**Iterator**

set<ll>st;

set<ll>:: iterator itt;

for(itt=st.begin();itt!=st.end();it++)

cout<<\*it<<" "<<endl;

**Permutation of a string**

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

**using namespace std;**

**void permute(string str, string out)**

**{**

**if (str.size() == 0)**

**{**

**cout << out << endl;**

**return;**

**}**

**for (int i = 0; i < str.size(); i++)**

**{**

**permute(str.substr(1), out + str[0]);**

**rotate(str.begin(), str.begin() + 1, str.end());**

**}**

**}**

**int main()**

**{**

**string str = "ABC";**

**permute(str, "");**

**return 0;**

**}**

**#Queue**

int main()

{

    ll n;

    cin>>n;

    queue<ll>qq;

    for(int i=0;i<n;i++)

    {

        ll x;

        cin>>x;

        qq.push(x);

    }

    qq.pop();// for remove top element

//cout<<qq.size();

    while(!qq.empty())

    {

         cout<<qq.front()<<" ";

        qq.pop();// it must give after cout

    }

}

**#Stack**

int main()

{

    ll n;

    cin>>n;

    stack<ll>stk;

    for(int i=0; i<n; i++)

    {

        ll x;

        cin>>x;

        stk.push(x);

    }

    //stk.pop();// for remove top element

//cout<<stk.size();

    while(!stk.empty())

    {

        cout<<stk.top()<<" ";

        stk.pop();// it must give after cout

    }

}

**#Map**

int main()

{

    ll n;

    vector<char>vc;

    cin>>n;

    for(int ii=0;ii<n;ii++)

    {

        string st;

        cin>>st;

        set<char>s;

        set<char> :: iterator itt;

        map<char, int>m;

        map<char, int>::iterator it;

        for(int i=0;i<st.length();i++)

        {

           if(st[i] == st[i+1]) i++;

           else{

            m[st[i]] = 1;

            s.insert(st[i]);

           }

        }

                for(itt = s.begin(); itt != s.end(); itt++)

                    cout << \*itt;

                cout<<endl;

                m.clear();

            s.clear();

    }

    return 0;

}

**#MultiSet**

#include <iostream>

#include <set>

int main ()

{

  std::multiset<int> mymultiset;

  std::multiset<int>::iterator itlow,itup;

  for (int i=1; i<8; i++) mymultiset.insert(i\*10); // 10 20 30 40 50 60 70

  itlow = mymultiset.lower\_bound (30);             //       ^

  itup = mymultiset.upper\_bound (40);              //             ^

  mymultiset.erase(itlow,itup);                    // 10 20 50 60 70

  std::cout << "mymultiset contains:";

  for (std::multiset<int>::iterator it=mymultiset.begin(); it!=mymultiset.end(); ++it)

    std::cout << ' ' << \*it;

  std::cout << '\n';

  return 0;

}

**#Pair**

**int** main() {

**int** a,b;

    // ..... something assigned to a and b.

    a = 10;

    b = 25;

    std::pair < **int**, **int** > v = make\_pair(a,b);

**int** c = v.first;

**int** d = v.second

    std::cout << c << std::endl;

    std::cout << d << std::endl;

**return** 0;

}

**#Reverse an Integer Number**

cin>>n;

**while**(n!=0)

  {

     rem=n%10;

     reverse=reverse\*10+rem;

     n/=10;

  }

 cout<<"Reversed Number: "<<reverse<<endl;

**#Convert Decimal to Ternary**

int main()

{

    long rem, base = 1, ternary = 0;

    long num;

    cout<<"Enter a Decimal Number: ";

    cin>>num;

    while(num > 0)

    {

        rem = num % 3;

        ternary = ternary + rem \* base;

        num /= 3;

        base = base \* 10;

    }

    cout<<"The Ternary number is: "<<ternary<<endl;

return 0;

}

**#Distinct Value from an Array & it’s Index**

int main()

{

     ll n,k;

    cin>>n>>k;

    ll arr[n+5];

    ll check[1005];

    memset(check,0,sizeof(check));

    for(int i=1;i<=n;i++)

    {

        ll x;

        cin>>x;

        if(!check[x])

