Raunak Gayen
2020EEB066

1. Write a C program to generate set of all prime numbers between 1 and *n*, where the number *n* has to be provided by the user.

```c
#include<stdio.h>
int main()
{
    int n,flag=0;
    printf("Enter any number\n");
    scanf("%d",&n);

    printf("The prime numbers upto %d are---\n\n",n);
    for(int i=2;i<=n;i++)
    {
        flag=0;
        for(int j=2;j<i;j++)
        {
            if(i%j==0)
            {
                flag=1;
                break;
            }
        }
        if(flag==0)
            printf("%d ",i);
    }
    return 0;
}
```

Output:

```
D:\compAssignment>gcc prime.c

D:\compAssignment>a.exe
Enter any number
20
The prime numbers upto 20 are---

2 3 5 7 11 13 17 19
```

2.1 Write a C program to find out the GCD of two numbers by repeated subtraction.

```c
#include<stdio.h>

int main()
{
    int num1, num2;
    printf("Enter 2 positive integer numbers\n");
    scanf("%d%d", &num1, &num2);
    num1 = (num1 < 0) ? -num1 : num1;
    num2 = (num2 < 0) ? -num2 : num2;
    printf("\nGreatest Common Divisor of %d and %d is ", num1, num2);
    while(num1 != num2)
    {
        if(num1 > num2)
        {
            num1 = num1 - num2;
        }
        else
        {
            num2 = num2 - num1;
        }
    }
    printf("%d.\n", num1);
    return 0;
}
```

Output:

```
D:\compAssignment>gcc gcd.c

D:\compAssignment>a.exe
Enter 2 positive integer numbers
9 3

Greatest Common Divisor of 9 and 3 is 3.
```

2.2 Write a C program to find out the GCD of two numbers by repeated division.

```c
#include<stdio.h>

int main()
{
    int n1, n2, i, gcd;

    printf("Enter two integers: ");
    scanf("%d %d", &n1, &n2);

    for(i=1; i <= n1 && i <= n2; ++i)
    {
        // Checks if i is factor of both integers
        if(n1%i==0 && n2%i==0)
            gcd = i;
    }

    printf("G.C.D of %d and %d is %d", n1, n2, gcd);

    return 0;
}
```

Output:

```
D:\compAssignment>gcc gcd2.c

D:\compAssignment>a.exe
Enter two integers: 8 2
G.C.D of 8 and 2 is 2
```

3. Write a C program to find the factorial of a number N using an iterative method. N should be taken as input from the user. Test with $N > 16$ and explain why are you getting wrong results.

```c
#include<stdio.h>

int main()
{
    int num,fact=1,i;

    printf("Enter any number\n");
    scanf("%d",&num);

    for(i=num; i>=1; i--)
    {
        fact=fact*i;
    }

    printf("Factorial of %d is %d\n",num,fact);

    return 0;
}
```

Output:

```
D:\compAssignment\assignemt 2>gcc factorial.c

D:\compAssignment\assignemt 2>a.exe
Enter any number
6
Factorial of 6 is 720
```

```
Enter any number
17
Factorial of 17 is -288522240
```

The anomalous answer for n>16 is due to the fact that integer data type is of 4 bytes on a 64-bit machine (like mine) which has a range of -2,147,483,648 to 2,147,483,647 and 16! = 2.092279e+13 which is within the range but 17! = 3.5568743e+14 which is beyond the range of int. So due to wrapping of data types, the answer comes in round circle and hence the answer is negative as if the end of the positive side is connected with the negative side. To avoid this error we can use long int(8 bytes) or long long int(8 bytes) according to our need of domain.

4. Write a C program to find the roots of a quadratic equation where the coefficients are entered by the user. Solve the quadratic equation for real and imaginary roots.

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

int main()
{
    int a , b , c , d ;
    float root1 = 0.0 , root2 = 0.0 , real , img ;

    printf("Enter the coefficients of quadratic equation : ");
    scanf("%d %d %d", &a , &b , &c );

    d = b*b - 4*a*c;

    if(d > 0)
    {
        printf("Roots are real and unequal\n");
        root1 = (float)(-b + sqrt(d))/(2*a);
        root2 = (float)(-b - sqrt(d))/(2*a);
        printf("The first root = %f , Second root = %f \n", root1 , root2 );
    }
    if( d == 0 )
    {
        printf("Roots are real and equal\n");
        root1 = root2 = (float)-b/(2*a);
        printf("The equal roots are %f and %f \n", root1 , root2 );
    }
    if( d < 0 )
    {
        printf("Roots are imaginary\n");
        real = (float)-b/(2*a);
        img = sqrt(-d)/(2*a);
        printf("The first root = %f + i%f \n", real , img );
        printf("The first root = %f - i%f \n", real , img );
    }
    return 0;
}
```

Output:

```
D:\compAssignment\assignemt 2>gcc quadratic.c

D:\compAssignment\assignemt 2>a.exe
Enter the coefficients of quadratic equation : 1 5 6
Roots are real and unequal
The first root = -2.000000 , Second root = -3.000000

D:\compAssignment\assignemt 2>a.exe
Enter the coefficients of quadratic equation : 1 4 5
Roots are imaginary
The first root = -2.000000 + i1.000000
The first root = -2.000000 - i1.000000
```

5. Write C program to print the sum of the following series upto *n*th term, where *x* or *n* have to be taken from the user. Do not use the library function *pow ()* for computation of x^n.

```c
// S= 1 - 2 + 3 - 4 + 5 upto nth term

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

int main()
{
    int n,sum=0;
    printf("Enter number of terms\n");
    scanf("%d",&n);
    if(n%2==0)
    {
        int end=n/2;
        sum= (-1)*end;

        printf("The sum is %d\n",sum);
    }
    else
    {
        int odd_end=(ceil)((double)n/2);
        int even_end=(floor)((double)n/2);
        sum = odd_end*odd_end - even_end*even_end - even_end;
        printf("The sum is %d\n",sum);
    }
    return 0;
}
```

Output:

```
D:\compAssignment\assignemt 2>gcc sumofseries1.c

D:\compAssignment\assignemt 2>a.exe
Enter number of terms
5
The sum is 3
```

5. Write C program to print the sum of the following series upto *n*th term, where *x* or *n* have to be taken from the user. Do not use the library function *pow ()* for computation of x^n.

```c
#include<stdio.h>

int pow_calc(int x,int n)
{
    if(n==0)
        return 1;
    else if(n==1)
        return x;
    else
        return x*pow_calc(x,n-1);
}

int main()
{
    int n,x;
    printf("Enter the number of terms\n");
    scanf("%d",&n);
    printf("Enter the value of x\n");
    scanf("%d",&x);

    int sum = (pow_calc(x,n)-1)/(x-1);

    printf("The sum is %d\n",sum);

    return 0;
}
```

Output:

```
D:\compAssignment\assignemt 2>gcc sumofseries2.c

D:\compAssignment\assignemt 2>a.exe
Enter the number of terms
5
Enter the value of x
2
The sum is 31
```

6. Write C programs to print the following pattern for *n* number of rows, here *n* is an input taken from keyboard. Use loops and ASCII codes where ever needed.

```c

#include<stdio.h>
int main()
{
    char ch;
    int n , c , a ;
    printf("Enter your choice among 'a' , 'b' and  'c' : ");
    scanf("%c", &ch);
    printf("Enter the number of lines : ");
    scanf("%d", &n );
    switch(ch)
    {
        case 'a':
            printf("The pattern 'a' : \n");
            for(int i = 1 ; i <= n ; i++ )
            {
                for(int sp = 4 ; sp >= i ; sp-- )
                {
                    printf(" \t");
                }
                for( int j = 1 ; j <= i ; j++ )
                {
                    printf("*\t");
                }
                printf("\n");
            }
            break;
        case 'b':
            printf("The pattern 'b' : \n");
            c = 1 ;
            for(int i = 1 ; i <= n ; i++ )
            {
                for( int j = 1 ; j <= i ; j++ , c++ )
                {
                    printf("%d\t" , c );
                }
                printf("\n");
            }
            break;
        case 'c':
            printf("The pattern 'c' : \n");
            a = 65 ;
            for(int i = 1 ; i <= n ; i++ )
            {
                for(int sp = 4 ; sp > i ; sp-- )
                {
                    printf(" \t");
                }
                for(int j = 1 ; j <= i ; j++ , a++ )
                {
                    printf(" %c\t",(char)(a));
                }
                printf("\n");
            }
            break;
        default:
            printf("Invalid Input :) ");
            break;
    }
}
```

Output:

```
D:\compAssignment\assignemt 2>gcc Pattern.c

D:\compAssignment\assignemt 2>a.exe
Enter your choice among 'a' , 'b' and  'c' : a
Enter the number of lines : 3
The pattern 'a' :
                *
            *       *
        *       *       *
```

```
Enter your choice among 'a' , 'b' and  'c' : b
Enter the number of lines : 3
The pattern 'b' :
1
2       3
4       5       6
```

```
Enter your choice among 'a' , 'b' and  'c' : c
Enter the number of lines : 3
The pattern 'c' :
                A
            B       C
        D       E       F
```