Programming Fundamentals
(BCL 103)

Lab Practical Report

Faculty name: Deepika Garg

Student Name: Bhumi Nabiyal

Roll No.: 24BCA046

Semester: First Semester

Group: 2 (Second)

Department of Computer Science and Engineering

NorthCap University, Gurugram- 122001, India

Session 2024-25

INDEX

S. No	Experiment	Page No.	Date of Experiment	Date of Submission	Marks	CO Covered	Signature

Experiment- 1

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First Sem.
Link to Code:
Date: 26 oct. 2024
Faculty Signature:

Objective

To familiarize the students about Number Systems

Program Outcome

- The students will be able to represent numbers in various numeration systems , convert a numeral from one base to another base and perform arithmetic operations in bases other than ten

Background Study:

A number system is defined as a system of writing for expressing numbers. It is the mathematical notation for representing numbers of a given set by using digits in a consistent manner. It provides a unique representation of every number and represents the arithmetic and algebraic structure of the figures. It also allows us to operate arithmetic operations like addition, subtraction, and division.

There are various types of the number system in mathematics. The four most common number system types are:

System	Base	Symbols

Decimal	10	0, 1, … 9
Binary	2	0, 1
Octal	8	0, 1, … 7
Hexa-decimal	16	0, 1, … 9, A, B, … F

Perform below Number system conversions, write all steps with the outcome:

1. Convert $(1056)_{16}$ to (? $)_8$
2. Convert $(11672)_8$ to (? $)_{16}$
3. Convert $(2724)_8$ to (? $)_{10}$
4. Convert $(3211)_4$ to (? $)_5$

5. Convert $(1001001100)_2$ to (? $)_6$

Experiment-2

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First Sem.
Link to Code:
Date: 26 oct. 2024
Faculty Signature:

Objective

To familiarize the students about Number Systems

Program Outcome

- The students will be able to represent numbers in various numeration systems , convert a numeral from one base to another base and perform arithmetic operations in bases other than ten

Background Study:

A number system is defined as a system of writing for expressing numbers. It is the mathematical notation for representing numbers of a given set by using digits in a consistent manner. It provides a unique representation of every number and represents the arithmetic and algebraic structure of the figures. It also allows us to operate arithmetic operations like addition, subtraction, and division.

There are various types of the number system in mathematics. The four most common number system types are:

System	Base	Symbols

Decimal	10	0, 1, … 9
Binary	2	0, 1
Octal	8	0, 1, … 7
Hexa-decimal	16	0, 1, … 9, A, B, … F

1. Convert the following 4-bit numbers from binary to decimal.
   a. 0101_2
   b. 0111_2
   c. 0011_2
   d. 1001_2
   e. 1011_2
   f. 1111_2
   g. 0000_2
   h. 1101_2
2. Convert the following 8-bit numbers from binary to decimal.
   a. 00010101_2
   b. 10110101_2
   c. 11010011_2
   d. 01101000_2
3. Convert the following 16-bit numbers from binary to decimal.
   a. $10110101\ 00010101_2$
   b. $01101000\ 11010011_2$

4. Determine whether the following statements are true or false.
   a. $1001_2 < 5_{10}$
   b. $0111_2 = 111_{10}$
   c. $0011_2 > 2_{10}$
   d. $1001_2 > 1101_2$

e. $1011_2 = 11_{10}$

f. $1111_2 = 15_{10}$

g. $0000_2 < 0_{10}$

h. $1101_2 > 1010_2$

Experiment-3

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First Sem.
Link to Code:
Date: 26 oct. 2024
Faculty Signature:

Objective

To familiarize the students about Number Systems

Program Outcome

- The students will be able to represent numbers in various numeration systems , convert a numeral from one base to another base and perform arithmetic operations in bases other than ten

Background Study:

A number system is defined as a system of writing for expressing numbers. It is the mathematical notation for representing numbers of a given set by using digits in a consistent manner. It provides a unique representation of every number and represents the arithmetic and algebraic structure of the figures. It also allows us to operate arithmetic operations like addition, subtraction, and division.

There are various types of the number system in mathematics. The four most common number system types are:

System	Base	Symbols
Decimal	10	0, 1, … 9
Binary	2	0, 1

Octal	8	0, 1, … 7
Hexa-decimal	16	0, 1, … 9, A, B, … F

1. What is the decimal equivalent of the number $3A_{16}$?
2. What is the 8 bit unsigned binary result of $56_{10} - 31_{10}$ = 25
3. What is the result of adding 7_{10} and -4_{10} using 8 bit signed binary notation?
4. Which of the following 4 bit Excess 3 numbers is equivalent to 5_{10}?
5. Consider the equation $(123)_5 = (x8)_y$ with x and y as unknown. The number of possible solutions is _____ .
6. Convert binary 111111110010 to hexadecimal.
7. The representation of octal number $(532.2)_8$ in decimal is _____
8. The decimal equivalent of the octal number $(645)_8$ is _____
9. The quantity of double word is _____
10. Octal to binary conversion: $(24)_8 =$?
11. Convert binary to octal: $(110110001010)_2 =$?
12. The octal number *(651.124)8* is equivalent to (_____)$_{10}$.
13. Convert the hexadecimal number $(1E2)_{16}$ to decimal:
14. Let r denote number system radix. The only value(s) of r that satisfy the equation $\sqrt{121_r} = 11_r$ is/are

Experiment- 4

Student Name and Roll Number: Bhumi Nabiyal (24bca046)	
Semester /Section: First Sem.	
Link to Code:	
Date: 26 oct. 2024	
Faculty Signature:	

Objective: Basic C Programming Questions

Program Outcome: Basic c programming understanding and able to perform simple programming questions

Background   Study: Input/Output   in   C can   be   achieved using scanf() and printf() functions. The printf and scanf are two of the many functions found in the C standard library. These functions are declared and related macros are defined in stdio.h header file. The printf function is used to write information from a program to the standard output device whereas scanf function is used to read information into a program from the standard input device.

Function Prototype of printf and scanf in C

Function name	Function prototype
Printf	int printf(const char* format, ...);
Scanf	int scanf(const char* format, ...);

Format Specifier of printf and scanf Functions

Format specifier	Description
%d	Signed decimal integer
%u	Unsigned decimal integer
%f	Floating point numbers
%c	Character
%s	Character String terminated by '\0'
%p	Pointer address

Q1. Write a C program to add two integer numbers.

Code:

```c
#include<stdio.h>
int main() {
    int num1, num2, sum;

    printf("Enter first integer: ");
    scanf("%d", &num1);

    printf("Enter second integer: ");
    scanf("%d", &num2);

    sum = num1 + num2;

    printf("Sum: %d", sum);

    return 0;
```

```
}
```

Output and screenshot:

```
Output                                          Clear

/tmp/Lt2vtChKOT.o
Enter first integer: 10
Enter second integer: 15
Sum: 25

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

Q2. Write a C Program to add two floating numbers.

Code:

```c
#include<stdio.h>
int main() {
    float num1, num2, sum;

    printf("Enter first integers: ");
    scanf("%f", &num1);

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

```
scanf("%f", &num2);


sum = num1 + num2;


printf("Sum: %.2f", sum);


return 0;

}
```

Output and screenshot:

Output	Clear
/tmp/yyWxUPWs7z.o Enter first integers: 1.50 Enter second integers: 3.75 Sum: 5.25 === Code Execution Successful ===	

Q3. Write a c program to display Hello World!

