LAB 6

Subject: programming fundamental

Submitted to: Dr. Kashif Bilal

Submitted by: **KASHIF KHAN**

Student Reg#: **FA22-BSE-O68**

Date: 16.04.2023

Q1: Ask user to enter 10 numbers and find the largest, smallest, Average, and Sum of the entered numbers.

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

int main()
{
    int number, sum=0, count = 1;
    // store max value in smaller and min value in larger
    int larger=INT_MIN, smaller = INT_MAX;
    float average;
    while(count <= 10)
    {
        printf("Enter Number %d : ", count);
        scanf("%d",&number);
        if(larger < number)
            larger = number;
        if(smaller > number)
            smaller = number;
        sum += number;
        count++;
    }
    average = sum/10.0;
    printf("The Larger Number is = %d\n",larger);
    printf("The Smaller Number is = %d\n",smaller);
    printf("The Sum is = %d\n",sum);
    printf("The Average is = %.2f\n",average);
}
```

Q2: Write a program to print all prime numbers from 1 to 300.

```c
#include <stdio.h>
int main()
{
    // for range
    int rangeStart, rangeEnd;
    printf("PleaseEnter the Strating number form which you want to print prime Number : ");
    scanf("%d",&rangeStart);
    printf("Please Enter the Ending number upto which you want to print prime Number : ");
    scanf("%d",&rangeEnd);
    // when user enter the smaller number In start and large in End
    if(rangeStart > rangeEnd){
        printf("Please Enter the  Ending number again, Ending Number must be greater than
Strating Number : ");
        scanf("%d",&rangeEnd);
    }
    // for prime number
    int num, isPrime;
    while (rangeStart<=rangeEnd)
    {
        // Initializing a isPrime with zero to checke prime
        isPrime = 0;
        // checke one by one number so num is rangeStart and it increment with iteration
        num = rangeStart;
        for (int i = 2; i <= num/2; i++)
        {
            if(num % i == 0)
            {
                isPrime++;
                break;
            }
        }
        if(isPrime == 0)
            printf("%d\t",num);

        rangeStart++;
    }
}
```

Q3: Write a program in C to display the n terms of harmonic series and their sum. 1 + 1/2 + 1/3 + 1/4 + 1/5 ... 1/n terms .

```c
#include <stdio.h>
int main()
{
    int nthTerm, count = 1;
    printf("Enter the nth term to print Harmonic series and it sum:  ");
    scanf("%d",&nthTerm);
    double sum;
    while (count <= nthTerm)
    {
        printf("1/%d + ",count);
        sum += 1.0/count;
        count++;
    }
    printf("\n");
    printf("The sum upto nth term of Harmonic series is %.3lf \n",sum);
}
```

Q4: Ask user to enter a decimal number and convert it to binary number format.

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

int main()
{
    int num, binary = 0;
    printf("Please Enter the Decimal Number to convert to Binary Number: ");
    scanf("%d",&num);
    // sotre the orignal number
    int tempNum = num;
    int power = 0;

    while (num > 0)
    {
        int remainder = num % 2;
        binary +=  remainder * (pow(10,power));
        num /= 2;
        power++;
    }
    printf("The Binary of %d is = %d", tempNum, binary);
}
```

Q5: Ask user to enter a binary number and show its decimal value.

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

int main()
{
    int num, decimal = 0;
    printf("Please Enter the Binary Number to Convert to Decimal Number : ");
    scanf("%d",&num);
    // sotre the orignal number
    int tempNum = num;
    int power = 0;

    while (num > 0)
    {
        int remainder = num % 10;
        decimal +=  remainder * (pow(2,power));
        num /= 10;
        power++;
    }
    printf("The decimal of %d is = %d", tempNum, decimal);
}
```

Q6: Write a program that uses for statements to print the following patterns separately, one below the other. Use for loops to generate the patterns. All asterisks (*) should be printed by a single statement of the form cout << '*'; (this causes the asterisks to print side by side). [Hint: The last two patterns require that each line begin with an appropriate number of blanks. Extra credit: Combine your code from the four separate problems into a single program that prints all four patterns side by side by making clever use of nested for loops.]

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

