

CC-112L

Programming Fundamentals

Laboratory 02

Introduction to C & Structured Program Development

Version: 1.0.0

Release Date: 24-07-2022

Department of Information Technology

University of the Punjab

Lahore, Pakistan

Contents:

- Learning Objectives
- Required Resources
- General Instructions
- Background and Overview
 - C Language
 - Program Sequence
 - Arithmetic Operators
 - Relational Operators
 - **if** Statement
 - **if...else** Statement
 - Increment & Decrement Operators
 - Microsoft ® Visual Studio
- Activities
 - Pre-Lab Activity
 - MinGW installation
 - A simple C Program
 - Comments
 - Preprocessor Directive
 - Escape sequences
 - Execution Sequence of a Program
 - Arithmetic Operators
 - Task 01: Adding Two Numbers
 - Task 02: Arithmetic Expressions
 - In-Lab Activity
 - Execute C program with Command Prompt
 - Relational Operators
 - **if** Statement
 - Multiple **if** Statement
 - **if...else** Statement
 - **if...else if** Statement
 - Increment and Decrement Operators
 - Task 01: Find Largest Number
 - Task 02: Fill in the blank area
 - Task 03: Dry Run Program
 - Task 04: Find and Resolve Errors
 - Bonus Task: Add Digits
 - Post-Lab Activity
 - Task 01: Simple Calculator
- Submissions
- Evaluations Metric
- References and Additional Material
- Lab Time and Activity Simulation Log

Learning Objectives:

- Writing First C program
- Understanding Comments
- Using Escape Sequences
- Execution Sequence of Program
- Using Arithmetic Operators
- Execution of a C Program through Command Prompt
- Using Relational Operators
- Using **if** Statement
- Using **if...else** Statement
- Using **if...else if** Statement
- Using Increment and Decrement Operators

Resources Required:

- Desktop Computer or Laptop
- Microsoft ® Visual Studio 2022

General Instructions:

- In this Lab, you are **NOT** allowed to discuss your solution with your colleagues, even not allowed to ask how is s/he doing, this may result in negative marking. You can **ONLY** discuss with your Teaching Assistants (TAs) or Lab Instructor.
- Your TAs will be available in the Lab for your help. Alternatively, you can send your queries via email to one of the followings.

Course Instructors:		
Teacher	Prof. Dr. Syed Waqar ul Qounain	s[REDACTED]
Teacher Assistants	Usman Ali	[REDACTED]
	Saad Rahman	bsef19m021@pucit.edu.pk

Background and Overview:

C Language: C language is a general-purpose computer programming language. It was created in the 1970s by Dennis Ritchie, and remains very widely used and influential.

© [https://en.wikipedia.org/wiki/C_\(programming_language\)](https://en.wikipedia.org/wiki/C_(programming_language))

Program Sequence: Statements are executed one after another in programming. The order in which these statements are executed is program sequence.

Arithmetic Operators: Operators which perform mathematical operations such as addition, subtraction, etc.

Relational Operators: Operators which create a relationship and compares values of the two operands.

if Statement: if statement executes a block of a code, if a specific condition is true.

if...else Statement: if...else statement executes two different blocks of code depending if a specific condition is true or false.

Increment & Decrement Operators: The increment operator (++) and the decrement operator (--) add one to and subtract one from an integer variable.

Microsoft ® Visual Studio: It is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps.

Microsoft ® Visual Studio supports 36 different programming languages and allows the code editor and debugger to support nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic, .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past.

