1. Write a C function to calculate NCR value.

**Code:**

#include <stdio.h>

int factorial(int n) {

int fact = 1;

for (int i = 1; i <= n; i++) {

fact \*= i;

}

return fact;

}

int nCr(int n, int r) {

if (r > n) {

return 0;

}

return factorial(n) / (factorial(r) \* factorial(n - r));

}

int main() {

int n, r;

printf("Enter the value of n: ");

scanf("%d", &n);

printf("Enter the value of r: ");

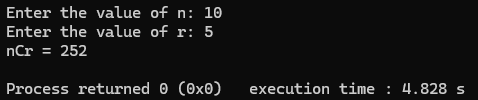
scanf("%d", &r);

printf("nCr = %d\n", nCr(n, r));

return 0;

}

**Output:**



1. Write a C function to transpose of a matrix.

**Code:**

#include <stdio.h>

void transposeMatrix(int rows, int cols, int matrix[rows][cols], int transpose[cols][rows]) {

for (int i = 0; i < rows; i++) {

for (int j = 0; j < cols; j++) {

transpose[j][i] = matrix[i][j];

}

}

}

int main() {

int rows, cols;

printf("Enter the number of rows: ");

scanf("%d", &rows);

printf("Enter the number of columns: ");

scanf("%d", &cols);

int matrix[rows][cols], transpose[cols][rows];

printf("Enter elements of the matrix:\n");

for (int i = 0; i < rows; i++) {

for (int j = 0; j < cols; j++) {

printf("Element [%d][%d]: ", i + 1, j + 1);

scanf("%d", &matrix[i][j]);

}

}

transposeMatrix(rows, cols, matrix, transpose);

printf("\nOriginal Matrix:\n");

for (int i = 0; i < rows; i++) {

for (int j = 0; j < cols; j++) {

printf("%d ", matrix[i][j]);

}

printf("\n");

}

printf("\nTranspose of the Matrix:\n");

for (int i = 0; i < cols; i++) {

for (int j = 0; j < rows; j++) {

printf("%d ", transpose[i][j]);

}

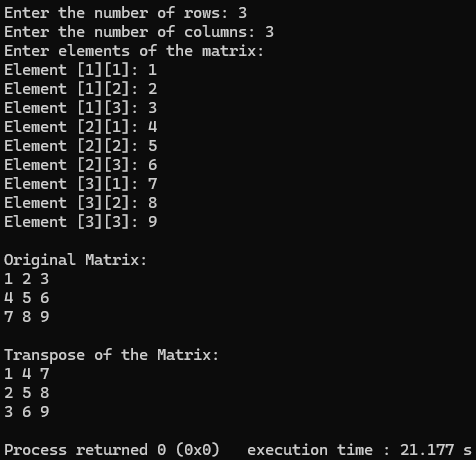
printf("\n");

}

return 0;

}

**Output:**



1. Write a recursive function to generate Fibonacci series.

**Code:**

#include <stdio.h>

int fibonacci(int n) {

if (n == 0 || n == 1) {

return n;

}

return fibonacci(n - 1) + fibonacci(n - 2);

}

int main() {

int n, i;

printf("Enter the number of terms: ");

scanf("%d", &n);

printf("Fibonacci Series: ");

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

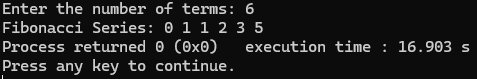
printf("%d ", fibonacci(i));

}

return 0;

}

**Output:**



1. Write a recursive function to find the LCM of two numbers.

**Code:**

#include <stdio.h>

int findLCM(int a, int b, int multiple) {

if (multiple % a == 0 && multiple % b == 0) {

return multiple;

}

return findLCM(a, b, multiple + b);

}

int main() {

int num1, num2;

printf("Enter two numbers: ");

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

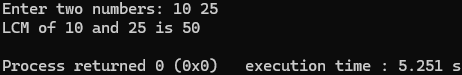
int lcm = findLCM(num1, num2, (num1 > num2 ? num1 : num2));

printf("LCM of %d and %d is %d\n", num1, num2, lcm);

return 0;

}

**Output:**



1. Write a recursive function to find the GCD of two numbers.

**Code:**

#include <stdio.h>

int gcd(int a, int b) {

if (b == 0)

return a;

return gcd(b, a % b);

}

int main() {

int num1, num2;

printf("Enter two numbers: ");

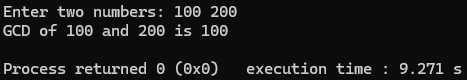
scanf("%d %d", &num1, &num2);

printf("GCD of %d and %d is %d\n", num1, num2, gcd(num1, num2));

return 0;

}

**Output:**



1. Write a recursive function to check whether the given number is palindrome or not.

**Code:**

#include <stdio.h>

int reverseNumber(int num, int rev) {

if (num == 0)

return rev;

return reverseNumber(num / 10, rev \* 10 + num % 10);

}

int isPalindrome(int num) {

if (num < 0)

return 0;

return num == reverseNumber(num, 0);

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (isPalindrome(num))

printf("%d is a palindrome.\n", num);

else

printf("%d is not a palindrome.\n", num);

return 0;

}

**Output:**

