**15B17CI371 – Data Structures Lab**

**ODD 2024**

**Week 3-LAB A**

**Practice Lab**

Q1. Find the first pair of repeating elements in a given array by either returning its

position(index) or a suitable statement if the set is not found or not repeated.

Assumption 1: indexing starts with value 1.

Assumption 2: There are ‘n’ number of elements in the array and you should allow

for it to be dynamically decided .

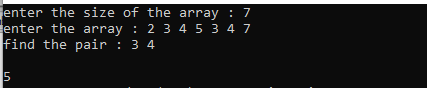
Example:

Input: A= {3,6,12,-10,3,3,6,34,0, -109,98,1}

Find the pair: 3,6

Output:6

Ans :



#include<iostream>

using namespace std;

int main()

{

int n;

cout<<"enter the size of the array : ";

cin>>n;

int \*arr = new int[n];

cout<<"enter the array : ";

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

{

cin>>arr[i];

}

int count=0;

int p1,p2,position;

cout<<"find the pair : ";

cin>>p1>>p2;

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

{

if(count==0)

{

if(arr[i]==p1 && arr[i+1]==p2)

{

count++;

continue;

}

}

if(count==1)

{

if(arr[i]==p1 && arr[i+1]==p2)

{

position = i+1;

}

}

}

cout<<"\n"<<position;

}

Q2.Given an integer array of ‘N’ elements. You need to find the maximum sum of

two elements such that the sum is closest to a given value, say zero (try with

visualizing the number line).

Example:

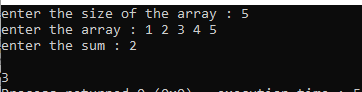
Input: N = 3, arr[] = {-5, -50, 56}

Output: 6

Explanation: Sum of two elements closest to zero is 6.

-55 + (-5) = -55 -5+56 = 51 -50 +56=6

Ans :



#include<iostream>

#include<cmath>

using namespace std;

void close(int \*bar,int a,int n)

{

int mini = bar[0] + bar[1];

int mini2 = abs((bar[0] + bar[1]) - a );

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

{

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

{

int sum = bar[i] + bar[i+1];

int dif = abs(a - sum);

if(dif<mini2)

{

mini = sum;

mini2 = dif;

}

}

}

cout<<"\n"<<mini;

}

int main()

{

int n,a;

cout<<"enter the size of the array : ";

cin>>n;

int \*arr = new int[n];

int \*sum\_arr = new int[n];

cout<<"enter the array : ";

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

{

cin>>arr[i];

}

cout<<"enter the sum : ";

cin>>a;

close(arr,a,n);

}

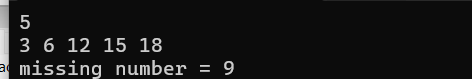
Q3. You are given an array ‘A’, which contains elements arranged in an Arithmetic

Progression. Within this pre-sorted set of elements (A.P.) one of the elements is

missing. Write a program by making a user-defined function called

‘findMissingAPval’ to find this missing element.

Ans :



#include<iostream>

#include<cmath>

using namespace std;

void posi(int \*arr, int n){

int d=arr[1]-arr[0];

int arr2[2];

arr2[0]=1;

arr2[1]=0;

int d2;

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

if((arr[i+1]-arr[i])!=d){

d2=arr[i+1]-arr[i];

arr2[1]++;

}

else arr2[0]++;

}

if(arr2[1]>arr2[0]) d=d2;

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

if((arr[i+1]-arr[i])!=d){

cout<<"missing number = "<<arr[i]+d;

}

}

}

int main(){

int n;

cin>>n;

int \*arr=new int[n];

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

cin>>arr[i];

}

posi(arr,n);

return 0;

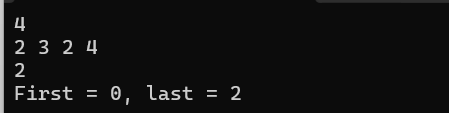
}

Q4. You are given a sorted array ‘A’ with possibly duplicate elements. The task is to

find indexes(positions) of the first and last occurrences of an element ‘x’ in the given

array. Write an efficient program to perform the above requirement.