        {

            check[x]=1;

            //cout<<x<<" "; //Distinct Value

            vc.push\_back(i); // here i is index of distinct Value

        }

    }

    //cout<<vc.size()<<endl;

        for(int i=0;i<vc.size();i++)

            cout<<vc[i]<<" ";

    return 0;

}

**#maximum number of houses having no two adjacent  of the same color.**

**1 2 3 3 2 1 2 2**

**Output : 4 (3,2,1,2)**

int main()

{

    ll n,k;

    cin>>n>>k;

    ll arr[n+5];

    ll cnt=0;

    ll mx=0;

    for(int i=0;i<n;i++)

    {

        cin>>arr[i];

    }

    for(int i=0;i<n-1;i++)

    {

        if(arr[i]!=arr[i+1])

        {

            cnt++;

            if(mx<cnt)

                mx=cnt;

        }

        else

            cnt=0;

    }

    cout<<mx+1<<endl;

    return 0;

}

**#**Maximum consecutive one’s (or zeros) in a binary array

int getMaxLength(bool arr[], int n)

{

int count = 0;

int result = 0;

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

{

if (arr[i] == 0)

count = 0;

else

{

count++;

result = max(result, count);

}

}

return result;

}

int main()

{

bool arr[] = {1, 1, 0, 0, 1, 0, 1, 0,

1, 1, 1, 1};

int n = sizeof(arr)/sizeof(arr[0]);

cout << getMaxLength(arr, n) << endl;

return 0;

}

**#Find Palindrome Number in A given range**

int main()

{

    for(int i=10;i<=100005;i++)

    {

        ll temp=i;

        ll flag=0;

        while(temp)

        {

            ll rem=temp%10;

            temp=temp/10;

            flag=flag\*10+rem;

        }

        if(i==flag)

        {

            cout<<i<<"\t";

        }

    }

    return 0;

}

**#Binary Search**

int search(int arr[],int n,int x)

{

    int start=0,end=n-1;

    while(start<=end)

    {

        int mid=(start+end)/2;

        if(x==arr[mid])

            return mid;

        else if(arr[mid]<x)

                start=mid+1;

            else

                end=mid-1;

        }

    return -1;

}

int main(void)

{

    int xx;

    cin>>xx;

    int arr[10005];

    for(int i=0;i<xx;i++)

    {

        cin>>arr[i];

    }

    int target=5;

    sort(arr,arr+xx);

    int index=search(arr,xx,target);

    if(index>0)

        cout<<index<<endl;

    else

        cout<<"Element Doesn't found"<<endl;

    return 0;

}

**Remove Higger ASCII from a String Template**

int main()

{

    int n;

    while(cin>>n)

    {

        string str;

        cin >> str;

        int m = str[0], flag=1;

        for(int i=1; i<n; i++){

            int x = str[i];

            m = max(m, x);

        }

        for(int i=0; i<n; i++){

            int x = str[i];

            if(m==x && flag) {flag=0; continue;}

            cout << str[i];

        }

        cout << endl;

    }

    return 0;

}

**#BigMod**

long long BigModRecursive(long long a, long long r, long long Mod)

{

    if(r==0)

        return 1;

    long long aka;

    aka=BigModRecursive(a,r/2,Mod);

    aka=aka\*aka;

    aka=aka%Mod;

    if(r%2==1)

    {

        aka=aka\*a;

        aka=aka%Mod;

    }

    return aka;

}

int main()

{

    long long n;

    long long xx=1378;

    long long yy=10;

    while(cin>>n)

    {

       cout<<BigModRecursive(xx,n,yy)<<endl;

    }

    return 0;

}

**#Find a number that is not Perfect Squre(Maximum)**

int main()

{

    vector<ll>vc;

    vector<ll>pstore;

    ll n;

    cin>>n;

    ll x;

    ll arr[n+5];

    for(int i=0;i<n;i++)

    {

        cin>>x;

        double dd=sqrt(x);

        ll intt=dd;

        if(dd==intt)

        {

            vc.push\_back(x);

        }

        else

            pstore.push\_back(x);

    }

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

        cout<<pstore[pstore.size()-1]<<" ";

    return 0;