Code:

```
#include <stdio.h>

int main() {

printf("Hello World!");

return 0;
```

}

Output and screenshot:

```
Output                                          Clear

/tmp/mGwjWcTuXI.o
Hello World!

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

Q4. Write a C Program to check if the number entered by the user is even or odd.Code:

#include <stdio.h>

int main() {

int num;

printf("Enter an number to check odd or even : ");

scanf("%d", &num);

if (num % 2 == 0)

printf("%d is even number", num);

else

printf("%d is odd number", num);

return 0;

}

Output and screenshot:

```
Output                                    Clear

/tmp/OFjxwhODMg.o
Enter an number to check odd or even : 15
15 is odd number

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

Q5. Write a program to check whether the character value entered by the user is Vowel or Consonant.

Code:

```
#include <stdio.h>

    int main() {

    char ch;

    printf("Enter a character: ");

    scanf("%c", &ch);

    if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' || ch == 'A' || ch ==
'E' || ch == 'I' || ch == 'O' || ch == 'U') {

    printf("%c is a vowel", ch);

    }

    else {

    printf("%c is a consonant", ch);

    }
```

```
    return 0;

}
```

Output and screenshot:

Output   Clear

/tmp/Eb9nkcM1pH.o
Enter a character: E
E is a vowel

=== Code Execution Successful ===

Experiment - 5

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First sem.
Link to Code:
Date: 26 oct. 2024
Faculty Signature:

Objective: To study pseudocode and flowcharts basics, symbols, practical usage.

Program Outcome: The various examples of the flowchart

Background Study: A flowchart is **a picture of the separate steps of a process in sequential order**. It is a generic tool that can be adapted for a wide variety of purposes, and can be used to describe various processes, such as a manufacturing process, an administrative or service process, or a project plan. A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows

Q1. Design a flowchart for adding two numbers entered by the user.

Q2: Design a flowchart for finding the largest among three numbers entered by the user.

Q3: Design a flowchart for calculating the profit and loss according to the cost price and income value entered by the user.

Q4: Draw a flowchart to calculate the average of two numbers.

Q5: Design a flowchart for the multiplication of three numbers entered by the user.

Q6: Design a flowchart for calculating the area of a rectangle.

Q7: Design a flowchart for calculating Simple Interest according to the value of principal amount, ROI, and time the user enters.

Q8: Design a flowchart for checking whether the number is positive or negative according to the number entered by the user.

Prog.Fund.-1 (BCL103), 2024-25

Practical No 6

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First Sem.
Link to Code:
Date: 26 oct. 2024
Faculty Signature:

Objective: To study different operators in c programming language.

Program Outcome: n operator is a symbol that tells the compiler to perform specific mathematical or logical functions. C language is rich in built-in operators and provides the following types of operators −

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Bitwise Operators
- Assignment Operators

Background Study: **Operators** are the foundation of any programming language. We can define operators as symbols that help us to perform specific mathematical and logical computations on operands. In other words, we can say that an operator operates the operands. For example, '+' is an operator used for addition, as shown below:
`c = a + b;`

Here, '+' is the operator known as the addition operator and 'a' and 'b' are operands. The addition operator tells the compiler to add both of the operands 'a' and 'b'.

An operator is a symbol that operates on a value or a variable. For example: + is an operator to perform addition.

C has a wide range of operators to perform various operations.

C Arithmetic Operators

An arithmetic operator performs mathematical operations such as addition, subtraction, multiplication, division etc on numerical values (constants and variables).

Operator	Meaning of Operator
+	Addition or unary plus
-	subtraction or unary minus
*	Multiplication
/	Division
%	remainder after division (modulo division)

C Increment and Decrement Operators

C programming has two operators increment ++ and decrement -- to change the value of an operand (constant or variable) by 1.

Increment ++ increases the value by 1 whereas decrement -- decreases the value by 1. These two operators are unary operators, meaning they only operate on a single operand.

C Relational Operators

A relational operator checks the relationship between two operands. If the relation is true, it returns 1; if the relation is false, it returns value 0.

Relational operators are used in decision making and loops.

Operator	Meaning of Operator	Example
==	Equal to	5 == 3 is evaluated to 0
>	Greater than	5 > 3 is evaluated to 1
<	Less than	5 < 3 is evaluated to 0
!=	Not equal to	5 != 3 is evaluated to 1
>=	Greater than or equal to	5 >= 3 is evaluated to 1
<=	Less than or equal to	5 <= 3 is evaluated to 0

C Logical Operators

An expression containing logical operator returns either 0 or 1 depending upon whether expression results true or false. Logical operators are commonly used in decision making in C programming.

Operator	Meaning	Example
&&	Logical AND. True only if all operands are true	If c = 5 and d = 2 then, expression ((c==5) && (d>5)) equals to 0.
||	Logical OR. True only if either one operand is true	If c = 5 and d = 2 then, expression ((c==5) || (d>5)) equals to 1.
!	Logical NOT. True only if the operand is 0	If c = 5 then, expression !(c==5) equals to 0.

List of Programs

Q1. Write a program to show arithmetic operators (+,-,* ,/and %)

Code:
#include <stdio.h>

```
int main() {
    int a, b;

    printf("Please enter the first number: ");
    scanf("%d", &a);

    printf("Please enter the second number: ");
    scanf("%d", &b);

    printf("The sum is: %d\n", a + b);
    printf("The difference is: %d\n", a - b);
    printf("The product is: %d\n", a * b);
    printf("The quotient is: %d\n", a / b);
    printf("The remainder is: %d\n", a % b);

    return 0;
}
```

Output and screenshot:

```
Output                                                      Clear

/tmp/DZPWooZCsL.o
Please enter the first number: 20
Please enter the second number: 54
The sum is: 74
The difference is: -34
The product is: 1080
The quotient is: 0
The remainder is: 20


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

Q2. Write a program that performs increment and decrement operators (Pre increment, Post-increment, Pre-Decrement, Post-decrement)

Code:

```c
#include <stdio.h>

int main() {
    int number;

    printf("Please enter a number: ");
    scanf("%d", &number);

    printf("Initial value: %d\n", number);

    printf("Pre-increment: %d\n", ++number);
    printf("Post-increment: %d\n", number++);
    printf("Value after post-increment: %d\n", number);

    printf("Pre-decrement: %d\n", --number);
    printf("Post-decrement: %d\n", number--);
    printf("Value after post-decrement: %d\n", number);

    return 0;
}
```

Output and screenshot:

