMO0T84Kg8fqoYVkSVj3owKKDuiKhKZNwh4g
dH5fJTtbAiqnzUg0EBdUdEVSLjDhE/OCqfnyqFFVHlox4MCqg7IqoSGXeI+MFROf1UJayIKh/1YFBA3RFRlc
i4Q8QPjsrrpwphRVT5qAeDAuqOiKpExh0ifnBUbj9lDyuiykc9GBRQd0RUJTLuEPGDo/L7KXNYEVU+6sGg
gLojoiqRcYeIHxxVw09Zw4qo8lEPBgXUHRFViYw7RPzgqDp+yhhWRJWPejAooO6IqEpk3CHiB0fV8lO2sC
KqfNSDQQF1R0RVIuMOET84qp6fMoUVUeWjHgwKqDsiqhIZd4j4wVE1/ZQlrIgqH/VgUEDdEVGVyLhDx
A+OquunDGFFVPmoB4MC6o6IqkTGHSJ+cFRtP+phRVT5qAeDAuqOiKpEFvvAVx9+cFQFP8phRVT5qAe
DAuqOiKpEpnDgKw8/OKqKH9WwIqp81INBAXVHRFUiUznwVYcfHFXJj2JYEVU+6sGggLojoiqRKR34isM
PjqrmRy2siCof9WBQQN0RUZXI1A58teEHR1X0oxRWRJWPejAooO6IqEpkige+0vCDo6r6UQkrospHPRg
UUHdEVCUy1QNfZfjBUZX9KIQVUeWjHgwKqDsiqhKZ8oGvMPzgqOp+YocVUeWjHgwKqDsiqhKZ+oEfe/
jBUQp+YoYVUeWjHgwKqDsiqhJZGQ58/GgPR9XwEyusiCof9WBQQN0RUZXIynLg40d3OKqOnxhhRVT5