}

**#X^Y and Y^X for big number**

int main()

{

    ld x,y;

    cin>> x >> y ;

    ld a=1.0\*y\*log2(1.0\*x) ;

    ld b=1.0\*x\*log2(1.0\*y) ;

    if(a>b)

        cout<<">"<<endl;

    else if(b>a)

        cout<<"<"<<endl;

    else

        cout<<"="<<endl;

    return 0;

}

**#Prime Factorization of a Given Number**

void primeFactors(ll n)

{

while (n%2 == 0)

{

n = n/2;

}

cout<<2<<endl;

for(ll i=3;i<=sqrt(n);i=i+2)

{

while (n%i==0)

{

n=n/i;

cout<<i<<endl;

}

}

if(n>2)

    {

        cout<<n<<endl;

    }

}

int main()

{

int n = 123 ;

primeFactors(n);

return 0;

}

#Finding LCM of a given Array

int gcd(int a,int b)

    {

        if(b==0)

            return a;

        return gcd(b,a%b);

    }

int main()

{

    int n;

    cin>>n;

    int arr[n+5];

    for(int i=0;i<n;i++)

    {

        cin>>arr[i];

    }

    int sum=arr[0];

    int xx,yy;

    for(int i=0;i<n;i++)

    {

        sum=((arr[i]\*sum)/gcd(arr[i],sum));

    }

    cout<<sum<<endl;

    return 0;

}

#Find LCM

int main()

{

    int a,b;

    cin>>a>>b;

    int lcm=(a\*b)/\_\_gcd(a,b);

    cout<<lcm<<endl;

    return 0;

}

#value convert into 100,50,20,10,5,1 takay

int main()

{

    ll t;

    cin>>t;

    while(t--)

    {

           ll taka=0;

           ll n;

           cin>>n;

           if(n>=100)

           {

               taka=taka+n/100;

               n=n%100;

           }

           if(n>=50)

           {

               taka=taka+n/50;

               n=n%50;

           }

           if(n>=10)

           {

               taka=taka+n/10;

               n=n%10;

           }

           if(n>=5)

           {

               taka=taka+n/5;

               n=n%5;

           }

           if(n>=2)

           {

               taka=taka+n/2;

               n=n%2;

           }

           if(n>=1)

           {

               taka=taka+n/1;

               n=n%1;

           }

           cout<<taka<<endl;

    }

    return 0;

}

 #**Find Factorial**

int main()

{

int arr[201],t,n,i,j,k,T;

cin>>T;

while(T--)

{

    cin>>n;

    for(int p=0;p<=200;p++)

    { arr[p]=0;

    }

    arr[200]=1;

    t=0;

     for(j=1;j<=n;j++)

    {

    for(i=200;i>=0;i--)

    {

      k=arr[i]\*j+t;

      arr[i]=k%10;

    t=k/10;

    }

    }

        int index=0,q=0;

        while(arr[q]==0)

        {

      q++;

        }

    cout<<endl;

    for(int r=q;r<=200;r++)

     { cout<<arr[r];

     }

}

cout<<endl;

return 0;

}

**#Nearest lucky Number ..(like 179 ans is 1 because 180 is lucky number.. 8 is there)**

ll checkeight(ll num)

{

    num=abs(num);

    while(num)

    {

        if(num%10==8)

            return 1;

        num/=10;

    }

    return 0;

}

int main()

{

    ll n;

    cin>>n;

    ll cnt=0;

    ll num=n+1;

    while(1)

    {

       if(checkeight(num))

       {

           ll ans=num-n;

           cout<<ans<<endl;

           return 0;

       }

       else

        num++;

    }

    return 0;

}

**#Maximum Repeated character in a string and its Index**

ll a[1000];  
int main()  
{  
    ll n;  
    cin>>n;  
    string st;  
    cin>>st;  
    ll cnt =0;  
    char temp;  
    for(int i=0;i<n;i++)  
    {  
        a[st[i]]++;  
        if(a[st[i]]>=cnt)  
        {  
            temp = st[i];  
            cnt = a[st[i]];  
        }  
        //cnt=max(a[st[i]], cnt);  
    }  
    //cout<<cnt<<endl;  
    //cout << temp << endl;  
  
    if(n==1)  
        cout<<"Yes"<<endl;  
   else if(cnt>=2)  
        cout<<"Yes"<<endl;  
    else  
        cout<<"No"<<endl;  
  
  
    return 0;  
}

**#Remove 0 before a number in string (0123 ans 123)**

#include<bits/stdc++.h>

#define ll long long

using namespace std;

int main()