```
Output                                          Clear

/tmp/zRKUjXKHMp.o
Please enter a number: 15
Initial value: 15
Pre-increment: 16
Post-increment: 16
Value after post-increment: 17
Pre-decrement: 16
Post-decrement: 16
Value after post-decrement: 15


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

Q3. Write a program that performs relational operators

Code:
```c
#include <stdio.h>

int main() {
    int a, b;

    printf("Please enter the first number: ");
    scanf("%d", &a);

    printf("Please enter the second number: ");
    scanf("%d", &b);

    printf("Is the first number equal to the second? : %s\n", a == b ? "True" : "False");
    printf("Is the first number not equal to the second? : %s\n", a != b ? "True" : "False");
    printf("Is the first number greater than the second? : %s\n", a > b ? "True" : "False");
    printf("Is the first number less than the second? : %s\n", a < b ? "True" : "False");
    printf("Is the first number greater than or equal to the second? : %s\n", a >= b ? "True" : "False");
    printf("Is the first number less than or equal to the second? : %s\n", a <= b ? "True" : "False");

    return 0;
}
```

Output and screenshot:

```
Output                                                    Clear

/tmp/n041HyhxdR.o
Please enter the first number: 17
Please enter the second number: 56
Is the first number equal to the second? : False
Is the first number not equal to the second? : True
Is the first number greater than the second? : False
Is the first number less than the second? : True
Is the first number greater than or equal to the second? : False
Is the first number less than or equal to the second? : True


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

Q5. Write a program that performs logical operators

Code:
```c
#include <stdio.h>

int main() {
    int num1, num2;

    printf("Enter two integers: ");
    scanf("%d %d", &num1, &num2);

    printf("%d > %d: ", num1, num2);
    if (num1 > num2) {
        printf("True\n");
    } else {
        printf("False\n");
    }

    printf("%d < %d: ", num1, num2);
    if (num1 < num2) {
        printf("True\n");
    } else {
        printf("False\n");
    }
```

```c
    printf("%d >= %d: ", num1, num2);
    if (num1 >= num2) {
        printf("True\n");
    } else {
        printf("False\n");
    }

    printf("%d <= %d: ", num1, num2);
    if (num1 <= num2) {
        printf("True\n");
    } else {
        printf("False\n");
    }

    printf("%d == %d: ", num1, num2);
    if (num1 == num2) {
        printf("True\n");
    } else {
        printf("False\n");
    }

    printf("%d != %d: ", num1, num2);
    if (num1 != num2) {
        printf("True\n");
    } else {
        printf("False\n");
    }

    return 0;
}
```

Output and screenshot:

Output   Clear

```
/tmp/jI7l8MXtJi.o
Enter two integers: 121
45
121 > 45: True
121 < 45: False
121 >= 45: True
121 <= 45: False
121 == 45: False
121 != 45: True


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

PRACTICAL NO. 7

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First Sem.
Link to Code:
Date: 28 oct. 2024
Faculty Signature:

	Objective To familiarize the students with operators in C
	Program Outcome Through this practical, students will learn the concept of operators in C programming

	Background Study:
	
	  An operator is a symbol that operates on a value or a variable. For example: + is an operator to perform addition.
	
	  C has a wide range of operators to perform various operations.
	
	  C Arithmetic Operators
	
	  An arithmetic operator performs mathematical operations such as addition, subtraction, multiplication, division etc on numerical values (constants and variables).

Operator	Meaning of Operator
+	Addition or unary plus
-	subtraction or unary minus
*	Multiplication
/	Division

%	remainder after division (modulo division)

C Increment and Decrement Operators

C programming has two operators increment ++ and decrement -- to change the value of an operand (constant or variable) by 1.

Increment ++ increases the value by 1 whereas decrement -- decreases the value by 1. These two operators are unary operators, meaning they only operate on a single operand.

C Relational Operators

A relational operator checks the relationship between two operands. If the relation is true, it returns 1; if the relation is false, it returns value 0.

Relational operators are used in decision making and loops.

Operator	Meaning of Operator	Example
==	Equal to	5 == 3 is evaluated to 0
>	Greater than	5 > 3 is evaluated to 1
<	Less than	5 < 3 is evaluated to 0
!=	Not equal to	5 != 3 is evaluated to 1
>=	Greater than or equal to	5 >= 3 is evaluated to 1
<=	Less than or equal to	5 <= 3 is evaluated to 0

C Logical Operators

An expression containing logical operator returns either 0 or 1 depending upon whether expression results true or false. Logical operators are commonly used in decision making in C programming.

Operator	Meaning	Example

&&	Logical AND. True only if all operands are true	If c = 5 and d = 2 then, expression ((c==5) && (d>5)) equals to 0.
||	Logical OR. True only if either one operand is true	If c = 5 and d = 2 then, expression ((c==5) || (d>5)) equals to 1.
!	Logical NOT. True only if the operand is 0	If c = 5 then, expression !(c==5) equals to 0.

Q1. What is the output of the following code snippet?

```c
#include<stdio.h>

main()
{
    int x = 5;

    if(x==5)
    {
        if(x==5) break;
        printf("Hello");
    }
    printf("Hi");
}
```

Q2. What is the output of the following code snippet?

```c
#include<stdio.h>

main()
{
    const int a = 5;

    a++;
    printf("%d", a);
}
```

Q3. What is the output of the following code snippet?

```c
#include<stdio.h>

main()
{
    char c = 'A'+255;

    printf("%c", c);
}
```

Q4. What is the output of the below code snippet?

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

```c
main()
{
    int a = 1;
    float b = 1.3;
    double c;

    c = a + b;
    printf("%.2lf", c);
}
```
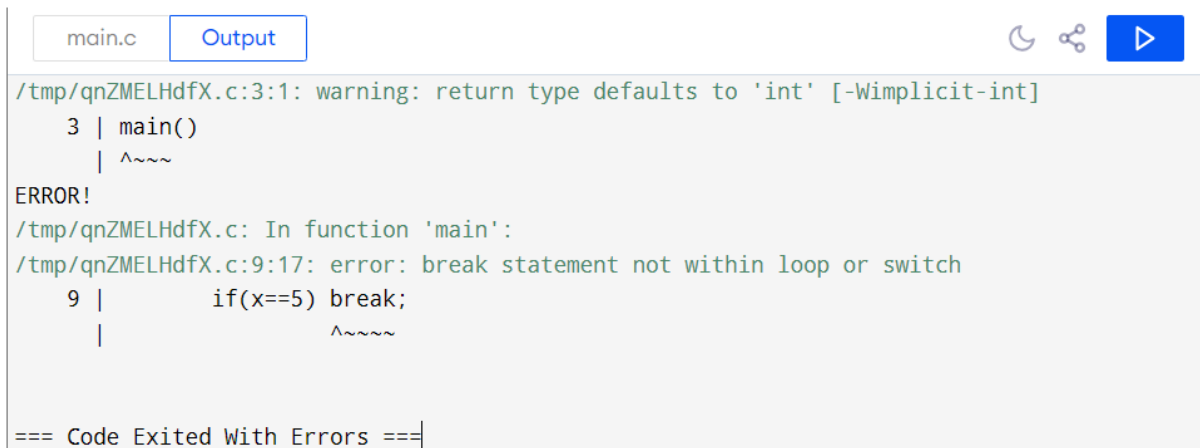
Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

Solution 1. The code will give a compilation error because the break statement is used incorrectly. In C, break can only be used inside loops (for, while, do-while) or switch statements. Here, break is placed inside an if statement without a loop, which is not allowed in C.