© https://en.wikipedia.org/wiki/Microsoft_Visual_Studio

Activities:

Pre-Lab Activities:

MinGW installation:

- Download MinGW from <https://sourceforge.net/projects/mingw/>

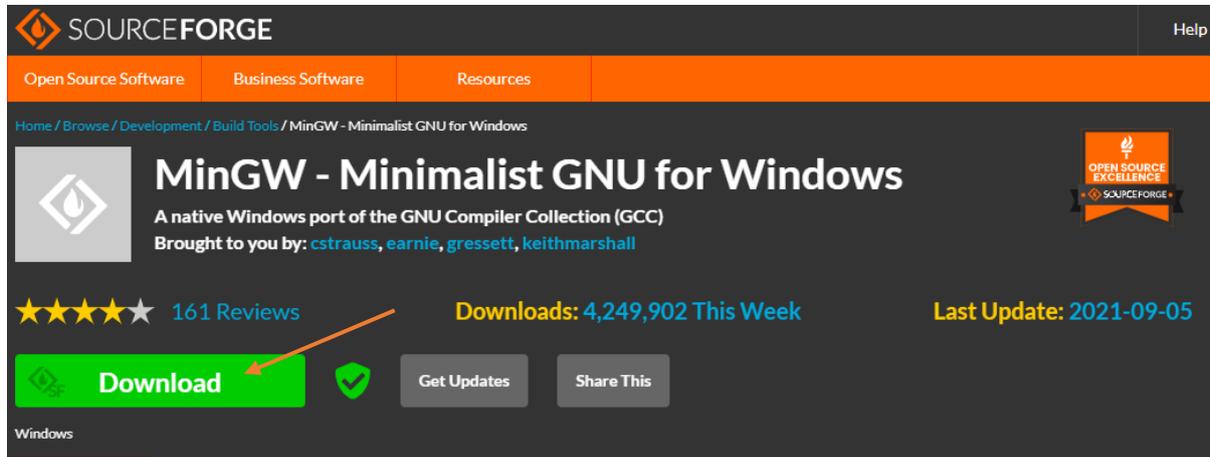


Fig. 01 (MinGW Installation)

- Double-click the downloaded file and follow the on-screen instructions for the installation
- When MinGW Installation Manager window opens, you'll see several packages in the upper-right panel. Check the boxes next to "mingw32-base" and "mingw-gcc-g++"

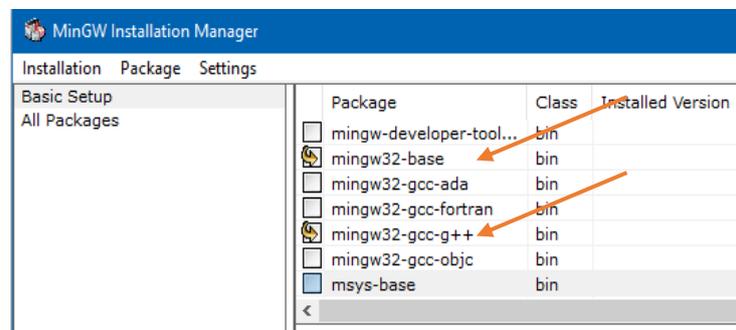


Fig. 02 (MinGW Installation)

- Click the "Installation" menu and select "Apply Changes"

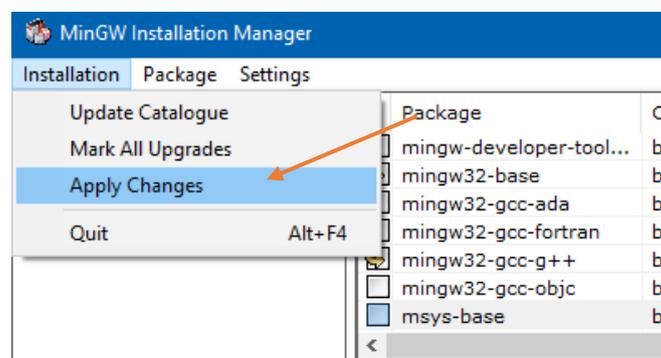


Fig. 03 (MinGW Installation)

- Press the “**Windows**” key and type “**environment**”
- Click “**Edit the system environment variables**”
- System Properties dialog will open. Click the “**Environment Variables...**” button

