Diet Plan

Problem Description

Arnold is planning to follow a diet suggested by his Nutritionist. The Nutritionist prescribed him the total protein, carbohydrates and fats, he should take daily. Arnold searches on an online food store and makes a list of protein, carbohydrates and fats contained in a single unit of various food items.

His target is to have the maximum protein, carbohydrates and fats in his diet without exceeding the prescribed limit. He also wants to have as much diverse food items as much as possible. That is, he does not want to have many units of one food item and 0 of others. Multiple combinations of 'units of food items' are possible to achieve the target. Mathematically speaking, diversity is more if the difference between the number of units of food item chosen the most and the number of units of another food item chosen the least, is as small as possible.

To solve this problem, he uses maximum possible number of units of all the items so that total amount of protein, carbohydrates and fats is not more than prescribed limit. For example - if total nutrition required is 100P, 130C and 130F (where P is Protein, C is Carbohydrates and F is Fats) and 2 different food items, viz. Item A and Item B, have following amount of nutrition:

Item A - 10P, 20C, 30F

Item B - 20P, 30C, 20F

then, he can have (maximum possible) 2 units of all the items as having 3 units will exceed the prescribed amount of Carbohydrates and fats.

Next, he chooses food items to fulfill the remaining nutrients. He chooses one more units of maximum number of food items. He continues this process till he cannot add a unit of any food item to his diet without exceeding the prescribed limit. In this example, he can choose one more unit of item B or one more unit of item A. In case he has two sets of food items then the priority is given to fulfill the requirements of Protein, Carbohydrates, and Fats in that order. So he chooses item B.

You will be provided the maximum nutrients required and the nutrients available in various food items. You need to find the amount of nutrients for which there is a shortfall as compared to the prescription, after making his selection using the process described above. In the example he still needs 20P, 0C, 10F to achieve his target.

Constraints

Number of Food Items <= 10

Maximum amount of Nutrients is less than 1500 i.e. x +y + z <= 1500

Amount of P, C, F in two food items will not be same

P, C, F values in input can be in any order

Output should be in order - P, C, F.

Input

First line contains the maximum limit of nutrients in the following format.

xP yC zF, where x, y and z are integers

Second line contains nutrient composition of different food items separated by pipe (|).
Output

Print the shortfall for each nutrient type, fulfilled separated by space.

E.g. If the output is 10P, 20 C, 30 F, then print "10 20 30" (without quotes).

Time Limit

1

Examples

Example 1

Input

100P 130C 130F

10P 20C 30F|20P 30C 20F

Output

20 0 10

Explanation

Having 2 units of item A and 3 units of item B provides - 2 * [10P 20C 30F] + 3 * [20P 30C 20F] = 100P, 130C, 120F. This is the best combination that can reduce the shortfall [20P, 0C, 10F] without exceeding his prescription. In contrast, if 3 units of A and 2 units of B are chosen then 3 * [10P 20C 30F] + 2 * [20P 30C 20F] = 70P, 120C, 130F produces a shortfall of [30P, 10C, 0F]. However, since protein shortfall in this case is more than the previous combination, this is not the right combination to follow.

Example 2

Input

130P 120C 110F

4P 9C 2F|4P 3C 2F|7P 1C 3F

Output

2 4 50

Explanation

Having 9 units of item A, 9 units of item B and 8 units of Item C provides - 9 * [4P 9C 2F] + 9 * [4P 3C 2F] + 8 * [7P 1C 3F] = 108P, 116C, 60F. This is the best combination that can reduce the shortfall [2P, 4C, 50F] without exceeding his prescription.

