

**Name Priyadarshi Prabhakar   SAP ID 590029237**

## **Experiment 6: Functions – C Programs, Pseudocode, and Algorithms**

### **1. FACT (Factorial) – Recursive & Non-Recursive + Binomial Coefficient**

Algorithm:

1. Start
2. Read n
3. If  $n = 0$ , return 1
4. Else factorial =  $n * \text{FACT}(n-1)$
5. End

Pseudocode (Recursive):

FUNCTION FACT(n)

    IF  $n == 0$  THEN

        RETURN 1

    ELSE

        RETURN  $n * \text{FACT}(n-1)$

END FUNCTION

Pseudocode (Iterative):

FUNCTION FACT\_ITER(n)

    fact = 1

    FOR i = 1 TO n

        fact = fact \* i

    END FOR

```
    RETURN fact
```

```
END FUNCTION
```

C Program:

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

```
long fact_rec(long n) {  
    if (n == 0) return 1;  
    return n * fact_rec(n - 1);  
}
```

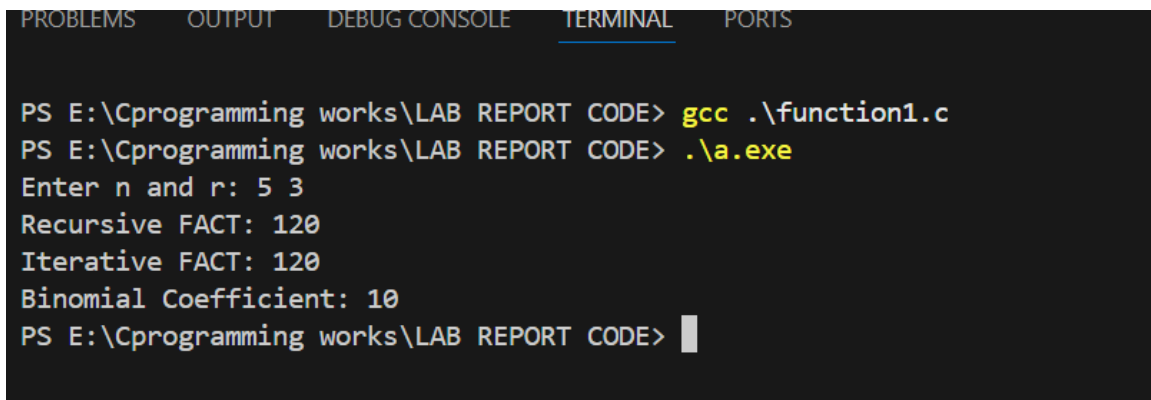
```
long fact_iter(long n) {  
    long f = 1;  
    for (long i = 1; i <= n; i++)  
        f *= i;  
    return f;  
}
```

```
long binomial(long n, long r) {  
    return fact_rec(n) / (fact_rec(r) * fact_rec(n - r));  
}
```

```
int main() {  
    long n, r;  
    printf("Enter n and r: ");  
    scanf("%ld %ld", &n, &r);
```

```
printf("Recursive FACT: %ld\n", fact_rec(n));  
printf("Iterative FACT: %ld\n", fact_iter(n));  
printf("Binomial Coefficient: %ld\n", binomial(n, r));  
return 0;  
}
```

## OUTPUT



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  
  
PS E:\Cprogramming works\LAB REPORT CODE> gcc .\function1.c  
PS E:\Cprogramming works\LAB REPORT CODE> .\a.exe  
Enter n and r: 5 3  
Recursive FACT: 120  
Iterative FACT: 120  
Binomial Coefficient: 10  
PS E:\Cprogramming works\LAB REPORT CODE> █
```

## 2. GCD – Recursive

Algorithm:

1. Start
2. Read num1, num2
3. If num2 == 0 return num1
4. Else return GCD(num2, num1 % num2)
5. Stop

Pseudocode:

FUNCTION GCD(a, b)

IF b == 0 THEN

RETURN a

ELSE

```
    RETURN GCD(b, a % b)
```

```
END FUNCTION
```

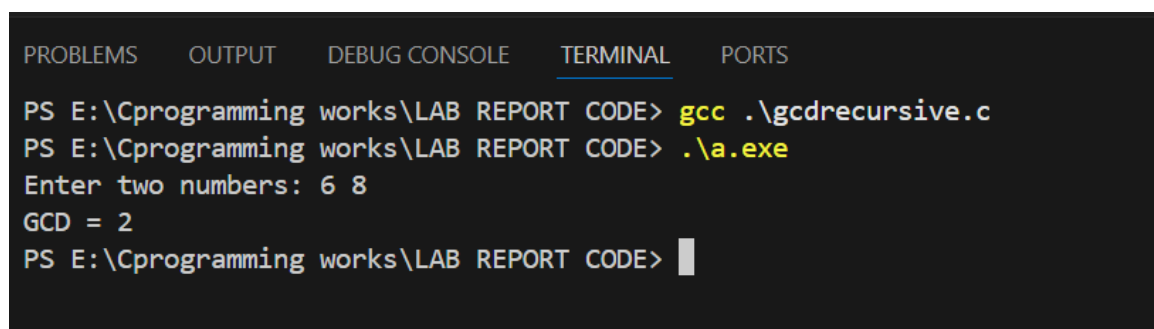
C Program:

```
#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: ");  
    scanf("%d %d", &x, &y);  
    printf("GCD = %d\n", gcd(x, y));  
    return 0;  
}
```

## OUTPUT



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  
PS E:\Cprogramming works\LAB REPORT CODE> gcc .\gcdrecursive.c  
PS E:\Cprogramming works\LAB REPORT CODE> .\a.exe  
Enter two numbers: 6 8  
GCD = 2  
PS E:\Cprogramming works\LAB REPORT CODE> █
```

### 3. Fibonacci – Recursive

Algorithm:

1. Start
2. Read n
3. If  $n \leq 1$  return n
4. Else return  $F(n-1) + F(n-2)$
5. Stop

Pseudocode:

FUNCTION FIB(n)

    IF  $n \leq 1$  THEN

        RETURN n

    ELSE

        RETURN FIB(n-1) + FIB(n-2)

END FUNCTION

C Program:

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

```
int fib(int n) {
```

```
    if ( $n \leq 1$ ) return n;
```

```
    return fib(n - 1) + fib(n - 2);
```

```
}
```

```
int main() {
```

```
    int n;
```

```
    printf("Enter number: ");
```

```
    scanf("%d", &n);
```

```
for (int i = 0; i < n; i++)  
    printf("%d ", fib(i));  
  
return 0;  
}
```

## OUTPUT

```
PS E:\Cprogramming works\LAB REPORT CODE> gcc .\fibonaacirecursive.c  
PS E:\Cprogramming works\LAB REPORT CODE> .\a.exe  
Enter number: 6  
0 1 1 2 3 5  
PS E:\Cprogramming works\LAB REPORT CODE> █
```

## 4. ISPRIME – Prime Number Checker

Algorithm:

1. Start
2. Read n
3. If  $n \leq 1$  return not prime
4. Check divisibility from 2 to  $n/2$
5. If divisible  $\rightarrow$  not prime, else prime
6. Stop

Pseudocode:

FUNCTION ISPRIME(n)

    IF  $n \leq 1$  THEN RETURN 0

    FOR  $i = 2$  TO  $n/2$

        IF  $n \% i == 0$  THEN RETURN 0

    RETURN 1

C Program:

```
#include <stdio.h>

int isPrime(int n) {
    if (n <= 1) return 0;
    for (int i = 2; i <= n/2; i++)
        if (n % i == 0)
            return 0;
    return 1;
}

int main() {
    int a, b;
    printf("Enter range: ");
    scanf("%d %d", &a, &b);

    for (int i = a; i <= b; i++)
        if (isPrime(i))
            printf("%d ", i);

    return 0;
}
```

## OUTPUT

```
PS E:\Cprogramming works\LAB REPORT CODE> gcc .\primecheck.c
PS E:\Cprogramming works\LAB REPORT CODE> .\a.exe
Enter range (a b): 1 11
2 3 5 7 11
PS E:\Cprogramming works\LAB REPORT CODE> █
```

## 5. REVERSE – Reverse a String

Algorithm:

1. Start
2. Read string
3. Set  $i = 0$ ,  $j = \text{length} - 1$
4. Swap characters while  $i < j$
5. Stop

Pseudocode:

FUNCTION REVERSE(str)

$i = 0$

$j = \text{len}(\text{str}) - 1$

    WHILE  $i < j$

        swap(str[i], str[j])

$i++$ ,  $j--$

END FUNCTION

C Program:

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

```
#include <string.h>
```



```
void reverse(char str[]) {  
    int i = 0, j = strlen(str) - 1;  
    while (i < j) {  
        char temp = str[i];  
        str[i] = str[j];  
        str[j] = temp;  
        i++;  
        j--;  
    }  
}  
  
int main() {  
    char s[100];  
    printf("Enter string: ");  
    scanf("%s", s);  
    reverse(s);  
    printf("Reversed string: %s\n", s);  
    return 0;  
}
```

## OUTPUT

```
PS E:\Cprogramming works\LAB REPORT CODE> gcc .\reverse.c  
PS E:\Cprogramming works\LAB REPORT CODE> .\a.exe  
Enter string: Priyadarshi]  
Reversed string: ]ihsradayirP  
PS E:\Cprogramming works\LAB REPORT CODE> █
```