Screenshot:


```
/tmp/qnZMELHdfX.c:3:1: warning: return type defaults to 'int' [-Wimplicit-int]
    3 | main()
      | ^~~~
ERROR!
/tmp/qnZMELHdfX.c: In function 'main':
/tmp/qnZMELHdfX.c:9:17: error: break statement not within loop or switch
    9 |         if(x==5) break;
      |                  ^~~~~


=== Code Exited With Errors ===
```

Solution 2. Since a is declared as a const int, trying to increment it with a++ will cause an error. Constants in C cannot be changed once set, so modifying a is not allowed, leading to a compilation error.

Screenshot:

```
main.c    Output                                           ☾  ∝°  ▷

/tmp/6WyW3xgcob.c:3:1: warning: return type defaults to 'int' [-Wimplicit-int]
    3 | main()
      | ^~~~
ERROR!
/tmp/6WyW3xgcob.c: In function 'main':
/tmp/6WyW3xgcob.c:7:5: error: increment of read-only variable 'a'
    7 |     a++;
      |      ^~


=== Code Exited With Errors ===|
```

Solution 3. The code will compile, but the output is unpredictable. This is because `'A' + 255` results in 320, which is too large for a `char` and causes overflow. So, `c` will store an unexpected character due to this overflow.

Screenshot:

```
main.c    Output                                           ☾  ∝°  ▷

/tmp/p2yl1ELMxu.c:3:1: warning: return type defaults to 'int' [-Wimplicit-int]
    3 | main()
      | ^~~~
/tmp/p2yl1ELMxu.c: In function 'main':
/tmp/p2yl1ELMxu.c:5:14: warning: overflow in conversion from 'int' to 'char' changes value
    from '320' to '64' [-Woverflow]
    5 |     char c = 'A'+255;
      |              ^~~
/tmp/p2yl1ELMxu.o
@

=== Code Execution Successful ===|
```
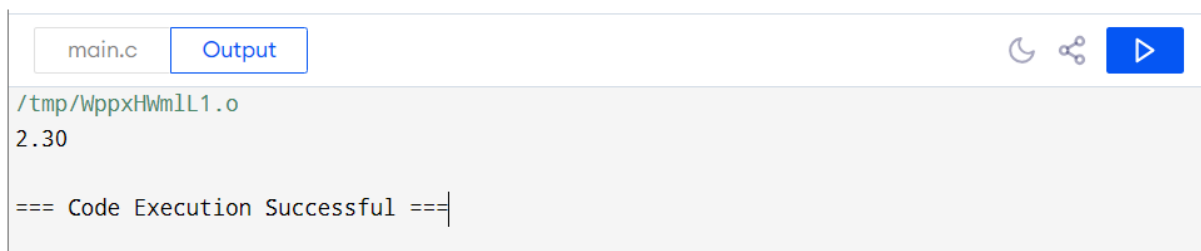
Solution 4.

- When a (an int) is added to b (a float), a is implicitly converted to a float, and the result is of float type.

- The result is then assigned to c, which is a double, so it is further converted to double for higher precision.

- printf("%.2lf", c); prints c with two decimal places.

Expected Output: 2.30

Screenshot:

Experiment- 8

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First Sem.
Link to Code:
Date: 28 oct. 2024
Faculty Signature:

Objective To familiarize the students with if-else loop.
Program Outcome The students will learn the concept of looping in C. They will be able to understand the different types of statements encountered in C.
Problem Statement 1. While purchasing certain items, a discount of 10% is offered if the quantity purchased is more than 1000. If quantity and price per item are input through the keyboard, write a program to calculate the total expenses.   2. The marks obtained by a student in 5 different subjects are input through the keyboard. The student gets a division as per the following rules: Percentage above or equal to 60 - First division Percentage between 50 and 59 - Second division Percentage between 40 and 49 - Third division Percentage less than 40 - Fail Write a program to calculate the division obtained by the student.   3. Write a program to check whether a triangle is valid or not, when the three angles of the triangle are entered through the keyboard. A triangle is valid if the sum of all the three angles is equal to 180 degrees. 4. Write a program in C to read the age and display whether the candidate is eligible to vote or not. Definition of Done:

- The program should ask the user to enter a value depicting his age.
- The program should use if-else statement

Background

The syntax of the if statement in C programming is:

if (test expression)

{

  // statements to be executed if the test expression is true

}

How if statement works?

The if statement evaluates the test expression inside the parenthesis ().

1. If the test expression is evaluated to true, statements inside the body of if are executed.
2. If the test expression is evaluated to false, statements inside the body of if are not executed**.**

Flipped Questions

1. What is the usage of nested if statements?

2. What will be the output of the C program?
#include<stdio.h>
int main()

```
    {
            int i = 5, j = 6, k = 7;
            if(i > j == k)
                        printf("%d %d %d", i++, ++j, --k);
            else
                        printf("%d %d %d", i, j, k);
            return 0;
    }
```

3. What will be the output of the C program?

```
#include<stdio.h>
int main()
{
            int i = 2;
            if(i == (1, 2))
                    printf("Hai");
            else
                    printf("No Hai");
            return 0;
}
```
a) Compilation error
b) Runtime error
c) Hai
d) No Hai

4. What will be the output of the C program?

```
#include<stdio.h>
int main()
{
            if(sizeof(0))
                    printf("Hai");
            else
                    printf("Bye");
            return 0;
}
```
a) Hai
b) Bye
c) Compilation Error

d) None

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

Solution to Problem Statements.

Solution 1.

