Course Name: Information and Communication Technologies Lab		
LAB # 5: (a) Introduction to Conditional statements in C Programming **(b)** Introduction to loops in C Programming		
Department	**Registration Number/Name**	**Semester/Section**
BS CEN	F24604018/Muhammad Hamzah Iqbal	1
Date	**Instructor's Name**	**Instructor's Signature**
14/10/2024	Iqra Ashraf	

Objectives:

- To understand the basics of C programming.
- To get familiar with Conditional statements.
- To get familiar with loops in C Programming.

Lab Tasks:

Lab 5(a)

Question 01: Design a calculator in C that can perform the following binary operations on two numbers obtained from user. The choice of operation should also be available to the user.

1. Addition

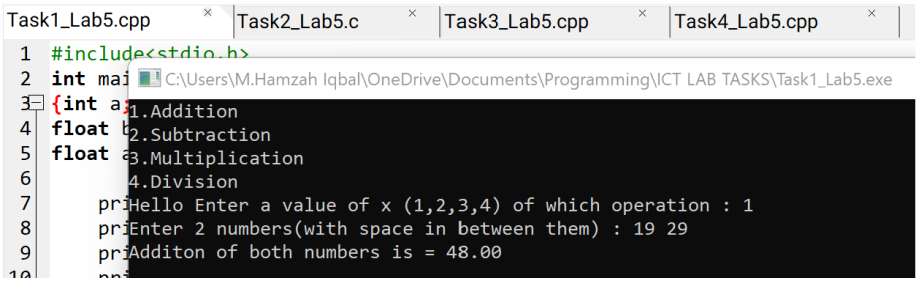
2. Subtraction

3. Multiplication

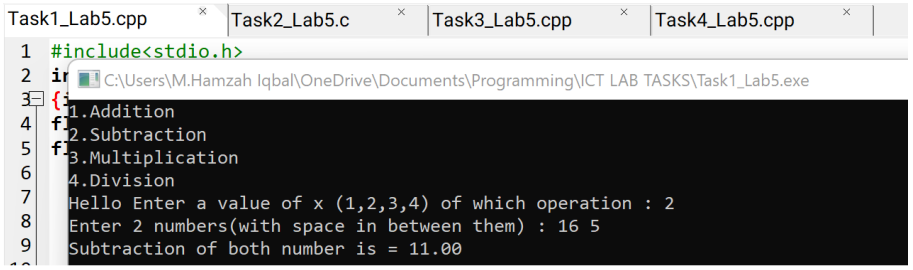
4. Division

Outputs:

