Name: Valluru Varshini

Class: CSE-F

Reg no:230701369

WEEK 6: COMPETITIVE PROGRAMMING

PROGRAM 1:

AIM: Find Duplicate in Array. Given a read only array of n integers between 1 and n, find one number that repeats.

ALGORITHM:

Step 1: Input the size of the array and the array elements.
Step 2: Sort the array using QuickSort.

Step 3: Search for the first repeated element. Step 4: Output the result.

PROGRAM:

#include<stdio.h> #include<stdlib.h>

```
int compare(const void* a, const void* b)
{
return (*(int*)a-*(int*)b);
}

int main()
{
int n,temp,p; scanf("%d",&n); int a[n];
for (int i=0;i<n;i++)
{
scanf("%d",&a[i]);
}

qsort(a,n,sizeof(int),compare); for(int i=0;i<n;i++)
{
if (a[i]==a[i+1])
{
```

```
p=1;
temp=a[i];
}
}
if (p==1)
{
printf("%d",temp);
}
}
```

OUTPUT:

	Input	Expected	Got	
✔	11 10 9 7 6 5 1 2 3 8 4 7	7	7	✔
✔	5 1 2 3 4 4	4	4	✔
✔	5 1 1 2 3 4	1	1	✔

Passed all tests! ✔

RESULT:

Thus the program executed successfully.

PROGRAM 2:

AIM; Find Duplicate in Array.

Given a read only array of n integers between 1 and n, find one number that repeats.

ALGORITHM:

Step 1: Start

Step 2: Read the value of n from the user and declare an array arr of size n. Step 3: Read the first value into t and assign it to arr[0].

Step 4: Iterate from index 1 to n-1, reading values into arr[i]. If the value of t matches arr[i], break the loop. Otherwise, update t to arr[i].

Step 5: After the loop, print the value of t. Step 6: End

PROGRAM:

```
#include<stdio.h> int main()
{
int n,t; scanf("%d",&n); int arr[n]; scanf("%d",&t); arr[0]=t;
for(int i=1;i<n;i++){ scanf("%d",&arr[i]); if(t==arr[i])
break; else t=arr[i];
}
printf("%d",t);
}
```

OUTPUT:

	Input	Expected	Got	
✔	11 10 9 7 6 5 1 2 3 8 4 7	7	7	✔
✔	5 1 2 3 4 4	4	4	✔
✔	5 1 1 2 3 4	1	1	✔

Passed all tests! ✔

RESULT:

Thus the program executes successfully

.

PROGRAM 3:

AIM: Find the intersection of two sorted arrays.

OR in other words, Given 2 sorted arrays, find all the elements which occur in both the arrays.

ALGORITHM:

Step 1: Start

Step 2: Read the number of test cases, t

Step 3: For each test case, read the sizes n1 and n2 and the elements of the arrays arr1 and arr2

Step 4: For each element in arr1, check if it exists in arr2. If it does, print the element Step 5: End

PROGRAM:

```
#include <stdio.h>


void intersection(int arr1[],int n1,int arr2[],int n2)
{
for (int i=0;i<n1;i++){ int element=arr1[i]; for (int
j=0;j<n2;j++){
if (arr2[j]==element) { printf("%d ",element); break;
}
}
}
printf("\n");
}
int main(){ int t;
scanf("%d",&t);
while(t--){ int n1,n2;
scanf("%d",&n1); int arr1[n1];
for(int i=0;i<n1;i++){ scanf("%d",&arr1[i]);
}
scanf("%d",&n2); int arr2[n2];
```

OUTPUT:

	Input	Expected	Got	
✔	11 10 9 7 6 5 1 2 3 8 4 7	7	7	✔
✔	5 1 2 3 4 4	4	4	✔
✔	5 1 1 2 3 4	1	1	✔

Passed all tests! ✔

RESULT: Thus the program executes successfully.

PROGRAM 4:

AIM:

To execute following program.

