1 (b) CTS Test 1

Test Summary

- No. of Sections: 1
- No. of Questions: 7
- Total Duration: 20 min

Section 1 - Automata Fix

Section Summary

- No. of Questions: 7
- Duration: 20 min

Additional Instructions:

None

Q1. The function **findMaxElement(int arr1[] ,int len1,int arr2[],int len2)** accepts two integer arrays arr1,arr2 of length len1,len2 respectively.
It is supposed to return the largest element in both the input arrays.
Another function **sortArray(int *arr,int len)** sorts the input array arr of length len in ascending order and returns the sorted array.
Your task is to use **sortArray(int *arr,int len)** function and complete the code in **findMaxElement(int arr1[],int len1,int arr2[],int len2)** so that it passes all test cases.

Sample Input

```
5 6
1 5 4 2 6
1 7 2 6 8 9
```

Sample Output

```
9
```

Time Limit: - ms Memory Limit: - kb Code Size: - kb

Q2. The function **matrixsum(int *matrix,int row,int col)** is supposed to return the sum of elements of the input array matrix having **row** number of rows and **col** number of columns. complete the function/method **matrixsum(int* matrix, int row, int col)** to get the desired output.

Sample Input

```
3 3
1 2 3
4 5 6
7 8 9
```

Sample Output

```
45
```

Time Limit: 2 ms Memory Limit: 256 kb Code Size: 256 kb

Q3. The Function **arrayReverse(int *arr,int len)** accepts an array arr of length len(len >=0) as an argument.The function is expected to reverse the elements of the input array in-place.
For example, if the input array arr is {20,30,10,40,50} the function is expected to return{50,40,10,30,20}
The function compiles successfully but fails to return the desired result due to logical errors

```
int arrayReverse(int *arr,int len){
int i,temp,originallen=len;
for(i=0;i<originallen;i++){
temp=arr[len-1];
arr[len-1]=arr[i];
arr[i]=temp;
len-=1;
}
Return arr;
}
```

Input Format

First Line of input contains the integer **n** - size of the array
Second line contains n - space seperated integers

Sample Input

Sample Output

```
8
1 2 3 4 5 6 7 8
```

```
8 7 6 5 4 3 2 1
```

Time Limit: 2 ms Memory Limit: 256 kb Code Size: 256 kb

Q4. The function/method **countElement(int arr[],int size,int num)** is supposed to return the number of elements in the input array arr which are greater than twice the input number **num.** complete the function/method **countElement(int arr[],int size,int num)** to give the desired output.

Input Format

First line of input consist of integer **size** - size of the array arr
Second line contains size number of space seperated integers
Third line contains the integer **num**

Sample Input

```
5
1 2 3 4 5
2
```

Sample Output

```
1
```

Time Limit: 2 ms Memory Limit: 256 kb Code Size: 256 kb

Q5. This method **countOccurance (int arr[], int val, int n)** is supposed to return the count of occurrences of a number value in the input array arr. The function compiles successfully but fails to return the desired result due to logical errors.

Note :
val - the value for which we require frequency
n - refers to the size of the array

Your task is to debug the program to pass all test cases.

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

Sample Input

```
5
1 2 2 2 3
2
```

Sample Output

```
3
```

Time Limit: 2 ms Memory Limit: 256 kb Code Size: 256 kb

Q6. In this challenge , You need to complete the provided function **printChars(int n)** to print the following pattern.

For Example:

Input : 5

Output :

a

ab
abc
abcd
abcde

Input Format

A Single integer **'N'**

Output Format

Print the above matrix in **'N'** Lines

Constraints

1 <= N <=26

Sample Input

```
6
```

Sample Output

```
a
ab
abc
abcd
```

Time Limit: 2 ms Memory Limit: 256 kb Code Size: 256 kb

Q7. The function **patternPrint(int n)** supposed to print n numberof lines in the following pattern
For n=4 the pattern should be:
1
1 1
1 1 1
 1 1 1 1
The function complies successfully but fails to return the desired results due to logical errors
Your task is to debug the program to pass all the testcases

