TCS Previous year Coding (2019) Solution

(For Study material and To read my blogs do Follow just entering your mail at https://technext21.wordpress.com)

TCS NQT 2019 | Set 1

Problems Statement

One programming language has the following keywords that cannot be used as identifiers:

break, case, continue, default, defer, else, for, func, goto, if, map, range, return, struct, type, var

Write a program to find if the given word is a keyword or not

SOLUTION (in C)

```
#include<stdio.h>
#include<string.h>
int main()
{
	char a[30];
	scanf("%s",&a);
//compare the sting with all keywords

if((strcmp(a,"defer")==0)||(strcmp(a,"if")==0)||(strcmp(a,"for")==0)||(strcmp(a,"ret
urn")==0)||(strcmp(a,"break")==0)||(strcmp(a,"case")==0)||(strcmp(a,"continue")=
=0)||(strcmp(a,"default")==0)||(strcmp(a,"else")==0)||(strcmp(a,"func")==0)||(strc
mp(a,"goto")==0)||(strcmp(a,"map")==0)||(strcmp(a,"range")==0)||(strcmp(a,"stru
ct")==0)||(strcmp(a,"type")==0)||(strcmp(a,"var")==0))
        {
                printf("%s is a keyword",a);
        }
        else                                    //print when keyword not matching
        {
                printf("%s is not a keyword",a);
        }
}
```

TCS NQT 2019 | Set 2

Problems Statement

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

Logic:

- ◆ Get the input as string because we need up to 100 digits.
- ◆ Then convert the char value to integer value.
- ◆ Add odd position values and even position values separately.
- ◆ Find the difference sing abs() function and print it.

SOLUTION (in C)

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int main()
{
    int a=0,b=0,i=0,n;
    char num[100];
    printf("Enter the number:");
    scanf("%s",&num);       //get the input up to 100 digit
    n=strlen(num);
    while(n>0)
    {
        if(i==0)                //add even digits when no of digit is even and vise versa
        {
            a+=num[n-1]-48;
            n--;
            i=1;
        }
        else                    //add odd digits when no of digit is even and vise versa
        {
            b+=num[n-1]-48;
            n--;
            i=0;
        }
    }
    printf("%d",abs(a-b)); //print the difference of odd and even
}
```

TCS NQT 2019 | Set 3

Problems Statement

Given a maximum of four digit to the base 17 (10 – A, 11 – B, 12 – C, 13 – D … 16 – G} as input, output its decimal value.

Logic:

- Get the input as string because it contains both integer and character.
- Then assign A=10,B=11………. G=16.
- Otherwise convert character to integer.

- Apply the rule for any number to decimal conversion.
- Then print the decimal value.

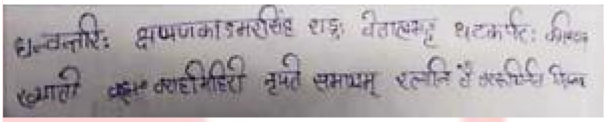
SOLUTION (in C)

```
#include<stdio.h>
#include<string.h>
int pow(int a,int n)
{
        int b=a;
        if(n==0)                //return 1 when n value is 0
        {
                return 1;
        }
        while(n>1)          //calculate the power value
        {
                a*=b;
                n--;
        }
        return a;
}
int    main()
{
        char a[50];
        int n,i,j=0,decimal=0;
        printf("Enter the base 17 number:");
        scanf("%s",&a);                                    //get the input as string
        n=strlen(a);
        while(n>0)
        {
            if(a[n-1]>64&&a[n-1]<72)           //check the char between A to G
                {
                    i=a[n-1]-55;                        //convert A=10,b=11 ..... G=16
                        decimal+=i*pow(17,j);         //calculate the decimal value
                        n--,j++;
                }
                else                                              //check for number 0 to 9
                {
                    i=a[n-1]-48;                    //convert char value to int
                    decimal+=i*pow(17,j);
                    n--,j++;
                }
        }
        printf("%d\n",decimal);                  // print the converted decimal
value
}
```

TCS NQT 2019 |Set 4

Problems Statement

Our hoary culture had several great persons since time immemorial and king vikramaditya’s nava ratnas (nine gems) belongs to this ilk.They are named in the following shloka:



धन्वन्तरिः क्षपणकाऽमरसिंह शङ्कु वेतालभट्ट घटकर्पर: कालि[illegible]
ख्यातो [illegible] वराहमिहिरो नृपते सभायाम् रत्नानि वै वररुचिर् [illegible]

Among these, Varahamihira was an astrologer of eminence and his book Brihat Jataak is recokened as the ultimate authority in astrology. He was once talking with Amarasimha,another gem among the nava ratnas and the author of Sanskrit thesaurus, Amarakosha. Amarasimha wanted to know the final position of a person, who starts from the origin 0 0 and travels per following scheme.

He first turns and travels 10 units of distance

His second turn is upward for 20 units

Third turn is to the left for 30 units

Fourth turn is the downward for 40 units

Fifth turn is to the right(again) for 50 units

… And thus he travels, every time increasing the travel distance by 10 units.

While Varahamitra could use his astrology skills to predict movement based on planetary positions, use your programming expertise to print the final position, given the number of turns(n);2<=n<=1000

SOLUTION:

```
#include<stdio.h>
int main()
{
        int x=10,y=20,d=0,n,i;
        scanf("%d",&n);
        for(i=2;i<n;i++)
        {
                d=10+i*10;              //each time extra 10 units are added
                if(i%2==0)
                {
                        if(x<0)
                        {
                                x=d+x;          //positive x axis direction
                        }
                        else
                        {
                                x=x-d;          //negative x axis direction
                        }
                }
                else
                {
                        if(y<0)
                        {
                                y=d+y;          //positive y axis direction
                        }
```

```
            else
            {
                y=y-d;      //negative y axis direction
            }
        }
    }
    printf("%d %d",x,y);
}
```