Experiment No-1

Experiment Name: Write a menu driven C program for calculator using do-while loop.

Objective:

- To create a simple calculator program that performs basic arithmetic operations: addition, subtraction, multiplication, and division.
- To implement a user-friendly menu-driven interface.
- To demonstrate the use of do-while loops for continuous operation until exit.
- To demonstrate the use of control structures like do-while loops and switch statements in C.

Problem analysis:

In this program, the objective is to design a menu-driven calculator that performs basic arithmetic operations such as addition, subtraction, multiplication, and division. The user selects an option from a displayed menu and enters two numbers for calculation. The program uses a do-while loop so that the menu keeps repeating until the user chooses the exit option. The variables a and b store the input numbers, while the variable choice determines which operation will be executed. A switch statement is used to process the selected operation and store the result in the variable result. Finally, the program displays the output on the screen. The calculator continues to operate until the user enters 5 to exit the program.

Input variable	Processing variable	Output variable	Header file
a,b(ch)	Result choice loop var	Result	< Stdio.h>

Algorithm:

Step1: Start.

Step2: take input

Step3: if the choice is between 1-4 perform the operation with two numbers

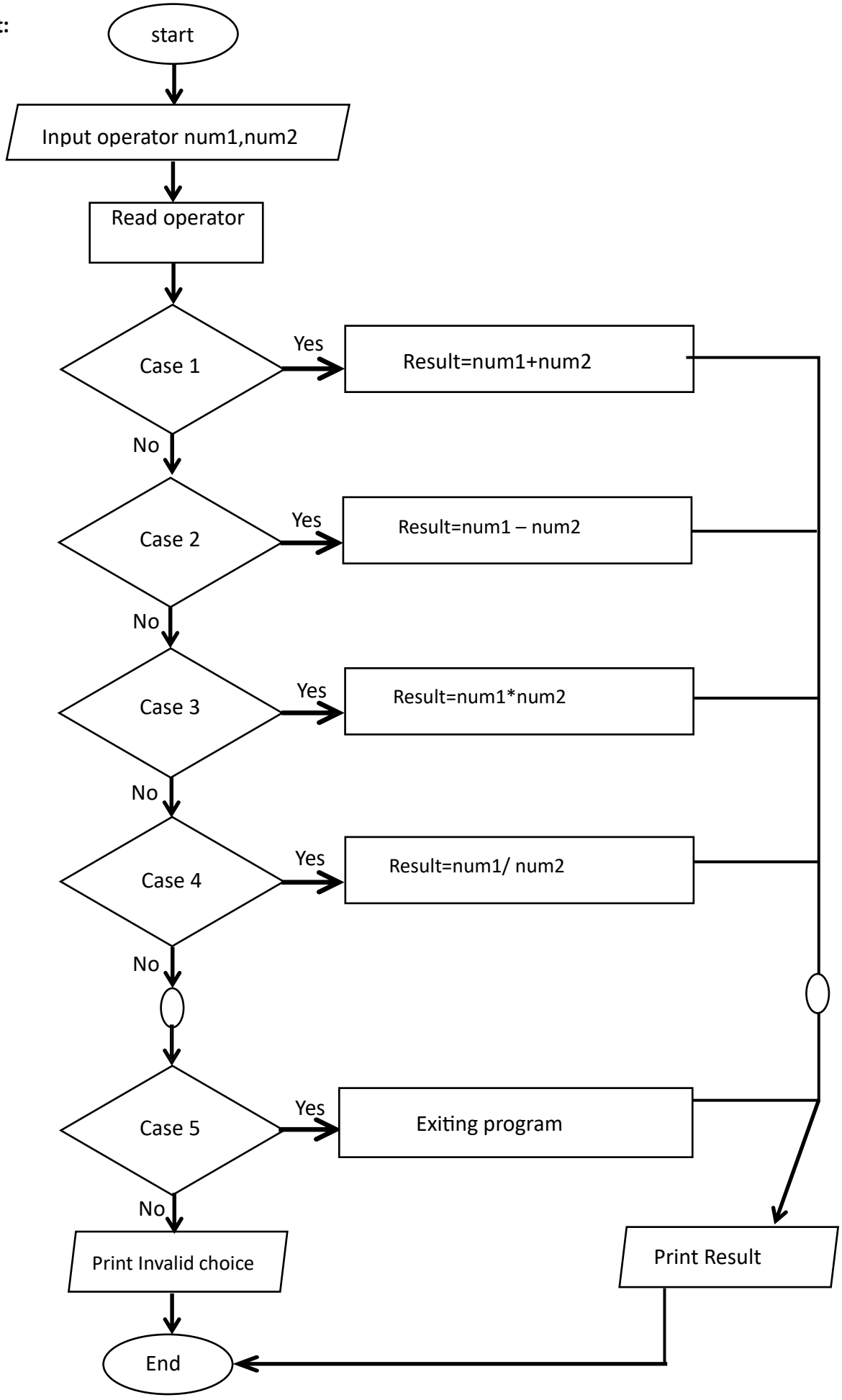
Step 4: if the choice is 5, stop the program

