Green University of Bangladesh
Department of Computer Science and Engineering

Program: CSE (Regular) **I** Spring 2023

Lab Report 3

Course Title: Computational Thinking and Problem Solving

Course Code: CSE 100

Section: 231 D1

Student Details

Name	ID No.
Mahmuda Akter Nadia	231002001

Lab Date: 30/05/23

Submission Date: 11/06/23

Course Instructor: Saurav Chandra Das

Lab Report Evaluation	
Marks:	Signature:
Comment:	Date:

Lab Report 3.1

Title: Find the sum of the first and last digits of any number

1. Introduction:

The objective of this lab is to develop a C program that calculates the sum of the first and last digits of any number. This lab report provides a detailed overview of the background, design, implementation, experimental setup, results and discussion, analysis, and summary of the program.

2. Background:

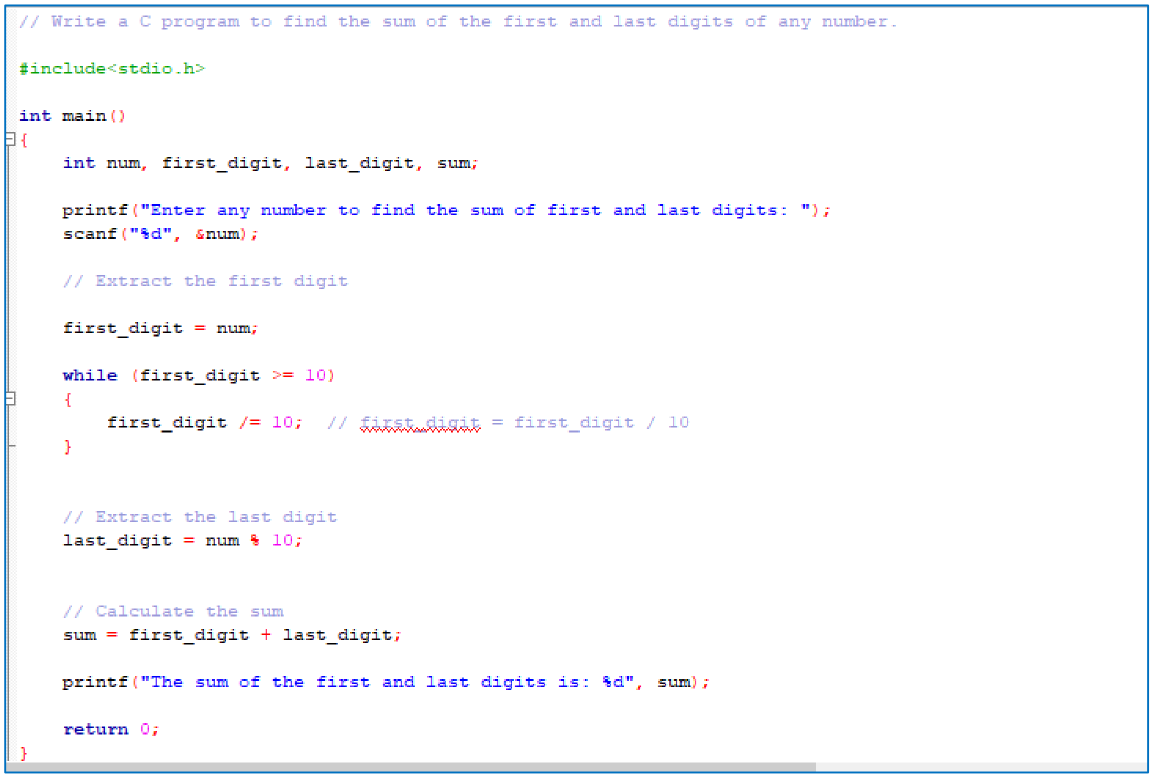
Finding the sum of the first and last digits of any number extracts the value of the first and last digits, computes their sum, and displays the result. The implementation involves basic arithmetic operations and the use of variables and control structures in C.

3. Design:

The design of the program involves the following key steps:

- Prompt the user to enter a number
- Read and store the input number
- Extract the first digit by repeatedly dividing the number by 10 until it becomes less than 10
- Extract the last digit using the modulus operator (%)
- Calculate the sum of the first and last digits
- Display the result to the user

4. Implementation:



```
// Write a C program to find the sum of the first and last digits of any number.

#include<stdio.h>

int main()
{
    int num, first_digit, last_digit, sum;

    printf("Enter any number to find the sum of first and last digits: ");
    scanf("%d", &num);

    // Extract the first digit

    first_digit = num;

    while (first_digit >= 10)
    {
        first_digit /= 10;   // first_digit = first_digit / 10
    }


    // Extract the last digit
    last_digit = num % 10;


    // Calculate the sum
    sum = first_digit + last_digit;

    printf("The sum of the first and last digits is: %d", sum);

    return 0;
}
```

In this program, we take an input number from the user. Then, we extract the first digit by continuously dividing the number by 10 until it becomes less than 10. We use the modulus operator (**%**) to extract the last digit by taking the remainder of the number when divided by 10. Finally, we calculate the sum of the first and last digits and display the result.