Code:

```c
#include <stdio.h>

int main() {
    int quantity;
    float price_per_item, total_expenses;

    printf("Please enter the quantity purchased: ");
    scanf("%d", &quantity);

    printf("Please enter the price per item: ");
    scanf("%f", &price_per_item);
```

```
    if (quantity > 1000) {

        total_expenses = quantity * price_per_item * 0.9;

    } else {

        total_expenses = quantity * price_per_item;

    }


    printf("The total expenses are: %.2f\n", total_expenses);


    return 0;

}
```

Screenshot:

```
Output                                                    Clear

/tmp/oocAPsscFs.o
Please enter the quantity purchased: 1500
Please enter the price per item: 100
The total expenses are: 135000.00


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

Solution 2.

Code:

```
#include <stdio.h>


int main() {

    int subject1, subject2, subject3, subject4, subject5;
```

```c
int total;

float percentage;


printf("Please enter the marks obtained in subject 1: ");

scanf("%d", &subject1);


printf("Please enter the marks obtained in subject 2: ");

scanf("%d", &subject2);


printf("Please enter the marks obtained in subject 3: ");

scanf("%d", &subject3);


printf("Please enter the marks obtained in subject 4: ");

scanf("%d", &subject4);


printf("Please enter the marks obtained in subject 5: ");

scanf("%d", &subject5);


total = subject1 + subject2 + subject3 + subject4 + subject5;

percentage = (float)total / 5;


printf("Total Marks: %d\n", total);
```

```c
    printf("Percentage: %.2f\n", percentage);


    if (percentage >= 60) {

        printf("Division: First Division\n");

    } else if (percentage >= 50 && percentage <= 59) {

        printf("Division: Second Division\n");

    } else if (percentage >= 40 && percentage <= 49) {

        printf("Division: Third Division\n");

    } else {

        printf("Division: Fail\n");

    }


    return 0;

}
```

Screenshot:

```
Output                                                          Clear

/tmp/s4YzCNBwrt.o
Please enter the marks obtained in subject 1: 80
Please enter the marks obtained in subject 2: 90
Please enter the marks obtained in subject 3: 98
Please enter the marks obtained in subject 4: 71
Please enter the marks obtained in subject 5: 91
Total Marks: 430
Percentage: 86.00
Division: First Division


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

Solution 3.

Code:

```c
#include <stdio.h>

int main() {
    int angle1, angle2, angle3, sum;

    printf("Please enter the first angle: ");
    scanf("%d", &angle1);

    printf("Please enter the second angle: ");
    scanf("%d", &angle2);

    printf("Please enter the third angle: ");
    scanf("%d", &angle3);

    sum = angle1 + angle2 + angle3;

    if (sum == 180) {
        printf("The triangle is valid.\n");
    } else {
        printf("The triangle is not valid.\n");
```

```
    }


    return 0;

}
```

Screenshot:

```
Output                                                    Clear

/tmp/VHAQNuVCZF.o
Please enter the first angle: 59
Please enter the second angle: 60
Please enter the third angle: 61
The triangle is valid.


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

Solution 4.

Code:

```c
#include <stdio.h>

int main() {
    int age;


    printf("Please enter your age: ");
    scanf("%d", &age);


    if (age >= 18) {
        printf("You are eligible to vote.\n");
```

```
    } else {

        printf("You are not eligible to vote.\n");

    }


    return 0;

}
```

Screenshot:

```
Output                                                    Clear

/tmp/smNzSLr3bG.o
Please enter your age: 21
You are eligible to vote.


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

Flipped Questions:

Solution 1. Nested if statements are used when multiple conditions need to be checked sequentially. If the outer if condition is true, the inner if is evaluated. This allows for more complex decision-making by checking multiple conditions within one another. Nested if statements help in handling complex conditions in a structured way.

Solution 2:

Output: 5 6 7

Screenshot:

```
main.c    Output

/tmp/Jf66x4NIKs.o
5 6 7

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

Solution 3:

Output: c) Hai

Explanation:

- In the condition i == (1, 2), the comma operator evaluates the expression to the last value, which is 2. Thus, the expression effectively becomes i == 2.

- Since i is initialized to 2, the condition evaluates to true, and the program prints "Hai".

Screenshot:

```
Output                                    Clear

/tmp/M9E9KHvoRk.o
Hai

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

Solution 4:

Output: a) Hai

Explanation:

In C, the sizeof operator evaluates the size of the operand in bytes. When you use sizeof(0), it returns the size of the integer type, which is typically 4 bytes on most platforms (though this can vary depending on the architecture).

- The expression sizeof(0) evaluates to a non-zero value (the size of an integer), which is treated as true in the context of the if statement.

- Since the condition is true, the program will execute the printf("Hai"); statement.

Screenshot:

```
Output                                                    Clear

/tmp/i7lr9Lp47y.o
Hai

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

Experiment- 9

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section: First Sem.
Link to Code:
Date: 29 oct. 2024
Faculty Signature:

Objective To familiarize the students with while and do while loop along with jumping statement.
Program Outcome The students will be able to understand iterative control statements and how to apply while and do-while loop in programs.
Problem Statement 1.   Find factorial of a number using while loop   2.   Write a program in C to print the table of a given number using do while loop.   3. Write a program to check whether a number is prime or not.

Background
while loop

The syntax of the while loop is:

while (testExpression)

{

　// statements inside the body of the loop

}

How while loop works?

- The while loop evaluates the test expression inside the parenthesis ().
- If the test expression is true, statements inside the body of while loop are executed. Then, the test expression is evaluated again.
- The process goes on until the test expression is evaluated to false.
- If the test expression is false, the loop terminates (ends).

do...while loop

The do..while loop is similar to the while loop with one important difference. The body of do...while loop is executed at least once. Only then, the test expression is evaluated.

The syntax of the do...while loop is:

do

{

　// statements inside the body of the loop

}

while (testExpression);

How do...while loop works?

- The body of do-while loop is executed once. Only then, the test expression is evaluated.
- If the test expression is true, the body of the loop is executed again and the test expression is evaluated.
- This process goes on until the test expression becomes false.

- If the test expression is false, the loop ends.

Question Bank

Suggested Question Bank

1. Differentiate between while and do while loop.

2. What will be the output of the C program?

a) void main( )

```
   {
    int j =1;
   while ( j <= 10 )
   {
    printf ( "\n%d", j ) ;
    j = j + 1 ; } }
```

b) void main( )

```
    {
     float x = 1.1 ;
     while ( x == 1.1 )
     {
      printf ( "\n%f", x ) ;
      x = x – 0.1 ; } }
```

Flipped Questions

1. What will be the output of the following?

a) #include <stdio.h>

```
int main()
{  int i = 0;
    while (i = 0)
        printf("True\n");
    printf("False\n");
}
```

b) #include<stdio.h>

```
int main()
{ int i = 0;
  while(i < 3, i = 0, i < 5)
  { printf("Loop ");
    i++; }
    return 0; }
```

c) #include<stdio.h>

```
int main()
{ int i = 0;
  while(i++)
  {
   printf("Loop ");
   if(i == 3) break; }
  return 0; }
```

Student Work Area

Algorithm/Flowchart/Code/Sample Outputs

Solution To Problem Statements:

1. Code:

```c
#include <stdio.h>

int main() {
    int num;
    long double factorial = 1.0;



