SEARCHING

Q1) binary search

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
using namespace std;
int solve(vector<ll>&arr,int n,int x){
   ll low = 0, high = n - 1;
          while(low <= high){//base condition
                    int mid = (low + high) / 2;
                    if(arr[mid] == x){
                              return mid;//case1
                    }else if(arr[mid] > x){
                              high = mid - 1; //case2
                    }
                    else{
                              low = mid + 1;//case3
                    }
          }
          return -1;

}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>m;//number to be searched
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
  }
  cout<<solve(v,n,m);
}
```

Q2)check for first occurrence using binary search

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
using namespace std;
int solve(vector<ll>&arr,int n,int x){
   ll low = 0, high = n - 1;
          while(low <= high){
                    ll mid = (low + high) / 2;
                    if(x > arr[mid]){
                              low = mid + 1;
                    }
                    else if(x < arr[mid]){
                              high = mid - 1;
                    }
                    else{
                              if(mid == 0 || arr[mid
- 1] != arr[mid]){
                                        return mid;
                              }
                              else{
                                        high = mid - 1;
                              }
                    }

          }

          return -1;

}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>m;//number to be searched
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
  }
  cout<<solve(v,n,m);

}
```

Q3)index of last occurrence

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
using namespace std;
int solve(vector<ll>&arr,int n,int x){
   ll low = 0, high = n - 1;
          while(low <= high){
                    ll mid = (low + high) / 2;
```

```
                if(x > arr[mid]){
                        low = mid + 1;
                }
                else if(x < arr[mid]){
                        high = mid - 1;
                }
                else{
                        if(mid == n-1 ||
arr[mid + 1] != arr[mid]){
                                return mid;
                        }
                        else{
                                low=mid+1;
                        }
                }

        }

        return -1;

}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>m;//number to be searched
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
  }
  cout<<solve(v,n,m);

}
```

Q4)count occurrence in sorted array

Create function:first occurrence and last occurrence then substract add +1 to it we can count occurrence.

Q5)count 1 in the binary array

quite similar to the index of the first occurrence of the element.

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
using namespace std;
int solve(vector<ll>&arr,int n,int x){
   ll low = 0, high = n - 1;
                while(low <= high){
                ll mid = (low + high) / 2;

                if(arr[mid] == 0){
                        low = mid + 1;
                }
                else{
                        if(mid == 0 || arr[mid
- 1] == 0){
                                return (n -
mid);
                        }
                        else{
                                high = mid -1;
                        }
                }
        }

        return 0;

}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>m;//number to be searched
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
  }
  cout<<solve(v,n,m);

}
```

Q6) Square root

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
```

```
using namespace std;
int solve(vector<ll>&arr,int n,int x){
   ll low = 0, high = x,ans=-1;
                    while(low <= high){
                    ll mid = (low + high) / 2;
                    ll mSq = mid * mid;
                    if(mSq == x){
                            return mid;
                    }
                    else if(mSq > x){
                            high = mid - 1;
                    }
                    else
                    {
                            low = mid + 1;
                            ans = mid;//storing
the nearest answer
                    }
           }

           return ans;

}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>m;//number to be searched
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
  }
  cout<<solve(v,n,m);

}
```

Q7)search in the infinite sized array

Imp for online contests as the array size can be in millions.
we do doubling of the positions and hence apply binary search in that indices that are lying in that range

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
using namespace std;
int bisearch(vector<ll>&arr,int low,int high,int
x){
           while(low <= high){
                    int mid = (low + high) / 2;
                    if(arr[mid] == x){
                            return mid;
                    }else if(arr[mid] > x){
                            high = mid - 1;
                    }
                    else{
                            low = mid + 1;
                    }
           }
           return -1;
}
void solve(vector<ll>arr,ll n,ll x){
   if(arr[0]==x){
      cout<<0;
   }
   //handle the 0 case;
           int i = 1;
           while(arr[i] < x){
                    i = i * 2;
           }
           if(arr[i]==x){
             cout<<i;
           }

           cout<< bisearch(arr, i / 2 + 1, i - 1, x);
}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>m;//number to be searched
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
  }
  solve(v,n,m);
}
```

Q8)search in a rotated array.

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

```
#define ll long long
#define endl  "\n"
using namespace std;
int solve(vector<ll>v,ll n,ll x){
   ll low=0,high=n-1;
   while(low<=high){
      ll mid=(low+high)/2;
      if(v[mid]==x){
         return mid;
      }//upto this normal binary search
      else if(v[mid]>v[low]){
         //left half sorted condition
         if(x>=v[low] && x<v[mid]){
            high=mid-1;
            //it means it lies in the range of left
half
         }else if (v[ low ] == x) {
            return low;
         }else{
            low=mid+1;
            //right half the element lies...
         }
      }
      else{//based on the right half
         if(x>v[mid] && x<=v[high]){
            low=mid+1;
         }else if (v[ high ] == x) {
            return high;
         }else{
            high=mid-1;
         }
      }
   }
   return -1;
}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>m;//number to be searched
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
 }
  cout<<solve(v,n,m);
}
```

Q9)Find the peak element

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
using namespace std;
int solve(vector<ll>arr,ll n){
  int low = 0, high = n - 1;
                  while(low <= high){
                        int mid = (low + high)
/ 2;
                        if((mid == 0 || arr[mid
- 1] <= arr[mid]) &&
                              (mid == n - 1
|| arr[mid + 1] <= arr[mid]))
                              return mid;
                        if(mid > 0 && arr[mid
- 1] >= arr[mid])
                              high = mid -1;
                        else
                              low = mid + 1;
                  }
            return -1;
}
int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T;
   vector<ll>v;
   cin>>n;
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
 }
  cout<<solve(v,n);
}
```

Q10)Two pointers approach