5. Experimental Setup:

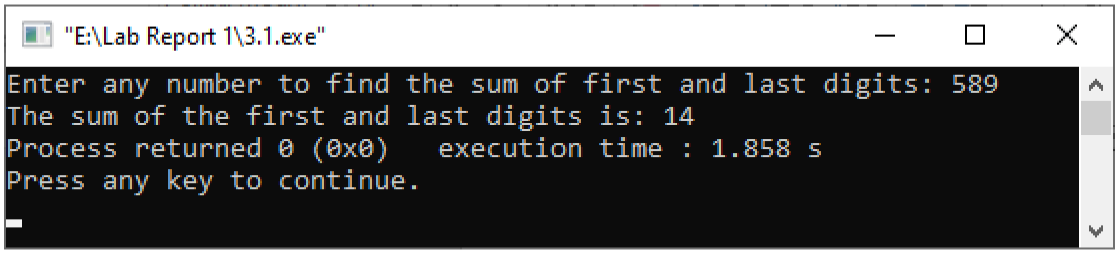
The program was implemented and tested using an Integrated Development Environment named "CodeBlocks" on a Microsoft operating system. The input for each test case was manually provided during the program execution.

6. Results and Discussion:

To ensure the correctness of the program, it has been conducted several test cases. Here are the results some:

Test Case 1:

- **Input:** 589
- **Output:** The sum of the first and last digits is: 14



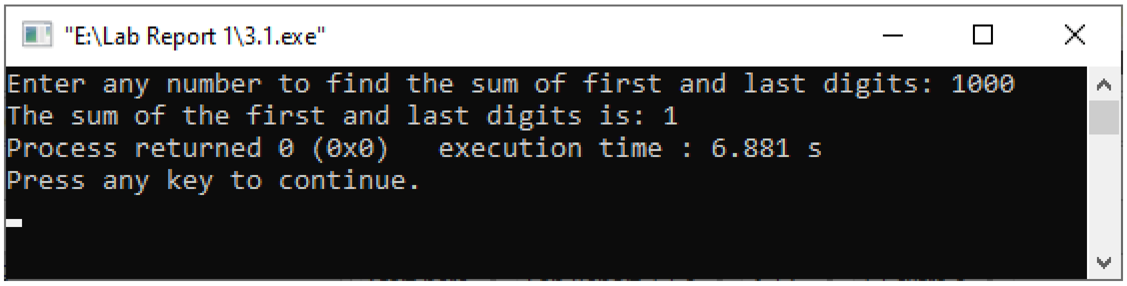
```
Enter any number to find the sum of first and last digits: 589
The sum of the first and last digits is: 14
Process returned 0 (0x0)   execution time : 1.858 s
Press any key to continue.
```

- **Explanation:**

In this case, the first and last digit of 589 is 5 and 9. So, the program correctly identifies 14 as the sum of the first and last digits.

Test Case 2:

- **Input:** 1000
- **Output:** The sum of the first and last digits is: 1



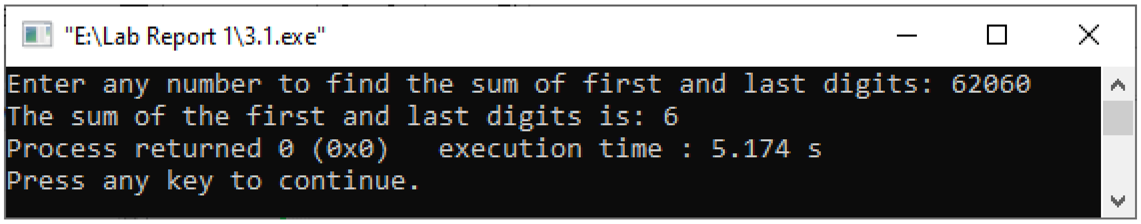
```
Enter any number to find the sum of first and last digits: 1000
The sum of the first and last digits is: 1
Process returned 0 (0x0)   execution time : 6.881 s
Press any key to continue.
```

- **Explanation:**

  In this case, the first and last digit of 1000 is 1 and 0. So, the program correctly identifies 1 as the sum of the first and last digits.

Test Case 3:

- **Input:** 62060
- **Output:** The sum of the first and last digits is: 6



"E:\Lab Report 1\3.1.exe"

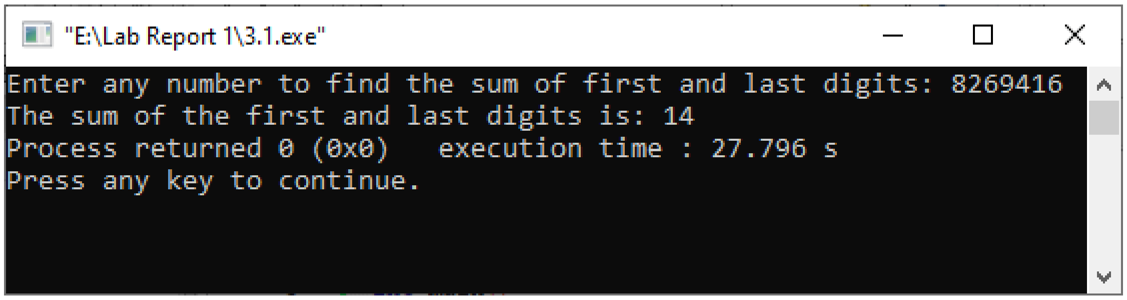
```
Enter any number to find the sum of first and last digits: 62060
The sum of the first and last digits is: 6
Process returned 0 (0x0)   execution time : 5.174 s
Press any key to continue.
```

- **Explanation:**

  The first and last digit of 62060 is 6 and 0. So, the program correctly displays 6 as the sum of the first and last digits.

