FACQ TCS Level1 Set2 Coding Solutions

Level-1 Set-2 : Easy to Medium complexity

#	Topic	Frequency
1	Convert base-17 value into Decimal equivalent	27 (similar)
2	Prepend zeroes while printing the numbers from m to n	23
3	Given a maximum of 100 digit numbers as input, find the difference between the sum of odd and even position digits	23
4	Case: Program to display the total number of mints with all the kids	
5	Find out whether the sum of the digits of the given positive integer number N is UNO or not	20
6	Case: Man invests money in bank… Display the date as on 183rd day	
7	Case: Find difference between adjacent digits… task here to find given number is CORRECT or INCORRECT.	
8	Program to remove numbers (ex: 3 and 97) from given string	17
9	Find the frequency of each digit in the given string	17
10	Program to find the count of numbers that consists of unique digits.	16
	Scenario-based questions	Refer to “FACQ TCS Level2 Coding” document

---

L1S2_tcs1_base17_decimal.c

```
/*
Convert base-17 value into Decimal equivalent
Given a maximum of four digit to the base 17(10 -> A, 11 -> B, 12 -> C, 16
-> G) as input,
output its decimal value.

Input: 35FD
Output: 16451
*/
#include <stdio.h>
#include <math.h>
#include <string.h>
int main(){
  char hex[17];
  long decimal;
  int i = 0, val, len;
  decimal = 0;
  scanf("%s",&hex);
  len = strlen(hex);
  len--;

  for(i = 0;hex[i]!='\0';i++)
  {
    if(hex[i]>='0'&& hex[i]<='9'){
      val = hex[i] - 48;
    }
    else if(hex[i]>='a'&& hex[i]<='g'){
      val = hex[i] - 97 + 10;
    }
    else if(hex[i]>='A'&& hex[i]<='G'){
      val = hex[i] - 65 + 10;
    }
    //printf("\n Len: %d Pow: %lf",len,pow(17,len));
    decimal = decimal + val * pow(17,len);
    len--;
  }
  printf(" %ld",decimal);
  return 0;    }
```

L1S2_tcs2_prepend_0s.c

```
/*
Given a pair of positive integers m and n (m < n; 0 < m < 999; 1 < n < =
999)
Write a program to prepend zeroes while printing the numbers from m to n.

Example-1
Input
3 11
Expected output
03 04 05 06 07 08 09 10 11

Example-2
Input
91 101
Expected output
091 092 093 094 095 096 097 098 099 100 101

Example-3
Input
1 7
*/

#include <stdio.h>
int main()
{
    int up,low;
    scanf("%d %d",&low,&up);
    for(int i=low; i<=up; i++)
    {
        if(up>=100)
            printf("%03d ",i);
        else if(up>=10)
            printf("%02d ",i);
        else
            printf("%d ",i);
    }
    return 0;
}
```

L1S2_tcs3_diff_sum_odd_even

```
/*
Given a maximum of 100 digit numbers as input, find the difference between
the sum of odd and even position digits.

Input 1:
4567
Expected output:
2

Explanation
The Sum of odd position digits 4 and 6 is 10. The Sum of even position
digits
5 and 7 is 12. The difference is 12-10=2.

Input #2:
9834698765123

*/

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

int main()
{
    int a = 0,b = 0,i = 0, n;
    char num[100];

    scanf("%s",&num);
    n = strlen(num);
    while(n>0)
    {
        if(i==0)
        {
            a+=num[n-1]-48;
            n--;
            i=1;
        }
```

```
        else
        {
            b+=num[n-1]-48;
            n--;
            i=0;
        }
    }
    printf("%d",abs(a-b));

    return 0;
}
```

L1S2_tcs4_mints_in_queue.c

```
/*
It was one of the places, where people need to get their provisions only
through
fair price (“ration”) shops. As the elder had domestic and official work
to attend to,
their wards were asked to buy the items from these shops. Needless to say,
there was a long queue of boys and girls. To minimize the tedium of
standing in the
serpentine queue, the kids were given mints. I went to the last boy in the
queue and
asked him how many mints he has. He said that the number of mints he has
is
one less than the sum of all the mints of kids standing before him in the
queue.
So I went to the penultimate kid to know how many mints she has.

She said that if I add all the mints of kids before her and subtract one
from it,
the result equals the mints she has. It seemed to be a uniform response
from everyone.
So, I went to the boy at the head of the queue consoling myself that
he would not give the same response as others. He said, “I have four
mints”.
```

```
Given the number of first kid's mints (n) and the length (len) of the
queue as input,
write a program to display the total number of mints with all the kids.

constraints:
2<n<10
1<len<20

Input#1:
4 2
Output:
7

Input#2:
14 4
Output:
105
*/

#include<string.h>
#include<stdio.h>

int main()
{
    int s,n;
    scanf("%d %d", &s, &n);
    int sum=s,prev;
    for(int i=1;i<n;i++)
    {
        prev=sum-1;
        sum+=prev;
    }
    printf("\n%d ",sum);
}
```

L1S2_tcs5_sum_digits_uno.c

```
/*
Let us find out whether the sum of the digits of the given positive
integer number
N is UNO or not. Given a positive integer number N, reduce the number of
digits of
N by computing the sum of all the digits to get a new number. If this new
number exceeds 9,
 then sum the digits of this new number to get another number and continue
this way until
 a single digit value is obtained as the 'digit sum'. The task here is to
find out whether
 the result of the digit sum done this way is '1' or not.

Input:1234 => 1+2+3+4 => 10 => 1+0 => 1
Output:UNO

Input:25631
Output:NOT UNO

*/
#include <stdio.h>

void main()
{
    int n,r,sum=0;
    scanf("%d",&n);
    while(n>0)
    {
        r=n%10;
        sum=sum+r;
        n=n/10;
    }
    if(sum%9==1)
        printf("UNO");
    else
        printf("NOT UNO");
}
```