Step5: if the choice is anything else show “invalid choice”

Step6: show the menu again

Step7: End

Flowchart:

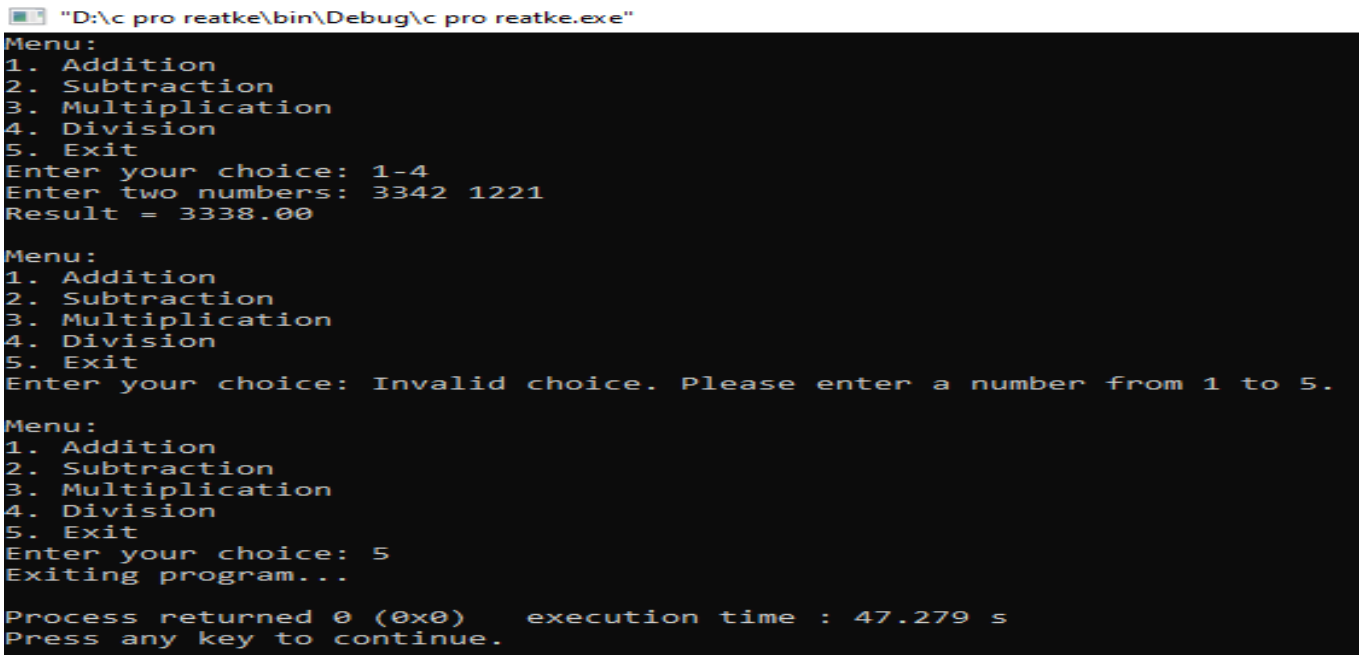

start
Input operator num1,num2
Read operator
Case 1
Yes
Result=num1+num2
No
Case 2
Yes
Result=num1 – num2
No
Case 3
Yes
Result=num1*num2
No
Case 4
Yes
Result=num1/ num2
No
Case 5
Yes
Exiting program
No
Print Invalid choice
Print Result
End

