CHITKARA UNIVERSITY

2023-2024

ETE Sample Coding Questions – March, 2023

Roll No: ……………… [Total No. of Pages:]

Programme: B.E. (CSE) **Time: 90 minutes**

Course Title: Problem Solving using C

Course Code: **Max. Marks: 60**

General Instructions:

o Follow the instructions given in each section.

Section – C

(Each question carries 5 marks)

Q1. Write a C++ Program to Count Number of Words in a String.

Take string size 100.

Input Format

Take Input string

Output Format

Print Number of Words in a String.

Sample Input

All the best

Sample Output

3

Code

```
#include<stdio.h>
#include<string.h>
int main()
{
char str[100], countw=0, i;
fgets(str,100,stdin);

int len=strlen(str);
for(i=0; i<len; i++)
{
if(str[i]==' ')
{
countw++;
}
}
printf("%d\n",countw+1);
return 0;
}
```

Q2. Given an array print the maximum value.

Input Format

First line contains integer n as size of array. Next n lines contains a single integer as element of array.

Constraints

N cannot be Negative.

Range of Numbers can be between -1000000000 to 1000000000

Output Format

Print the required output.

Sample Input

4

2

8

6

4

Sample Output

8

Explanation

Arrays= {2, 8, 6, 4} => Max value = 8 .

```
#include<stdio.h>
int maximum(int arr[],int n)
{
long int max= -1000000000;

for(int i=0;i<n;i++)
{
if(arr[i]>max)
{
max=arr[i];
}
}
return max;
}

int main() {
int arr[100000];
long int n;
scanf("%d",&n);
for(int i=0;i<n;i++)
{
scanf("%d",&arr[i]);
}
printf("%d",maximum(arr,n));
return 0;
}
```

Q3. Given a string, count the number of Vowels, Consonants, Digits, and Spaces present in a string.

NOTE: String length will not exceed 100.

Input format : First line take string input

Output format : Print the count in the following order :

vowels

Consonants

Digits

Spaces

Sample Input :

```
Ram Got 100 Marks in Hindi
```

Sample Output :

```
6
12
3
5
```

Explanation :

In given input : Ram Got 100 Marks in Hindi

Vowels : 6 - a o a i i i

Consonants : 12 - R m G t M r k s n H n d

Digits : 3 - 1 0 0

Spaces : 5 – Each word is separated by 1 space.

Code

```
#include<stdio.h>
#include<string.h>
void freqcount(char *str)
{
int cv=0, cc=0, cd=0, cs=0;

for(int i=0;str[i]!=0;i++)
{
if(str[i]=='a' ||str[i]=='A' || str[i]=='e' ||str[i]=='E' || str[i]=='i' ||str[i]=='I' ||
str[i]=='o' ||str[i]=='O' || str[i]=='u' ||str[i]=='U')
{
cv++;
}
else if(str[i]>='a' && str[i]<='z' || str[i]>='A' && str[i]<='Z')
{
cc++;
}
else if(str[i]==' ')
{
cs++;
}
else if(str[i]>='0' && str[i]<='9')
{
cd++;
}
}
printf("%d\n%d\n%d\n%d\n",cv,cc,cd,cs);
}

int main()
{
char str[100];
fgets(str,100,stdin);
freqcount(str);
return 0;
}
```

Q4. Write a recursive function which can find Nth Fibonacci number.

Input format : First line take input – N

Output format : Print the Nth Fibonacci number.

Sample Input

6

Sample output

8

Explanation

Fibonacci Numbers are : 0 1 1 2 3 5 8 13 …….

Hence 6th Fibonacci number is - 8

Code :

```
#include<stdio.h>
int fibno(long int n)
{
if(n==0 || n==1)
{
return n;
}
return fibno(n-1) + fibno(n-2);
}
int main()
{
long int n;
scanf("%ld",&n);
printf("%ld\n",fibno(n));
return 0;
}
```

Q.5 You are given two strings word1 and word2. Merge the strings by adding letters in alternating order, starting with word1. If a string is longer than the other, append the additional letters onto the end of the merged string.