ALGORITHM:

Step 1: Start

Step 2: Read the number of test cases, t

Step 3: For each test case, read the sizes n1 and n2, then read elements of the arrays arr1 and arr2

Step 4: Use two pointers to iterate through arr1 and arr2, printing the common elements Step 5: End

PROGRAM:

```
#include <stdio.h>

void intersection(int arr1[], int n1, int arr2[], int n2)
{
int i=0,j=0;
while (i<n1 && j<n2){ if (arr1[i]<arr2[j]){ i++;
}
else if (arr2[j]<arr1[i]){ j++;
}
else{
printf("%d ",arr1[i]); i++;
```

```
j++;
}
}
printf("\n");
}

int main(){ int t;
scanf("%d",&t);
while (t--){ int n1,n2;
scanf("%d", &n1); int arr1[n1];
for (int i=0;i<n1;i++){ scanf("%d",&arr1[i]);
}
scanf("%d",&n2); int arr2[n2];
for (int i=0;i<n2;i++){ scanf("%d", &arr2[i]);
}
intersection(arr1,n1,arr2,n2);
}
}
```

OUTPUT:

	Input	Expected	Got	
✔	1 3 10 17 57 6 2 7 10 15 57 246	10 57	10 57	✔
✔	1 6 1 2 3 4 5 6 2 1 6	1 6	1 6	✔

Passed all tests! ✔

RESULT:

Thus the program executes successfully.

PROGRAM 5:

AIM: Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that A[j] - A[i] = k, i != j.

ALGORITHM:

Step 1: Start

Step 2: Declare n, k and read the input values

Step 3: Create an array arr of size n and read its values

Step 4: Iterate through the array using nested loops to check if there is any pair whose difference is equal to k. If found, return 1.

If no such pair is found, return 0

Step 5: Print the result and end the program

PROGRAM:

```
#include <stdio.h>
```

```
int checkpair(int arr[],int n,int k){ for (int i=0;i<n;i++){
for (int j=i+1;j<n;j++){ if(arr[j]-arr[i]==k){
return 1;
}
else if(arr[j]-arr[i]>k){ break;
}
}
}
return 0;
}

int main(){ int n, k;
scanf("%d", &n); int arr[n];
for (int i=0;i<n;i++) { scanf("%d",&arr[i]);
}
scanf("%d",&k);
int result=checkpair(arr,n,k); printf("%d\n",result);
}
```

OUTPUT:

	Input	Expected	Got	
✔	3 1 3 5 4	1	1	✔
✔	10 1 4 6 8 12 14 15 20 21 25 1	1	1	✔
✔	10 1 2 3 5 11 14 16 24 28 29 0	0	0	✔
✔	10 0 2 3 7 13 14 15 20 24 25 10	1	1	✔

Passed all tests! ✔

RESULT:

Thus the program executes successfully.

PROGRAM 6:

AIM: Given an array A of sorted integers and another non negative integer k, find if there exists 2 indices i and j such that A[j] - A[i] = k, i != j.

ALGORITHM:

Step 1: Start

Step 2: Declare n, k and read the input values

Step 3: Create an array arr of size n and read its values

Step 4: Use two pointers i and j to iterate through the array, checking if the difference between arr[j] and arr[i] is equal to k.

Adjust the pointers based on the value of the difference
Step 5: Print the result

PROGRAM:

```
#include <stdio.h>

int checkpair(int arr[],int n,int k){ int i=0,j=1;
while(j<n){
int diff=arr[j]-arr[i]; if (diff==k && i!=j){
return 1;
}
else if(diff<k){ j++;
}
else{
i++;
}
if(i==j){ j++;
}
}
return 0;
}

int main(){ int n,k;
```

```
scanf("%d",&n); int arr[n];
for (int i=0;i<n;i++){
scanf("%d",&arr[i]);
}
scanf("%d",&k);
int result=checkpair(arr,n,k); printf("%d\n",result);
}
```

OUTPUT:

	Input	Expected	Got	
✔	3 1 3 5 4	1	1	✔
✔	10 1 4 6 8 12 14 15 20 21 25 1	1	1	✔
✔	10 1 2 3 5 11 14 16 24 28 29 0	0	0	✔
✔	10 0 2 3 7 13 14 15 20 24 25 10	1	1	✔

Passed all tests! ✔

RESULT:

Thus the program executes successfully.