Source code:

```c
#include <stdio.h>
int main() {
    int choice;
    float a, b, result;
    do {
        printf("\n Menu:\n");
        printf("1. Addition\n");
        printf("2. Subtraction\n");
        printf("3. Multiplication\n");
        printf("4. Division\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        if (choice >= 1 && choice <= 4) {
            printf("Enter two numbers: ");
            scanf("%f %f", &a, &b);
        }
        switch (choice) {
            case 1: result = a + b;
                    printf("Result = %.2f\n", result);
                    break;
            case 2: result = a - b;
                    printf("Result = %.2f\n", result);
                    break;
            case 3: result = a * b;
                    printf("Result = %.2f\n", result);
                    break;
            case 4:
                    if (b != 0)
                        result = a / b;
                    else
                        printf("Division by zero not allowed!\n");
                    printf("Result = %.2f\n", result);
                    printf("Result = %.2f\n", result);
                    break;
            case 5:
                    printf("Exiting program...\n");
                    break;
            default:
                    printf("Invalid choice!\n");
        }
    } while (choice != 5);
    return 0;
}
```

Output:



Discussion:

This program allows the user to perform different operations repeatedly using a do-while loop. It handles invalid choices and prevents division by zero. The use of a switch-case structure ensures readability and manageability. The program successfully demonstrates how to implement a calculator using a do-while loop and switch-case statements. It provides a user-friendly way to perform calculations until the user chooses to exit.

Experiment No-2

Experiment Name: Write a C program to find sum of the square of all natural numbers from 1 to N using while loop.

Series: $1^2+2^2+3^2+4^2+.....+N^2$

Objective:

- To write a C program that calculates the sum of the squares of all natural numbers from 1 to a user given number N using a while loop.

Problem analysis:

The problem involves calculating the sum: $Sum=1^2+2^2+3^2+...+N^2$

This is a classic iterative problem and is best approached using a loop that increments a counter from 1 to N, squaring each number and accumulating the result.

We'll use:

- A loop (while) to iterate through each number.
- A variable to keep the running total.
- Another variable to count from 1 to N.

Input variable	Processing variable	Output variable	Header file
N (last natural number)	i,sum	Sum of squares	#include<stdio.h>

Algorithm:

Step1: start

Step2: take input N

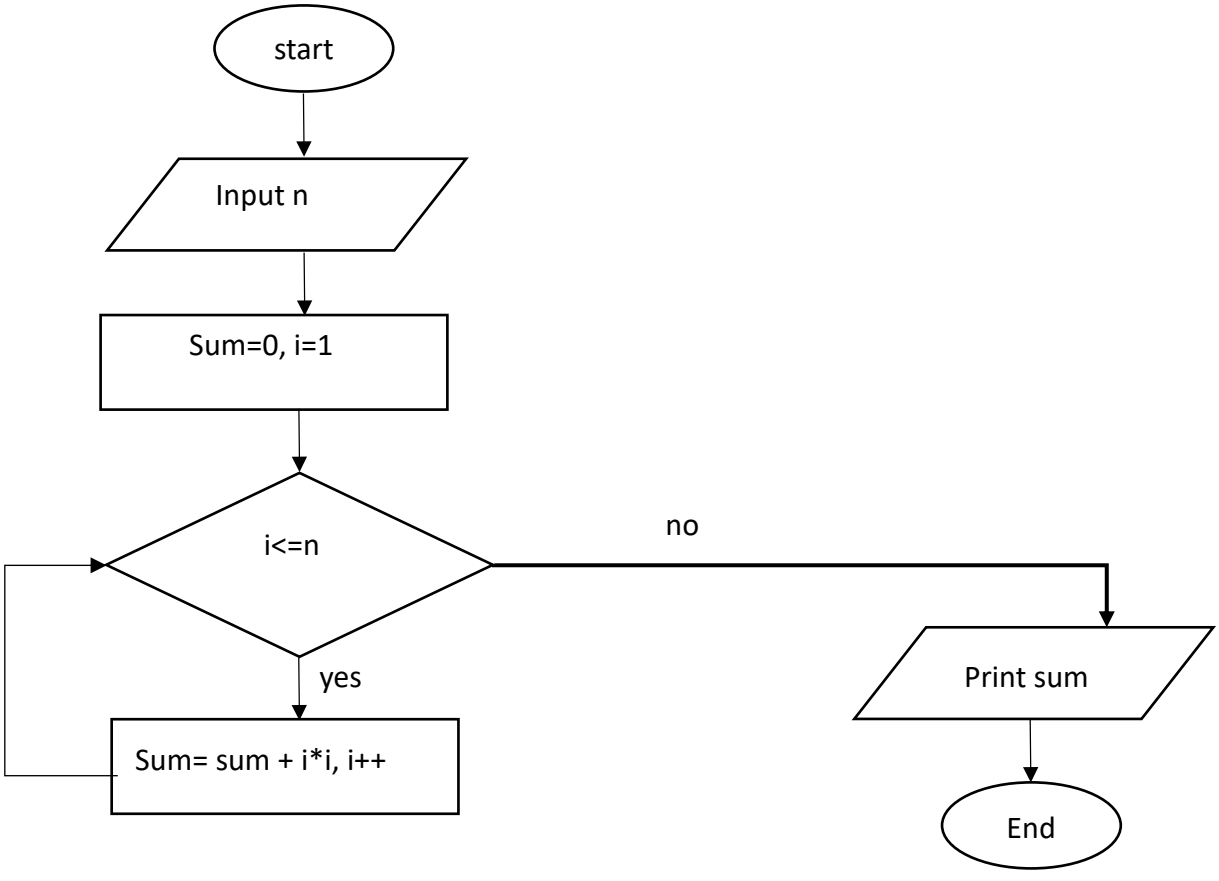
Step3: Initialize i=1, sum=0.