Return the merged string.

Input format
First string
Second string
Output format
Merged output string

Sample Input 1:
abc pqr
Sample Output 1:
apbqcr
Explanation: The merged string will be merged as so:
word1: abc word2:   pqr merged: apbqcr

Sample Input 2:
ab pqrs
Sample Output 2: apbqrs

Code :-

```
#include<stdio.h>

#include<string.h> #include<stdlib.h> char *
mergeAlternately(char * word1, char * word2){     char *result =
(char *)calloc(strlen(word1) + strlen(word2) + 1, sizeof(char));
int id1 = 0, id2 = 0, idx = 0;     while(id1 < strlen(word1) || id2 <
strlen(word2))
   {

  if(id1 < strlen(word1))
result[idx++] =
word1[id1++];

if(id2 < strlen(word2))
result[idx++] =
word2[id2++];

   }

   return result;

}



int main()

{

 char s1[100],s2[100];
scanf("%s",s1);
scanf("%s",s2);
printf("%s",mergeAlternat
ely(s1,s2));
}
```

Q.6 You are given an integer array nums. The unique elements of an array are the elements that appear exactly once in the array.

Return the sum of all the unique elements of nums.

Constraint: All the numbers in the array lie between 1 to 100.

Input format

An integer N

N elements of array

Output format

Sum of all unique elements

Sample Input 1:

```
4
1 2 3 2
```

Sample Output 1:

```
4
```

Explanation: The unique elements are [1,3], and the sum is 4.

Sample Input 2:

```
5
1 1 1 1 1
```

Sample Output 2:

```
0
```

```
#include <stdio.h> int
sumOfUnique(int*
nums, int n) {


 int uni[101] = {0};

    int i, j;

    int sum = 0;



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

 uni[nums[i-1]]++;

 for(j = 0; j < 101; j++)

            if(uni[j] == 1)

                        sum += j;

 return sum;

}



int main() {

    int n;

    scanf("%d",&n);

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

Q7 Write a program in C to calculate sum of every row and column in a 2D matrix.

Q.8 Given an circular array which represent a id of your company's customers you need to decrypt
That data by replacing the number with sum of next 3 numbers

Input format

An integer N

N elements of array

Output format

Resultant array

Sample Input 1:

5

1 2 3 4 5

Sample Output 1:

9 12 10 8 6

Explanation :-

(2+3+4) (3+4+5) (4+5+1) (5+1+2) (1+2+3)

9 12 10 8 6

Code :-

```
#include <stdio.h>

int
mai
n()
{
    int n;

scanf("d"
,&n);
int arr[n];
    int ans[n];

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

    for(int
i=0;i<n;i++)
ans[i]=0;


    for(int i=0;i<n;i++)
    {         int
sum=0;
for(int
j=1;j<=3;j+
+)
sum+=arr[(i
+j)%n];

        ans[i] = sum;
    }

    for(int i=0;i<n;i++)
    printf("%d ",ans[i]);


}
```

Q9 Write a recursive function which can find Nth Fibonacci number.

Input format : First line take input – N

Output format : Print the Nth Fibonacci number.

Sample Input

6

Sample output

8

Code :

```
#include<stdio.h>
int fibno(long int n)
{
if(n==0 || n==1)
{
return n;
}
return fibno(n-1) + fibno(n-2);
}
int main()
{
long int n;
scanf("%ld",&n);
printf("%ld\n",fibno(n));
return 0;
}
```

Q10. Write a function named **countsetbits** that can take an integer value as an argument and return a number of 1s(set bits) in the binary representation of a given integer value.

Input Format

Input N = Number of Test Cases, followed by N numbers.

Output Format

Number of 1s(set bits) in each number in a new line.

Sample Input

```
3

5

4

15
```

Sample Output

```
2

1

4
```

Explanation :

The binary of 5 is 1001 hence number of 1s(set bits) = 2.
The binary of 4 is 1000 hence number of 1s (set bits) = 1.

The binary of 15 is 1111 hence number of 1s (set bits)= 4.

Code:

```
#include<stdio.h>
int countsetbits(int num)
{
int count=0;
while(num>0)
{
if(num%2==1)
{
count++;
}
num=num/2;
}
return count;
}

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

Try solving this using bitwise operators.