Test Case 4:

- **Input:** 8269416
- **Output:** The sum of the first and last digits is: 14



"E:\Lab Report 1\3.1.exe"

```
Enter any number to find the sum of first and last digits: 8269416
The sum of the first and last digits is: 14
Process returned 0 (0x0)   execution time : 27.796 s
Press any key to continue.
```

- **Explanation:**

  The program correctly identifies 8 as the first and 6 as the last digit. Then, it displays that the sum of the first and last digits is 14.

For any value of the numbers, this program can determine the sum of the first and last digits.

7. Analysis and Discussions:

The program successfully achieves its objective of the sum of the first and last digits of any number. It utilized basic arithmetic operations, variables, and control structures to extract the necessary digits and compute their sum.

One possible limitation of the program is that it assumes the user will input a valid integer. If the user enters a non-integer value, unexpected behavior may occur. To address this, input validation techniques such as checking for valid input and handling error cases could be incorporated.

Overall, the program's logic and implementation were sound, providing an efficient solution to the problem at hand.

8. Summary:

The lab successfully produced a C program capable of finding the sum of the first and last digits of any input number. The program demonstrated the effective utilization of variables, arithmetic operations, and control structures in C programming. This exercise provided practical experience in developing basic programs, which will serve as a foundation for future programming tasks.

Lab Report 3.2

Title: Swap the first and last digits of any number

1. Introduction:

Swapping the first and last digits of a number is a common task in programming. The program uses basic arithmetic operations to achieve the desired result. The report provides an overview of the program's structure, explanations of the key algorithms used, and a discussion of the program's performance and limitations.

2. Background:

The objective of this lab is to develop a program in C language to swap the first and last digits of any number. By understanding the underlying algorithms and logic, we aim to gain a deeper knowledge of arithmetic operations and their applications in programming.

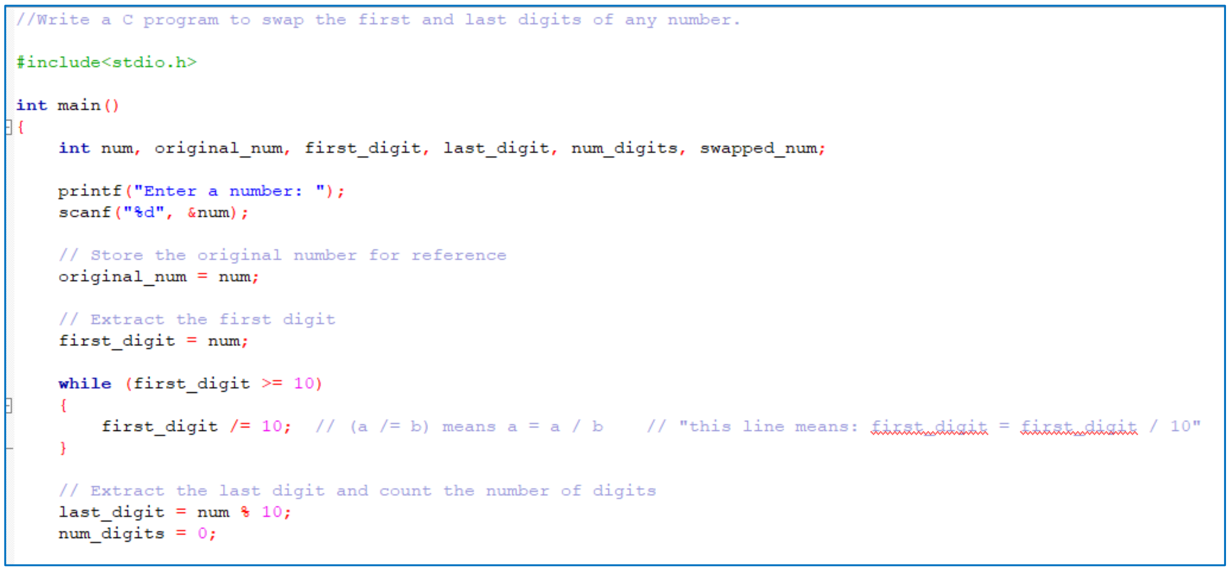
3. Design:

To swap the first and last digits in C, the following steps are used:

