

EXPERIMENT NO. : 6

Functions

- 1). Develop a recursive and non-recursive function $FACT(n)$ 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.

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

```
unsigned long long int factorial_recursive (int n)
{
    if (n == 0)
        return 1;
    return n * factorial_recursive (n-1);
}
```

```
unsigned long long int factorial_non_recursive
(int n) {
    unsigned long long int result = 1;
    for (int i = 1; i <= n; i++)
    {
        result * = i;
    }
    return result;
}
```

Teacher's Signature _____

```

unsigned long long int binomial - coefficient
(int n, int r) {
    if (r > n)
        return 0;
    return factorial - non - recursive / factorial -
        non - recursive(r) * factorial - non - recursive(n-r);
}

```

```

int main () {
    int n, r;

```

```

    printf ("Calculating Binomial Coefficients for values
    of n (0-10) and r (0 to n) : \n");
    printf ("n\t r\t C(n,r) \n");
    printf ("----- \n");

```

```

    for (n = 0; n <= 10; n++)
    {

```

```

        for (r = 0; r <= n; r++)
        {

```

```

            printf ("%d\t %d\t %llu \n", n, r,
                binomial - coefficients (n, r));

```

```

        }

```

```

    printf ("----- \n");
}

```

```

    return 0;
}

```


Output :- Calculating Binomial coefficients for values of $n(0-10)$ and $r(0 \text{ to } n)$;

n	r	$C(n, r)$
-----	-----	-----------

0	0	1
---	---	---

1	0	1
---	---	---

1	1	1
---	---	---

2	0	1
---	---	---

2	1	2
---	---	---

2	2	1
---	---	---

3	0	1
---	---	---

3	1	3
---	---	---

3	2	3
---	---	---

3	3	1
---	---	---

4	0	1
---	---	---

4	1	4
---	---	---

4	2	6
---	---	---

4	3	4
---	---	---

4	4	1
---	---	---

5	0	1
---	---	---

5	1	5
---	---	---

5	2	10
---	---	----

5	3	10
---	---	----

5	3	10	9	0	1
5	4	5	9	1	9
5	5	1	9	2	36
			9	3	84
6	0	1	9	4	126
6	1	6	9	5	126
6	2	15	9	6	84
6	3	20	9	7	36
6	4	15	9	8	9
6	5	6	9	9	1
6	6	1			

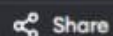
			10	0	1
7	0	1	10	1	16
7	1	7	10	2	45
7	2	21	10	3	120
7	3	35	10	4	210
7	4	35	10	5	252
7	5	21	10	6	210
7	6	7	10	7	120
7	7	1	10	8	45

10	9	10
10	10	1

8	0	1
8	1	8
8	2	28
8	3	56
8	4	70
8	5	56
8	6	28
8	7	8
8	8	1

Teacher's Signature _____

main.c



Run

Output

Clear

```
1 #include <stdio.h>
2
3 unsigned long long int factorial_recursive(int n) {
4     if (n == 0)
5         return 1;
6     return n * factorial_recursive(n - 1);
7 }
8
9 unsigned long long int factorial_non_recursive(int n) {
10     unsigned long long int result = 1;
11     for (int i = 1; i <= n; i++) {
12         result *= i;
13     }
14     return result;
15 }
16
17 unsigned long long int binomial_coefficient(int n, int r) {
18     if (r > n)
19         return 0;
20     return factorial_non_recursive(n) / (factorial_non_recursive(r) *
21         factorial_non_recursive(n - r));
22 }
23
24 int main() {
25     int n, r;
26
27     printf("Tabulating Binomial Coefficients for values of n (0-10) and r (0 to n
28         ):\n");
29     printf("n\t r\t C(n, r)\n");
30     printf("-----\n");
```

Tabulating Binomial Coefficients for values of n (0-10) and r (0 to n):