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


    int i = 1;
    while (i <= num) {
        factorial *= i;
        i++;
    }


    printf("Factorial of %d is %.0Lf\n", num, factorial);
```
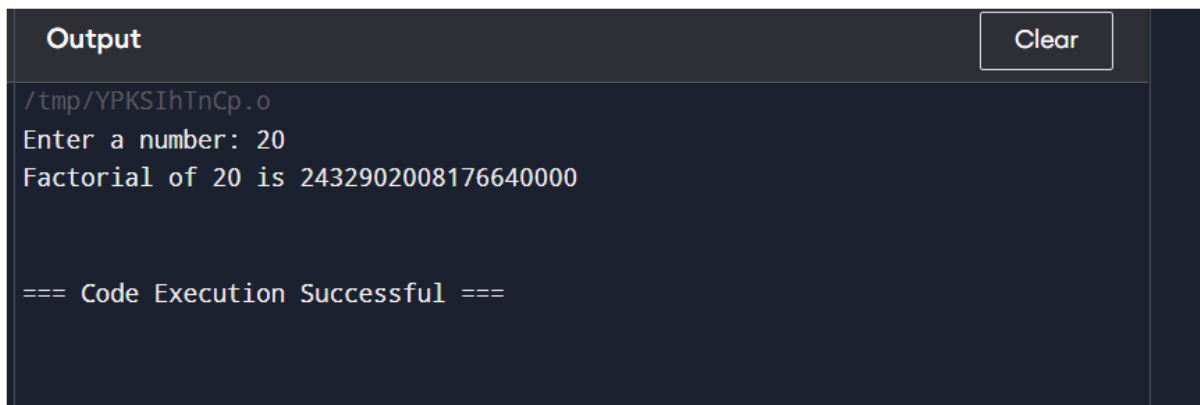
```
    return 0;

}
```

Screenshot:

2. Code:

```c
#include <stdio.h>

int main() {
    int number, i = 1;

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

    do {
        printf("%d x %d = %d\n", number, i, number * i);
```
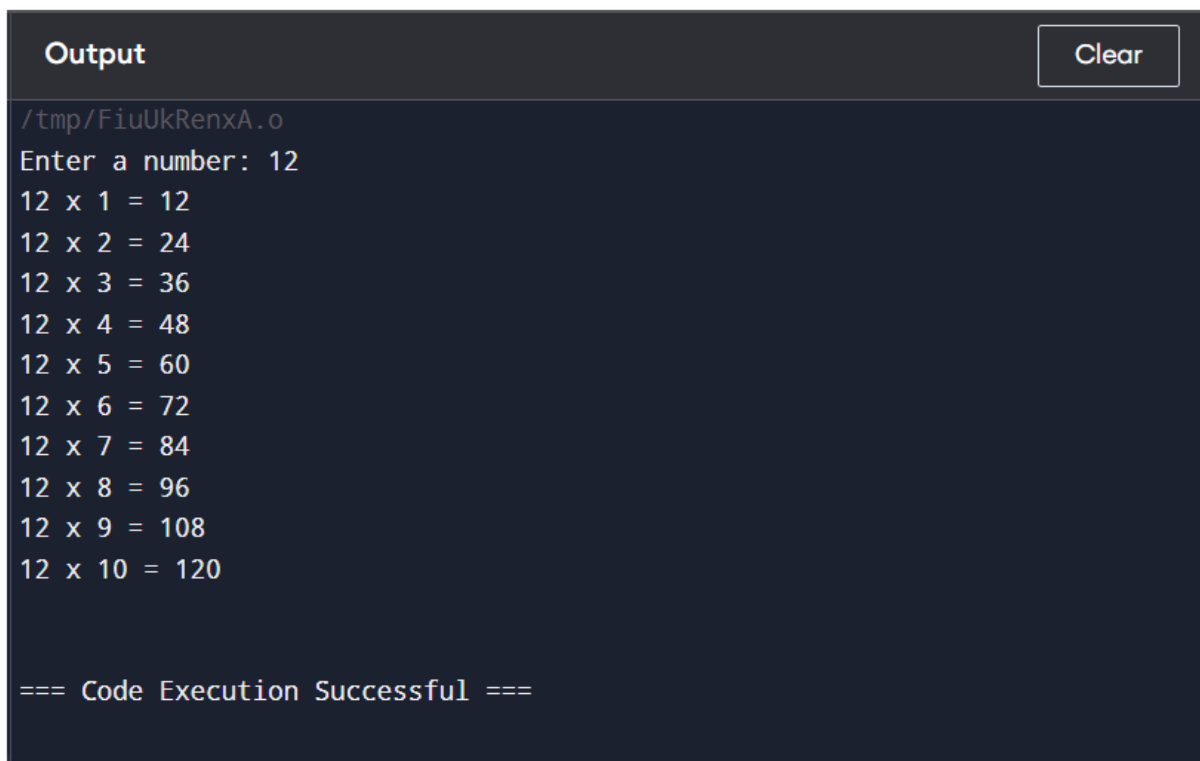
```
        i++;

    } while (i <= 10);



    return 0;

}
```

Screenshot:

3. Code:

```
#include <stdio.h>

int main()

{
```

```c
int n, i, flag = 0;
printf("Enter a positive integer: ");
scanf("%d", &n);
for (i = 2; i <= n / 2; ++i)
{
    if (n % i == 0)
    {
        flag = 1;
        break;
    }
}
if (n == 1)
{
    printf("1 is neither prime nor composite.");
}
else
{
    if (flag == 0)
        printf("%d is a prime number.", n);
    else
        printf("%d is not a prime number.", n);
}
```
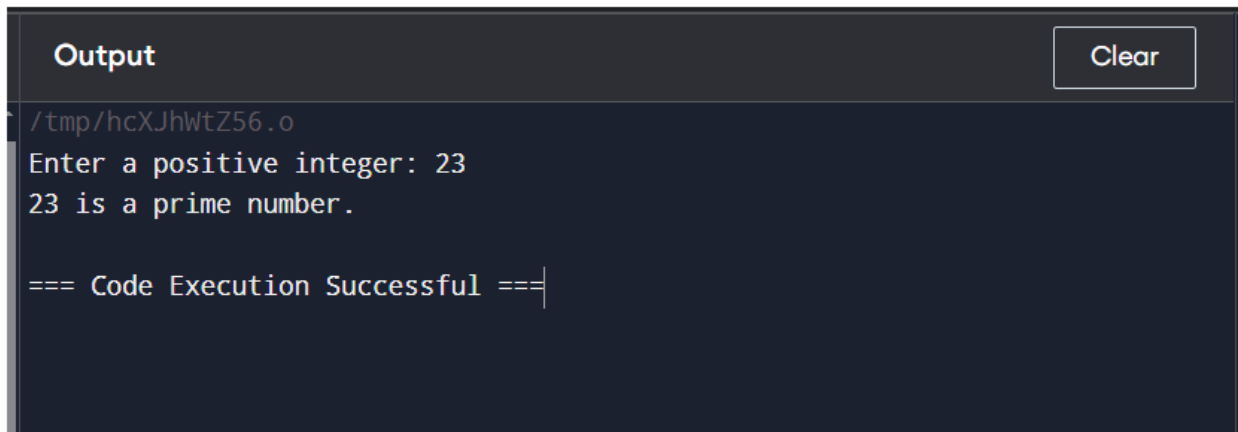
```
    return 0;