PROGRAM:

```
#include<stdio.h>
void patternPrint(int n)
{

   for(int i=0;i<n;i++){
       for(int j=0;j<n;j++){
           printf("1");
       }
       printf("\n");
   }
}
int main()
{
   int n;
   scanf("%d",&n);
   patternPrint(n);
}
```

Sample Input

```
5
```

Sample Output

```
1
11
111
1111
```

Time Limit: 2 ms Memory Limit: 256 kb Code Size: 256 kb

Answer Key & Solution

Section 1 - Automata Fix

Q1

Test Case

Input

```
10 6
1 8 5 7 2 4 12 8 9 11
3 5 7 12 98 89
```

Output

```
98
```

Weightage - 25

Input

```
5 5
1 2 3 4 5
5 4 3 2 6
```

Output

```
6
```

Weightage - 25

Input

```
5 9
8 99 89 90 11
78 999 888 77 666 545 67 54 23
```

Output

```
999
```

Weightage - 50

Sample Input

```
5 6
1 5 4 2 6
1 7 2 6 8 9
```

Sample Output

```
9
```

Solution

Header

```
#include<stdio.h>
#include<stdlib.h>
int * sortArray(int *arr, int length)
{
int x=0,y=0,n=length;
for(x=0;x<n;x++)
{
int index_of_min = x;
for(y=x;y<n;y++)
{
if(arr[index_of_min]>arr[y])
{
index_of_min=y;
}
}
int temp=arr[x];
arr[x]=arr[index_of_min];
arr[index_of_min]=temp;
}
return  arr;

}
void findMaxElement(int arr1[], int arr2[], int len1, int len2){
```

```
void findMaxElement(int arr1[],int arr2[],int len1,int len2){



    int index;
    sortArray(arr1,len1);
    sortArray(arr2, len2);
    int max=0;
    if(arr1[len1-1]>arr2[len2-1]){
        max=arr1[len1-1];
    }
    else{
        max=arr2[len2-1];
    }
   printf("%d",max);
```

Footer

```
}
int main()
{
    int len1,len2;
    scanf("%d %d",&len1,&len2);
    int arr1[len1],arr2[len2];
    for(int i=0;i<len1;i++){
        scanf("%d",&arr1[i]);
    }
    for(int i=0;i<len2;i++){
        scanf("%d",&arr2[i]);
    }
    findMaxElement(arr1,arr2,len1,len2);
    return 0;
}
```

Q2

Test Case

Input

```
2 2
1 4
6 8
```

Output

```
19
```

Weightage - 25

Input

```
5 5
1 2 3 67 67
90 87 65 43 21
1 2 3 4 5
```

Output

```
511
```

Weightage - 75

Sample Input

```
3 3
1 2 3
4 5 6
7 8 9
```

Sample Output

```
45
```

Solution

Header

```
#include<stdio.h>
#include<stdlib.h>
#define SIZE 100
int matrixsum(int* arr,int row,int col)
{




    int sum=0;
    for (int i = 0; i < row; i++)
      for (int j = 0; j < col; j++)
        sum=sum+ *(arr + i*col + j);
    return sum;
```

Footer

```
}
int main()
{
    int row,col;
    scanf("%d %d",&row,&col);
    int *arr = (int *)malloc(row * col * sizeof(int));
    int i, j, count = 0;
    for (i = 0; i <row; i++)
      for (j = 0; j<col; j++)
         scanf("%d",(arr + i*col + j));

    printf("%d",matrixsum(arr,row,col));
}
```

Q3

Test Case

Input

```
10
4 9 4 7 5 9 3 6 0 8
```

Output

```
8 0 6 3 9 5 7 4 9 4
```

Weightage - 25

Input

```
3
1 1 1
```

Output

```
1 1 1
```

Weightage - 25

Input

```
5
10 56 76 89 45
```

Output

```
45 89 76 56 10
```

Weightage - 25

Input

```
10
12 13 14 16 17 18 19 23 25 27
```

Output

```
27 25 23 19 18 17 16 14 13 12
```

Weightage - 25

Sample Input

```
8
1 2 3 4 5 6 7 8
```

Sample Output

```
8 7 6 5 4 3 2 1
```

Solution

Header

```
#include<stdio.h>
int* arrayReverse(int *arr,int len){




int i,temp,originallen=len;
for(i=0;i<originallen/2;i++){
temp=arr[len-1];
arr[len-1]=arr[i];
arr[i]=temp;
len-=1;
}
return arr;
```