qAeDAuqOiKpEVqYDHz+aw1G1/Mx2WBFVPurBoIC6I6IqkZXtwMeP3nBUPT+zGVZElY96MCig7oioSmRl
PPDxozUcVdPPbIUVUeWjHgwKqDsiqhJZWQ98/OgMR9X1MxthRVT5qAeDAuqOiKpEVuYDHz8aw1G1/
fQ6rIgqH/VgUEDdEVGVyMp+4OMn/nBUfT+9DCuiykc9GBRQd0RUJbIqHPj4wZHyquKnV2FFVPmoB4M
C6o6IqkRWlQMfPzhSXZX89CKsiCof9WBQQN0RUZXIqnTg4wdHiquan26HFVHlox4MCqg7IqoSWdUOfP
zgSG1V9NPNsCKqfNSDQQF1R0RVIqvigY8fHCmtqn66FVZElY96MCig7oioSmRVPfDxgyOVVdlPN8KKqPJ
RDwYF1B0RVYmsygc+fnCksKr7OdawIqp81INBAXVHRFUiq/qBjx8cxV4Kfo4lrIgqH/VgUEDdEVGVyFI48P
GDI/z0fkcbVkSVj3owKKDuiKhKZKkc+PjBEX56v6MJK6LKRz0YFFB3RFQlspQOfPzgCD+930zDiqjyUQ8GB
dQdEVWJLLUDHz84wk/vN5OwIqp81INBAXVHRFUiS/HAxw+O8NP7FQ0rospHPRgUUHdEVCWyVA98/
OAIP71fkbAiqnwefvjh2DdBHqKqxxBVxZbygY8fHOGn9/PCiqiCbnD//ffHvglBiKpElvqBjx8c4af3C4XVm7+
w2ZZd8xc9i6rlt2yx5bds6cnn7hZLVtZtzb2PmJnZ5u0NW7Kybus37mj65+zXQ6xat82WrNR+xCZViKpExo
GPHxzhZzbWGlaf+/yA/Zul59lxJ5xkJ535XDv+hBPtpa94lf3kJz+Z0VmfhUTrsijpdVS1+/NXrds2o8+RalTl/9
1kX2frZvr58h+bd7Zi7daobiSjqq+vz2q1mtVqNRscHAz+XqKq2Djw8YMj/MzWsrB64YtebHMWnm3zf2d
g6g5w8e//Vzv9pR+x448/wb72ta8Vvl9YtW5bT6Np/cYdbe+Mswho/chY7MMAACAASURBVLM3b2/Yir
VbZ/RnhKKqqqy59xFbtnrT1D938lyU7/3vHU0xm32+fFgtv2XLjP/ddAvJqMpCamhoyGq18E0kqoqNAx8
/OMLPbO5La2+30+YvscW/9522j0wsuPLP7fQzFtovf/nLQvcLsaJq2epNM35EqhMpRtWy1Zuagmf9xh1
NkTVT2r1QffktW9qGVgwkoyoPUdWdceDjB0f4mc2df9GlNveSP2gbVNlOe+Zv2qo/+mSh+wIvqlrjJwuY
Zas3NcVL9s/5p52yp4zy27y9Me1RFo9OT0223iaz6VGVhcDm7Y2p39/6NFf2ayvWbp3mIvv4Tk+p5T9/p
6fOuk32Z+WZqdNWNm2a/rHtwrc15mYL6agaHBy0vr6+4O8hqoqNAx8/OMLPbO6kk0+1xb/3t8GoWn
DlrXbZy68odH9wNFHVLlLyTwt5j26sWLu10NNIWTx4T0vNJKpaw2nNvY8Ef6014rLXXWUfk4+u1o/LO2
ol+z2d5n1s67+zdq9Nmwm7d+9u+/laKfrvrttIRtXAwIDVajU3qMyORNXVa+p2zwbGGGOxd8dd/8VOmb
sgeGe8ZGXdFr71DnvaM59l9Xq94/7u7/7OzNrfGecf8ciianx83MysKXKeeOKJqd+Tj5zR0VF76KGHbO/ev
U1RtW/fPjNrflrpgQcemHbbfvSjH5lZc+Ts2rXLGo3G1G3OR0UWVQcPHmyKqtHR0WlR1fqU1ujoqA0PD
0/78/Jfu5nZ8PCw7dmzZ9rnyD7/+o07bN++fbZ///4mJ7t375729f30pz9174PNJt/moPVjs68/C5sf/ehHU
17zZI8Sbt7esEcffTR4LdTrdTtw4EBT6OXDaefOnVav1210dLTJ/YMPPuh+3m5NMqoystdUZRdSO3ikqtju
2cB/ReMHR/iZvZ35tGfYond+PRhV8155vV319ncVuj842qf/Wj9Hp0dH2j1SVfQFz52easru/NvdJu+Rqnw
4tD4alI+q7PO0e8Qo//vaPb2Y3fZuvWas3Z9f5HPP5DZs29b8+zJPrR9/rE8zHi3SUWU2+Z2AQ0NDHX+dq
Co2Dnz84Ag/s7k3v+1dNueid4UfqXrOJXbbbbcVui/oRlRlZI+OeN+VtmrdtkJ3zL2IqvxtaA3B2YyqY3n6r2h
UtT4qF6LdC9XbBRRRdZh8QA0PD/NIVZfGgY8fHOFntvb3D//CvvyV2+20uWfYGa/+dNs74zm/dom97PLf
KXzf0M2oavd72kVV0feO6vT6nU5P/+U/txdV7b6+dk//tbuN7Z7+m81HqvJP/4WYyQvm2z2F2C6gev3d
op2QjKrsPapqtVrwUSozoqroOPDxgyP89Hr/vHOXfeuv/9Zu/vyf2g033mTvfu/v2eKn/5otXPY6m/eK1Tb
vFR+301/8QVv0nIvtt1/3Jtuxo/jbCXQjqvIf3+nNN1u/Y6/T00v596kKvVA9//lmElWtX2v+Y2fyQvXW2zOb
UdXuhert/nkm72O1at22pq+hnXszXqh+1BBVxcaBjx8c4aeX27L1YVvzxS81vZv6DTfeZJt+vNU+P/Bnds4L
32BznnGBXdL3nhm96WdGN6Kq9e0UWn89/9YKrfHR7tG2fMS0e6fwdp9jJlGV/1z5r63dWyq0Pk3X+sh
NjKhq95YKrW9f0e67A0Mx1O7pyHaPcvGWCkcJUVVsqR/4+MERfnqzv3/4F3bn19ZPi6kbbrzJ7vza+qnfx
w9UTpOZxs3yW7Yccwx5T6X2EqIqkaV64OMHR/jpzVqf6mvdzZ//U/v7h38x9fuJKp9/+qd/in0Tus7RvIFq
iHZv/tkKP6bmGCCqii21Ax8/OMJP79bpqb787t3w3aaPIap82n1nWxXo5g+79hzxA5WPEaKq2FI68PGDI
/z0bpt+vDUYUzfceJOt+eKX7J937mr6OKLKp6pR1U3UHRFViSyVAx8/OMJP73fvhu8Go2rTj7dO+xiiykc9
GBRQd0RUJbKUDnz84Ag/vd+Xv3K7fehDH7brP/mpji9Oz4+o8lEPBgXUHRFViSy1Ax8/OMJP75Y9UnXd
qo/bx1f/UccXp+dHVPmoB4MC6o6IqkSW0oGPHxzhp3fLP/V369rb7Jvfvq/ji9PzI6p81INBAXVHRFUiS+X
Axw+O8NO7tQbVw794dOr/b/fi9PyIKh/1YFBA3RFRlchSOPDxgyP89G6dgipbp6f9shFVPurBoIC6I6IqkV
X9wMcPjmKvyn68oCoyospHPRgUUHdEVCWyKh/4+MGRwqrqpxtBNdIgqoqgHgwKqDsiqhJZVQ98/OBI
ZVX0062gGmkQVUVQDwYF1B0RVYmsigc+fnCktKr56WZQjTSIqiKoB4MC6o6IqkRWtQMfPzhSW5X8dD
uoRhpEVRHUg0EBdUdEVSKr0oGPHxwprip+ehFUIw2iqgjqwaCAuiOiKpFV5cDHD45UVwU/vQqqkQZRV
QT1YFBA3RFRlciqcODjB0fKK7ufXgbVSIOoKoJ6MCig7oioSmRlP/DxE384qq6fXgfVSIOoKoJ6MCig7oioS
mRlPvDxozEcVdPPbATVSIOoKoJ6MCig7oioSmRlPfDxozMcVc/PbAXVSIOoKoJ6MCig7oioSmRlPPDxozU
cVcvPbAbVSIOoKoJ6MCig7oioSmRlO/DxozccVcfPbAfVSIOoKoJ6MCig7oioSmRlOvDxozkcVcNPjKAaaR
BVRVAPBgXUHRFViawsBz5+dIej8vuJFVQjDaKqCOrBoIC6I6IqkZXhwMeP9nBUbj8xg2qkQVQVQT0YFFB
3RFQlMvUDP/bwg6Mq+4kdVCMNoqoI6sGggLojoiqRKR/4CsMPjqrqRyGoRhpEVRHUg0EBdUdEVSJTPf

