

1 Introduction

The C programming language, developed in the early 1970s by Dennis Ritchie at Bell Labs, is one of the most influential and widely utilized programming languages. It serves as the foundation for numerous contemporary languages, including C++, Java, and Python. In this report, we will elucidate five fundamental C programs, demonstrating essential programming concepts such as input/output operations, arithmetic computations, and error handling. Each program comprises a comprehensive algorithm, flow diagram, code, and output, which facilitates a thorough understanding of these fundamental principles.

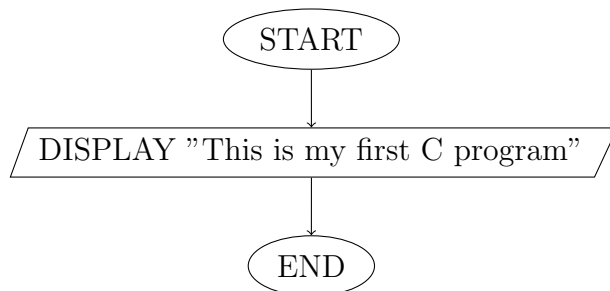
2 C Programs and Flowcharts and Algorithms

2.1 Task: Print "Hello, World"

Algorithm:

1. START.
2. DISPLAY "This is my first C program".
3. END.

Flowchart:



Code:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("This is my first C program\n");
5     return 0;
6 }
```

Listing 1: Hello, World Program

Output:

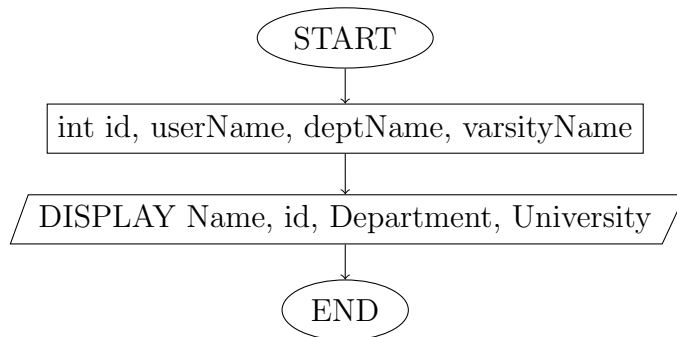
```
● shoruv@Shahidurs-MacBook-Air CSE 1104 - Lab 1 % cd
s/GitHub/CSE 1104 - Lab 1/"task1
This is my first C program
○ shoruv@Shahidurs-MacBook-Air CSE 1104 - Lab 1 % █
```

2.2 Task: Print Student Information

Algorithm:

1. START.
2. DECLARE id, userName, deptName, and varsityName.
3. DISPLAY id, userName, deptName, varsityName.
4. END.

Flowchart:



Code:

```
1 #include <stdio.h>
2
3 int main(){
4     int id = 240126;
5     char userName[] = "MD. Shahidur Rahman";
6     char deptName[] = "Computer Science and Engineering";
7     char varsityName[] = "Pabna University of Science and Technology";
8
9     printf("Name: %s\n", userName);
10    printf("id: %d\n", id);
11    printf("Department: %s\n", deptName);
12    printf("University: %s\n", varsityName);
13
14    return 0;
15 }
```

Listing 2: Printing Student Information

Output:

```
shoruv@Shahidurs-MacBook-Air CSE 1104 - Lab 1 % cd "/Users/shoruv/
s/GitHub/CSE 1104 - Lab 1/"task2
Name: MD. Shahidur Rahman
id: 240126
Department: Computer Science and Engineering
University: Pabna University of Science and Technology
shoruv@Shahidurs-MacBook-Air CSE 1104 - Lab 1 %
```

2.3 Task: Adding Two Integers

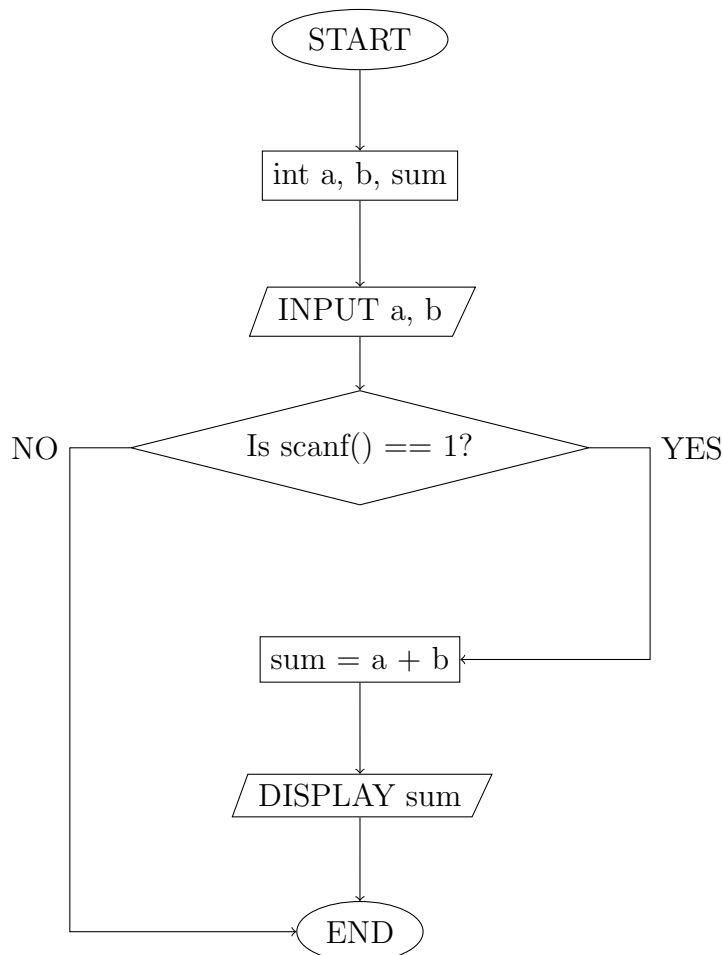
Algorithm:

1. START.
2. DECLARE `a`, `b`, and `sum`.
3. INPUT `a` , `b`.
4. IF `scanf() == 1` THEN

 `sum = a + b;`

 ELSE
 RETURN 0;
 END IF
5. DISPLAY `sum`.
6. END.

Flowchart:



Code:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Adding two numbers \n");
5     printf("-----\n");
6
7     int a,b,sum;
8
9     printf("Enter two numbers to add: ");
10    if(!scanf("%d %d", &a, &b)) {
11        printf("Invalid Input!\n");
12        return 0;
13    }
14
15    sum = a + b;
16    printf("%d + %d = %d\n", a, b, sum);
17
18    return 0;
19 }
```