```c
int main()
{
    int row;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d", &row);

    printf("Patter of Question 6 a\n\n\n");
    for (int i = 1; i <= row; i++)
    {
        for (int j = 1; j <= i; j++)
        {
            printf("*");
        }
        printf("\n");
    }

    printf("\n\n\nPatter of Question 6 b\n\n\n");
    for (int i = row; i >= 1; i--)
    {
        // for star
        for (int j = 1; j <= i; j++)
        {
            printf("*");
        }

        printf("\n");
    }
    printf("\n\n\nPatter of Question 6 c \n\n\n");
    for (int i = 1; i <= row; i++)
    {
        for (int j = 1; j <= i; j++)
        {
            printf(" ");
        }
        for (int j = 1; j <= row - i; j++)
        {
            printf("*");
        }
        printf("\n");
    }
    printf("\n\n\nPatter of Question 6 d\n\n\n");
    for (int i = row; i >= 1; i--)
    {
        for (int j = 1; j <= i; j++)
        {
            printf(" ");
        }
        for (int j = 1; j <= row - i; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

Q7: Write a program to print (up to n. Here n=5)
1
12
123
1234
12345

```
#include <stdio.h>

int main()
{
    int row;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d",&row);
    for (int i = 1; i <= row; i++)
    {
        for(int j=1; j<=i; j++)
        {
            printf("%d",j);
        }
        printf("\n");
    }
}
```

Q8: Write a program to print a pyramid

```
#include <stdio.h>

int main()
{
    int row;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d",&row);

    for(int i=1; i<=row; i++)
    {
        // space
        for(int j=1; j<=row-i; j++)
        {
            printf(" ");
        }
        // print odd star
        for(int j=1; j<=2*i-1; j++)
        {
            printf("*");
        }
        printf("\n");

    }
}
```

Q9: Write a program to print Number pyramid.

```
#include <stdio.h>

int main()
{
    int row;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d",&row);

    for(int i=1; i<=row; i++)
    {
        // spaces
        for(int j=1; j<=row-i; j++)
        {
            printf(" ");
        }
        // print number
        for(int j=1; j<=2*i-1; j++)
        {
            printf("%d",j);
        }
        printf("\n");

    }
}
```

Q10: Write a program to print Diamond

```
#include <stdio.h>

int main()
{
    int row;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d",&row);

    // upper part
    for(int i=1; i<=row; i++)
    {
        // space
        for(int j=1; j<=row-i; j++)
        {
            printf(" ");
        }
        // print odd number star
        for(int j=1; j<=2*i-1; j++)
        {
            printf("*");
        }
        printf("\n");

    }
    // lower part
    for(int i=row; i>=1; i--)
    {
        // space
        for(int j=1; j<=row-i; j++)
        {
            printf(" ");
        }
        // print odd number star
        for(int j=1; j<=2*i-1; j++)
        {
            printf("*");
        }
        printf("\n");

    }
}
```

Q11: (Pythagorean Triples) A right triangle can have sides that are all integers. A set of three integer values for the sides of a right triangle is called a Pythagorean triple. These three sides must satisfy the relationship that the sum of the squares of two of the sides is equal to the square of the hypotenuse. Find all Pythagorean triples for side1, side2 and hypotenuse all no larger than 500. Use a triple-nested for loop that tries all possibilities. ($a_2 + b_2 = c_2$)

```
#include<stdio.h>

int main(){
    int sideA, sideB, sideC;
    int lowerLimit, upperLimit;
    printf("Enter the starting number: ");
    scanf("%d",&lowerLimit);
    printf("Enter the Ending number: ");
    scanf("%d",&upperLimit);
    // Check if the starting number is greater than ending number
    if(lowerLimit > upperLimit)
    {
        printf("Please Enter the Ending number again Ending Number Must be Greater than
Starting: ");
        scanf("%d",&upperLimit);
    }
    // For loop for side A
    for(sideA=lowerLimit; sideA * sideA<= upperLimit; sideA++)
    {
        //  For loop for side B
        for(sideB=lowerLimit; sideB * sideB <= upperLimit; sideB++)
        {
            // For loop for side C
            for(sideC= lowerLimit; sideC * sideC <= upperLimit; sideC++)
            {
                // Check if the sum of square of side A and side B is equal to square of side
C
                if(sideA * sideA + sideB * sideB == sideC * sideC)
                {
                    printf("Side A %d + Side B %d = Side C %d\n",sideA, sideB, sideC);
                }
            }
        }
    }
}
```

Q12: Write a program in C to print.