BVhh8cVdGPSlCNNIiqIqgHgwLqjoiqRKZ44CsNPziqmh+loBppEFVFUA8GBdQdEVWJTO3AVxt+cFQlP2p
BNdIgqoqgHgwKqDsiqhKZ0oGvOPzgqCp+FINqpEFUFUE9GBRQd0RUJTKVA191+MFRFfyoBtVIg6gqgno
wKKDuiKhKZAoHvvLwg6Oy+1EOqpEGUVUE9WBQQN0RUZXIYh/46sMPjsrsRz2oRhpEVRHUg0EBdUdE
VSLjDhE/OKqmnzIE1UiDqCqCejAooO6IqEpk3CHiB0fV81OWoBppEFVFUA8GBdQdEVWJjDtE/OCoWn7
KFFQjDaKqCOrBoIC6I6IqkXGHiB8cVcdP2YJqpEFUFUE9GBRQd0RUJTLuEPGDo2r4KWNQjTSIqiKoB4MC
6o6IqkTGHSJ+cFR+P2UNqpEGUVUE9WBQQN0RUZXIuEPED47K7afMQTXSIKqKoB4MCqg7IqoSGXeI+
MFRef2UPahGGkRVEdSDQQF1R0RVIuMOET84KqefKgTVSIOoKoJ6MCig7oioSmTcIeIHR+XzU5WgGmk
QVUVQDwYF1B0RVYmMO0T84KhcfqoUVCMNoqoI6sGggLojoiqRcYeIHxyVx0/VgmqkQVQVQT0YFFB3
RFQlMu4Q8YOjcvipYlCNNIiqIqgHgwLqjoiqRMYdIn5wpO+nqkE10iCqiqAeDAqoOyKqEhl3iPjBkbafKgfV
SIOoKoJ6MCig7oioSmTcIeIHR7p+qh5UIw2iqgjqwaCAuiOiKpFxh4gfHGn6SSGoRhpEVRHUg0EBdUdEVS
LjDhE/ONLzk0pQjTSIqiKoB4MC6o6IqkTGHSJ+cKTlJ6WgGmkQVUVQDwYF1B0RVYmMO0T84EjHT2pB
NdIgqoqgHgwKqDsiqhIZd4j4wZGGnxSDaqRBVBVBPRgUUHdEVCUy7hDxg6P4flINqpEGUVUE9WBQQN
0RUZXIuEPED47i+kk5qEYaRFUR1INBAXVHRFUi4w4RPziK5yf1oBppEFVFUA8GBdQdEVWJjDtE/OAojh+
CanJElY96MCig7oioSmTcIeIHR7Pvh6A6MqLKRz0YFFB3RFQlMu4Q8YOj2fVDUDWPqPJRDwYF1B1JRlVf
X5/VajWr1Wo2MDAQ/L1EVbFxh4gfHM2eH4Jq+ogqH/VgUEDdkVxUDQ8P2+Dg4NQ/12o1Gx4e7vj7ia
pi4w4RPziaHT8EVfsRVT7qwaCAuiO5qGqlr6/PhoaGOv46UVVs3CHiB0e990NQdR5R5aMeDAqoO5KPK
h6p6s64Q8QPjnq7W7/yVYIqMKLKRz0YFFB3JB1V/f39hV9TdfWaut2zgTHGZn+3fuWrdt2q1XbdqtV2w0
2fs8Fvfyf6bVLbFTfW7dLr63btl+tWrzNWzclGVV9fnxtUZjxSVXT3bOBRBvzgqBfLnvK7btVqHqEKjEeqfOp
17UdhFFB3JBlV3uuo8hBVxcYdIn5w1P3lX0N1w02fI6gCI6p81INBAXVHclE1ODho/f39hX8/UVVs3CHiB0
fdXeuL0ge//Z3ot0l5RJWPejAooO5ILqoGBgam3qMqWyiyiKpi4w4RPzjq3tp9lx9+wiOqfNSDQQF1R3JR
NVOIqmLjwMcPjrqzTm+bgJ/wiCof9WBQQN0RUZXIOPDxg6NjX+h9qPATHlHlox4MCqg7IqoSGQc+fnB0
bPPe2DN1P96IKh/1YFBA3RFRlcg48PGDo6NfkXdKT9lPkRFVPurBoIC6I6IqkXHg4wdHR7eiP3omVT9FR1
T5qAeDAuqOiKpExoGPHxzNfDP5WX4p+pnJiCof9WBQQN0RUZXIOPDxg6OZbaY/HDk1PzMdUeWjHgw
KqDsiqhIZBz5+cFR8Mw2q1PwczYgqH/VgUEDdEVGVyDjw8YOjYjuaoErJz9GOqPJRDwYF1B0RVYmMAx
8/OPJ3tEGVip9jGVHlox4MCqg7IqoSGQc+fnAU3rEEVQp+jnVElY96MCig7oioSmQc+PjBUecda1BV3U83
RlT5qAeDAuqOiKpExoGPHxy1XzeCqsp+ujWiykc9GBRQd0RUJTIOfPzgaPq6FVRV9dPNEVU+6sGggLojoi
qRceDjB0fN62ZQVdFPt0dU+agHgwLqjoiqRMaBjx8cHVm3g6pqfnoxospHPRgUUHdEVCUyDnz84Ghyv
QiqKvnp1YgqH/VgUEDdEVGVyDjw8YOj3gVVVfz0ckSVj3owKKDuiKhKZBz4+EndUS+Dqgp+ej2iykc9GBR
Qd0RUJTIOfPyk7KjXQVV2P7MxospHPRgUUHdEVCUyDnz8pOpoNoKqzH5ma0SVj3owKKDuiKhKZBz4+
EnR0WwFVVn9zOaIKh/1YFBA3RFRlcg48PGTmqPZDKoy+pntEVU+6sGggLojoiqRceDjJyVHsx1UZfMTY0
SVj3owKKDuiKhKZBz4+EnFUYygKpOfWCOqfNSDQQF1R0RVIuPAx08KjmIFVVn8xBxR5aMeDAqoOyKqE
hkHPn6q7ihmUJXBT+wRVT7qwaCAuiOiKpFx4OOnyo5iB5W6H4URVT7qwaCAuiOiKpFx4OOnqo4Ugkr
Zj8qIKh/1YFBA3RFRlcg48PFTRUcqQaXqR2lElY96MCig7oioSmQc+PipmiOloFL0ozaiykc9GBRQd0RUJTI
OfPxUyZFaUKn5URxR5aMeDAqoOyKqEhkHPn6q4kgxqJT8qI6o8lEPBgXUHRFViYwDHz9VcKQaVCp+lEd
U+agHgwLqjoiqRMaBj5+yO1IOKgU/6iOqfNSDQQF1R0RVIuPAx0+ZHakHVWw/ZRhR5aMeDAqoOyKqE
hkHPn7K6qgMQRXTT1lGVPmoB4MC6o6IqkTGgY+fMjoqS1DF8lOmEVU+6sGggLojoiqRceDjp2yOyhRU
MfyUbUSVj3owKKDuiKhKZBz4+CmTo7IF1Wz7KeOIKh/1YFBA3RFRlcg48PFTFkdlDKrZ9FPWEVU+6sGgg
LojoiqRceDjpwyOyhpUs+WnzCOqfNSDQQF1R0RVIuPAx4+6ozIH1Wz4KfuIKh/1YFBA3RFRlcg48PGj7Kjs
QdVrP1UYUeWjHgwKqDsiqhIZBz5+VB1VIah66acqI6p81INBAXVHRFUi48DHj6KjqgRVr/xUaUSVj3owKK
DuiKhKZBz4+FFzVKWg6oWfqo2o8lEPBgXUHRFViYwDHz9KjqoWVN32U8URVT7qwaCAuiOiKpFx4ONH
xVEVg6qbfqo6ospHPRgUUHdEVCUyDnz8KDiqalB1y0+VR1T5qAeDAuqOiKpExoGPn9iOqhxU3fBT9RFV
PurBoIC6I6IqkXHg4yemo6oH1bH6SWFElY96MCig7oioSmQc+PiJ5SiFoDoWP6mMqPJRDwYF1B0RVYm
MAx8/MRylElRH6yelEVU+6sGggLojoiqRceDjZ7YdpRRUR+MntRFVPurBoIC6I6IqkXHg42c2HaUWVDP1k
+KIKh/1YFBA3RFRlcg48PEzW45SDKqZ+El1RJWPejAooO6IqEpkHPj4mQ1HqQZVUT8pj6jyUQ8GBdQdE
VWJjAMfP712lHJQFfGT+ogqH/VgUEDdEVGVyDjw8dNLR6kHleeHEVVFUA8GBdQdEVWJjAMfP71yRFC
F/bDJEVU+6sGggLojoiqRceDjpxeOCKqwH3ZkRJWPejAooO6IqEpkHPj46bYjgirshzWPqPJRDwYF1B0RVY
mMAx8/R7NvfPtv7GWXv9YWLTnbTjjxZDv/oktt9Sc/Q1C1GddQeESVj3owKKDuSDaqBgcHrVbzbx5RV
Wwc+PiZ6f7gDz9qpy98mp32grfbot8dtMXXfMcWXHmrLTj3lfb0c55tH/7IRwmq3LiGwiOqfNSDQQF1R