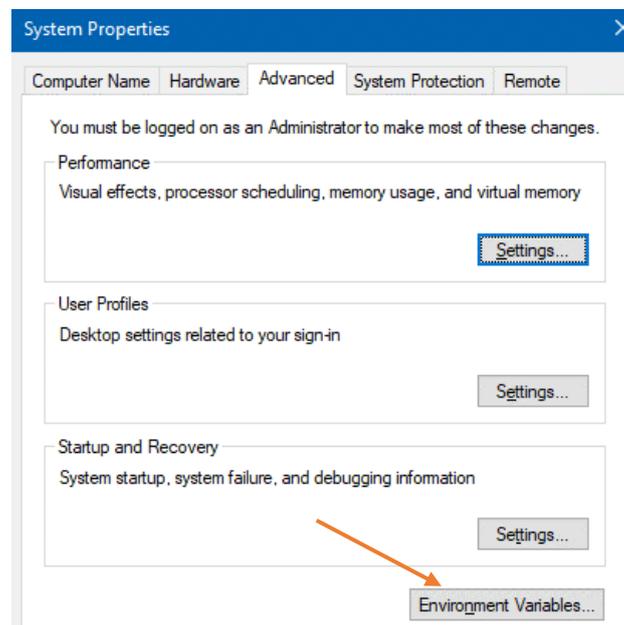


Fig. 04 (Environment Variable)

- Double Click "**Path**" option under "**System variables**"
- Click “**New**”
- Type “**C:\MinGW\bin**” and click OK

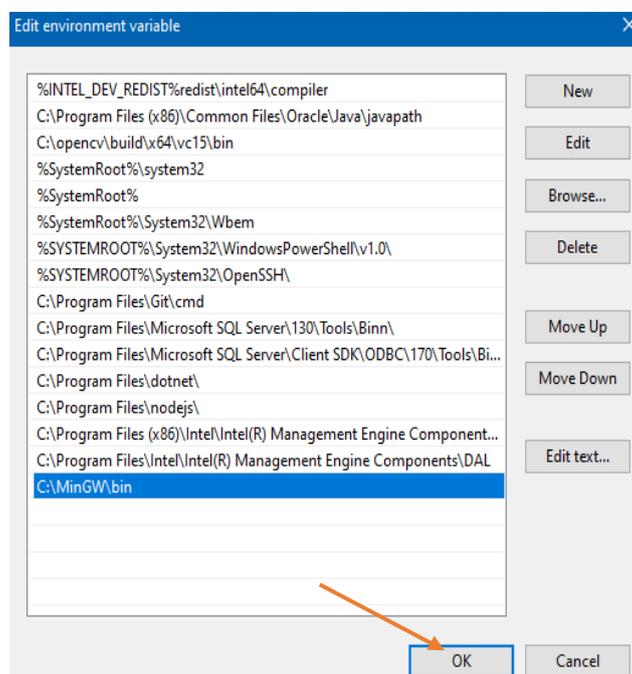


Fig. 05 (Environment Variable)

A simple C program:

A C program which prints a text.

```

1  #include <stdio.h>
2
3  int main()
4  {
5
6  printf("This is my first program.");
7  }

```

Fig. 06 (C program)

The output of above program will be a text line “**This is my first program.**”.

```

Microsoft Visual Studio Debug Console
This is my first program.
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 17176)
exited with code 0.
Press any key to close this window . . .

```

Fig. 07 (C program output)

Comments:

In C, line beginning with “//” is considered as a comment. Comments are added to improve the readability of the code. For commenting multiple lines “/* ... */” is used where /* is used in start and /* is used in the end. The green lines in the following code are comments.

```

1  #include <stdio.h>
2
3  // Start of execution
4  int main()
5  {
6
7  //Print a line
8  printf("This is my first program.");
9
10 /*
11 printf("This code is commented.");
12 printf("This code is commented.");
13 */
14 }

```

Fig. 08 (Comments in C)

Preprocessor Directive:

“**#include <stdio.h>**” is a C preprocessor directive. The preprocessor handles lines beginning with # before compilation. It tells the preprocessor to include the contents of the “**standard input/output header ()**”. The contents of headers will be explained in more detail in the upcoming manuals.