{

    ll n;

    cin>>n;

    for(int i=0;i<n;i++)

    {

        string st;

        cin>>st;

        ll len=st.length();

            reverse(st.begin(),st.end());

        ll flag=1;

        for(int i=0;i<len;i++)

        {

            if(flag==1&&st[i]!='0')

                {flag=0;}

            if(flag==0)

            {

                cout<<st[i];

            }

        }

        cout<<endl;

        if(flag)

            cout<<0<<endl;

    }

    return 0;

}

**#Remove 0 before a number(0123 ans 123)**

#include<bits/stdc++.h>

using namespace std;

int main()

{

    string st1,st2;

    cin>>st1>>st2;

    long long len=st1.length();

    int arr[len+5];

    long long res;

    if(st1==st2)

        cout<<0<<endl;

    else

    {

         for(int i=0;i<len;i++)

    {

        res=abs(st1[i]-st2[i]);

        arr[i]=res;

    }

        long long flag=1;

        for(int i=0;i<len;i++)

        {

            if(flag==1&&arr[i]!=0)

            flag=0;

        if(flag!=1)

            {

            printf("%d",arr[i]);

            }

        }

    }

   return 0;

    }

**# (Finding maximum number of char in  a string & counting it)**

int main()

{

    string st;

    int cnt=0;

    cin>>st;

    int flag=0;

    int ll=st.length();

    sort(st.begin(),st.end());

    //cout<<st<<endl;

    int mx=0;

   for(int i=0;i<=ll;i++)

   {

       if(st[i]>mx)

       {

           mx=st[i];

           flag=1;

       }

       else if(st[i]==mx)

        flag++;

   }

   for(int i=0;i<flag;i++)

   {

       //cout<<flag<<endl;

       cout<<char(mx);

   }

**Iterator**

set<ll>st;

set<ll>:: iterator itt;

for(itt=st.begin();itt!=st.end();it++)

cout<<\*it<<" "<<endl;

**Permutation of a string**

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

**using namespace std;**

**void permute(string str, string out)**

**{**

**if (str.size() == 0)**

**{**

**cout << out << endl;**

**return;**

**}**

**for (int i = 0; i < str.size(); i++)**

**{**

**permute(str.substr(1), out + str[0]);**

**rotate(str.begin(), str.begin() + 1, str.end());**

**}**

**}**

**int main()**

**{**

**string str = "ABC";**

**permute(str, "");**

**return 0;**

**}**

**#Queue**

int main()

{

    ll n;

    cin>>n;

    queue<ll>qq;

    for(int i=0;i<n;i++)

    {

        ll x;

        cin>>x;

        qq.push(x);

    }

    qq.pop();// for remove top element

//cout<<qq.size();

    while(!qq.empty())

    {

         cout<<qq.front()<<" ";

        qq.pop();// it must give after cout

    }

}

**#Stack**

int main()

{

    ll n;

    cin>>n;

    stack<ll>stk;

    for(int i=0; i<n; i++)

    {

        ll x;

        cin>>x;

        stk.push(x);

    }

    //stk.pop();// for remove top element

//cout<<stk.size();

    while(!stk.empty())

    {

        cout<<stk.top()<<" ";

        stk.pop();// it must give after cout

    }

}

**#Map**

int main()

{

    ll n;

    vector<char>vc;

    cin>>n;

    for(int ii=0;ii<n;ii++)

    {

        string st;

        cin>>st;

        set<char>s;

        set<char> :: iterator itt;

        map<char, int>m;

        map<char, int>::iterator it;

        for(int i=0;i<st.length();i++)

        {

           if(st[i] == st[i+1]) i++;

           else{

            m[st[i]] = 1;

            s.insert(st[i]);