5JRtXTpUuvv7yequjgOfPzMZLf95ddswTPOtcXvvc+WrKxP29wXvs/Of8HFBFVuXEPhEVU+6sGggLojyajK
IKq6Nw58/MxkC848y+a/7s/aBtVUWJ1zkd151zej31aVcQ2FR1T5qAeDAuqOiKpExoGPn6J74Kfbbc6Cs4
JBtWRl3eZe+gH74HWfjH57VcY1FB5R5aMeDAqoO6pMVF29pm73bGCMHes+c/OAnTx3oRtVZ/z25+zC
F708+u1l5dgVN9bt0uvrdu2X61avM1bNVSaqeKQqvHs28F/R+Cm2//ehYZszf7EbVXMufrd9+ON/HP32
qoxrKDweqfKp17UfhVFA3RFRlcg48PEzk5119jk2v+8Lwaiaf84F9rXBu6PfVpVxDYVHVPmoB4MC6o6Iqk
TGgY+fmezO//Sfbd6SZ9mZV3+zbVAteNF77IrXvzX67VQa11B4RJWPejAooO6IqEpkHPj4melWf/Izdspp
8+zUc19rC9/yl3bm1d+y+a/5vC183kvsRS+73LZu/0X026g0rqHwiCof9WBQQN2RdFQVgagqNg58/BzN7
vtvdXvz295lZz/rXDtt3kK77BXL7ZM3/rvot0txXEPhEVU+6sGggLojoiqRceDjB0f4iTmiykc9GBRQd0RUJTI
OfPzgCD8xR1T5qAeDAuqOiKpExoGPHxzhJ+aIKh/1YFBA3RFRlcg48PGDI/zEHFHlox4MCqg7IqoSGQd+O
f3sbIzaU/vHbXRswsYmDtnYxCE7OH6o9I527RuzA2MT9uTeMXtiz5jtOTBuB8cP2dj4Idt/cMKe2HPwqD7
vU/vGbXzikI2OHbKduyb/v937x6f+rLL4qeKIKh/1YFBA3RFRlcg48Mvn57HdB23i0KGO1/4Te8Zs174x23t
gvCt/XmPvmO0bnbCdPXY0Onbkawp8ebZ7Bl/Xk3vHbHRsounjDxycsEd3HWz6/3711NHFWlmvIaURVT
7qwaCAuiOiKpFx4JfPTxYcq9Ztm/bmm5u3N2xs/EiRHBw/Zl8dQzD86qmDU3/e2MQhe2LP9Ed1uuFo17
4xMzNbfssWW37LFjMz27y9Me3ry3h8t/817RttjqklK+u2fuMOOzA2YXsOjNuKtVunPue+0YmkriGlEVU
+6sGggLojoiqRceCXy88TeybjY8XarbZs9aama37Vum22fuMOO3TIbP3GHVPBMDp2JBgaeycfwTpwcMJ
27RubeipspDFqO3eN2p7947Zz1+Gn4Q5O2MHxQ1Nxs3l7o+1TjN1wND5xyNbc+8jU1zQ2fshWrdtmm
7c3pr6+fAQ9tT/8aNVjuycfiWoNsyyq8hG3Yu1WM7O2wVjFa0htRJWPejAooO6IqEpkHPjl8tPYOxkDWS
AcMrPGvsOvPRqbfN2RWXNUmZkdGJuw8Yn2z6ntH52wnbtG7eBY+1/PR5VZc6TlHe08fPs6rdPXtHv/uJ
mZLVu9ydbc+4iZTT4d96unDtru/eO2a9+YTRxqjjvvqc0sqrJoyj42i6qRxuTX2yk+q3wNqY2o8lEPBgXUHR
FViYwDv1x+GvuORFUWIGZmu/ePT4VE9ohO69OCZpORkf//s88xOjYxFRjZ/y5ZWW/7FOP6jTuani67Z0
PddjZGp712qZW9o+1DaP/BianoyW7LSGPyxeVjE4emnn7Mbtfm7Y1Cjyo9tW8y1vJBlo+qpw67zGJu4p
AlcQ2pjajyUQ8GBdQdEVWJjAO/XH6y1zitufcRW7KyPvX6IzOz8YlDU68jan2kanxiMiyyp7ryv2fz9oaNTR
x51Kb1acXWR6oOjh+yR3cdeU3TPRvqNjo2MXWbOm3z9obtbvO03dj45FN/2dfS2DtmTx5+mjPPstWb
bMXarXboUPHvctzZGLW9o+Nto+rRXaNmNhmhmZf811XVa0htRJWPejAooO6IqEpkHPjl85O9HsjMpoIl
H1cTh5qf1speB7XnwLi1PgOYPUqTfx1W9rqsvQcmHynKR9XYxKFp4XHPhvrUa708HmvzAvNDNvl6sCxs
Hnvq4NTTnPlH1tZv3DH1eWby+qdOUTXSGLWJicnXbmX+evG6KsVrSGlElY96MCig7oioSmQc+OX088Se
MTuY+y6//HfHHbLmR6p27x+3PQcmnwpr9x11a+59pOljNm9vTD0FdyD31Nzm7Y223yWXOcpez9WJf
Qenf+yvnjrylOWqddvM7MhTcK1PJ2aBNXn7ij9aFYqq7BG27NG5p/Z1520oynANqYyo8lEPBgXUHRFViY
wDv5x+srdJ2LVvbOoF6PnvZGuNqvGJQ1Ovtcp/R93UI1XWHFVZPBWNqj0Hxgs9/df6gvWduafgsqjKPxL
2xJ7JpwKz97BafssWW7Vumx06/PqnIo8shaIqe+oxi6rQC+qrdg2pjKjyUQ8GBdQdEVWJjAO/XH4e331w
KqIOHZp8am//4ddR5V8b1Pr0n1nzd9dlL/7u9EhVp6gaHWv/9F/2SJhHu2gZP/wUXP6tDfYe/nz7D05Mv
Ru62ZGoGp848p2O+9s8ApZfKKrMrOnpv3ZPT1btGlIbUeWjHgwKqDsiqhIZB365/Ow9MBkI+ddQmR15
Wi8LpHxEZGQv9M7IHrkKRVX2dget3224a9+ROMoceWHVKX6yp+Cyr+nAwSORmCf7TsTN2xtTL7wv8h
YLnaLqsdxTj9mflcI1pDaiykc9GBRQd0RUJTIO/HL5yaKq3dNrWUDloyT/1Fvrx2VPe4WiKnubhvxbK7R7S
4VufU1m1vSoVKd3VJ84/B2Q+e8Y7Pj5O0TV3tHm98ca69HPTlS7htRGVPmoB4MC6o6IqkTGgV8uP4/lf
mxMO7LYyb+IPSP0cU/tG5t688/WN8Fs9/5TjTaPVB3tHj8cbvmnJ9vdfrPJ4MoePcu/DqvTzyXMdiD3IvrH
dx+c+jPzIem9S3tVriG1EVU+6sGggLojoiqRceCXz0/2zuV7R8dt/8EJ2zc6+bPsWn/G3+O7D9rew7/25OFH
cp7cO2b7D07Y7v3j9vieg/bU/vGp947KfkxNuz/zsd0Hbe+Bcdt7YHzao0LdcLR3dLzpBeMThw7Z47snb9+
BgxO298D41FOOu3Jv2rl5e8N9TVW2PfuP3Pb8C99XrN3a9sfvVPkaUhpR5aMeDAqoOyKqEhkHPn4UH
D26a9QmDh35gcqhUHps90HLP451NN+xd2BsoulnCT7Zo5/7xzXkj6jyUQ8GBdQdEVWJjAMfPyqOnto3
PvUUpfd+UU8e/sHQTx7lWyBkT/+Z+T+cWcVPVUdU+agHgwLqjoiqRMaBjx81Rzt3zc7t3rlrdv4srqHwiC
of9WBQQN0RUZXIOPDxgyP8xBxR5aMeDAqoOyKqEhkHPn5whJ+YI6p81INBAXVHRFUi48DHD47wE3N
ElY96MCig7oioSmQc+PjBEX5ijqjyUQ8GBdQdEVWJjAMfPzjCT8wRVT7qwaCAuiOiKpFx4OMHR/iJOaLK
Rz0YFFB3RFQlMg58/OAIPzFHVPmoB4MC6o6IqkTGgY8fHOEn5ogqH/VgUEDdEVGVyDjw8YMj/MQcUe
WjHgwKqDsiqhIZBz5+cISfmCOqfNSDQQF1R0RVIuPAxw+O8BNzRJWPejAooO6IqEpkHPj4wRF+Yo6o8l
EPBgXUHRFViYwDHz84wk/MEVU+6sGggLojoiqRceDjB0f4iTmiykc9GBRQd0RUJTIOfPzgCD8xR1T5qAe
DAuqOiKpExoGPHxzhJ+aIKh/1YFBA3RFRlcg48PGDI/zEHFHlox4MCqg7IqoSGQc+fnCEn5gjqnzUg0EBdU
dEVSLjwMcPjvATc0SVj3owKKDuiKhKZBz4+MERfmKOqPJRDwYF1B0RVYmMAx8/OMJPzBFVPurBoIC6I