Listing 3: Adding Two Integers

Input: 5 8

Output:

```
Adding two numbers
-----
Enter two numbers to add: 5 8
5 + 8 = 13
shoruv@Shahidurs-MacBook-Air CSE 1104 - Lab 1 % >
```

2.4 Task: Adding Two Floating-Point Numbers

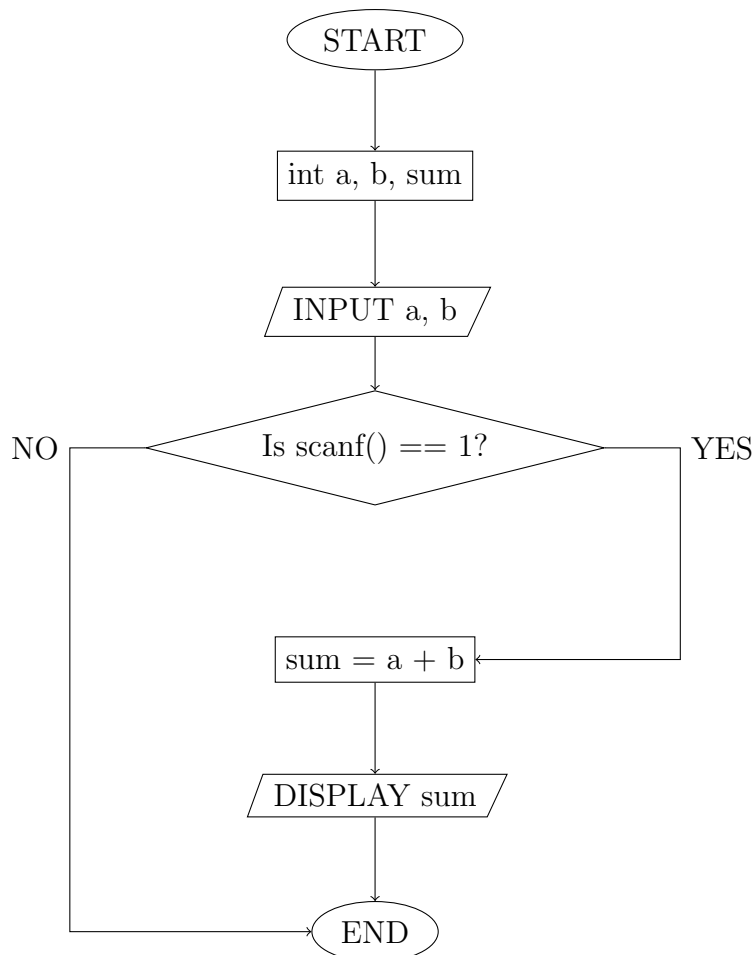
Algorithm:

1. START
2. DECLARE float **a**, **b**, and **sum**.
3. INPUT **a**, **b**.
4. IF `scanf() == 1` THEN

 `sum = a + b;`

 ELSE
 RETURN 0;
 END IF
5. DISPLAY **sum**.
6. END.