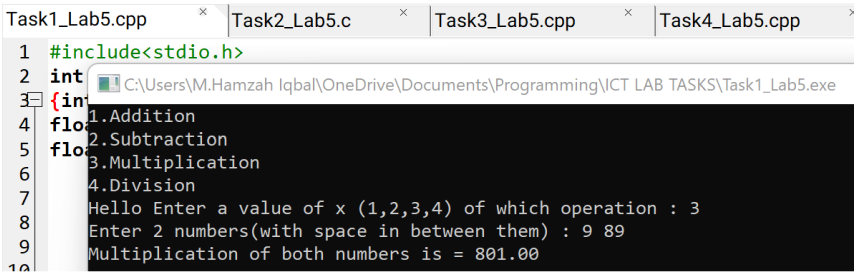
Addition



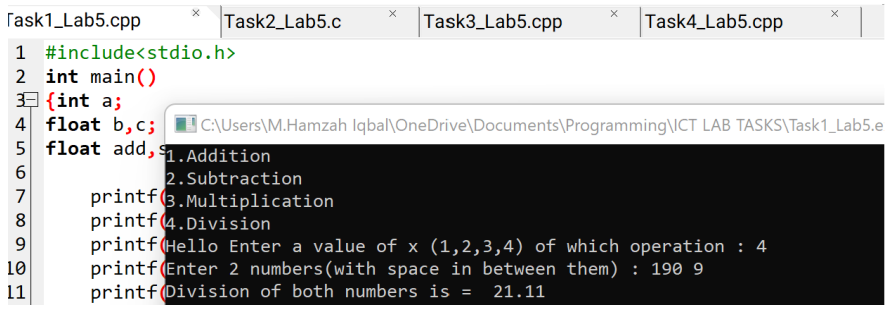
Subtraction



Multiplication



Division



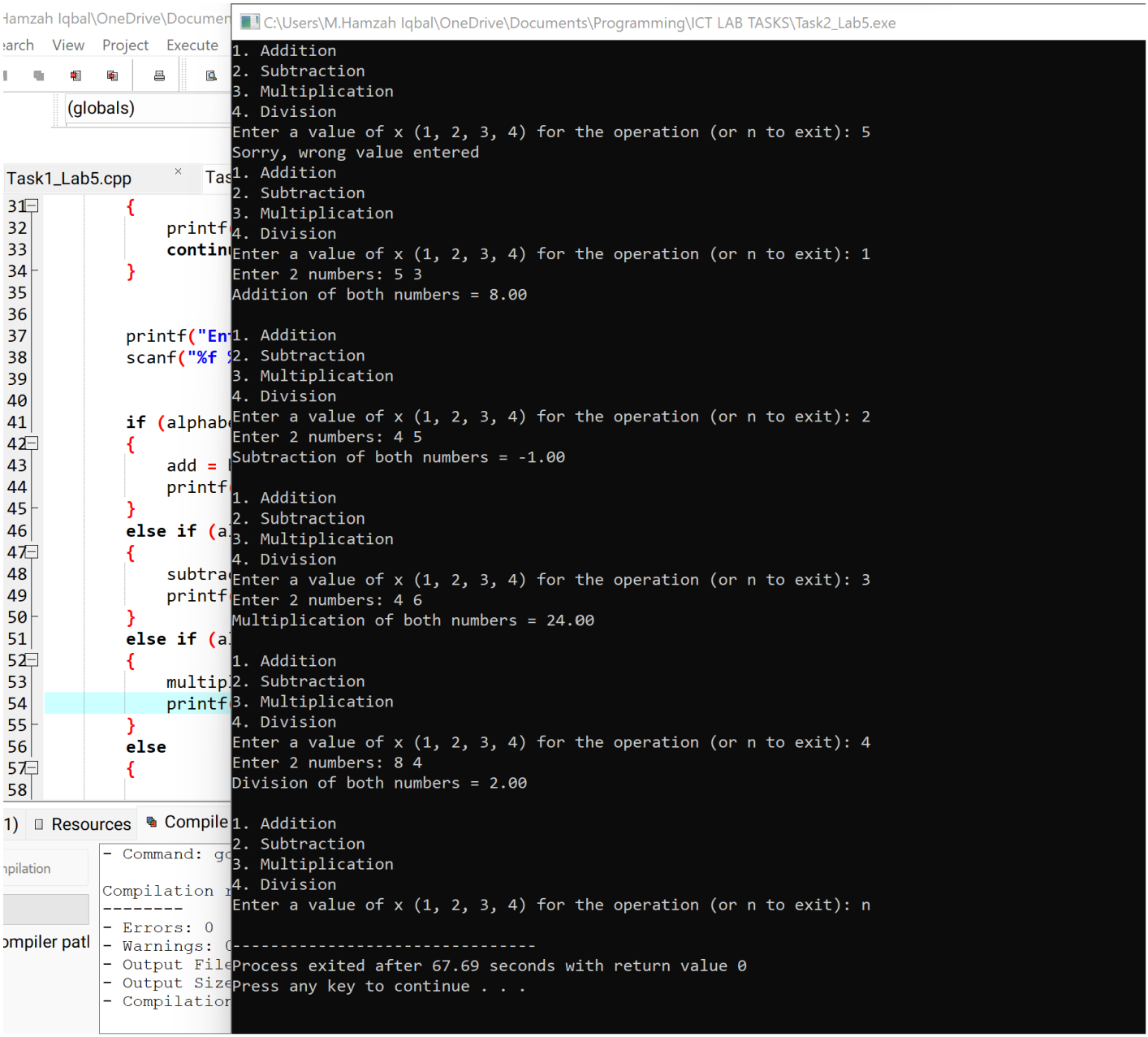
Code :

```
#include<stdio.h>
int main()
{int a;
float b,c;
float add,subtract,multiply,divide;
printf("1.Addition \n");
printf("2.Subtraction \n");
printf("3.Multiplication \n");
printf("4.Division \n");
printf("Hello Enter a value of x (1,2,3,4) of which operation : ");
scanf("%d",&a);

printf("Enter 2 numbers(with space in between them) : ");
scanf("%d %d",&b,&c);
if(a==1) //Addition
{
add=b+c;
printf("Additon of both numbers is = %.2f \n",add);}

else if(a==2) //Subtraction
{subtract=b-c;
printf("Subtraction of both number is = %.2f",subtract);}

else if(a==3) //Multiplication
{multiply=b*c;
printf("Multiplication of both numbers is = %.2f",multiply);}

else   //Division
{
divide=b/c;
printf("Division of both numbers is =  %.2f",divide);
}
return 0;
}
```

Question 02:

Modify your design of the calculator such that it won't terminate after just one execution but instead asks from the user to enter a specific key to terminate. Otherwise, your calculator should keep on asking for the new numbers and operation.