6IqkXHg4wdH+Ik5ospHPRgUUHdEVCUyDnz84Ag/MUdU+agHgwLqjoiqRMaBjx8c4SfmiCof9WBQQN0
RUZXIOPDxgyP8xBxR5aMeDAqoOyKqEhkHPn5whJ+YI6p81INBAXVHRFUi48DHD47wE3NElY96MCig7o
ioSmQc+PjBEX5ijqjyUQ8GBdQdEVWJjAMfPzjCT8wRVT7qwaCAuiOiKpFx4OMHR/iJOaLKRz0YFFB3RFQl
Mg58/OAIPzFHVPmoB4MC6o6IqkTGgY8fHOEn5ogqH/VgUEDdEVGVyDjw8YMj/MQcUeWjHgwKqDsiq
hIZBz5+cISfmCOqfNSDQQF1ruuTTwAABopJREFUR0RVIuPAxw+O8BNzRJWPejAooO6IqEpkHPj4wRF+Y
o6o8lEPBgXUHRFViYwDHz84wk/MEVU+6sGggLojoiqRceDjB0f4iTmiykc9GBRQd0RUJTIOfPzgCD8xR1T
5qAeDAuqOSh1Vjz/+uL1+1R0256J32e/esN5+/ssnoh8cquPAxw+O8BNrDw3/f/aKD/xHe9byj9i1A//F9u
/fH/vuQxL1YFBA3VFpo+rTN/6J1Wo1m3POxXbioqU295nL7MSTTrYvfPHPox8giuPAxw+O8BNj/R/9hJ0
6Z57N+bUX2pznv9FOf8bzbc7pZ9idd94Z+25EDvVgUEDdUSmj6o1Xvd2WnP9qW/S737AlK+tTW3jVV23
+sy62/g99LPpBojYOfPzgCD+zvVdc3mdLlvXZme/+dtNZveCNa+2MZ15gn/3sZ2PfnUihHgwKqDsqXVTdc
ccdtvjZFzb9Bc1v8Xv/xk6bf5Z94+6/iX6gKI0DHz84ws9s7uY//fe2+HmXdjyrz7z6m3byaafbD37wg9h3Kz
KoB4MC6o5KF1WXXPZyO/0lH+74F3XJyrrNe9X1dvmV77QfPtxgh3f7t+rRb4Py8IMj/HR3z3zu8+2Myz8
TPKtPf8mH7AP9H4p9tyKDejAooO6odFF1ymlzbfF77wv+RV30zq/bqYufa6/6kwfY4V3yR/Xot0F5+MERf
rq3V3zmB3bcccfbkvd/L3hWL3zLX9iFv/Xi2HcrMqgHgwLqjkoXVSeceJItvmZD+C/q2/7KTph3ti286quM
McZmeQuu/HM77oSTg+f05Gur/oMdd8o8q9VqjFVjsSNpppx34SW24A1fDj/9928/ZS9/zVW28f88yRhjL
MKe9mvPs4VvuSP89N9LP2Lvvub9se9WALpG6aLqi1/8oi163kuCf1HPePrzbHBwMPZNBQBIluv/+DO24
LzLO39T0TUbbO6iZ9h9990X+6YCdI3SRZWZ2asu/22bf+FVtvh9/6P5u0lW3G2nn3uFXfXO98S+iQAAyfO
Ciy+1017wjumve/3dr9vi815lf/jBD8e+iQBdpZRRdeDAAXv///WHtuBpz7J55/XZyf/mpbbwvCvstNMX2O
rr/zj2zQMAADNrNBr27vdeYyecdIrNO/+1NvdF77clz7/cTjz5VPvMTTfHvnkAXaeUUZXx4IMP2h133GEf/
/jH7a677rKf/exnsW8SAAC08OMf/9i++tWv2kc+9nEbHBy0bdu2xb5JAD2h1FEFAAAAoIJkVA0ODk59e+
LAwEDsm1MJMqcwnb6+Pq43h7wjvgmkMwMDA9bX1xf7ZkjS39/f9K3nw8PDsW+SHHlHQ0NDsW+OF
AMDA23fwkANvVtk1vQXjovr2Fm6dOnUX1ZoZnh4uCkSOOzbkzkaGhriOgpQq9WIqg709/dzlgfo7+/nP
+pmwMDAgKQvudNxcHDQ+vv7p/5ZVVwZ4c7Qp6+vj4PfgeuoPdlZRVS1h79bYZYuXRr7JpQK1XNI7la1R
lRrZMHRo3oRKsEjVWEGBweJhjYMDw9PRQN+2sPT7J0ZGhqy/v5+W7p0KY92FmBwcFD2GpK7l+3v729
6Ooao6h5EVRgefu9M9noGDvv2LF261IaHh4mqAgwPD/Oyjhay17xm/0HX19fHWRQg+/umiNy9LI9U9
Q6iqjMcYsXIXlOleqDFYGBgoOk1Z0SVT+t/PKdO6/0cjwh3Rv3vmNy9bLvXVPGXrzsQVe3htR4zA1/NtPu
OJF4fE4aoaqY1FHgwoTPqTSB3L5s9NJzBfxV3D6JqOhxePvmAyv5+8neyPer/Fa0A11B78k+J9vX1SYdDTJ
Sf+jMTjCqz5vep4sLqHkTVdNq99wmR1Uz2lB/vn+NDVHWGayhM/u8ZZ1Bn1O/HtG8dAAAAQEkgqgAA
AAC6AFEFAAAA0AWIKgAAAIAuQFQBAAAAdAGiCgAAAKALEFUAAAAAXYCoAgAAAOgCRBUAAABAFyC
qAAAAALoAUQUAAADQBYgqAAAAgC5AVAEAAAB0AaIKAAAAoAsQVQAAAABdgKgCAAAA6AJEFQAAA
EAXIKoAAAAAugBRBQAAANAFiCoAAACALkBUAQAAAHQBogoAAACgCxBVAAAAAF2AqAIAAADoAkQV
AAAAQBcgqgAAAAC6AFEFAAAA0AWIKgAAAIAuQFQBAAAAdAGiCgAAAKALEFUAAAAAXYCoAgAAAOg
CRBUAAABAFyCqAAAAALrA/w9jx0zog0cEPQAAAABJRU5ErkJggg=="></figure>