Q11 Write a C program which can convert Lowercase to Uppercase of given string.

NOTE: String length will not exceed limit of 100.

Input format : First line take string input

Output format : Print the converted string

Sample Input 1 :

raamMohan

Sample Output 1 :

RAAMMOHAN

Sample Input 2 :

raamMohan

Sample Output 2 :

RAAMMOHAN

```
#include<stdio.h>
void uppercase(char str[])
{
int i=0;
while(str[i]!='\0')
{
if((str[i]>='a' && str[i]<='z'))
{
str[i]=str[i]-32;
}
i++;
}
}

int main()
{
char str[100];
fgets(str,100,stdin);
uppercase(str);
printf("%s\n",str);
return 0;
}
```

Q12 Take N as input. Print the sum of its odd placed digits and sum of its even placed digits.

Constraints

0 < N <= 1000000000

Sample input

3526

Sample Output

11

5

Explanation

6 is present at 1st position, 2 is present at 2nd position, 5 is present at 3rd position and 3 is present at 4th position.

Sum of odd placed digits on first line. 6 and 5 are placed at odd position. Hence odd place sum is 6+5=11

Sum of even placed digits on second line. 2 and 3 are placed at even position. Hence even place sum is 2+3=5

Code :

```
#include<stdio.h>
int main()
{
int n, esum=0, osum=0 ,i=1;;
scanf("%d",&n);
while(n>0)
{
if(i%2==0)
{
esum=esum+n%10;
}
else
{
osum=osum+n%10;
}
n=n/10;
i++;
}
printf("%d\n%d\n",osum,esum);
return 0;
}
```

Q13 You have just graduated from college and decided to start your own business.

You decided to be a trader of iron rods. The iron rods of specific lengths have

fixed price in the market. Given the prices of rods of different length from l unit

to n units. You have to find the maximum profit that can be earned by cutting

rod into pieces of different length.

Input Format:

First line will take an integer n i.e. length of the rod

Next n lines will take input n elements of the array i.e. prices of rod of length 1 to n

Output Format:

Print the optimal profit that can be earned

Input :

```
4
1 1 1 6
```

Output : 6

Explanation : It can be seen that performing no cuts and taking the entire rod as a whole can lead to the optimal answer of 6.

```
#include <stdio.h>
#include<limits.h>
int max(int a, int b)
{
    return a>b ? a : b;
}
int maxProfit(int* prices,int n)
{
    if(n<=0)
    return 0;

    int ans=INT_MIN;

    for(int i=1;i<=n;i++)
    {
        int currentProfit=prices[i-1]+maxProfit(prices,n-i);
        ans=max(ans,currentProfit);
    }

    return ans;
}
int main() {
    int n;
    scanf("%d",&n);
    int prices[n];
    for(int i=0;i<n;i++)
    scanf("%d",&prices[i]);


    printf("%d",maxProfit(prices,n));


}
```

Q14 Given a string, check if it is a palindrome.

Q15 Given an array, print the maximum sum of any subarray (kadane’s Algorithm)

Q16 https://leetcode.com/problems/kth-missing-positive-number/description/

Q17 James want to climb n steps from ground level. At each step he can either take a jump of 1 step or a jump of 2 steps or a jump of 3 steps. Find the total number of in which James can reach the nth step.

Input : 2
Output: 2
Explanation : James can reach the 2nd step in 2 ways if at each step he can take a jump of 1,2 or 3. In this reach 2nd step by the following ways step 0 to step 2 directly or 0 to 1 then step 1 to step 2

Input: 3
Output: 4
Explanation : James can reach the 3rd step in 4 ways if at each step he can take a jump of 1,2 or 3.