Output:



Code:

```
#include<stdio.h>
#include<stdlib.h>
int main()
{  float a, b, c;
   int add, subtract, multiply;
   float divide;
   char alphabet;
   int i = 1;
 while (i > 0)
  { printf("1. Addition \n");
     printf("2. Subtraction \n");
     printf("3. Multiplication \n");
     printf("4. Division \n");
     printf("Enter a value of x (1, 2, 3, 4) for the operation (or n to exit): ");
      scanf(" %c", &alphabet);
 if (alphabet == 'n') // Exit condition if specific character entered
     {  exit(0); }
     // Handling wrong input by user
     if (alphabet < '1' || alphabet > '4')
     {
        printf("Sorry, wrong value entered \n");
        continue;  // Skip to the next iteration of the loop
     }
  printf("Enter 2 numbers: "); //Taking input from user
     scanf("%d %d", &b, &c);
     if (alphabet == '1')
     { add = b + c;
        printf("Addition of both numbers = %d \n \n", add);}
     else if (alphabet == '2')
     {subtract = b - c;
        printf("Subtraction of both numbers = %d \n \n", subtract); }
     else if (alphabet == '3')
     {   multiply = b * c;
        printf("Multiplication of both numbers = %d \n \n", multiply);    }
     else
        {divide = b / c;
        printf("Division of both numbers = %.2f \n \n", divide);   }
     i++;//increament in value of i, to satisfy loop condition
  }
  return 0;    }
```

Lab 5(b)

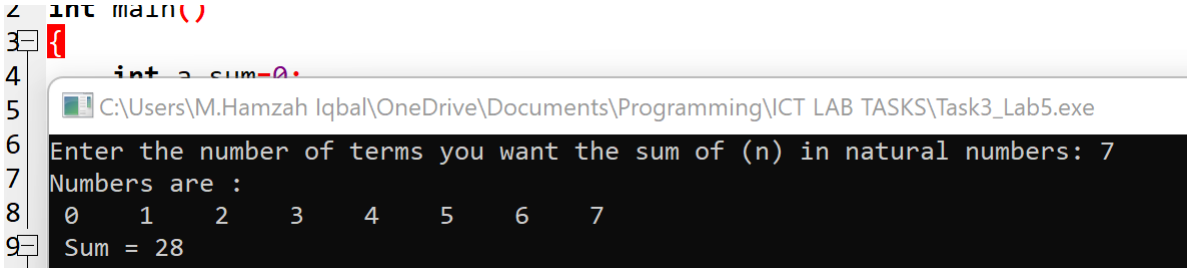
Question 01:

Write a program in C to display n terms of natural number and their sum.

Code:

```
#include<stdio.h>
int main()
{
	int a,sum=0;
	printf("Enter the number of terms you want the sum of (n) in natural numbers: ");
	scanf("%d",&a);
	printf("Numbers are : \n");
	for(int i=0;i<=a;i++)
	{
		printf(" %d  ",i);
		sum=sum+i;
	}

	printf(" \n Sum = %d",sum);
	return 0;
}
```

Output:



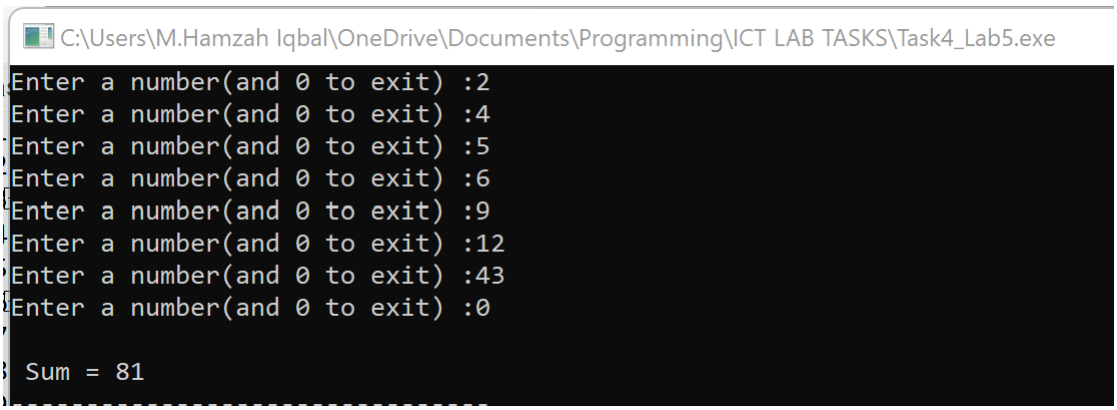
Question 02:

Write a Program to add numbers until the user enters zero. (Hint: Use do while)

Code:

```
#include<stdio.h>
int main()
{int num,sum=0;

do
{
printf("Enter a number(and 0 to exit) :");
scanf("%d",&num);
sum=sum+num;
}

while(num!=0);
printf("\n");
printf(" Sum = %d",sum);
return 0;
}
```

Output:



Conclusion:

To sum up, the four exercises aimed to improve basic C programming abilities, especially regarding basic input/output, loops, and control structures. In Task 1, a calculator with binary operations was shown, allowing users to do addition, subtraction, multiplication, and division. In Task 2, the calculator was improved to run continuously until the user chose to stop using it. Task 3 involved the usage of loops by introducing natural number sequences and their total. Lastly, Task 4 asked user for integers as input until a termination condition was satisfied by using a do-while loop. When combined, these exercises improved understanding about C's user interface, arithmetic operations, and iteration.