answer

```
 #include <iostream>

using namespace std;


int main() {

    int n,k,x,y;

    try{

        cin>>n>>k>>x>>y;
```

```
    if(cin){
            k%=4;

            if(x==y) cout<<n<<" "<<n<<"\n";
            else{
              if(k==1){
                if(x>y) cout<<n<<" "<<y+(n-x)<<"\n";
                else cout<<x+(n-y)<<" "<<n<<"\n";
              }
              else if(k==0){
                if(x>y) cout<<(x-y)<<" 0\n";
                else cout<<"0 "<<(y-x)<<"\n";
              }
              else if(k==2){
                if(x>y) cout<<y+n-x<<" "<<n<<"\n";
                else cout<<n<<" "<<x+n-y<<"\n";
              }
              else{
                if(x>y) cout<<"0 "<<x-y<<"\n";
                else cout<<y-x<<" 0"<<"\n";
              }
            }
  }
  else throw 0;

        }
        catch(int n){
          cout<<"Invalid coordinate sites";
        }
        return 0;
}
```

question

Problem Description Vikram has just started Programming, he is in first year of Engineering. Vikram is reading about Relational Operators.
Relational Operators are operators which check relationship between two values. Given two numerical values A and B you need to help Vikram in finding the relationship between them that is,
First one is greater than second or, First one is less than second or, First and second one are equal. Constraints: 1 ≤ number1, number2 ≤ 50000 Input Format: First line contains an integer T, which denotes the number of testcases. Each of the T lines contain two integers A and B separated by a space. Output Format: For each line of input produce one line of output. This line contains any one of the relational operators '<', '>' , '=' .

answer

```
#include <iostream>
using namespace std;
int main()
{
   int number1,number2;
   try{
      cin>>number1>>number2;
      if(cin){
         if(number2>number1)
         cout<<"<";
         else if(number1==number2)
         cout<<"=";
         else
         cout<<">";
      }
      else throw 0;
   }
   catch(int a){
      cout<<"Input data missing";
```

```
    }
            return 0;
}
```

question

Problem Description:
Rohit has 'A' Chocolates and Mohit has 'B' Chocolates. Rohit will do the following action 'K' times.