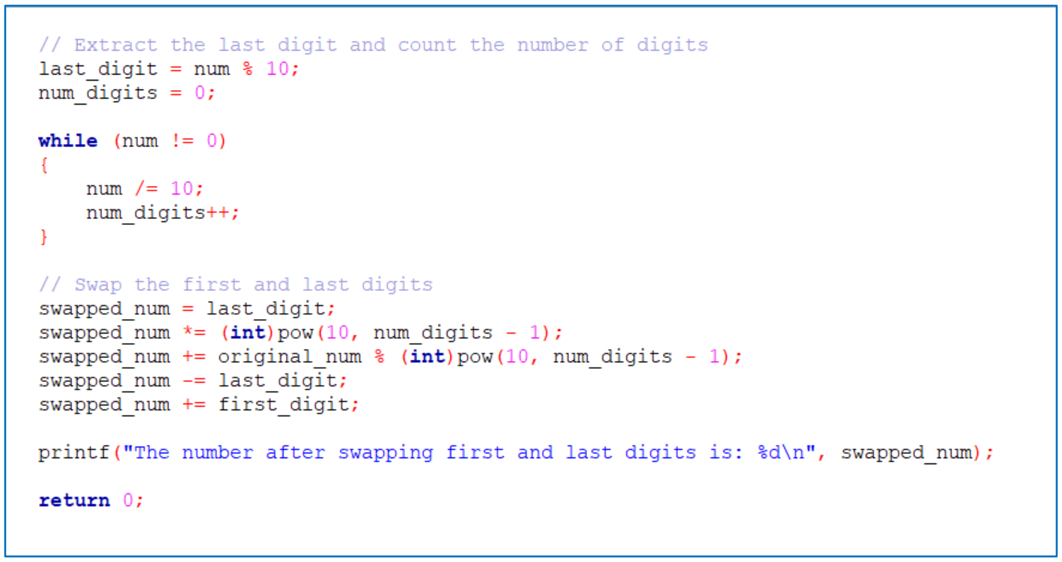
1. The program prompts the user to enter a number and read the input using the **scanf** function, then stored it in the **num** variable.
2. To preserve the original number for reference, the program assigns the value of **num** to the new variable **original_num**.
3. The last digit of the number is extracted by calculating **number % 10** and assigned to the **last_digit** variable.
4. The remaining digits of the number are obtained by dividing the number by 10 and assigning the result to the **first_digit** variable.

5. The program counts the number of digits in the given number by iteratively dividing the number by 10 until it becomes zero. The count is stored in the **num_digits** variable.
6. To compute the swapped number, the program initializes the **swapped_num** variable with the last digit (**last_digit**).
7. Using arithmetic operations, the program completes the swap operation and prints the result of **swapped_num** using the **printf** function.

4. Implementation:



```c
//Write a C program to swap the first and last digits of any number.

#include<stdio.h>

int main()
{
    int num, original_num, first_digit, last_digit, num_digits, swapped_num;

    printf("Enter a number: ");
    scanf("%d", &num);

    // Store the original number for reference
    original_num = num;

    // Extract the first digit
    first_digit = num;

    while (first_digit >= 10)
    {
        first_digit /= 10;  // (a /= b) means a = a / b    // "this line means: first_digit = first_digit / 10"
    }

    // Extract the last digit and count the number of digits
    last_digit = num % 10;
    num_digits = 0;
```



```c
    // Extract the last digit and count the number of digits
    last_digit = num % 10;
    num_digits = 0;

    while (num != 0)
    {
        num /= 10;
        num_digits++;
    }

    // Swap the first and last digits
    swapped_num = last_digit;
    swapped_num *= (int)pow(10, num_digits - 1);
    swapped_num += original_num % (int)pow(10, num_digits - 1);
    swapped_num -= last_digit;
    swapped_num += first_digit;

    printf("The number after swapping first and last digits is: %d\n", swapped_num);

    return 0;
```

5. Experimental Setup:

The program was implemented and tested using an Integrated Development Environment named “CodeBlocks” on a Microsoft operating system. The input for each test case was manually provided during the program execution.

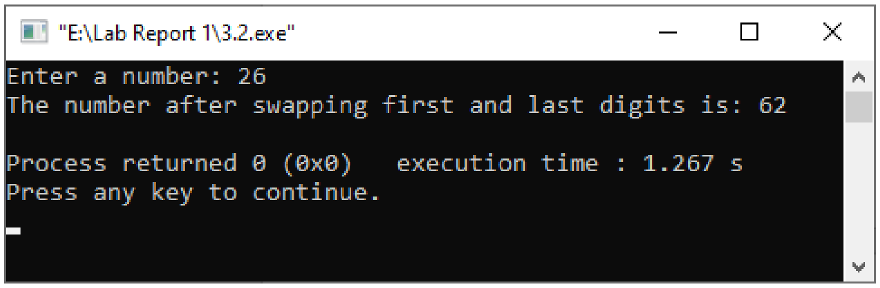
6. Results and Discussion:

The implemented program was tested with various inputs to verify its correctness and accuracy. The following results were obtained:

Test Case 1:

- **Input:** 26

- **Output:** The number after swapping first and last digits is: 62



```
"E:\Lab Report 1\3.2.exe"
Enter a number: 26
The number after swapping first and last digits is: 62

Process returned 0 (0x0)   execution time : 1.267 s
Press any key to continue.
```

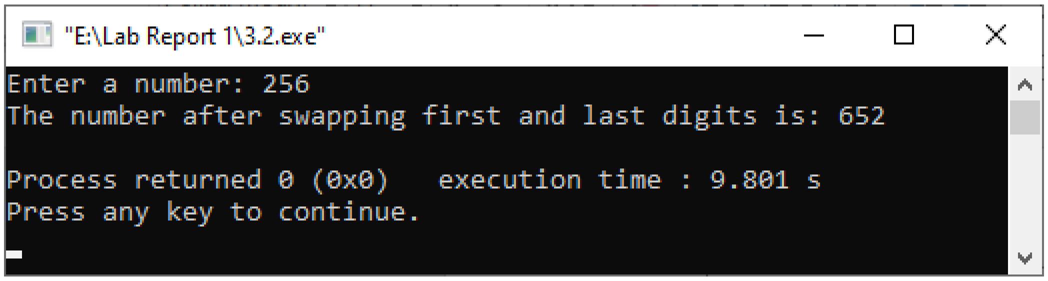
- **Explanation:**

  In this case, the program correctly swaps the number 26 as 62.