Step4: While i <= N, do sum = sum + i*i, increment = i++

Step5: display sum

Step6: End

Flowchart:



Source code:

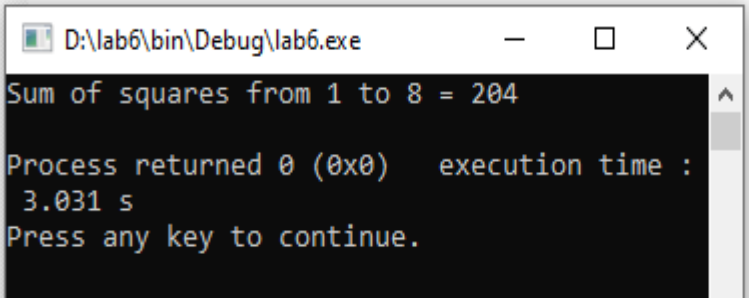
```c
#include <stdio.h>

int main() {
    int N, i = 1, sum = 0;

    printf("Enter the value of N: ");
    scanf("%d", &N);

    while (i <= N) {
        sum += i * i;
        i++;
    }

    printf("Sum of squares from 1 to %d = %d\n", N, sum);
    return 0;
}
```



Discussion:

The program effectively calculates the sum of the squares of natural numbers using while loop. The loop iterates until the condition i<=n become false. In each intention the square of current value of i is added to the sum. And finally, the calculated some is printed to the console. This program demonstrates the fundamental concept of using loops to perform repetitive calculations in C.

Experiment No-3

Experiment Name: Write a C program to find the sum of Natural Number/Factorial of Number of all natural numbers from 1 to N using for loop. Series: 1/1! + 2/2! + 3/3! + 4/4! + ...+ N/N!

Objective:

- To write a C program that computes the sum of a mathematical series of the form:
  S = 1/1! + 2/2! + 3/3! + ... + N/N!
  using a for loop, and to understand basic concepts of loops and factorials in C.

Problem analysis:

We are given a natural number N, and we need to compute the sum:

$$S = \Sigma(i/i!) for i = 1 to N$$

To compute this:

- We need to calculate the factorial of each i (i.e., i!)
- Then divide i by i!
- Accumulate the result in a sum variable

Import variable	Processing variable	Output variable	Header file
N(int)	i, fact, sum	Sum(float)	#include<stdio.h>