Ans :



#include<iostream>

#include<cmath>

using namespace std;

void posi(int \*arr, int n, int x)

{ bool f=true,l=true;

int first=-1,last=-1;

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

if(arr[i]==x && f){

first=i;

f=false;

}

if(arr[n-1-i]==x && l){

last=n-1-i;

l=false;

}

}

cout<<"First = "<<first<<", last = "<<last <<endl;

}

int main()

{

int n;

cin>>n;

int \*arr=new int[n];

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

{

cin>>arr[i];

}

int x;

cin>>x;

posi(arr,n,x);

return 0;

}

Q5. Given a sorted array of size N and an integer K, find the position at which K is

present in the array using interpolation search.

Ans :



#include<iostream>

#include<cmath>

using namespace std;

void posi(int \*arr, int n, int x)

{

int lo=0;

int hi=n-1;

int mid=lo+((hi-lo)\*(x-arr[lo]))/(arr[hi]-arr[lo]);

while(lo<=hi)

{

if(arr[mid]==x)

{

cout<<"element present at "<<mid;

return;

}

else if(arr[mid]>x)

{

hi=mid;

}

else

{

lo=mid;

}

}

}

int main()

{

int n;

cin>>n;

int \*arr=new int[n];

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

{

cin>>arr[i];

}

int x;

cin>>x;

posi(arr,n,x);

return 0;

}

Q6. Use Binary search to find the maximum length subarray such that its first

element is greater than or equal to the last element of the subarray in a given array

arr[0..n-1] of length ‘n’.

Example:

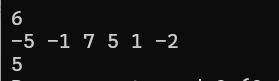
Input : arr[] = {-5, -1, 7, 5, 1, -2}

Output : 5

Hint: Subarray {-1, 7, 5, 1, -2} forms a maximum length subarray with its first

element greater than last.

Ans :

\

#include <iostream>

using namespace std;

bool canFindSubarray(const int arr[], int n, int length)

{

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

{

if (arr[i] >= arr[i + length - 1])

{

return true;

}

}

return false;

}

int findMaxLengthSubarray(const int arr[], int n)

{

int left = 1, right = n;

while (left <= right)

{

int mid = (left + right) / 2;

if (canFindSubarray(arr, n, mid))

{

left = mid + 1;

}

else

{

right = mid - 1;

}

}

return right;

}

int main()

{

int n;

cin>>n;

int \*arr=new int[n];

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

{

cin>>arr[i];

}

cout<<findMaxLengthSubarray(arr,n);

return 0;

}

Q7. Given an array of Strings and a String x, find an index of x if it is present in the

array.

Examples:

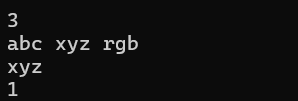
Input: arr[] = {”Hi”, ”Folks”, ”ide”, ”for”,”practice”}, x = ”ide”

Output: 2, The String x is present at index 2.

Input : arr[ ] = {”Hi”, ”Folks”, ”ide”, ”for”, ”practic”}, x = ”zz”

Output: 0 (Hint: substring not present.)

Ans :



#include <iostream>

using namespace std;

int main()

{

int n;

cin>>n;

string \*arr=new string[n];

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

{

cin>>arr[i];

}

string s;

cin>>s;

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

{

if(arr[i]==s)

{

cout<<i;

return 0;

}

}

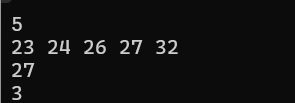
return 0;

}

Q8. Write a program using linear search to check whether the inputted

element belongs to it or not.

Ans :



#include <iostream>

using namespace std;

int main()

{

int n;

cin>>n;

int \*arr=new int[n];

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

{

cin>>arr[i];

}

int s;

cin>>s;

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

{

if(arr[i]==s)

{

cout<<i;

return 0;

}

}

return 0;

}