           }

        }

                for(itt = s.begin(); itt != s.end(); itt++)

                    cout << \*itt;

                cout<<endl;

                m.clear();

            s.clear();

    }

    return 0;

}

**#MAPP**

#include<bits/stdc++.h>

using namespace std;

//vector<pair<int,int>>v;

//int ara[10005];

map<string,int>mp;

int main()

{

    int n;

    cin>>n;

    while(n--)

    {

        int cre;

        string name;

        cin>>name>>cre;

        mp[name]=cre;

    }

    int q;

    cin>>q;

    int sum=0;

    while(q--)

    {

        int a,b;

        cin>>a>>b;

        int sum=mp[a]+mp[b];

        cout<<sum<<endl;

    }

    return 0;

}

**#MultiSet**

#include <iostream>

#include <set>

int main ()

{

  std::multiset<int> mymultiset;

  std::multiset<int>::iterator itlow,itup;

  for (int i=1; i<8; i++) mymultiset.insert(i\*10); // 10 20 30 40 50 60 70

  itlow = mymultiset.lower\_bound (30);             //       ^

  itup = mymultiset.upper\_bound (40);              //             ^

  mymultiset.erase(itlow,itup);                    // 10 20 50 60 70

  std::cout << "mymultiset contains:";

  for (std::multiset<int>::iterator it=mymultiset.begin(); it!=mymultiset.end(); ++it)

    std::cout << ' ' << \*it;

  std::cout << '\n';

  return 0;

}

**#Pair**

**int** main() {

**int** a,b;

    // ..... something assigned to a and b.

    a = 10;

    b = 25;

    std::pair < **int**, **int** > v = make\_pair(a,b);

**int** c = v.first;

**int** d = v.second

    std::cout << c << std::endl;

    std::cout << d << std::endl;

**return** 0;

}

**#Reverse an Integer Number**

cin>>n;

**while**(n!=0)

  {

     rem=n%10;

     reverse=reverse\*10+rem;

     n/=10;

  }

 cout<<"Reversed Number: "<<reverse<<endl;

**#Convert Decimal to Ternary**

int main()

{

    long rem, base = 1, ternary = 0;

    long num;

    cout<<"Enter a Decimal Number: ";

    cin>>num;

    while(num > 0)

    {

        rem = num % 3;

        ternary = ternary + rem \* base;

        num /= 3;

        base = base \* 10;

    }

    cout<<"The Ternary number is: "<<ternary<<endl;

return 0;

}

**#Distinct Value from an Array & it’s Index**

int main()

{

     ll n,k;

    cin>>n>>k;

    ll arr[n+5];

    ll check[1005];

    memset(check,0,sizeof(check));

    for(int i=1;i<=n;i++)

    {

        ll x;

        cin>>x;

        if(!check[x])

        {

            check[x]=1;

            //cout<<x<<" "; //Distinct Value

            vc.push\_back(i); // here i is index of distinct Value

        }

    }

    //cout<<vc.size()<<endl;

        for(int i=0;i<vc.size();i++)

            cout<<vc[i]<<" ";

    return 0;

}

**#maximum number of houses having no two adjacent  of the same color.**

**1 2 3 3 2 1 2 2**

**Output : 4 (3,2,1,2)**

int main()

{

    ll n,k;

    cin>>n>>k;

    ll arr[n+5];

    ll cnt=0;

    ll mx=0;

    for(int i=0;i<n;i++)

    {

        cin>>arr[i];

    }

    for(int i=0;i<n-1;i++)

    {

        if(arr[i]!=arr[i+1])

        {

            cnt++;

            if(mx<cnt)

                mx=cnt;

        }

        else

            cnt=0;

    }

    cout<<mx+1<<endl;

    return 0;

}

**#**Maximum consecutive one’s (or zeros) in a binary array

int getMaxLength(bool arr[], int n)

{

int count = 0;

int result = 0;

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

{

if (arr[i] == 0)

count = 0;

else

{

count++;

result = max(result, count);

}

}

return result;

}

int main()

{

bool arr[] = {1, 1, 0, 0, 1, 0, 1, 0,

1, 1, 1, 1};

int n = sizeof(arr)/sizeof(arr[0]);

cout << getMaxLength(arr, n) << endl;

return 0;