}
```

Screenshot:

Solution For Question Bank:

1. The main difference between a while loop and a do-while loop is when they check the condition:

- While loop: It checks the condition before running the loop's code. If the condition is false from the start, the code inside the loop won't run at all.

  Example:

  while (condition) {

      // code

  }

The loop runs only if the condition is true.

- Do-while loop: It runs the code first, and then checks the condition. This means the code inside the loop will always run at least once, even if the condition is false.

Example:

do {

   // code

} while (condition);

The loop runs once, and then checks the condition to decide whether to continue.
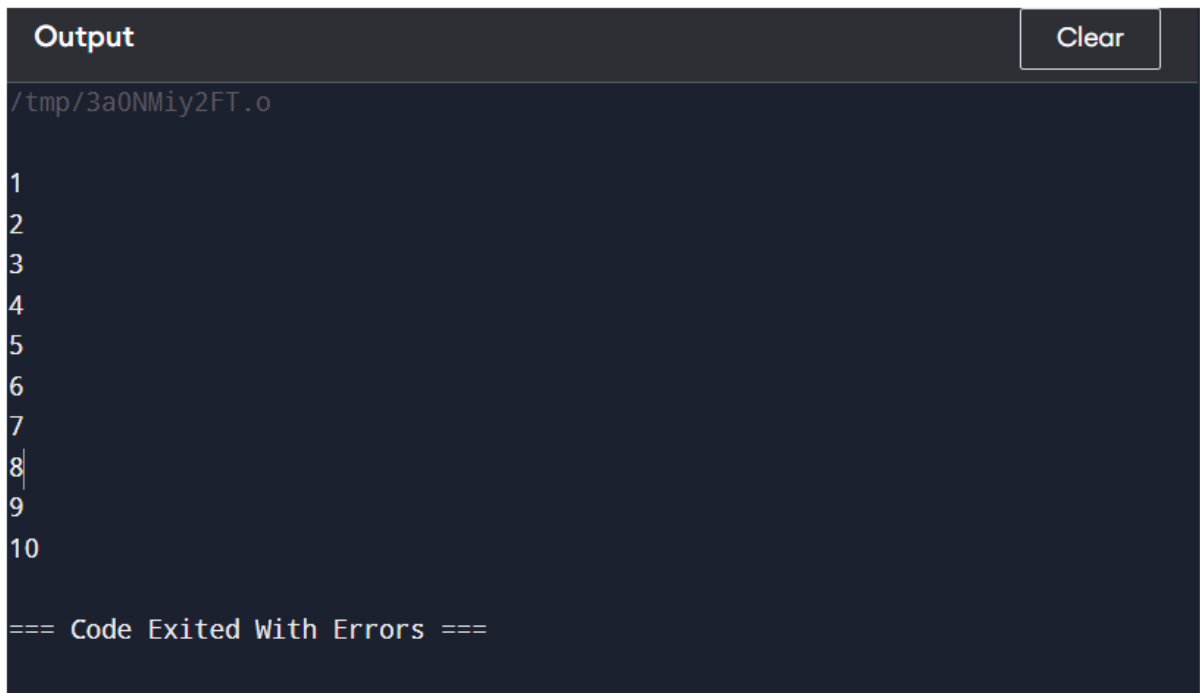
2.

a.) Output:

1

2

3

4

5

6

7

8

9

10

Screenshot:

b). The loop will not run, and nothing will be printed. The program simply terminates without any output.

Solution to flipped Questions:

1. a.) Output : False

   Screenshot:

b.) Output: The output will continuously print "Loop " indefinitely because i will never be able to increment properly.

Screenshot:

c.) Output: Because i starts at 0, the loop condition while(i++) is false on the first check, so nothing is printed. The program terminates without output.

Screenshot:

```
Output                                          Clear

/tmp/0z8IlWAN47.o


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

Experiment- 10

Student Name and Roll Number: Bhumi Nabiyal (24bca046)
Semester /Section:
Link to Code:
Date: 29 oct. 2024
Faculty Signature:

Objective To familiarize the students with for loop
Program Outcome The students will be able to understand iterative control statements and how to apply for loop in programs.
Problem Statement **Q1. Write a c program to print numbers from 1 to 10** **Code :**   **Q2. Program to calculate the sum of first n natural numbers** **// Positive integers 1,2,3...n are known as natural numbers** **Code:**   3.   Write a program to print the following pattern.

```
1

 1  2

 1  2  3

 1  2  3  4
```

Background

for Loop

The syntax of the for loop is:

for (initializationStatement; testExpression; updateStatement)

{   // statements inside the body of loop }

How for loop works?

- The initialization statement is executed only once.
- Then, the test expression is evaluated. If the test expression is evaluated to false, the for loop is terminated.
- However, if the test expression is evaluated to true, statements inside the body of for loop are executed, and the update expression is updated.
- Again the test expression is evaluated.
- This process goes on until the test expression is false. When the test expression is false, the loop terminates.

Question Bank
1. To find the sum of individual digits of a positive integer and test given number is palindrome.

2. That will read in a positive integer value and determine whether its prime or Fibonacci.

Flipped Questions

1. How many times "FOCP" is get printed?

```c
#include<stdio.h>
int main()
{
   int x;
   for(x=-1; x<=10; x++)
   {
      if(x < 5)
         continue;
      else
         break;
      printf("FOCP");
   }
   return 0;
}
```
2 What will be the output of the program?

```c
#include<stdio.h>
int main()
{
   int i=0;
   for(; i<=5; i++);
      printf("%d", i);
   return 0;
}
```
3. Point out the error, if any in the for loop.

```c
#include<stdio.h>
int main()
{
   int i=1;
```

```
   for(;;)
   {
     printf("%d\n", i++);
     if(i>10)
        break;
   }
   return 0;
}
```