Test Case 2:

- **Input:** 256

- **Output:** The number after swapping first and last digits is: 652



```
"E:\Lab Report 1\3.2.exe"
Enter a number: 256
The number after swapping first and last digits is: 652

Process returned 0 (0x0)   execution time : 9.801 s
Press any key to continue.
```

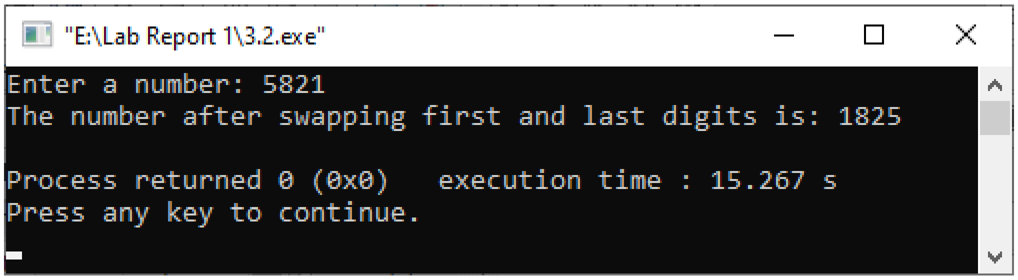
- **Explanation:**

  Here, it displays 652 after swapping between the first digit 2 and the last digit 6.

Test Case 3:

- **Input:** 5821

- **Output:** The number after swapping first and last digits is: 1825



```
"E:\Lab Report 1\3.2.exe"
Enter a number: 5821
The number after swapping first and last digits is: 1825

Process returned 0 (0x0)   execution time : 15.267 s
Press any key to continue.
```

- **Explanation:**

  In this case, the program correctly displays 1825 after swapping 5 and 1.

The program displayed accurate results for a variety of test cases, proving its correctness and functionality.

7. Analysis and Discussions:

Upon analyzing the program and its results, the following observations can be made:

- The program successfully identifies the first and last digit and displays the correct output of any given number after swapping the first and last digit by implementing a C program.
- The implementation has been done by using several arithmetic operations and while statements.

8. Summary:

This lab demonstrates an understanding of extracting digits, counting digits, and performing the swap operation during implementing this program.

Lab Report 3.3

Title: Calculate the product of the digits of any number

1. **Introduction:**

   The objective of this lab is to develop a C program that calculates the product of the digits of any given number. This lab provides an opportunity to practice fundamental programming concepts such as variables, loop and arithmetic operations in C.

2. **Background:**

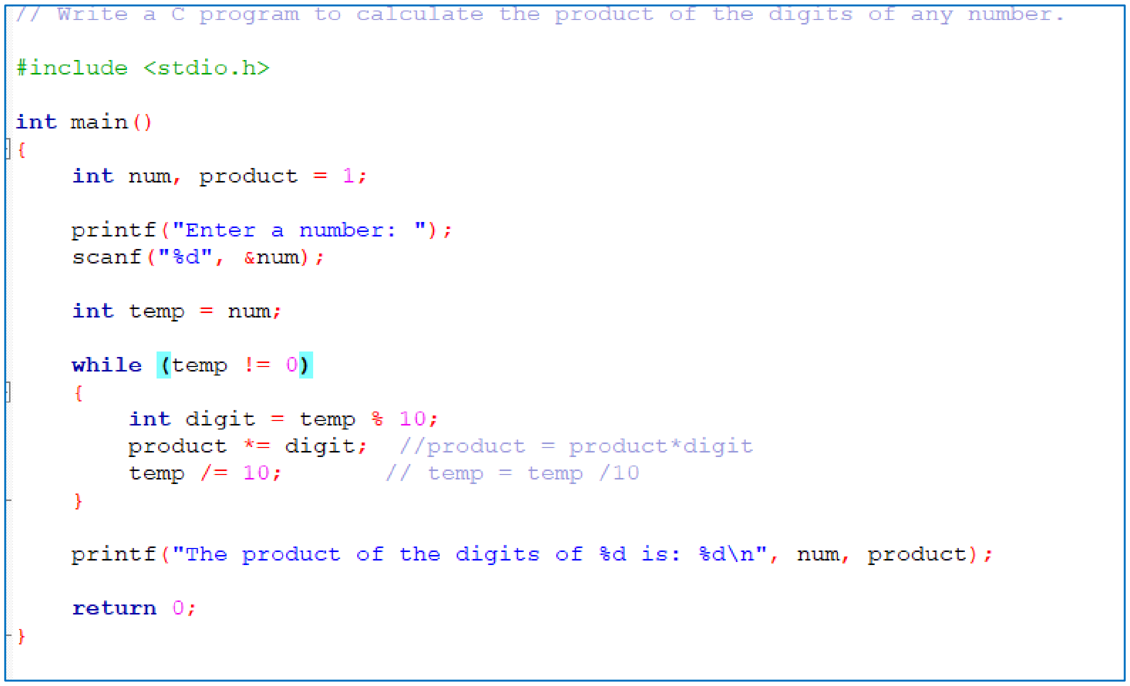
   The program should prompt the user to enter a number, compute the product of its digits using a loop, and display the result.

