EXPERIMENT-6

QUESTION-1: Develop a recursive and non-recursive function FACT(num) to find the factorial of a number, n!, defined by FACT(n) = 1, if n = 0. Otherwise, FACT(n) = n * FACT(n-1). Using this function, write a C program to compute the binomial coefficient. Tabulate the results for different values of n and r with suitable messages

CODE:

```
#include <stdio.h>

// Recursive factorial function
long fact_rec(int n)
{
if(n == 0)
return 1;
return n * fact_rec(n - 1);
}

// Non-recursive factorial function
long fact_nonrec(int n)
{
long fact = 1;
for(int i = 1; i <= n; i++)
fact *= i;
return fact;
}

int main()
{
int n, r;

printf("Enter n and r: ");
if(scanf("%d %d", &n, &r)!=2)
{
printf("invalid input");
```

```
return 0;
}
if(r > n || n < 0 || r < 0)
{
printf("Invalid input. r must be <= n and both non-negative.\n");
return 0;
}

long ncr = fact_rec(n) / (fact_nonrec(r) * fact_rec(n - r));
printf("factorial of %d is %ld:\n",n,fact_rec(n));
printf("Binomial Coefficient C(%d, %d) = %ld\n", n, r, ncr);

return 0;
}
```

OUTPUT:

(a) Enter n and r: j

invalid input

(b) Enter n and r: 5 4

factorial of 5 is 120:

Binomial Coefficient C(5, 4) = 5

(c) Enter n and r: 0 0

factorial of 0 is 1:

Binomial Coefficient C(0, 0) = 1

(d) Enter n and r: -1 -1

Invalid input. r must be <= n and both non-negative.

QUESTION-2: Develop a recursive function GCD(num1,num2) that accepts two integer arguments. Write a C program that invokes this function to find the greatest common divisor of two given integers.

CODE:

```c
#include <stdio.h>

int GCD(int a, int b)
{
if(b == 0)
return a;
return GCD(b, a % b);
}

int main()
{
int x, y;

printf("Enter two numbers: ");
if(scanf("%d %d", &x, &y)!=2 ||(x==0 && y==0))
{
printf("invalid input");
return 0;
}

printf("GCD of %d and %d is %d\n", x, y, GCD(x, y));

return 0;
}
```

OUTPUTS:

(a) Enter two numbers: 0 0

invalid input

(b) Enter two numbers: a b

invalid input

(c) Enter two numbers: -7 -7

GCD of -7 and -7 is -7

(d) Enter two numbers: 12 0

GCD of 12 and 0 is 12

QUESTION-3: Develop a recursive function FIBO(num) that accepts an integer argument. Write a C program that invokes this function to generate the Fibonacci sequence up to num.

CODE:

```
#include <stdio.h>

int FIBO(int n)
{
if(n == 0) return 0;
if(n == 1) return 1;
return FIBO(n-1) + FIBO(n-2);
}

int main()
{
int n;
printf("Enter how many terms: ");
if(scanf("%d", &n)!=1 || n<0 || n==-0)
{
printf("invalid");
return 0;
}
printf("Fibonacci series:\n");
for(int i = 0; i < n; i++)
printf("%d \n", FIBO(i));
```

```
return 0;
}
```

OUTPUT:

(a) Enter how many terms: 6

Fibonacci series:

0

1

1

2

3

5

(b) Enter how many terms: h

invalid

(c) Enter how many terms: -6

invalid

(d) Enter how many terms: -0

invalid

QUESTION-4: Develop a C function ISPRIME(num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given ranges.

CODE:

```c
#include <stdio.h>
int ISPRIME(int n)
{
if(n < 2)
return 0;
for(int i = 2; i <= n/2; i++)
if(n % i == 0)
return 0;
return 1;
}

int main()
{
int start, end;
if( scanf("%d %d", &start, &end)!=2 || start>end)
{
printf("invalid");
return 0;
}
printf("Prime numbers in range:\n");
for(int i = start; i <= end; i++)
if(ISPRIME(i))
printf("%d ", i);

return 0;
}
```

OUTPUT:

(a) Enter starting and ending numbers i.e. range: 8

7

invalid

(b) Enter starting and ending numbers i.e. range: 1

20

Prime numbers in range:

2 3 5 7 11 13 17 19

(c) Enter starting and ending numbers i.e. range: 5

5

Prime numbers in range:

5