}

**#Find Palindrome Number in A given range**

int main()

{

    for(int i=10;i<=100005;i++)

    {

        ll temp=i;

        ll flag=0;

        while(temp)

        {

            ll rem=temp%10;

            temp=temp/10;

            flag=flag\*10+rem;

        }

        if(i==flag)

        {

            cout<<i<<"\t";

        }

    }

    return 0;

}

**#Binary Search**

int search(int arr[],int n,int x)

{

    int start=0,end=n-1;

    while(start<=end)

    {

        int mid=(start+end)/2;

        if(x==arr[mid])

            return mid;

        else if(arr[mid]<x)

                start=mid+1;

            else

                end=mid-1;

        }

    return -1;

}

int main(void)

{

    int xx;

    cin>>xx;

    int arr[10005];

    for(int i=0;i<xx;i++)

    {

        cin>>arr[i];

    }

    int target=5;

    sort(arr,arr+xx);

    int index=search(arr,xx,target);

    if(index>0)

        cout<<index<<endl;

    else

        cout<<"Element Doesn't found"<<endl;

    return 0;

}

**Remove Higger ASCII from a String Template**

int main()

{

    int n;

    while(cin>>n)

    {

        string str;

        cin >> str;

        int m = str[0], flag=1;

        for(int i=1; i<n; i++){

            int x = str[i];

            m = max(m, x);

        }

        for(int i=0; i<n; i++){

            int x = str[i];

            if(m==x && flag) {flag=0; continue;}

            cout << str[i];

        }

        cout << endl;

    }

    return 0;

}

**#BigMod**

long long BigModRecursive(long long a, long long r, long long Mod)

{

    if(r==0)

        return 1;

    long long aka;

    aka=BigModRecursive(a,r/2,Mod);

    aka=aka\*aka;

    aka=aka%Mod;

    if(r%2==1)

    {

        aka=aka\*a;

        aka=aka%Mod;

    }

    return aka;

}

int main()

{

    long long n;

    long long xx=1378;

    long long yy=10;

    while(cin>>n)

    {

       cout<<BigModRecursive(xx,n,yy)<<endl;

    }

    return 0;

}

**#Find a number that is not Perfect Squre(Maximum)**

int main()

{

    vector<ll>vc;

    vector<ll>pstore;

    ll n;

    cin>>n;

    ll x;

    ll arr[n+5];

    for(int i=0;i<n;i++)

    {

        cin>>x;

        double dd=sqrt(x);

        ll intt=dd;

        if(dd==intt)

        {

            vc.push\_back(x);

        }

        else

            pstore.push\_back(x);

    }

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

        cout<<pstore[pstore.size()-1]<<" ";

    return 0;

}

**#X^Y and Y^X for big number**

int main()

{

    ld x,y;

    cin>> x >> y ;

    ld a=1.0\*y\*log2(1.0\*x) ;

    ld b=1.0\*x\*log2(1.0\*y) ;

    if(a>b)

        cout<<">"<<endl;

    else if(b>a)

        cout<<"<"<<endl;

    else

        cout<<"="<<endl;

    return 0;

}

**#Prime Factorization of a Given Number**

void primeFactors(ll n)

{

while (n%2 == 0)

{

n = n/2;

}

cout<<2<<endl;

for(ll i=3;i<=sqrt(n);i=i+2)

{

while (n%i==0)

{

n=n/i;

cout<<i<<endl;

}

}

if(n>2)

    {

        cout<<n<<endl;

    }

}

int main()

{

int n = 123 ;

primeFactors(n);

return 0;

}

#Finding LCM of a given Array

int gcd(int a,int b)

    {

        if(b==0)

            return a;

        return gcd(b,a%b);

    }

int main()

{

    int n;

    cin>>n;

    int arr[n+5];

    for(int i=0;i<n;i++)

    {

        cin>>arr[i];

    }

    int sum=arr[0];

    int xx,yy;

    for(int i=0;i<n;i++)

    {

        sum=((arr[i]\*sum)/gcd(arr[i],sum));

    }

    cout<<sum<<endl;

    return 0;

}

#Find LCM

int main()