Flowchart:



Code:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Adding two float numbers \n");
5     printf("-----\n");
6
7     float a,b,sum;
8
9     printf("Enter two float numbers to add: ");
10    if(!scanf("%f %f", &a, &b)) {
11        printf("Invalid Input!\n");
12        return 0;
13    }
14
15    sum = a + b;
16    printf("%.2f + %.2f = %.2f\n", a, b, sum);
17
18    return 0;
19 }
```

Listing 4: Adding Two Floating-Point Numbers

Input: 10.5 20.3

Output:

```
Adding two float numbers
-----
Enter two float numbers to add: 10.5 20.3
10.50 + 20.30 = 30.80
○ shoruv@Shahidurs-MacBook-Air CSE 1104 - Lab 1 %
```

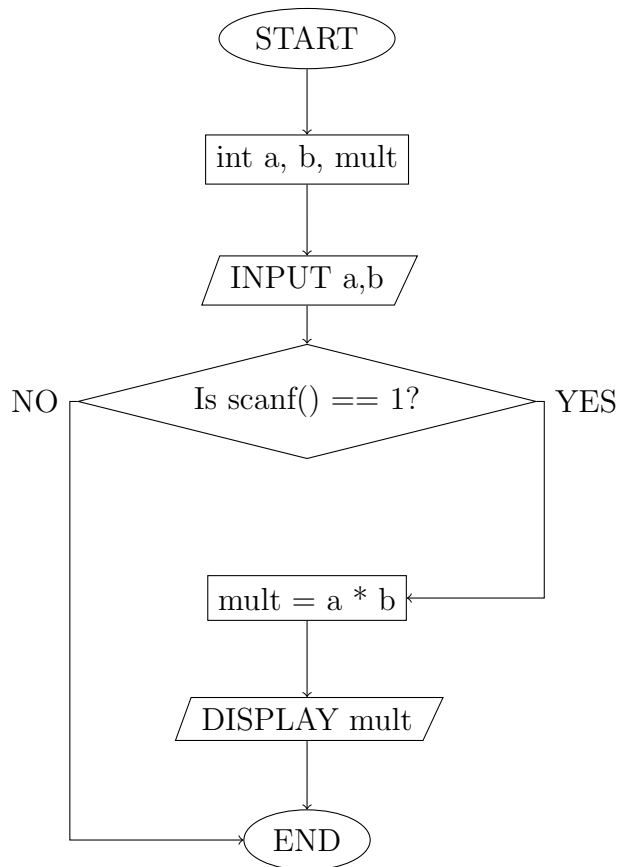
2.5 Task: Multiplying Two Numbers

Algorithm:

1. START.
2. DECLARE int a, b, and mult.
3. INPUT a , b.
4. IF scanf() == 1 THEN
 mult = a * b;

 ELSE
 RETURN 0;
 END IF
5. If the input is valid, calculate the multiplication of a and b and store it in mult.
6. DISPLAY mult
7. END.

Flowchart:



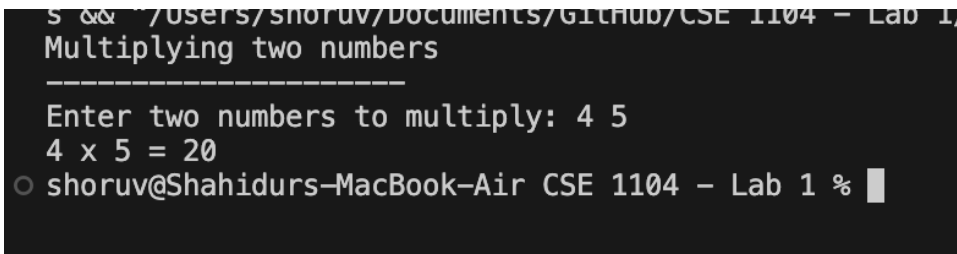
Code:

```
1 #include <stdio.h>
2
3 int main() {
4     printf("Multiplying two numbers \n");
5     printf("-----\n");
6
7     int a,b,mult;
8
9     printf("Enter two numbers to multiply: ");
10    if(!scanf("%d %d", &a, &b)) {
11        printf("Invalid Input!\n");
12        return 0;
13    }
14
15    mult = a * b;
16    printf("%d x %d = %d\n", a, b, mult);
17
18    return 0;
19 }
```

Listing 5: Multiplying Two Numbers

Input: 4 5

Output:



```
S && ~7users/shoruv/Documents/GitHub/CSE 1104 - Lab 1/
Multiplying two numbers
-----
Enter two numbers to multiply: 4 5
4 x 5 = 20
shoruv@Shahidurs-MacBook-Air CSE 1104 - Lab 1 %
```

3 Discussion

In this report, we have delved into five fundamental C programs that concentrate on handling input/output operations, performing arithmetic operations utilizing algorithms, flowcharts, and C code. These concepts form the foundation of structured programming. It is imperative to exercise caution with syntax, as common errors such as missing semicolons and misspelling of reserved keywords can lead to linker command failures. For example, I mistakenly typed “mian()” instead of “main(),” resulting in a linker command error. Therefore, it is crucial to memorize and exercise caution while typing the code. When compiling C code for the first time, it is necessary to install a compiler such as MinGW or GCC to support the code. Computers lack the ability to understand the code on their own; hence, it is essential to provide proper input to obtain the desired output.