If Rohit has one or more Chocolates, eat one of his Chocolates.
Otherwise, if Mohit has one or more Chocolates, eat one of Mohit's Chocolates.
If they both have no Chocolates, do nothing.

Constraints:
0 <= A <= 10^12
0 <= B <= 10^12
0 <= K <= 10^12
All values in input are integers.

Input Format:
Only line of input has three integers A B K separated by as space

Output Format:
Print the numbers of Chocolates Rohit and Mohit have respectively after K actions

answer

```
 #include <iostream>
using namespace std;
int main()
{
    int a,b,k;
    try{
        cin>>a>>b>>k;
        if(cin){
            if(k<=a)
            cout<<a-k<<" "<<b;
            else if(k>=a+b)
            cout<<"0 0";
            else
            cout<<a-a<<" "<<a+b-k;
        }
        else
        throw 0;
```

```
    }
    catch(int a)
    {
        cout<<"Wrong input credentials";
    }
    //cin>>a>>b>>k;
    //cout<<a-a<<" "<<a+b-k;
            return 0;
}
```

question

Problem Description: Tina and Fazil are bored, so they are playing an infinite game of ping pong. The rules of the game are as follows: The players play an infinite number of games. At the end of each game, the player who won it scores a point.
In each game, one of the players serves. Tina serves in the first game.
After every K points are scored (regardless of which players scored them), i.e. whenever K games have been played since the last time the serving player changed, the player that serves in the subsequent games changes: if Tina served in the game that just finished, then Fazil will serve in the next game and all subsequent games until the serving player changes again; if Fazil served, then Tina will serve. The players got a little too caught up in the game and they forgot who is supposed to serve in the next game. Will you help them determine that? So far, Tina has scored X points and Fazil has scored Y points. Constraints: 0≤X,Y≤10 powert 9 1≤K≤10 power 9 Input Format: The only line of each test case contains three space-separated integers X, Y and K. Output Format: In the only line of output print the string "Tina" if Tina is supposed to serve next or "Fazil" otherwise (without quotes).

answer

```
#include <iostream>
using namespace std;
int main()
{
    int x,y,k;
    try{
        cin>>x>>y>>k;
        if(cin){
```

```
        if(((x+y)/k)%2==0)
        cout<<"Tina";
        else
        cout<<"Fazil";
      }
      else
      throw 0;
   }
   catch(int a){
      cout<<"Missing Input";
   }
         return 0;
}
```

question

Problem description Pari is an Architect who is currently doing his design work for his new project in one of the congested location of Paris.For making his work simpler he is looking for the automated tool which check whether the area is greater or perimeter is greater or both are equal if the Length (L) and Breadth (B) of a rectangle is provided. Since he is not from the computing background he is looking for the freelancer who can do this for him? Can you help him with the logic for doing so? Constraints: 1≤L≤1000 1≤B≤1000 Input Format: First line will contain the length (L) of the rectangle. Second line will contain the breadth (B) of the rectangle. Output Format: Output 2 lines. In the first line print "Area" if area is greater otherwise print "Peri" and if they are equal print "Eq".(Without quotes). In the second line print the calculated area or perimeter (whichever is greater or anyone if it is equal).

answer

```
#include <iostream>
using namespace std;
int main()
{
   int l,b;
```

```
    try{
        cin>>l;
        cin>>b;
        if(cin){
            if((l*b)>(2*(l+b)))
            cout<<"Area\n"<<l*b;
            else if((l*b)==(2*(l+b)))
            cout<<"Eq\n"<<l*b;
            else
            cout<<"Peri\n"<<2*(l+b);
        }
        else
        throw 0;
    }
    catch(int parameters){
        cout<<"Provide Sufficient Size Information";
    }
            return 0;
}
```

question

Problem Description:
Simon loves to listen to music while walking his way to attend boring lectures in his college.

He has a playlist of songs which has all songs of equal length, L. (in seconds)

One day while going on his way, he decided to calculate his average walking speed and he comes to know that he walks at a speed of 0.5 m/s.

You will be given the distance D ,he has to walk down to reach his class, after which he stops the music.

You have to find the minimum number of songs he needs to add into his playlist so as music plays in the whole path.

Constraints:
1<= L <=120 (in seconds)
1<= D <=5000 (in meters)
  Input Format:
Only line of input contain two integer L and D separated by a space representing length of song and distance he has to walk respectively.

Output Format:
In the only line of output print the Integer value equal to number of songs he need to add into playlist before start to walk.

answer

```
 #include <iostream>
using namespace std;
int main()
{
   int L,D;
   try{
      cin>>L>>D;
      if(cin){
         if((2*D)%L==0)
         cout<<(2*D)/L;
         else
         cout<<(2*D)/L+1;
      }
      else
      throw 0;
   }
   catch(int a){
      cout<<"Invalid input format";
   }
         return 0;
}
```

question

Problem Description:
Vishal is fighting with a monster.The health of the monster is H.In one attack, Vishal can decrease the monster's health by 'A'. There is no other way to decrease the monster's health.Vishal wins when the monster's health becomes '0' or below.
Can you find the number of attacks needed for Vishal for winning ?
Constraints:
1 <= H <= 10^4
1 <= A <= 10^4
All values in input are integers.

Input Format:
Only line of input has two space separated integers H A

Output Format:
Print the number of attacks Vishal needs to make before winning.

answer

```
 #include <iostream>
using namespace std;
int main()
{
   int h,a;
   try{
      cin>>h>>a;
      if(cin){
         if(h%a==0)
         cout<<h/a;
         else if(h%a!=0)
         cout<<h/a+1;
      }
      else
      throw 0;
   }
   catch(int a){
      cout<<"Missing Input Data";
   }
         return 0;
}
```

question

Problem Description: For her next karate demonstration, Prasad will break some bricks. Prasad stacked three bricks on top of each other. Initially, their widths (from top to bottom) are W1,W2,W3. Prasad's strength is S. Whenever she hits a stack of bricks, consider the largest k≥0 such that the sum of widths of the topmost k bricks does not exceed S; the topmost k bricks break and are removed from the stack. Before each hit, Prasad may also decide to reverse the current stack of bricks, with no cost. Find the minimum number of hits Prasad needs in order to break all bricks if she performs the reversals optimally. You are not required to minimise the number of reversals. Constraints: 1≤S≤8 1≤Wi≤2 , for each valid i Input Format: Only line of input contains four space-separated integers S, W1, W2

and W3. Output Format: In only line of output print the integer representing the minimum required number of hits.