Input: 4
Output: 7
Explanation : James can reach the 4th step in 7 ways if at each step he can take a jump of 1,2 or 3.

Code :

```
#include<stdio.h>
int ways(int n)
{       if(n==0)
        return 1;
    if(n<0)
    return 0;
    return ways(n-1)+ways(n-2)+ways(n-3);
} int main(){
    int n;
    scanf("%d",&n);     printf("%d",ways(n));
}
```

Q18
https://hack.codingblocks.com/app/contests/3942/200/problem

Q19

https://hack.codingblocks.com/app/contests/3942/380/problem

Q1 Given two strings, check whether they are anagrams or not. Two strings are said to be anagrams if they contain same characters.

Sample input : Take input (string1 and string2) from each player in a new line.

Sample Output : Print Yes or No according to the condition.

Sample Input 1:

cat

tac

SampleOutput 1:

Yes

Explanation :

cat and tac

Both the strings have the same set of characters.

Code

```
#include<stdio.h>
#include <stdbool.h>
#include<string.h>
bool isAnagram(char * s, char * t){

   int freq[26]={0};
```

```
    int n=strlen(s);

    if(strlen(s)!=strlen(t))
    return false;

    for(int i=0;i<n;i++)
    {
        char ch=s[i];
        int idx=ch-'a';
        freq[idx]++;
    }


    for(int i=0;i<n;i++)
    {
        char ch=t[i];
        int idx=ch-'a';
        freq[idx]--;
    }

    for(int i=0;i<26;i++)
    {
        if(freq[i]!=0)
        return false;
    }

    return true;

}


int main()
{
    char str1[100],str2[100];

         scanf(“%s”,str1);
         scanf(“%s”,str2);
    if(isAnagram(str1,str2))
        printf("Yes");
    else
    printf("No");
}
```

Q2 https://leetcode.com/problems/minimum-time-to-type-word-using-special-typewriter/

Q3 The n-queens puzzle is the problem of placing n queens on an n x n chessboard such that no two queens attack each other.
Given an integer n, count all the *distinct solutions to the* n-queens puzzle.
Print the total number of solutions possible.

Input : 3

Output : 0

Input : 4

Output : 2

Input : 5

Output : 10

Code:

```
#include <stdio.h>
#include<stdbool.h>
// to check whether we can place n queens in a nXn board or not.

bool isSafe(int board[][10],int x,int y, int n)
{
```

```
// check whether x,y cell is safe or not

// current col

// col
for(int i=x-1;i>=0;i--)
{
   if(board[i][y]==1)
   return false;
}

// left diagonal
int i=x-1;
int j=y-1;

while( i>=0 && j>=0 )
{
   if(board[i][j]==1)
      return false;
   i--;
   j--;
}

// right diagonal
i=x-1;
j=y+1;
```

```
    while(i>=0 && j<n)
    {
        if(board[i][j]==1)
            return false;


        i--;
        j++;
    }


    return true;


}


void printMatrix(int mat[][10],int n)
{
    for (int i = 0; i < n; ++i)
    {
        for (int j= 0; j < n; ++j)
        {
            printf("%d ",mat[i][j]);
        }
        printf("\n");
    }


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

```
int cnt=0;
bool nqueen(int board[][10],int currRow,int n)
{
    // base conditions
    if(currRow==n)
    {
        cnt++;
        return false;
    }



    // Standing Currently Row currRow->0 to currRow->3

    for(int i=0;i<n;i++)
    {
        // traversing current row

        if(isSafe(board,currRow,i,n))
        {
            board[currRow][i]=1;
            bool canPlaceRemainingQueens=nqueen(board,currRow+1,n);


            if(canPlaceRemainingQueens)
                return true;
            board[currRow][i]=0;
        }



    }


    return false;
}
int main()
{
    int board[10][10]={0};
    int n;
    scanf("%d",&n);

    nqueen(board,0,n);

    printf("%d",cnt);

}
```