n	r	C(n, r)
---	---	---------

0	0	1
---	---	---

1	0	1
---	---	---

1	1	1
---	---	---

2	0	1
---	---	---

2	1	2
---	---	---

2	2	1
---	---	---

3	0	1
---	---	---

3	1	3
---	---	---

3	2	3
---	---	---

3	3	1
---	---	---

4	0	1
---	---	---

4	1	4
---	---	---

4	2	6
---	---	---

4	3	4
---	---	---

4	4	1
---	---	---

5	0	1
---	---	---

5	1	5
---	---	---

5	2	10
---	---	----

5	3	10
---	---	----

5	4	5
---	---	---

5	5	1
---	---	---



```
11 for (int i = 1; i <= n; i++) {
12     result *= i;
13 }
14 return result;
15 }
16
17 unsigned long long int binomial_coefficient(int n, int r) {
18     if (r > n)
19         return 0;
20     return factorial_non_recursive(n) / (factorial_non_recursive(r) *
21         factorial_non_recursive(n - r));
22 }
23
24 int main() {
25     int n, r;
26
27     printf("Tabulating Binomial Coefficients for values of n (0-10) and r (0 to n
28         ):\n");
29     printf("n\t r\t C(n, r)\n");
30     printf("-----\n");
31     for (n = 0; n <= 10; n++) {
32         for (r = 0; r <= n; r++) {
33             printf("%d\t %d\t %llu\n", n, r, binomial_coefficient(n, r));
34         }
35         printf("-----\n");
36     }
37     return 0;
38 }
```

```
6 0 1
6 1 6
6 2 15
6 3 20
6 4 15
6 5 6
6 6 1
```

```
-----
7 0 1
7 1 7
7 2 21
7 3 35
7 4 35
7 5 21
7 6 7
7 7 1
```

```
-----
8 0 1
8 1 8
8 2 28
8 3 56
8 4 70
8 5 56
8 6 28
8 7 8
8 8 1
```

```
-----
9 0 1
9 1 9
9 2 36
```




```
11 for (int i = 1; i <= n; i++) {
12     result *= i;
13 }
14 return result;
15 }
16
17 unsigned long long int binomial_coefficient(int n, int r) {
18     if (r > n)
19         return 0;
20     return factorial_non_recursive(n) / (factorial_non_recursive(r) *
21         factorial_non_recursive(n - r));
22 }
23
24 int main() {
25     int n, r;
26
27     printf("Tabulating Binomial Coefficients for values of n (0-10) and r (0 to n
28         ):\n");
29     printf("n\t r\t C(n, r)\n");
30     printf("-----\n");
31     for (n = 0; n <= 10; n++) {
32         for (r = 0; r <= n; r++) {
33             printf("%d\t %d\t %llu\n", n, r, binomial_coefficient(n, r));
34         }
35         printf("-----\n");
36     }
37     return 0;
38 }
```

```
8 6 28
8 7 8
8 8 1
```

```
-----
9 0 1
9 1 9
9 2 36
9 3 84
9 4 126
9 5 126
9 6 84
9 7 36
9 8 9
9 9 1
```

```
-----
10 0 1
10 1 10
10 2 45
10 3 120
10 4 210
10 5 252
10 6 210
10 7 120
10 8 45
10 9 10
10 10 1
```

--- Code Execution Successful ---

- 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.

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

```
int gcd (int num1, int num2)
```

```
{
```

```
    if (num2 == 0)
```

```
{
```

```
        return num1;
```

```
}
```

```
    return gcd (num2, num1 % num2);
```

```
}
```

```
int main ( )
```

```
{
```

```
    int num1, num2;
```

```
    printf ("Enter two integers : ");
```

```
    scanf ("%d %d", &num1, &num2);
```

```
    int result = gcd (num1, num2);
```

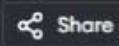
```
    printf ("The GCD of %d and %d is : %d\n",  
           num1, num2, result);
```

```
    return 0;
```

```
}
```


Output :- Enter two integers : 5, 6

The GCD of 5 and 6 is : 1



```
1 #include <stdio.h>
2
3 int gcd(int num1, int num2) {
4     if (num2 == 0) {
5         return num1;
6     }
7     return gcd(num2, num1 % num2);
8 }
9
10 int main() {
11     int num1, num2;
12
13     printf("Enter two integers: ");
14     scanf("%d %d", &num1, &num2);
15
16     int result = gcd(num1, num2);
17
18     printf("The GCD of %d and %d is: %d\n", num1, num2, result);
19
20     return 0;
21 }
```

Enter two integers: 5 6
The GCD of 5 and 6 is: 1

=== Code Execution Successful ===



- 3) Develop a recursive function $FIBO(num)$ that accepts an integer arguments. Write a C program that involves this function to generate the Fibonacci sequence up to num .