3. **Design:**

   To calculate the product of the digits of any number in C, the following steps are used:

   1. Prompt the user to enter a number.
   2. Read and store the input in the variable 'num'.
   3. Initialize a variable 'product' to 1.
   4. Create a temporary variable 'temp' and assign it the value of 'num'.
   5. Start a loop that continues until 'temp' becomes 0.
   6. Inside the loop, extract the rightmost digit of 'temp' using the modulo operator and store it in a variable 'digit'.
   7. Multiply 'digit' with 'product' and assign the result back to 'product'.
   8. Divide 'temp' by 10 to remove the rightmost digit.
   9. Repeat steps 7-9 until the loop completes.
   10. Display the value of 'product' as the product of the digits of 'num'.

4. Implementation:



```c
// Write a C program to calculate the product of the digits of any number.

#include <stdio.h>

int main()
{
    int num, product = 1;

    printf("Enter a number: ");
    scanf("%d", &num);

    int temp = num;

    while (temp != 0)
    {
        int digit = temp % 10;
        product *= digit;   //product = product*digit
        temp /= 10;        // temp = temp /10
    }

    printf("The product of the digits of %d is: %d\n", num, product);

    return 0;
}
```

5. Experimental Setup:

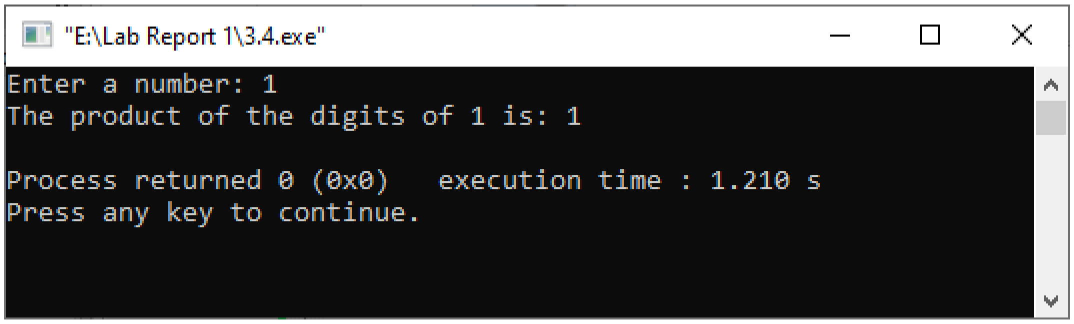
The program was implemented and tested using an Integrated Development Environment named “CodeBlocks” on a Microsoft operating system. The input for each test case was manually provided during the program execution.

6. Results and Discussion:

The implemented program was tested with various inputs to verify its correctness and accuracy. The following results were obtained:

Test Case 1:

- **Input:** 1
- **Output:** The product of the digits of 1 is: 1



"E:\Lab Report 1\3.4.exe"

```
Enter a number: 1
The product of the digits of 1 is: 1

Process returned 0 (0x0)   execution time : 1.210 s
Press any key to continue.
```

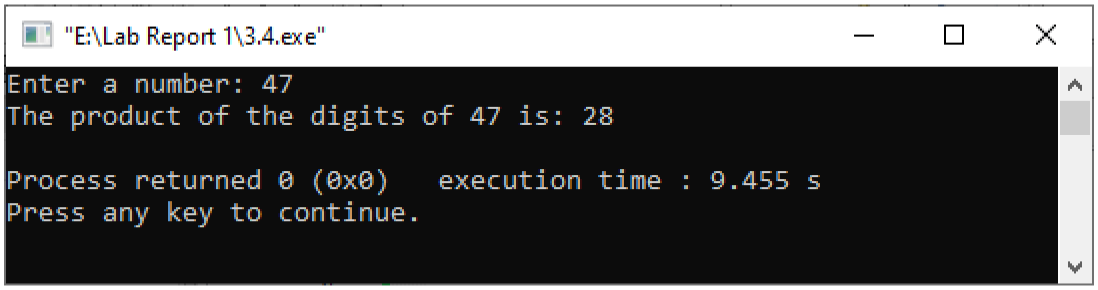
- **Explanation:**

  In this case, the digit of given number is only 1. The program correctly identifies 1 as the product of the digit of the given number 1.

Test Case 2:

- **Input:** 47

- **Output:** The product of the digits of 47 is: 28



"E:\Lab Report 1\3.4.exe"

```
Enter a number: 47
The product of the digits of 47 is: 28

Process returned 0 (0x0)   execution time : 9.455 s
Press any key to continue.
```

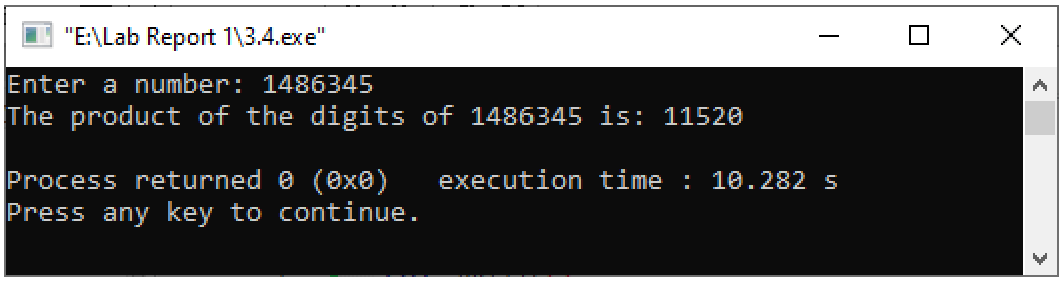
- **Explanation:**

  In this case, the given number has two digits 4 and 7. The program shows the product of the digits of the given number 28.