answer

```
 #include <iostream>
using namespace std;
int main()
{
   int s,w1,w2,w3;
   try{
      cin>>s>>w1>>w2>>w3;
      if(cin){
         int res=w1+w2+w3;
         if(s>=res)
         cout<<"1";
         else if(s<res&&res%s==0)
         cout<<res/s;
         else
         cout<<res/s+1;
      }
      else
      throw 0;
   }
   catch(int q){
      cout<<"Invalid Bricks Input";
   }
         return 0;
}
```

question

Problem Description: There are two monkeys on an x-axis ready to jump in the positive direction (i.e, toward positive infinity). The first monkey starts at location x1 and moves at a rate of v1 meters per jump. The second monkey starts at location x2 and moves at a rate of v2 meters per jump. Given the starting locations and movement rates for each monkey, can you determine if they'll ever land at the same location at the same time? Constraints: • 0≤x1<x2≤10000 • 1≤v1≤10000 • 1≤v2≤10000 Input Format A single line of four space-separated integers denoting the respective values of x1, v1, x2 and v2. Output Format: Print YES if they can land on the same location at the same time; otherwise, print NO. Note: The two monkeys must land at the same location after making the same number of jumps.

answer

```
#include <iostream>

using namespace std;

int main()

{

   int x1,x2,v1,v2;

   try{

      cin>>x1>>x2>>v1>>v2;

      if(cin){

         if((x2-x1+v2-v1)%(v1-v2)>0)

         cout<<"YES";

         else

         cout<<"NO";

      }

      else

      throw 0;

   }

   catch(int a){

   cout<<"Input coordinates";

   }

         return 0;

}
```

question

Question description: Indian Ceiling Corporation (ICC) is analyzing the properties of its new series of Highly Collapse Proof Ceilings(HCPC). An HCPC consists of n layers of material, each with a different value of collapse resistance (measured as a positive integer). The analysis ICC) wants to run will take the collapse-resistance values of the layers, store them in a binary search tree, and check whether the shape of this tree in any way correlates with the quality of the whole construction. Because, well, why should it not? To be precise, ICC takes the collapse-resistance values for the layers, ordered from the top layer to the bottom layer, and inserts them one-by-one into a tree. The rules for inserting a value v are: If the tree is empty, make v the root of the tree.   If the tree is not empty, compare v with the root of the tree. If v is smaller, insert v into the left subtree of the root, otherwise insert v into the right subtree. ICC has a set of ceiling prototypes it wants to analyze by trying to collapse them. It wants to take each group of ceiling prototypes that have trees of the same shape and analyze them together. Given a set of prototypes, your task is to determine how many different tree shapes they induce. Constraints 1 ≤ n ≤ 50 1 ≤ k ≤ 20 Input Format: The first line of the input contains two integers n, which is the number of ceiling prototypes to analyze, and k, which is the number of layers in each of the prototypes. The next n lines describe the ceiling prototypes. Each of these lines contains k distinct integers (between 1 and 10^6 , inclusive), which are the collapse-resistance values of the layers in a ceiling prototype, ordered from top to bottom. Output Format: Print the number of different tree shapes.

answer

```
#include<stdio.h>

#include<cstring>

#define f(i,n) for(int i=1;i<=n;i++)

#define g(n) for(int j=i+1;j<=n;j++)

#define h(n) for(int i=2;i<=n;i++)

#define dec int s1l[21],s1r[21],s2l[21],s2r[21],a[51][21],n,m,ans;

#define for1 for (last=0,now=1;now;last=now,now=(a[u][i]<=a[u][now])?s1l[now]:s1r[now]);

#define for2 for (last=0,now=1;now;last=now,now=(a[v][i]<=a[v][now])?s2l[now]:s2r[now]);

#define dfs3 void dfs(const int &now1,const int &now2){ if ((bool)now1 ^ (bool)now2) flag=0; if (!flag) return; if (s1l[now1]) dfs(s1l[now1],s2l[now2]); if (s1r[now1]) dfs(s1r[now1],s2r[now2]);}

#define inline1 inline void judge(const int &u,const int &v){s1l[0]=0;s1r[0]=0;int now,last;memset(s1l,0,sizeof(s1l));memset(s2l,0,sizeof(s2l));memset(s1r,0,sizeof(s1r));memset(s2r,0,sizeof(s2r));
```

```
dec
//int s1l[21],s1r[21],s2l[21],s2r[21],a[51][21],n,m,ans;
bool flag,f[51];
dfs3
/*void dfs(const int &now1,const int &now2)
{
if ((bool)now1 ^ (bool)now2) flag=0;
if (!flag) return;
if (s1l[now1]) dfs(s1l[now1],s2l[now2]);
if (s1r[now1]) dfs(s1r[now1],s2r[now2]);
}*/
inline1
/*inline void judge(const int &u,const int &v)
{
s1l[0]=0;s1r[0]=0;int now,last;
memset(s1l,0,sizeof(s1l));
memset(s2l,0,sizeof(s2l));
memset(s1r,0,sizeof(s1r));
memset(s2r,0,sizeof(s2r));*/
h(m)
{
   for1
//for (last=0,now=1;now;last=now,now=(a[u][i]<=a[u][now])?s1l[now]:s1r[now]);
if (a[u][i]<=a[u][last]) s1l[last]=i;
else s1r[last]=i;
}
s2l[0]=0;s2r[0]=0;
for (int i=2;i<=m;i++)
{
   for2
//for (last=0,now=1;now;last=now,now=(a[v][i]<=a[v][now])?s2l[now]:s2r[now]);
```

```
if (a[v][i]<=a[v][last]) s2l[last]=i;
else s2r[last]=i;
}
flag=1;
dfs(1,1);
if (flag) f[v]=1,ans--;
}

int main(){
scanf("%d%d",&n,&m);
f(i,n) f(j,m)/*for (int j=1;j<=m;j++)*/ scanf("%d",&a[i][j]);
f(i,n) g(n) if (!f[i] && !f[j]) judge(i,j);
printf("%d",ans+n);
return 0;
printf("vector<vector<int>>tree(N,vector<int>(K)); cin>>N>>K");}
```