Solution
(C ++)

```
#include<bits/stdc++.h>
using namespace std;
typedef  long long  ll;
ll inf=1000000000000000000,mod=1000000007,BS,k;
#define en printf("\n");
int main(){
	string a,b;
	getline(cin,a);int res=0,p,c,f,a_len,b_len;
	a_len=a.length();
	for(int i=0;i<a_len;i++){
	   if(a[i]==' ')continue;
	   if(isdigit(a[i])){
	      res=res*10+a[i]-'0';
	   }
	   else{
	      if(a[i]=='P')
	      p=res;
	      else if(a[i]=='C')
	      c=res;
	      else
	      f=res;
	      res=0;
	   }
	}
	getline(cin,b);int pp,cc,ff;
	vector<pair<int,pair<int,int>>>ans;res=0;
	int mi=INT_MAX,sump=0,sumc=0,sumf=0;
	b_len=b.length();
	for(int i=0;i<b_len;i++){
	   if(b[i]==' ')continue;
	   if(b[i]=='|'){
	      ans.push_back({pp,{cc,ff}});
	      pp=0;ff=0;cc=0;
	   }
	   if(isdigit(b[i])){
	      res=res*10+b[i]-'0';
	   }
	   else{
	      if(b[i]=='P')
```

```
            pp=res,sump+=pp;
            else if(b[i]=='C')
            cc=res,sumc+=cc;
            else
            ff=res,sumf+=ff;
            res=0;
        }
    }ans.push_back({pp,{cc,ff}});
    if(sump)
    mi=min(mi,p/sump);
    if(sumc)
    mi=min(mi,c/sumc);
    if(sumf)
    mi=min(mi,f/sumf);
    if(mi==INT_MAX)mi=0;
    int nn=ans.size();
    pair<int,pair<int,int>>lef,ma={0,{0,0}},curr;
    // cout<<p<<" "<<c<<" "<<f;
    lef={p-mi*sump,{c-mi*sumc,f-mi*sumf}};
    // cout<<lef.first<<" "<<lef.second.first<<" "<<lef.second.second;
    for(int i=0;i<(1<<nn);i++){
        curr={0,{0,0}};
        for(int j=0;j<nn;j++){
            if(i&(1<<j))
            {
                curr.first+=ans[j].first;
                curr.second.first+=ans[j].second.first;
                curr.second.second+=ans[j].second.second;
            }
        }
        if(curr.first<=lef.first&&curr.second.first<=lef.second.first&&curr.second.second<=l
ef.second.second)
        ma=max(ma,curr);
    }
    cout<<p-sump*mi-ma.first<<" "<<c-sumc*mi-ma.second.first<<" "<<f-sumf*mi-
ma.second.second<<endl;
  return 0;
}
```

Count Pairs | TCS CodeVita 9 Solution (Zone 1) 2020 | By CodingHumans |

by CodingHumans *on* 22:19 *in* TCS codevita IX Zone 1

Count Pairs

Problem Description

Given an array of integers A, and an integer K find number of happy elements.

Element X is happy if there exists at least 1 element whose difference is less than K i.e. an element X is happy,
if there is another element in the range [X-K, X+K] other than X itself.

Constraints

1 <= N <= 10^5

0 <= K <= 10^5

0 <= A[i] <= 10^9

Input

First line contains two integers N and K where N is size of the array and K is a number as described above

Second line contains N integers separated by space.

Output

Print a single integer denoting the total number of happy elements.

Time Limit

1

Examples
Example 1

Input

6 3
5 5 7 9 15 2

Output

5

Explanation

Other than number 15, everyone has at least 1 element in the range [X-3, X+3].
Hence they are all happy elements. Since these five are in number, the output is 5.

Example 2

Input

3 2
1 3 5

Output

3

Explanation

All numbers have at least 1 element in the range [X-2, X+2]. Hence they are all happy elements.
Since these three are in number, the output is 3.

Solution
(C ++)

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

```
typedef long long ll;

void test()
{
   ll n,k,l;
   scanf("%lld",&n);

   set<ll> s; // to remove duplicate elements from array
   map<ll,int> m; // to keep track of frequency of each element in given array

   for(int i=0;i<n;i++)
   {scanf("%lld",&l);
    s.insert(l);
    m[l]++;
   }

  //copying the set to array a
  n=s.size();
  ll a[n],i=0,count=0;

  for(ll x : s) {a[i]=x;
                 i++; }
 // loop to check  (adjacent diff <=k )
 if(a[1]-a[0]<=k) count+=m[a[0]];
 for(int i=1;i<n-1;i++)
 {
   if( a[i+1]-a[i]<=k || abs(a[i]-a[i-1])<=k) count+=m[a[i]]; // when this condition satisfies we
increment the count by that element frequency.
 }
 if(a[n-1]-a[n-2]<=k) count+=m[a[n-1]];

  cout<<count; // printing output

}
int main()
{ test();

}
```

Minimum Gifts

Problem Description

A Company has decided to give some gifts to all of its employees. For that, company has given some

rank to each employee. Based on that rank, company has made certain rules to distribute the gifts.

The rules for distributing the gifts are:

Each employee must receive at least one gift.

Employees having higher ranking get a greater number of gifts than their neighbours.

What is the minimum number of gifts required by company?

Constraints

$1 < T < 10$

$1 < N < 100000$

$1 < Rank < 10^9$

Input

First line contains integer T, denoting the number of testcases.

For each testcases:

First line contains integer N, denoting number of employees.

Second line contains N space separated integers, denoting the rank of each employee.

Output

For each testcase print the number of minimum gifts required on new line.