The main Function:

“**int main() { ... }**” is a part of every C program. The parentheses after main indicate that main is a **function** (building block of a program). C programs must contain a main function. Execution of every program begins at the main function. Functions can return information. The keyword “**int**” to the left of main indicates that main returns an integer value.

Escape sequences:

The “**backslash (\)**” is an escape character in a string. It indicates that printf should do something different. Backslash combined with the next character to form an **escape sequence**. Some common escape sequences are:

Escape sequence	Description
<code>\n</code>	Moves the cursor to the beginning of the next line.
<code>\t</code>	Moves the cursor to horizontal tab space.
<code>\\</code>	<code>\\</code> is required to insert a backslash character in a string.
<code>\"</code>	Strings are enclosed in double quotes, <code>\"</code> is required to insert a double-quote character in a string.

Following example demonstrates the use of “`\n`” escape sequence. It is used in the **printf** statement to display text on two lines.

```

1  #include <stdio.h>
2
3  // Start of execution
4  int main()
5  {
6      printf("Text on first line.\nText on second line.");
7  }
```

Fig. 09 (Escape Sequence Example)

Execution Sequence of a Program:

A program is executed line by line in a top to bottom fashion. The execution of a program starts from the **main()** function.

Fig. 10 (Execution Sequence)

The execution process of above program is as follows:

- The execution starts at line 4
- On line 6, an integer is defined
- Then line 7 is executed which prints “**Enter integer:**” on the console and on the execution of line 8 program stops for user input

Fig. 11 (Execution Sequence)

- User enters an integer and press enter
- Then line 9 will be executed which prints the integer entered by the user
- The program ends

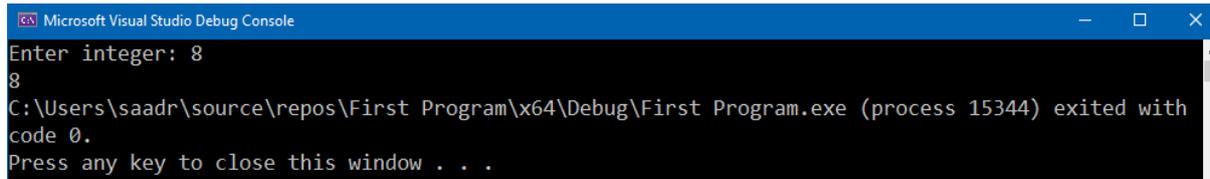


Fig. 12 (Execution Sequence)

Arithmetic Operators:

Operation	Arithmetic Operator
Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus/Remainder	%

The following program demonstrates the multiplication of two numbers and print the result on the console.

```

1  #include <stdio.h>
2
3  // Start of execution
4  int main()
5  {
6      int a = 10;
7      int b = 20;
8
9      int multiply = a * b;
10     printf("Result: %d",multiply);
11 }
    
```

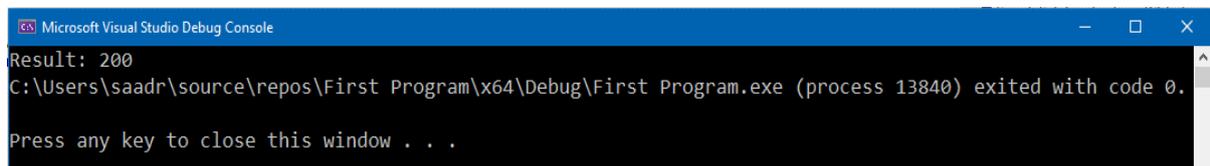


Fig. 13 (Arithmetic in C)

The “**Modulus (%) Operator**” returns the remainder after division. For example, if we divide 9 by 2 the remainder will be 1. Example is shown in the figure below:

```

1  #include <stdio.h>
2
3  int main()
4  {
5      int remainder = 9 % 2;
6      printf("Remainder: %d",remainder);
7  }
    
```