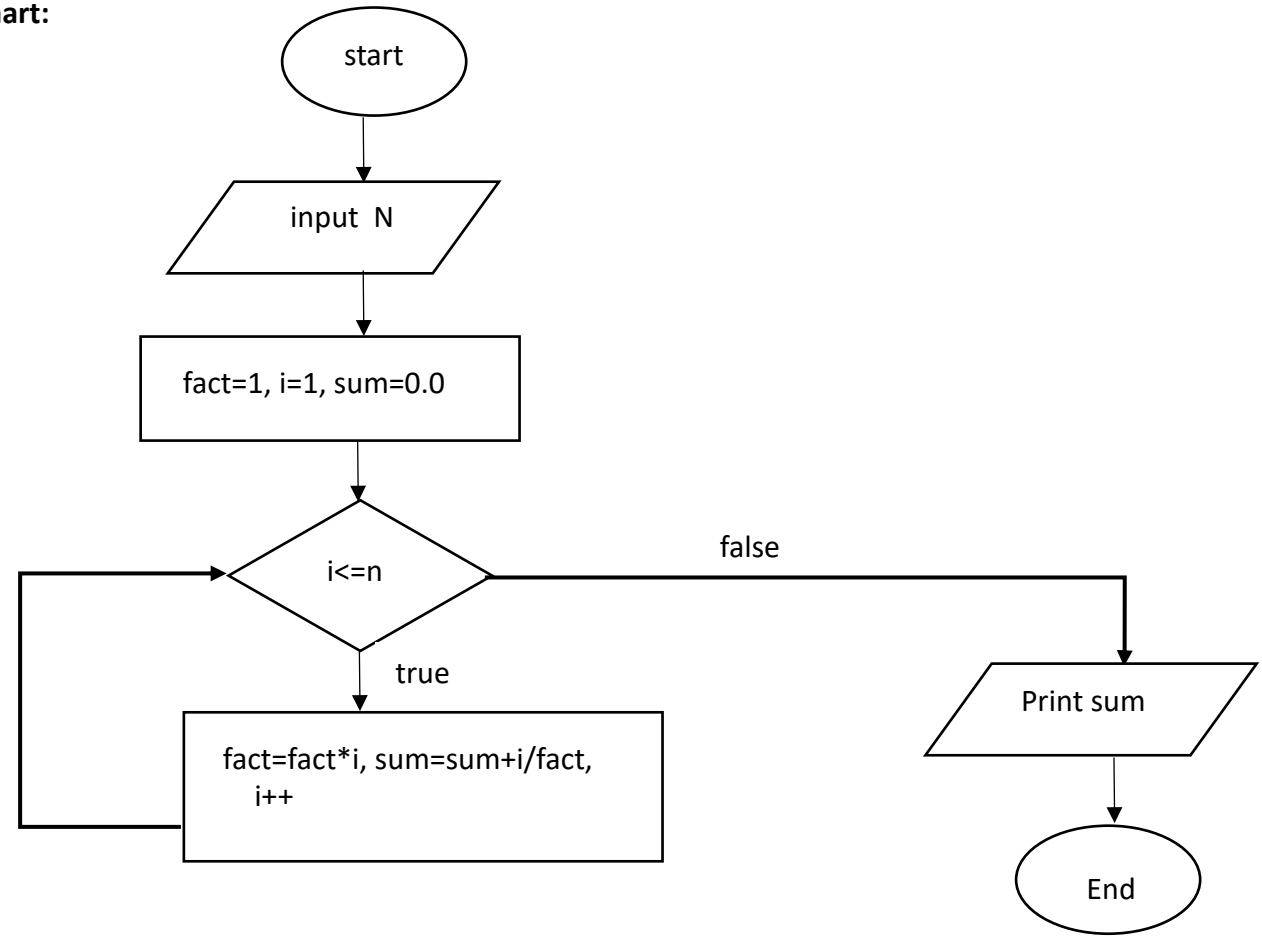
Algorithm:

Step1: start
Step2: input N
Step3: initialize sum=0
Step4: use for loop from 1 to N:

- Calculate factorial
- Add i/fact to sum

Step5**:** display sum
Step6: End

Flowchart:



Source code:

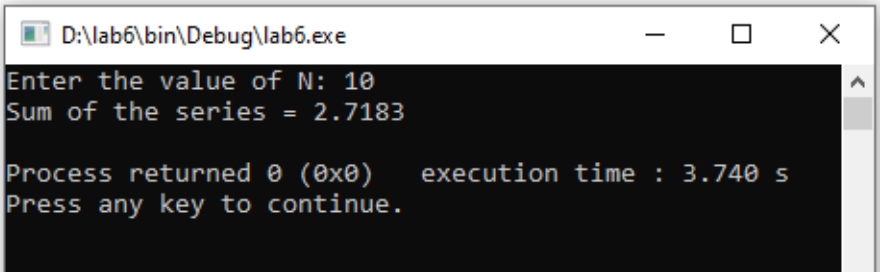
```c
#include <stdio.h>

int main() {
    int N, i, j, fact;
    float sum = 0.0;

    printf("Enter the value of N: ");
    scanf("%d", &N);

    for (i = 1; i <= N; i++) {
        fact = 1;
        for (j = 1; j <= i; j++) {
            fact *= j;
        }
        sum += (float)i / fact;
    }

    printf("Sum of the series = %.4f\n", sum);
    return 0;
}
```



Discussion:

The program calculates each factorial using a nested for loop. The division is cast to float to avoid integer division. The output converges close to the mathematical constant e as N increases.

So, This program successfully calculates the sum of the given factorial series using a for loop. It demonstrates the use of nested loops, factorial computation, and handling floating-point arithmetic in C.