Student Work Area
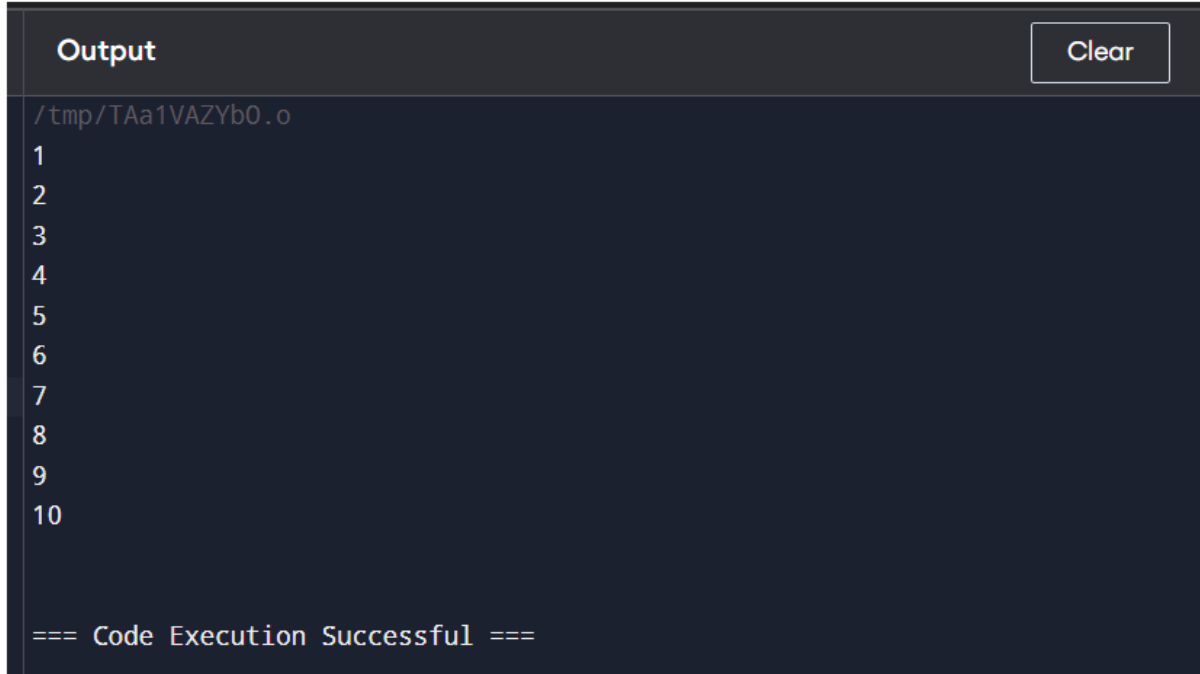
Algorithm/Flowchart/Code/Sample Outputs

Solutions to Problem Statements:

1. Code:

```
#include <stdio.h>

int main() {
   for (int i = 1; i <= 10; i++) {
      printf("%d\n", i);
   }
   return 0;
}
```

Screenshot:

```
Output                                              Clear

/tmp/TAa1VAZYbO.o
1
2
3
4
5
6
7
8
9
10


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

2. Code:

#include <stdio.h>

int main() {

    int n, sum = 0;

    printf("Enter a number: ");

    scanf("%d", &n);

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

        sum += i;

    }
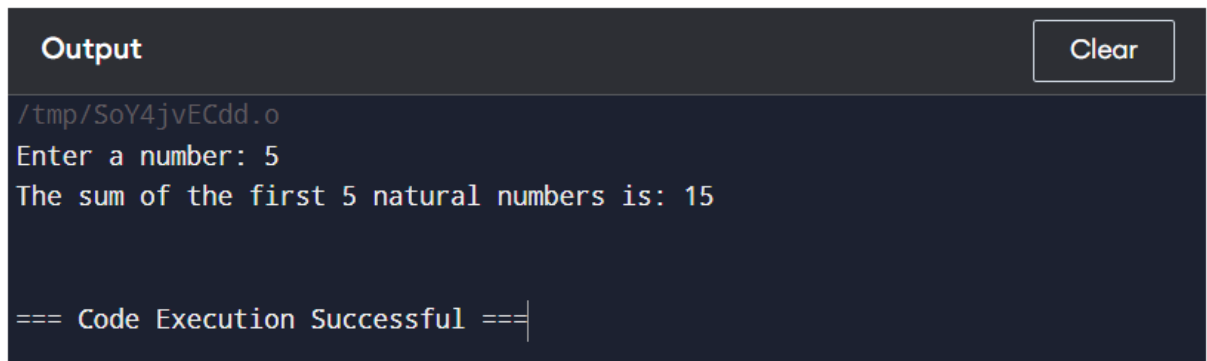
```c
    printf("The sum of the first %d natural numbers is: %d\n", n, sum);


    return 0;

}
```

Screenshot:

```
Output                                            Clear

/tmp/SoY4jvECdd.o
Enter a number: 5
The sum of the first 5 natural numbers is: 15


=== Code Execution Successful ===
```
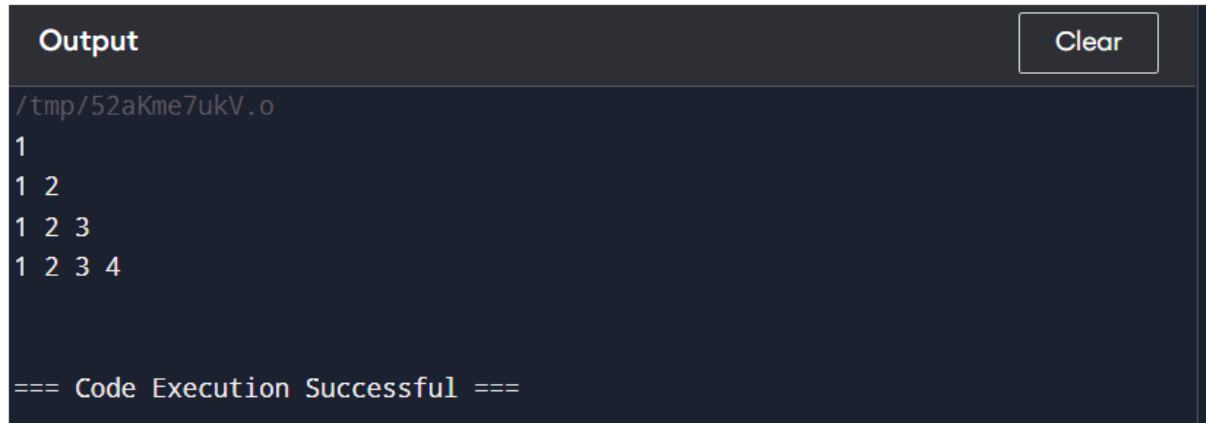
3. Code:

```c
#include <stdio.h>


int main() {
    for (int i = 1; i <= 4; i++) {
        for (int j = 1; j <= i; j++) {
            printf("%d ", j);
        }
        printf("\n");
    }
```

```
    return 0;

}
```

Screenshot:

Solutions to Question Bank

1. Code:

```
#include<stdio.h>


int main() {
    int num, originalNum, reversedNum = 0, remainder, sumOfDigits = 0;


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



    originalNum = num;
```

```c
    while(num != 0) {

        remainder = num % 10;

        sumOfDigits += remainder;

        reversedNum = reversedNum * 10 + remainder;

        num /= 10;

    }



    printf("Sum of digits: %d\n", sumOfDigits);



    if(originalNum == reversedNum) {

        printf("The number is a palindrome.\n");

    } else {

        printf("The number is not a palindrome.\n");

    }


    return 0;

}
```

Screenshot:

```
Output                                    Clear

/tmp/akTKFWQq1d.o
Enter a positive integer: 123454321
Sum of digits: 25
The number is a palindrome.


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

2. Code: ____Unable to form the logic____

Solutions to flipped Questions.

1. The code does not print "FOCP" at all. Zero times.

2. Output: 6

Screenhot:

```
Output                                    Clear

/tmp/uneZTYEVAe.o
6

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

3. There is no error in the loop. It will print numbers from 1 to 10, each on a new line, and then exit the loop.

Output Screenshot:

```
Output                                          Clear

/tmp/ZuHmYQJVHz.o
1
2
3
4
5
6
7
8
9
10


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