L1S2_tcs6_date2invest.c

```
/*
A man invests a certain amount on monthly basis in a bank. He withdraws
that money once in 4 years which is a leap year, to make a big scale
purchase .He starts next investment exactly 183 days after the purchase .

Initially, he makes a note of his purchase date
Given the date(dd) and month(mm) of his purchase. The task here is to help
him find the date and month to start his investment.

His next investment date is calculated from the next day of his purchase.
Display the date as on 183rd day.
*/

#include <stdio.h>
void getDate(int d, char m)
{
    int days[] = {31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    char month[12] = {1,2,3,4,5,6,7,8,9,10,11,12};
    int count = 183;
    int current_month = 0;
    for(int i = 0; i < 12; i++)
      if (m == month[i])
            current_month = i;
    int current_date = d;

    while (1)
         {
              while (count > 0 && current_date <= days[current_month])
              {
               count -= 1;
                current_date += 1;
              }
              if (count == 0)
              break;
              current_month = (current_month + 1) % 12;
              current_date = 1;
         }
    printf("\nDate to Invest dy/mo:
%d/%d",current_date,month[current_month]);
```

```
    }

int main(char args[])
    {
        int D;
        char M;
        printf("Enter D: "); scanf("%d",&D);
        printf("\nEnter M: "); scanf("%d",&M);
        getDate(D, M);
        return 0;
    }
```

L1S2_tcs7_diff_adjacent_digits.cpp

```
/*
Find difference between adjacent digits
Kids up to the age of 7 are confused formation of letters and numbers,
teacher uses the different methodologies to make the concepts of
mathematics clear to the students. One of the methods the teacher uses to
emphasis on the addition and recognition of numbers is that she muddles up
the numbers randomly and then asks the students to find difference between
adjacent digits. The task here to find given number is CORRECT or
INCORRECT.
*/
//C++
#include <iostream>
#include <bits/stdc++.h>

using namespace std;

int main()
{
    int number;
    cin>>number;
    number=abs(number);

    int remainder1=number%10;
    number=number/10;
    while(number>0)
    {
```

```
        int remainder2=number%10;
        if(abs(remainder1-remainder2)==1)
        { remainder1=remainder2;
            number=number/10;
        }
        else
            break;
    }
    if(number>0)
        cout<<"INCORRECT";
    else
        cout<<"CORRECT";
    return 0;
}
```

```
//C  WIP - Terminating at run time. Use C++ verson.
#include<stdio.h>
#include<stdlib.h>

int main()
{
    int number;
    scanf("%d",number);
    number=abs(number);

    int remainder1=number%10;
    number=number/10;
    while(number>0)
    {
        int remainder2=number%10;
        if(abs(remainder1-remainder2)==1)
        { remainder1=remainder2;
            number=number/10;
        }
        else
            break;
    }
    if(number>0)
        printf("INCORRECT");
```

```
    else
        printf("CORRECT");
    return 0;
}
```

L1S2_tcs8_remove_nums_in_string.c

```
/* Write a program to remove 3 and 97 from given string.
Ex:   String='This3isa97correction'
      Output: 'Thisisacorrection'
      String='This3isa9corr7ection'
      Output: 'Thisisa9corr7ection'
*/
#include <stdio.h>
#include <string.h>
int main()
{    //char c[100] ="This3isa97correction";
    char c[100];
    printf("Enter a string including numbers 3 9 7: "); scanf("%s",&c);
    int n=strlen(c);
    int i;
        for(i=0;i<n;i++)
        {
            if(c[i]=='3')
            {
                c[i] = '\0' ;
            }
            if(c[i]=='9' && c[i+1]=='7')
            {
                c[i] = '\0' ;
                c[i+1]='\0';
            }
        }
        for(i=0;i<n;i++)
        {
           printf("%c",c[i]);
        }
    return 0;   }
```

L1S2_tcs9_freq_digits_in_string.c

```c
/*
Given a string, consisting of alphabets and digits,
find the frequency of each digit in the given string.

Constraint: All English alphabets & decimal numeric digits
Output: Print ten space-separated integers in a single line denoting the
frequency of each digit
Sample:
This99is111aPro0000gram
4 3 0 0 0 0 0 0 0 2
*/
#include<stdio.h>
#include<string.h>

int main ()
{
  char a[1000], freq[256] = {0};
  int i, n, j, count = 0;
  scanf("%[^\n]s", a);    //input from user
  n = strlen(a);          // finding string length
  char ch = '0';
  for (i = 0; i < 10; i++)
    {
          for (j = 0; j < n; j++)
          {
            if (a[j] == ch)
            {
              count++;
            }
          }
      printf("%d ", count);
      count = 0;
      ch++;
    }

  return 0;
}
```

L1S2_tcs10_unique_digits.c

```c
/*
Write a program to find the count of numbers that consists of unique
digits.
Input: Two integers - Lower & Upper range
Output: Count of unique digits in the range

*/

#include<stdio.h>
#define true 1
#define false 0

void printUnique(int l, int r)
{
    int count=0;
    for (int i=l ; i<=r ; i++)
    {
        int num = i;
        int visited[10] = {false};

        while (num)
        {

            if (visited[num % 10])
                { //printf("\n %d Break",num%10);
                break; }

            visited[num%10] = true;
            num = num/10;
        }

        if (num == 0)
            count++;
    }
    if(count>0)
     printf("\nCount of Unique numbers: %d",count);
    else
        printf("\nNo Unique Number");
```

```
}

int main()
{
    int l,r;
    scanf("%d %d",&l,&r);
    printUnique(l, r);
    return 0;
}
```