Footer

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

Q4

Test Case

Input

```
10
11 12 13 14 15 16 17 18 19 20
8
```

Output

```
4
```

Weightage - 50

Input

```
6
12  12  13  14  23
6
```

Output

```
6
```

Weightage - 50

Sample Input

```
5
1  2  3  4  5
2
```

Sample Output

```
1
```

Solution

Header

```
#include<stdio.h>
int countElement(int arr[],int size,int num)
{



 int count=0;
    for(int i=0;i<size;i++)
    {
        if(arr[i]>2*num)
        {
            count++;
        }
    }
    return count;
```

Footer

```
}
int main()
{
    int size;
    scanf("%d",&size);
    int arr[size];
    for(int i=0;i<size;i++)
    {
        scanf("%d ",&arr[i]);
    }
    int num;
    scanf("%d",&num);
    printf("%d",countElement(arr,size,num));
}
```

Q5

Test Case

Input

```
10
1  2  9  0  87  87  56  5  78  23
87
```

Output

```
2
```

Weightage - 25

Input

```
5
1 1 1 1 1
9
```

Output

```
0
```

Weightage - 50

Input

```
1
1
1
```

Output

```
1
```

Weightage - 25

Sample Input

```
5
1 2 2 2 3
2
```

Sample Output

```
3
```

Solution

Header

```
#include<stdio.h>
int countOccurences(int arr[],int val,int n)
{




    int count=0;
    for(int i=0;i<n;i++)
    {
        if(arr[i]==val){
            count++;
        }
    }
    return count;
```

Footer

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

    printf("%d",countOccurences(arr,val,n));
}
```

}

Q6

Test Case

Input

```
26
```

Output

```
a
ab
abc
abcd
```

Weightage - 50

Input

```
18
```

Output

```
a
ab
abc
abcd
```

Weightage - 25

Input

```
14
```

Output

```
a
ab
abc
abcd
```

Weightage - 25

Sample Input

```
6
```

Sample Output

```
a
ab
abc
abcd
```

Solution

Header

```
import java.io.*;
import java.util.Scanner;
class Main
{
    public static void printChars(int n)
    {
```

Header

```
#include<stdio.h>
void printcharacterpattern( int num)
     {
```

```
import java.io.*;
import java.util.Scanner;
class Main
{
    public static void printChars(int n)
    {

        int i, j;
        char num='a';
        for(i=0; i<n; i++)
        {
            num='a';
            for(j=0; j<=i; j++)
            {
```

```
#include<stdio.h>
void printcharacterpattern( int num)
     {
         int i,j;
         char s;
        for(i=0;i<num;i++)
        {
            s='a';
            for(j=0;j<=i;j++)
            {
                 printf("%c", s++);

            }
             printf("\n");
        System.out.print(num);
        num++;
      }
      System.out.println();
    }
  }
  public static void main(String args[])
  {
    int n;
    Scanner myObj = new Scanner(System.in);
    n = myObj.nextInt();
    printChars(n);
  }
}
```

Footer

```
  }
  public static void main(String args[])
  {
    int n;
    Scanner myObj = new Scanner(System.in);
    n = myObj.nextInt();
    printChars(n);
  }
}
```

```
        }

    }
int main()
{
    int n;
    scanf("%d",&n);
    printcharacterpattern(n);
    return 0;
}
```

Footer

```
 }
int main()
{
    int n;
    scanf("%d",&n);
    printcharacterpattern(n);
    return 0;
}
```

Q7

Test Case

Input

```
6
```

Output

```
1
11
111
1111
```

Weightage - 50

Input

```
20
```

Output

```
1
11
111
1111
```

Weightage - 50

Sample Input

```
5
```

Sample Output

```
1
11
111
1111
```

Solution

Header

```
#include<stdio.h>
void patternPrint(int n)
{
    for(int i=0;i<n;i++){
        for(int j=0;j<=i;j++){
            printf("1");
        }
        printf("\n");
    }
```

Footer

```
}
int main()
{
    int n;
    scanf("%d",&n);
    patternPrint(n);
}
```