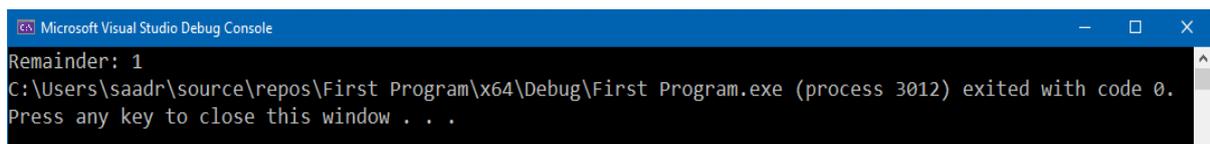
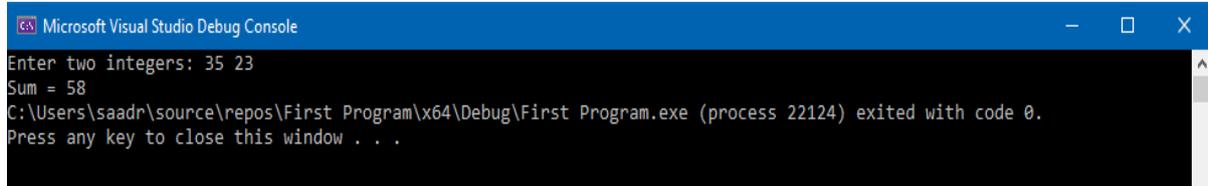


Fig. 14 (Arithmetic in C)

Task 01: Adding Two Numbers**[Estimated 15 minutes / 10 marks]**

Create a C program which:

- Takes two integers as an input from user
- Add both integers
- Display the Sum of integers on the Console as shown in figure



```
Microsoft Visual Studio Debug Console
Enter two integers: 35 23
Sum = 58
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 22124) exited with code 0.
Press any key to close this window . . .
```

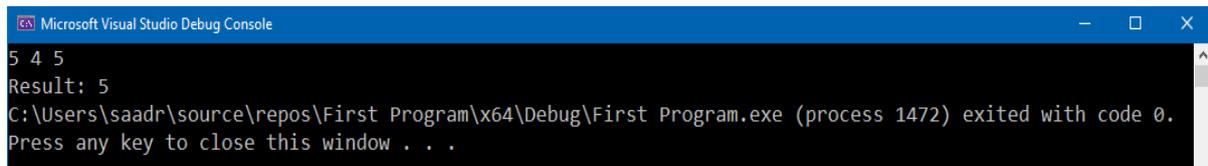
Fig. 15 (Pre-Lab Task)

- Submit “.c” file named your “**Roll No**” on Google Classroom

Task 02: Arithmetic Expression**[Estimated 15 minutes / 10 marks]**

Create a C program which:

- Takes take three integers as an input from user
- Solve the arithmetic expression “ $(a + b \times c) / 5$ ”
- Display the Result on the Console as shown in figure



```
Microsoft Visual Studio Debug Console
5 4 5
Result: 5
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 1472) exited with code 0.
Press any key to close this window . . .
```

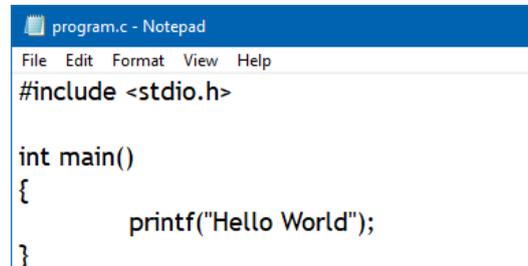
Fig. 16 (Pre-Lab Task)

- Submit “.c” file named your “**Roll No**” on Google Classroom

In-Lab Activities:

Execute C program with Command Prompt:

