**Experiment 3.3**

**Student Name: Ayushi Aggarwal UID:21MCA2806**

**Branch: MCA Section/Group8B**

**Semester: 2ND Date of Performance:24-04-2022**

**Subject Name: Computing Aptitude Subject Code:21CAP-654**

1. **Aim/Overview of the practical: Apply the concept of c++**

**Task to be done:**

**Q1.** You are given a stack of N integers such that the first element represents the top of the stack and the last element represents the bottom of the stack. You need to pop at least one element from the stack. At any one moment, you can convert stack into a queue. The bottom of the stack represents the front of the queue. You cannot convert the queue back into a stack. Your task is to remove exactly K elements such that the sum of the K removed elements is maximized.

Input format :

• The first line consists of two space-separated integers N and K.

• The second line consists of N space-separated integers denoting the elements of the stack.

Output format :

• Print the maximum possible sum of the K removed elements

Constraints :

• 1≤N≤105

• 1≤K≤N

• 1≤Ai≤109

**Steps/Commands involved to perform practical:**

#include<stdio.h>

int main(){

long int n,k;

scanf("%ld %ld",&n,&k);

long long int a[n];

long long int sum\_s=0,sum\_l=0,max=0;

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

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

if(i >= (n-k))

sum\_l+=a[i];

}

for(long int i=0;i<k;i++){

sum\_s += a[i];

sum\_l -= a[n-k+i];

if(max < (sum\_s + sum\_l))

  max = sum\_s + sum\_l;

}

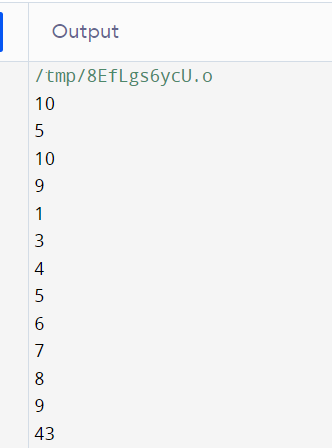
printf("%lld",max);

return 0;

}

**Result/Output/Writing Summary:**

.



**Q2.** The king of ghosts is really disappointed when he sees that all the human beings on Planet Earth have stopped fearing the ghost race. He knows the reason for this. The existing ghost race has become really lazy and has stopped visiting Planet Earth to scare the human race. Hence, he decides to encourage the entire ghost race into scaring the humans by holding a competition. The king, however, never visits 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.

On each day of the competition, ghosts have to visit Planet Earth to scare people. At the end of each day, a "Ghost of the Day" title is awarded to the ghost who scares the most number of humans on that particular day. However, the king of ghosts believes in consistency. Once this title has been given, the ghost who has won the most number of such titles until that particular moment is presented with a "Consistency Trophy". If there are many such ghosts, the oldest among them is given the trophy. Note that this "Title Giving" and "Trophy Giving" happens at the end of each day of the competition.

You will be given the age of the ghost who won the "Ghost of the Day" title on each day of the competition. Your job is to find out the age of the ghost who was awarded with the "Consistency Trophy" on each day of the competition.

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

**Steps/Commands involved to perform practical:**

#include<stdio.h>

#include<stdlib.h>

int age[100010];

int c[100010];

int a[100010];

int b[100010];

int compare(int \*a, int\*b)

{

           return \*a-\*b;

}

int binarysearch(int n,int low,int high)

{

           int mid;

           if (low > high)

                      return -1;

          mid = (low + high)/2;

          if(n == age[mid])

          {

          return mid;

   }

          if(n < age[mid])

   {

                       high = mid - 1;

          return binarysearch(n,low,high);

   }

          if(n > age[mid])

   {

                       low = mid + 1;

          return binarysearch(n,low,high);

   }

}

int main()

{

           int n, m, i, j, max[2];

           scanf("%d%d", &n, &m);

           for(i=0; i<n; i++)

           {

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

                       b[i]=a[i];

           }

           qsort(b, n, sizeof(int), compare);

           age[0]=b[0];

           j=1;

           for(i=1; i<n; i++)

           {

                       if(b[i]!=age[j-1])

                       {

                                   age[j]=b[i];

                                   j++;

                       }

           }

           int M=j;

           for(i=0; i<M; i++)

           {

                       c[i]=0;

           }

           printf("\n");

           max[0]=-1;

           max[1]=-1;

           for(i=0; i<n; i++)

           {

                       int x = binarysearch(a[i], 0, M);

                       c[x]++;

                       if(c[x]>max[0])

                       {

                                   max[0]=c[x];

                                   max[1]=age[x];

                       }

                       else if(c[x]==max[0])

                       {

                                               if(max[1]<age[x])

                                                           max[1]=age[x];

                       }

                       printf("%d %d\n", max[1], max[0]);

           }

           return 0;

}

**Result/Output/Writing Summary:**