Test Case 3:

- **Input:** 1486345

- **Output:** The product of the digits of 1486345 is: 11520



"E:\Lab Report 1\3.4.exe"

```
Enter a number: 1486345
The product of the digits of 1486345 is: 11520

Process returned 0 (0x0)   execution time : 10.282 s
Press any key to continue.
```

- **Explanation:**

  In this case, the given number has seven digits. The program correctly shows that the product of the digits of the given number is 1*4*8*6*3*4*5 = 11520.

The program displayed accurate results for a variety of test cases, proving its correctness and functionality.

7. Analysis and Discussions:

Upon analyzing the program and its results, the following observations can be made:

- The program accurately identifies each digits of any given number.
- The implementation uses a conditional statement and arithmetic operation to ensure accurate results.
- The implementation of the program is concise and efficient, requiring minimal lines of code.

8. Summary:

The lab was successfully completed by implementing a C program to calculate the product of the digits of any given number. In summary, the importance of this program lies in its ability to identify separately each digit of the number, application of decision-making and conditional statements, its relevance in mathematics, its emphasis on input validation, its potential for code reusability, and its educational value for learning programming concepts.

Lab Report 3.4

Title: Find the sum of the series 1 + 11 + 111 + 1111 + .. n terms

1. **Introduction:**

   The objective of this lab is to develop a C program to calculate the sum of the series 1 + 11 + 111 + 1111 + ... up to 'n' terms. This lab provides an opportunity to practice fundamental programming concepts such as loops, variables, and arithmetic operations in C.

2. **Background:**

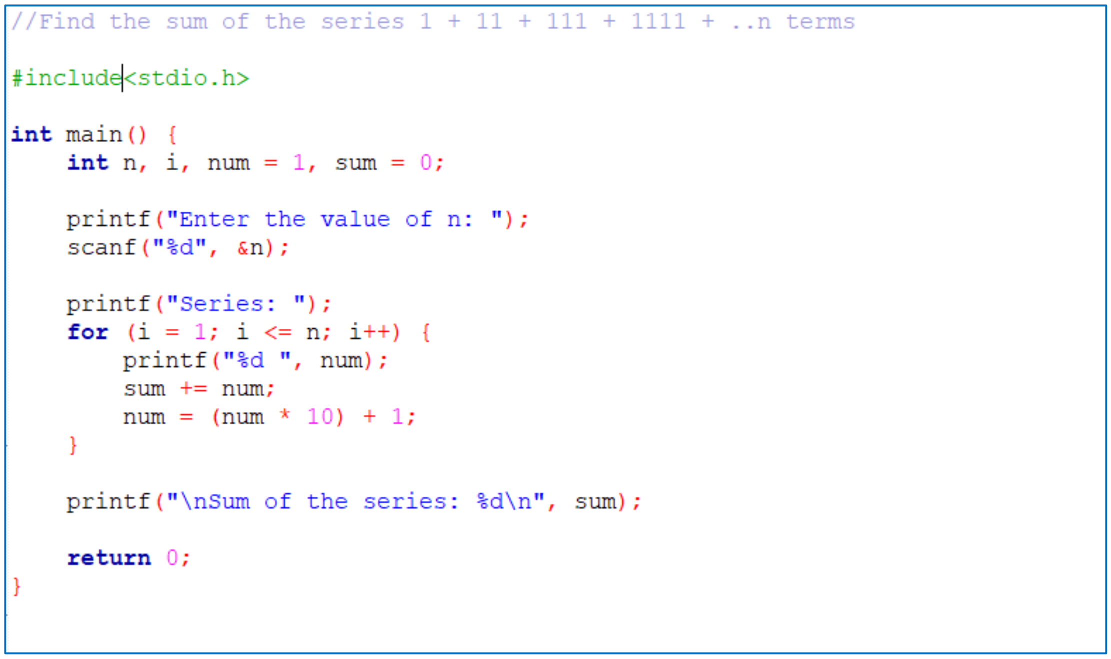
   This program follows a unique pattern: each term in the series is formed by concatenating the digit '1' to the previous term. For example, the first term is '1', the second term is '11', the third term is '111', and so on. This program prompts the user to enter the number of terms in the series, compute the sum using a loop, and display the result.

3. **Design:**

   To calculate the sum of the mentioned series in C, the following steps are used:

   1. Prompt the user to enter the number of terms in the series
   2. Read and store the value of 'n'
   3. Initialize a variable 'num' to 1 and 'sum' to 0
   4. Start a loop with a variable 'i' from 1 to 'n'.
   5. Inside the loop, add 'num' to 'sum'.
   6. Update the value of 'num' by multiplying it by 10 and adding 1.
   7. Repeat steps 6 and 7 until the loop completes.
   8. Print the value of 'sum' as the sum of the series.