Time Limit

1

Examples
Example 1

Input

```
2
5
1 2 1 5 2
2
1 2
```

Output

7
3

Explanation

For testcase 1, adhering to rules mentioned above,

Employee # 1 whose rank is 1 gets one gift

Employee # 2 whose rank is 2 gets two gifts

Employee # 3 whose rank is 1 gets one gift

Employee # 4 whose rank is 5 gets two gifts

Employee # 5 whose rank is 2 gets one gift

Therefore, total gifts required is 1 + 2 + 1 + 2 + 1 = 7

Similarly, for testcase 2, adhering to rules mentioned above,

Employee # 1 whose rank is 1 gets one gift

Employee # 2 whose rank is 2 gets two gifts

Therefore, total gifts required is 1 + 2 = 3

Solution
(C ++)

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
   int t;
   cin>>t;
   while(t--){
 long n;
 cin>>n;
    long a[n],b[n],s1=0;
 for(int i=0;i<n;i++)
 {
 cin>>a[i];
  b[i]=1;
 }

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

```
 if(a[i]>a[i+1] && b[i]<=b[i+1])
 {
                b[i]=b[i+1]+1;
                for(int j=i;j>0;j--)
                {
                   if(a[j-1]>a[j] && b[j-1]<=b[j])
                      b[j-1] = b[j]+1;
                   else
                      break;
                }
             }
             else if(a[i]<a[i+1] && b[i+1]<=b[i])
             {
                b[i+1] = b[i]+1;

             }
}
      for(int i=0;i<n;i++)
      {
    s1+=b[i];
}

    cout<<s1<<endl;}

       return 0;
}
```

Minimize The Sum

Problem Description

Given an array of integers, perform atmost K operations so that the sum of elements of final array is minimum. An operation is defined as follows -

Consider any 1 element from the array, arr[i].

Replace arr[i] by floor(arr[i]/2).

Perform next operations on updated array.

The task is to minimize the sum after atmost K operations.

Constraints

1 <= N, K <= 10^5.

Input

First line contains two integers N and K representing size of array and maximum numbers of operations that can be performed on the array respectively.

Second line contains N space separated integers denoting the elements of the array, arr.

Output

Print a single integer denoting the minimum sum of the final array.

Time Limit

1

Examples
Example 1

Input

4 3

20 7 5 4

Output

17

Explanation

Operation 1 -> Select 20. Replace it by 10.

New array = [10, 7, 5, 4]

Operation 2 -> Select 10. Replace it by 5.

New array = [5, 7, 5, 4].

Operation 3 -> Select 7. Replace it by 3.

New array = [5,3,5,4].

Sum = 17.

Solution
(C++)

```
#include<bits/stdc++.h>
using namespace std;
#define FIO ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
#define ull unsigned long long
int main()
{
 FIO;
 ull n,k,x,y,s=0; cin>>n>>k;
 priority_queue<int> p;
 for(ull i=0;i<n;i++)
 {
   cin>>x;
   p.push(x);
 }
 while(k--)
 {
   x=p.top();
   p.pop();
   x=floor(x/2);
   p.push(x);
 }
 while(!p.empty())
 {
   s+=p.top();
   p.pop();
 }
 cout<<s;
 return 0;

}
```

Railway Station

Problem Description

Given schedule of trains and their stoppage time at a Railway Station, find minimum number of platforms needed.
Note - If Train A's departure time is x and Train B's arrival time is x, then we can't accommodate Train B on the same platform as Train A.

Constraints

1 <= N <= 10^5
0 <= a <= 86400 0 < b <= 86400
Number of platforms > 0

Input

First line contains N denoting number of trains. Next N line contain 2 integers, a and b, denoting the arrival time and stoppage time of train.

Output

Single integer denoting the minimum numbers of platforms needed to accommodate every train.

Time Limit

1

Examples

Example 1

Input

3 10 2 5 10 13 5

Output

2

Explanation

The earliest arriving train at time t = 5 will arrive at platform# 1.
Since it will stay there till t = 15, train arriving at time t = 10 will arrive at platform# 2. Since it will depart at time t = 12, train arriving at time t = 13 will arrive at platform# 2.

Solution
(C ++)

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

int solve(ll a[],ll b[],int n)
{
   sort(a,a+n);
   sort(b,b+n);

   int p=1,r=1;
   for(int i=1,j=0; i<n && j<n;) {

       if (a[i]<=b[j])
       {p++;
        i++;}

       else if (a[i]>b[j])
       { p--;
         j++;}
       r=max(r,p);
   }
   return r;
}

void test()
{
   ll n;
   scanf("%lld",&n);

   ll a[n],b[n],s,t;


   for(int i=0;i<n;i++)
   {scanf("%lld%lld",&s,&t);
    a[i]=s;
    b[i]=s+t;
   }

   cout<<solve(a,b,n);

}
int main()
{ test();
}
```