1
01
101
0101
10101

```
#include <stdio.h>

int main()
{
    int row;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d",&row);
    for (int i = 1; i <= row; i++)
    {
        for(int j=1; j<=i; j++)
        {
            if((i+j) % 2 == 0)
                printf("1 ");
            else
                printf("0 ");
        }
        printf("\n");
    }
}
```

Q13: Write a program to find the sum of following series.

11!+ 12!+ 13!+ 14!+ 15!+ 16!

Write the above program for n number (e.g., n=6 in above case)

```
#include <stdio.h>

int main()
{
     int nthTerm, count = 1, factorial, number;
     double sum = 0;
    printf("Enter the nth term to print Harmonic series and it sum:  ");
    scanf("%d",&nthTerm);
    while(count <= nthTerm)
    {
        printf("1/%d! + ",count);
        factorial = 1;
        number = count;
        for(int i=number; i>=1;i--)
        {
            factorial *= i;
        }
        sum += 1.0/factorial;
        count++;
    }
    printf("\nThe sum is = %.3lf ",sum);
}
```

Q14: Write a program to generate all combinations of 1, 2 and 3 using for loop. (e.g., 111, 112, 113, 121, …..)

```
#include <stdio.h>

int main()
{
    // Print all possible combinations of 1, 2, 3
    for (int i = 1; i <= 3; i++)
    {
        for (int j = 1; j <= 3; j++)
        {
            for (int k = 1; k <= 3; k++)
            {
                printf("%d%d%d, ",i, j, k);
            }
        }
    }
}
```

Q15:

Write a program to produce the following output:

```c
#include <stdio.h>

int main()
{
    int row;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d",&row);
    int toPrint = 1;
    for(int i=1; i<= row; i++)
    {
        for(int j=1; j<=row - i; j++)
        {
            printf("  ");
        }
        for (int  j = 1; j <=i; j++)
        {
            printf("%d   ",toPrint++);

        }
        printf("\n");

    }
}
```

Q16: the entries in Pascal's Triangle are called the *binomial coefficients*. The rows are represented by **n** and columns by **k. Always start with n=0 and k=0.**

There's a pretty simple formula for figuring out the binomial coefficients:

n!

[n:k] = --------

k! (n-k)!

6 * 5 * 4 * 3 * 2 * 1

For example, [6:3] = ------------------------ = 20.

3 * 2 * 1 * 3 * 2 * 1

Write a program to produce the following output:

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

```c
#include <stdio.h>
// Function to find factorial
int findFactorial(int number){
    int fact  = 1;
    for(int i=1; i<=number; i++){
        fact *= i;
    }
    return fact;
}
int main()
{
    int row, toPrint;
    printf("Enter the Number of Row you want To print: ");
    scanf("%d",&row);

    for(int i=0; i<=row; i++)
    {
        // for spaces
        for(int j=0; j<= row -i; j++)
        {
            printf("  ");
        }
        // for Pascal's Triangle
        for(int j=0; j<=i; j++)
        {     // Formula for Pascal's Triangle Using Factorial function
            toPrint = findFactorial(i)/(findFactorial(j)*findFactorial(i-j));
            printf("%d   ",toPrint);
        }
        printf("\n");
    }
}
```

Q 17: The natural logarithm can be approximated by the following series.

$$\frac{x-1}{x}+\frac{1}{2}\left(\frac{x-1}{x}\right)^2+\frac{1}{2}\left(\frac{x-1}{x}\right)^3+\frac{1}{2}\left(\frac{x-1}{x}\right)^4+\ldots.$$

If **x** is input through the keyboard, write a program to calculate the sum of first seven terms of this series.

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

int main()
{
    int number, count=1, power=1, nthTerm;
    double sum=0,term;
    printf("Enter the Value of X : ");
    scanf("%d",&number);
    printf("Enter the Value of nth term  : ");
    scanf("%d",&nthTerm);

    while (count<=nthTerm)
    {
        //
        term =(double) (number-1)/number;
        if(power == 1)
            sum += term;
        else
            sum += (0.5) * (pow(term,power));

        power++;
        count++;
    }
    printf("The sum is = %lf \n",sum);


}
```

Q18: Check whether a number is a Strong Number or not. A number is strong when sum of factorial of its digits is equal to number itself. E.g., 145 (1! + 4! + 5! = 145)