```
#include <bits/stdc++.h>
#define ll long long
#define endl  "\n"
using namespace std;
bool solve(vector<ll>arr,ll n,ll x){
   int i=0, j=n-1; //i=0,j=n-1
   while (i<j){
      if (arr[i] + arr[j] == x){
         return true;
      }
```

```
      else if (arr[i] + arr [j] <x){
         i++;
      }
      else{
      j--;
     }
   }

   return false;
}

int main() {
   ios_base::sync_with_stdio(false);
   cin.tie(NULL);
   cout.tie(0);
   ll n,m,temp1,temp2,T,x;
   vector<ll>v;
   cin>>n;
   cin>>x;//element
    for(ll i=0;i<n;i++){
    cin>>temp1;
    v.push_back(temp1);
 }
  if(solve(v,n,x)){
   cout<<"found";
  }else{
     cout<<"not found";
  }
}
```

Q11)three pointers approach

Here this is just a subset of the two pointers while checking bit by bit and checking the pair on the right half of it …

Q12)median of two sorted array

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

double getMed(vector<ll>a1, vector<ll>a2, ll n1, ll n2){
   if (n1 > n2) {
      swap(a1, a2);
      swap(n1, n2);
   }
         int begin1 = 0, end1 = n1;
         while(begin1 <= end1){
                  int i1 = (begin1 + end1) / 2;
                  int i2 = (n1 + n2 + 1) / 2 - i1;
                  int min1 = (i1 == n1)?INT_MAX:a1[i1];
                  int max1 = (i1 == 0)?INT_MIN:a1[i1 - 1];
                  int min2 = (i2 == n2)?INT_MAX:a2[i2];
                  int max2 = (i2 == 0)?INT_MIN:a2[i2 - 1];
                  if(max1 <= min2 && max2 <= min1){
                           if((n1 + n2) % 2 == 0)
                                    return ((double)max(max1, max2) + min(min1, min2)) / 2;
                           else
                                    return (double)max(max1, max2);
                  }
                  else if(max1 > min2)
                           end1 = i1 - 1;
                  else
                           begin1 = i1 + 1;
         }
         return 0.0;
}
int main(){
   ll a,b,m,n,temp1,temp2;
   cin>>m>>n;
   vector<ll>v1;
   vector<ll>v2;
   for(ll i=0;i<m;i++){
     cin>>temp1;
     v1.push_back(temp1);
   }
   for(ll i=0;i<n;i++){
      cin>>temp2;
      v2.push_back(temp2);
   }
   cout<<getMed(v1,v2,m,n);
}
```

Q13)Repeating element

Through bool expression:

```
#include<bits/stdc++.h>
#define ll long long
using namespace std;
int main(){
  ll a,b,c,m,n,temp;
  vector<ll>v;
```

```
  cin>>n;
  for(ll i=0;i<n;i++){
     cin>>temp;
     v.push_back(temp);
  }
  bool visit[n];
  memset(visit,false,sizeof(visit));
  for(ll i=0;i<n;i++){
     if(visit[v[i]]){
        cout<<v[i];
        break;
     }
     visit[v[i]]=true;
  }

}
```

Through binary search:

DOUBT:

It doesn't pass the test case that's written only looping conditions get passed rest dont

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

int repeat(int arr[], int n){
   int slow = arr[0], fast = arr[0];
   do{
      slow = arr[slow];
      fast = arr[arr[fast]];
   } while (slow != fast);

   fast = arr[0];

   while (slow != fast)
   {
      slow = arr[slow];
      fast = arr[fast];
   }

   return slow;
}

int main()
{
   int arr[] = {1, 3, 3, 4, 6, 7};
   int n = sizeof(arr) / sizeof(arr[0]);

   int result = repeat(arr, n);

   if (result != -1)
   {
      cout << "Repeating element: " << result
<< endl;
   }
   else
   {
      cout << "No repeating element found." <<
endl;
   }

   return 0;
}
```

Q14)Allocate minimum pages

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

bool isFeasible(int arr[],int n,int k, int ans){
   int req=1,sum=0;
   for(int i=0;i<n;i++){
      if(sum+arr[i]>ans){
         req++;
         sum=arr[i];
      }
      else{
         sum+=arr[i];
      }
   }
   return (req<=k);
}

int minPages(int arr[],int n, int k){
   int sum=0,mx=0;
   for(int i=0;i<n;i++){
      sum+=arr[i];
      mx=max(mx,arr[i]);
   }
   int low=mx,high=sum,res=0;

   while(low<=high){
      int mid=(low+high)/2;
      if(isFeasible(arr,n,k,mid)){
         res=mid;
         high=mid-1;
      }else{
         low=mid+1;
      }
   }
```

```
    return res;
}

int main()
{
    int arr[]={10,20,10,30};
    int n=sizeof(arr)/sizeof(arr[0]);
    int k=1;

    cout<<minPages(arr,n,k);
}
```

Questions:

Q17)Minimize the maximum(Codechef-1650)
Q18)Quests(codeforces)
https://codeforces.com/problemset/problem/1760/F

Q19)minions chefs and bananas(codechef-1750)
https://www.codechef.com/practice/course/binary-search/INTBINS01/problems/MINEAT