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

```
int fibo (int num)
```

```
{
```

```
if num
```

```
{
```

```
return num;
```

```
}
```

```
return fibo (num-1) + fibo (num-2);
```

```
}
```

```
int main () {
```

```
int num;
```

```
printf ("Enter an integer to generate Fibonacci  
sequence up to that number: ");
```

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

```
printf ("Fibonacci sequence up to %d:\n", num);
```

```
for (int i = 0; i <= num; i++) {
```

```
int result = fibo (i);
```

```
printf ("%d", result);
```

```
}
```

```
printf ("\n");
```

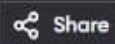


```
return 0;  
}
```

Output :- Enter an integer to generate Fibonacci sequence up to that number: 19

Fibonacci sequence up to 19 :

0	1	1	2	3	5	8	13	21	34	55
89	144	233	377	610	987	1597				
2584	4181									



```
1 #include <stdio.h>
2
3 int fibo(int num) {
4     if (num <= 1) {
5         return num;
6     }
7     return fibo(num - 1) + fibo(num - 2);
8 }
9
10 int main() {
11     int num;
12
13     printf("Enter an integer to generate Fibonacci sequence up to that number: ");
14     scanf("%d", &num);
15
16     printf("Fibonacci sequence up to %d:\n", num);
17
18     for (int i = 0; i <= num; i++) {
19         int result = fibo(i);
20         printf("%d ", result);
21     }
22
23     printf("\n");
24     return 0;
25 }
```

Enter an integer to generate Fibonacci sequence up to that number: 19
Fibonacci sequence up to 19:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181

=== Code Execution Successful ===



- 4) Develop a C function ISPRIME (num) that accepts an integer argument and return 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.

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

```
int ISPRIME (int num)
```

```
{
```

```
if (num <= 1) {
```

```
    return 0;
```

```
}
```

```
for (int i = 2; i * i <= num; i++) {
```

```
    if (num % i == 0) {
```

```
        return 0;
```

```
    }
```

```
}
```

```
return 1;
```

```
}
```

```
int main ( ) {
```

```
    int start, end;
```

```
    printf ("Enter the range (start and end): ");
```

```
    scanf ("%d %d", &start, &end);
```

```
    printf ("Prime numbers between %d and %d are:\n", start, end);
```



```
for (int num = start; num <= end; num++)  
{  
    if (ISPRIME) {  
        printf ("%d", num);  
    }  
    printf ("\n");  
}  
return 0;  
}
```

Output :- Enter the range (start and end):
11 90

Prime numbers between 11 and 90 are:

11	13	17	19	23	29	31	37
41	43	47	51	53	59	61	67
71	73	79	83	89			



```
1 #include <stdio.h>
2
3 int ISPRIME(int num) {
4     if (num <= 1) {
5         return 0;
6     }
7     for (int i = 2; i * i <= num; i++) {
8         if (num % i == 0) {
9             return 0;
10        }
11    }
12    return 1;
13 }
14
15 int main() {
16     int start, end;
17
18     printf("Enter the range (start and end): ");
19     scanf("%d %d", &start, &end);
20
21     printf("Prime numbers between %d and %d are:\n", start, end);
22
23     for (int num = start; num <= end; num++) {
24         if (ISPRIME(num)) {
25             printf("%d ", num);
26         }
27     }
28     printf("\n");
29
30     return 0;
```

```
Enter the range (start and end): 11 90
Prime numbers between 11 and 90 are:
11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89
```

```
=== Code Execution Successful ===
```

5) Develop a function REVERSE (str) that accepts a string argument, write a C program that invokes this function to find the reverse of a given string

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

```
void REVERSE (char str [])
```

```
{
```

```
    int n = strlen (str);
```

```
    char temp ;
```

```
    for (int i = 0 ; i < n / 2 ; i++)
```

```
        temp = str [i];
```

```
        str [i] = str [n-i-1];
```

```
        str [n-i-1] = temp ;
```

```
    }
```

```
}
```

```
int main ( )
```

```
{
```

```
    char str [100];
```

```
    printf ("Enter a string : ");
```

```
    fgets (str, sizeof (str), stdin);
```

```
    str [strlen (str), "\n"] = 0;
```

```
    REVERSE (str);
```



```
printf ("Reversed string : %s\n", str);
```

```
return 0;
```

```
}
```

Output :- Enter a string : 783

Reversed string : 387



```
1 #include <stdio.h>
2 #include <string.h>
3
4 void REVERSE(char str[]) {
5     int n = strlen(str);
6     char temp;
7
8     for (int i = 0; i < n / 2; i++) {
9         temp = str[i];
10        str[i] = str[n - i - 1];
11        str[n - i - 1] = temp;
12    }
13 }
14
15 int main() {
16     char str[100];
17
18     printf("Enter a string: ");
19     fgets(str, sizeof(str), stdin);
20
21     str[strcspn(str, "\n")] = 0;
22
23     REVERSE(str);
24
25     printf("Reversed string: %s\n", str);
26
27     return 0;
28 }
```

Enter a string: 783
Reversed string: 387

=== Code Execution Successful ===