```
#include <stdio.h>
int main()
{
    int number, tempNumber, strongNumber = 0;
    printf("Please Enter the Number: ");
    scanf("%d",&number);
    tempNumber = number;

    while(number != 0)
    {
        int fact = 1;
        int remindar = number % 10;
        // find factorial of the obtained reminder which is the last digit of the number
        for(int i= remindar; i>=1; i--)
            fact *= i;

        strongNumber += fact;
        number /= 10;
    }

    if(tempNumber == strongNumber)
        printf("%d is Strong Number\n",tempNumber);
    else
        printf("%d is Not Strong Number\n",tempNumber);

}
```

Q19: The greatest common divisor of two positive integers, n and m, is the largest number, d, which divides evenly into both n and m. There are several algorithms that can be used to solve this problem, including: Initialize d to the smaller of m and n. Keep decreasing d, until you find a number that divides both of the numbers.

```c
#include <stdio.h>

int main()
{
    int number1, number2, smaller, gcd;
    printf("Please Enter Number 1 : ");
    scanf("%d",&number1);
    printf("Please Enter Number 2 : ");
    scanf("%d",&number2);
    // store the smaller number in smaller of num1 and num2
    if(number1 < number2)
        smaller = number1;
    else
        smaller = number2;
    /*find the GCD of num1 and num2 using the smaller number
    and decrementing it until we find the both numbers are divisible by it.
    */
    for(int i=smaller; i>=1; i--)
    {
        if(number1 % i == 0 && number2 % i == 0){
            gcd = i;
            break;
        }
    }
    printf("The GCD is %d", gcd);
}
```

Q20: Find GCD from the conventional method.

```c
#include <stdio.h>

int main()
{
    int number1, number2;
    printf("Please Enter Number 1 : ");
    scanf("%d",&number1);
    printf("Please Enter Number 2 : ");
    scanf("%d",&number2);
    // sotre for later use
    int tempNum1, tempNum2;
    tempNum1 = number1;
    tempNum2 = number2;
    /*finding GCD using Euclidean Algorithm */
    while(number2 != 0){
        int tempNum = number2;
        number2 = number1 % number2;
        number1 = tempNum;
    }
    printf("The GCD of %d and %d is %d",tempNum1, tempNum2, number1);
}
```

Q21: Find LCM of two numbers using conventional method.

```c
#include <stdio.h>
int main()
{
    int number1, number2, greater, lcm = 1;
    printf("Please Enter Number 1 : ");
    scanf("%d",&number1);
    printf("Please Enter Number 2 : ");
    scanf("%d",&number2);
    // sotre for later use
    int tempNum1, tempNum2;
    tempNum1 = number1;
    tempNum2 = number2;
    //
    while(number1 != 1 || number2 != 1)
    {
        // find the greater number and store it in greater variable
        greater = number1 > number2 ? number1 : number2;
        // iterate from 2 to greater number
        for(int i=2; i<= greater; i++){
            // check if i is divisible by both or number1 or number2
            if(number1 % i == 0 || number2 % i == 0){
                lcm *= i;
                // if number1 is divisible by i then divide it by i
                if(number1 % i == 0)
                    number1 /= i;
                // if number2 is divisible by i then divide it by i
                if(number2 % i == 0)
                    number2 /= i;
            }
        }
    }
    printf("The LCM of %d and %d is %d",tempNum1, tempNum2, lcm);

}
```

Q22: Find LCM by using algorithm: to get the least common multiple of two numbers is to multiply them and divide the result by their greatest common divisor.

```c
#include <stdio.h>

int main()
{
    int number1, number2,lcm;
    printf("Please Enter Number 1 : ");
    scanf("%d",&number1);
    printf("Please Enter Number 2 : ");
    scanf("%d",&number2);
    // sotre for later use
    int tempNum1, tempNum2;
    tempNum1 = number1;
    tempNum2 = number2;
    // use Euclidean Algorithm to find GCD
    while(number2 !=0)
    {
        int tempNum = number2;
        number2 = number1 % number2;
        number1 = tempNum;
    }
    // use GCD to find LCM becacuse LCM = (num1 * num2) / GCD
    lcm = (tempNum1 * tempNum2) / number1;
    printf("The LCM of %d and %d is %d",tempNum1, tempNum2, lcm);
}
```