4. Implementation:



```c
//Find the sum of the series 1 + 11 + 111 + 1111 + ..n terms

#include<stdio.h>

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

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

    printf("Series: ");
    for (i = 1; i <= n; i++) {
        printf("%d ", num);
        sum += num;
        num = (num * 10) + 1;
    }

    printf("\nSum of the series: %d\n", sum);

    return 0;
}
```

5. Experimental Setup:

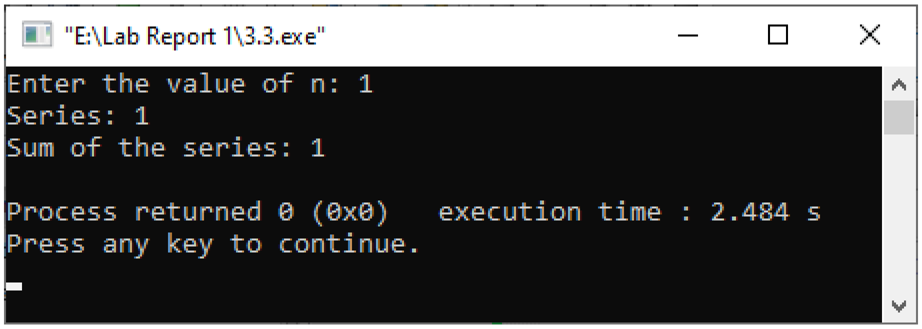
The program was implemented and tested using an Integrated Development Environment named "CodeBlocks" on a Microsoft operating system. The input for each test case was manually provided during the program execution.

6. Results and Discussion:

The implemented program was tested with various inputs to verify its correctness and accuracy. The following results were obtained:

Test Case 1:

- **Input:** 1
- **Output:** 1



"E:\Lab Report 1\3.3.exe"

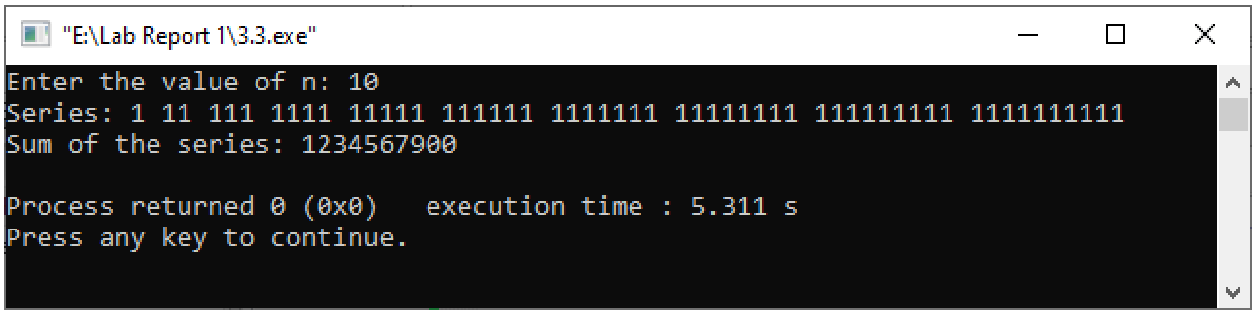
```
Enter the value of n: 1
Series: 1
Sum of the series: 1

Process returned 0 (0x0)   execution time : 2.484 s
Press any key to continue.
```

- **Explanation:**

  In this case, the series is 1. So, it correctly displays the sum of the series 1.

Test Case 2:

- **Input:** 10
- **Output:** 1234567900



"E:\Lab Report 1\3.3.exe"

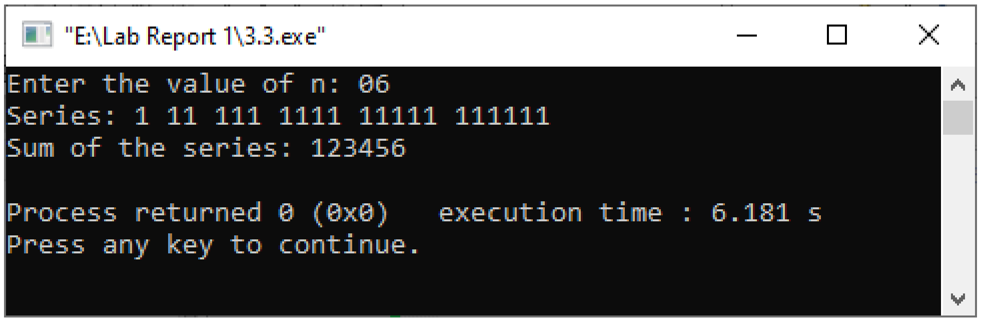
```
Enter the value of n: 10
Series: 1 11 111 1111 11111 111111 1111111 11111111 111111111 1111111111
Sum of the series: 1234567900

Process returned 0 (0x0)   execution time : 5.311 s
Press any key to continue.
```

- **Explanation:**

  In this case, the n series has continued to 10. The program correctly shows the output of the sum of the given series 1234567900.

Test Case 3:

- **Input:** 06
- **Output:** 123456



"E:\Lab Report 1\3.3.exe"

```
Enter the value of n: 06
Series: 1 11 111 1111 11111 111111
Sum of the series: 123456

Process returned 0 (0x0)   execution time : 6.181 s
Press any key to continue.
```

- **Explanation:**

  The n series has continued to 06. The program correctly shows the output of the sum of the given series 123456.

The program displayed accurate results for a variety of test cases, proving its correctness and functionality.

7. Analysis and Discussions:

Upon analyzing the program and its results, the following observations can be made:

- The program accurately determines the series based on the given number.
- The implementation uses a conditional statement to evaluate the sum of the mentioned series, which is an efficient approach for this task.
- The implementation of the program is concise and efficient, requiring minimal lines of code.

8. Summary:

The program successfully computes the sum of the series 1 + 11 + 111 + 1111 + ... up to 'n' terms. It demonstrates an understanding of loop structures, variables, and basic arithmetic operations in C. By executing the program, we can easily determine the sum of the series for any given number of terms.