{

    int a,b;

    cin>>a>>b;

    int lcm=(a\*b)/\_\_gcd(a,b);

    cout<<lcm<<endl;

    return 0;

}

#value convert into 100,50,20,10,5,1 takay

int main()

{

    ll t;

    cin>>t;

    while(t--)

    {

           ll taka=0;

           ll n;

           cin>>n;

           if(n>=100)

           {

               taka=taka+n/100;

               n=n%100;

           }

           if(n>=50)

           {

               taka=taka+n/50;

               n=n%50;

           }

           if(n>=10)

           {

               taka=taka+n/10;

               n=n%10;

           }

           if(n>=5)

           {

               taka=taka+n/5;

               n=n%5;

           }

           if(n>=2)

           {

               taka=taka+n/2;

               n=n%2;

           }

           if(n>=1)

           {

               taka=taka+n/1;

               n=n%1;

           }

           cout<<taka<<endl;

    }

    return 0;

}

 #**Find Factorial**

int main()

{

int arr[201],t,n,i,j,k,T;

cin>>T;

while(T--)

{

    cin>>n;

    for(int p=0;p<=200;p++)

    { arr[p]=0;

    }

    arr[200]=1;

    t=0;

     for(j=1;j<=n;j++)

    {

    for(i=200;i>=0;i--)

    {

      k=arr[i]\*j+t;

      arr[i]=k%10;

    t=k/10;

    }

    }

        int index=0,q=0;

        while(arr[q]==0)

        {

      q++;

        }

    cout<<endl;

    for(int r=q;r<=200;r++)

     { cout<<arr[r];

     }

}

cout<<endl;

return 0;

}

**#Nearest lucky Number ..(like 179 ans is 1 because 180 is lucky number.. 8 is there)**

ll checkeight(ll num)

{

    num=abs(num);

    while(num)

    {

        if(num%10==8)

            return 1;

        num/=10;

    }

    return 0;

}

int main()

{

    ll n;

    cin>>n;

    ll cnt=0;

    ll num=n+1;

    while(1)

    {

       if(checkeight(num))

       {

           ll ans=num-n;

           cout<<ans<<endl;

           return 0;

       }

       else

        num++;

    }

    return 0;

}

**#Maximum Repeated character in a string and its Index**

ll a[1000];  
int main()  
{  
    ll n;  
    cin>>n;  
    string st;  
    cin>>st;  
    ll cnt =0;  
    char temp;  
    for(int i=0;i<n;i++)  
    {  
        a[st[i]]++;  
        if(a[st[i]]>=cnt)  
        {  
            temp = st[i];  
            cnt = a[st[i]];  
        }  
        //cnt=max(a[st[i]], cnt);  
    }  
    //cout<<cnt<<endl;  
    //cout << temp << endl;  
  
    if(n==1)  
        cout<<"Yes"<<endl;  
   else if(cnt>=2)  
        cout<<"Yes"<<endl;  
    else  
        cout<<"No"<<endl;  
  
  
    return 0;  
}

**#Remove 0 before a number in string (0123 ans 123)**

#include<bits/stdc++.h>

#define ll long long

using namespace std;

int main()

{

    ll n;

    cin>>n;

    for(int i=0;i<n;i++)

    {

        string st;

        cin>>st;

        ll len=st.length();

            reverse(st.begin(),st.end());

        ll flag=1;

        for(int i=0;i<len;i++)

        {

            if(flag==1&&st[i]!='0')

                {flag=0;}

            if(flag==0)

            {

                cout<<st[i];

            }

        }

        cout<<endl;

        if(flag)

            cout<<0<<endl;

    }

    return 0;

}

**#Remove 0 before a number(0123 ans 123)**

#include<bits/stdc++.h>

using namespace std;

int main()

{

    string st1,st2;

    cin>>st1>>st2;

    long long len=st1.length();

    int arr[len+5];

    long long res;

    if(st1==st2)

        cout<<0<<endl;

    else

    {

         for(int i=0;i<len;i++)

    {

        res=abs(st1[i]-st2[i]);

        arr[i]=res;

    }

        long long flag=1;

        for(int i=0;i<len;i++)

        {

            if(flag==1&&arr[i]!=0)

            flag=0;

        if(flag!=1)

            {

            printf("%d",arr[i]);

            }

        }

    }

   return 0;

    }

**# (Finding maximum number of char in  a string & counting it)**

int main()

{

    string st;

    int cnt=0;

    cin>>st;

    int flag=0;

    int ll=st.length();

    sort(st.begin(),st.end());

    //cout<<st<<endl;

    int mx=0;

   for(int i=0;i<=ll;i++)

   {

       if(st[i]>mx)

       {

           mx=st[i];

           flag=1;

       }

       else if(st[i]==mx)

        flag++;

   }

   for(int i=0;i<flag;i++)

   {

       //cout<<flag<<endl;

       cout<<char(mx);

   }

1. Prime Factorization

prime(int n)

{

    while (n%2 == 0)

    {

        printf("%d ", 2);

        n = n/2;

    }

    for (int i = 3; i <= sqrt(n); i = i+2)

    {

        while (n%i == 0)

        {

            printf("%d ", i);

            n = n/i;

        }

    }

    if (n > 2)

        printf ("%d ", n);

}

2. sieve\_algorithm:

void SieveOfEratosthenes(int n)

{

    bool prime[n+1];

    memset(prime, true, sizeof(prime));

    for (int p=2; p\*p<=n; p++)

    {

        if (prime[p] == true)

        {

            for (int i=p\*2; i<=n; i += p)

                prime[i] = false;

        }

    }

    // Print all prime numbers

    for (int p=2; p<=n; p++)

       if (prime[p])

          cout << p << " ";

}

void search(char\* pat, char\* txt)

{

    int M = strlen(pat);

    int N = strlen(txt);

    for (int i = 0; i <= N - M; i++) {

        int j;

                for (j = 0; j < M; j++)

            if (txt[i + j] != pat[j])

                break;

        if (j == M)            printf("Pattern found at index %d \n", i);

    }

}

3.DFS:

void find\_max(int s)

{

**int v=0,l=0;int level[100005];**

    queue<int>q;

     q.push(s);

    vis[s]=1;

**level[s]=0;**

    while(!q.empty())

    {

       int d=q.front();

        q.pop();

**if(level[d]%2)**

            v++;

            else

                l++;

        for(int i=0;i<store[d].size();++i)

**{**

            if(!vis[store[d][i]])

            {

                level[store[d][i]]=level[d]+1;

                vis[store[d][i]]=1;

**q.push(store[d][i]);**

          }

        }

  }

    result+=max(l,v);

**}**

4. BFS Algo

class Graph

{

    int numVertices;

    list \*adjLists;

    bool\* visited;

public:

    Graph(int vertices);

    void addEdge(int src, int dest);

    void BFS(int startVertex);

};

Graph::Graph(int vertices)

{

    numVertices = vertices;

    adjLists = new list[vertices];

}

void Graph::addEdge(int src, int dest)

{

    adjLists[src].push\_back(dest);

    adjLists[dest].push\_back(src);

}

void Graph::BFS(int startVertex)

{

    visited = new bool[numVertices];

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

     visited[i] = false;

    list queue;

    visited[startVertex] = true;

    queue.push\_back(startVertex);

    list::iterator i;

    while(!queue.empty())

    {

     int currVertex = queue.front();

     cout << "Visited " << currVertex << " ";

     queue.pop\_front();

     for(i = adjLists[currVertex].begin(); i != adjLists[currVertex].end(); ++i)

     {

         int adjVertex = \*i;

         if(!visited[adjVertex])

         {

             visited[adjVertex] = true;

             queue.push\_back(adjVertex);

         }

     }

    }

}

5.Heaps Algo.

void printArr(int a[],int n)

{

    for (int i=0; i<n; i++)

        cout << a[i] << " ";

    printf("\n");

}

void heapPermutation(int a[], int size, int n)

{

    if (size == 1)

    {

        printArr(a, n);

        return;

    }

    for (int i=0; i<size; i++)

    {

        heapPermutation(a,size-1,n);

        if (size%2==1)

            swap(a[0], a[size-1]);

        else

            swap(a[i], a[size-1]);

    }

}