- Create a text file
- Name the text file to “**Program**” with extension “.c”
- Add the following code to the file



```

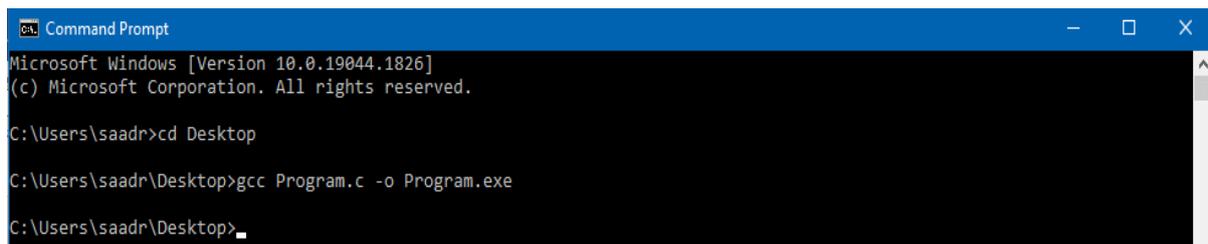
#include <stdio.h>

int main()
{
    printf("Hello World");
}

```

Fig. 17 (Execution in CMD)

- Open “**CMD**”
- Change directory to the path where you created “**Program.c**” file
- Type “**gcc Program.c -o Program.exe**”, where “**Program.c**” is your file name and “**Program.exe**” is the name, you want to give to the compiled program with “.exe” extension
- Press “**Enter**”



```

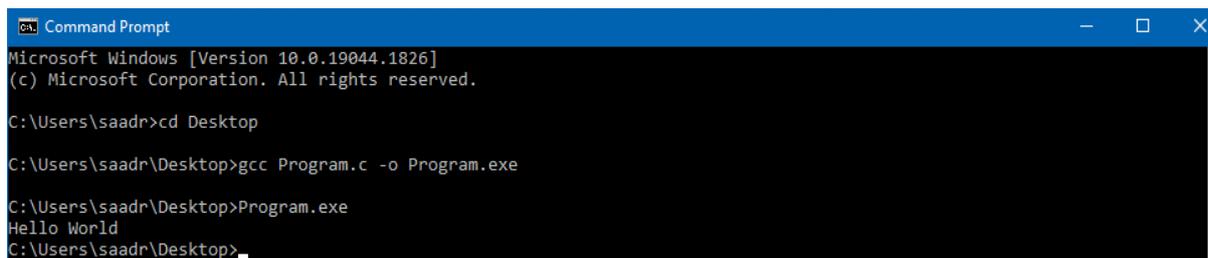
Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\saadr>cd Desktop
C:\Users\saadr\Desktop>gcc Program.c -o Program.exe
C:\Users\saadr\Desktop>_

```

Fig. 18 (Execution in CMD)

- Type “**Program.exe**”. Press “**Enter**”. Your program will be executed



```

Microsoft Windows [Version 10.0.19044.1826]
(c) Microsoft Corporation. All rights reserved.

C:\Users\saadr>cd Desktop
C:\Users\saadr\Desktop>gcc Program.c -o Program.exe
C:\Users\saadr\Desktop>Program.exe
Hello World
C:\Users\saadr\Desktop>_

```

Fig. 19 (Execution in CMD)

Relational Operators:

Operation	Relational Operator
Greater than	>
Smaller than	<
Greater than equal to	>=
Smaller than equal to	<=
Equal to	==
Not equal to	!=

if Statement:

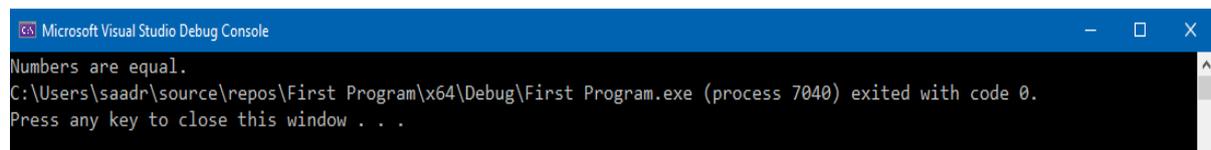
The if Statement runs a block of code if a condition is true. Its syntax is “**if (condition) {...}**”

In the figure below if Statement is used to compare (using “==” relational operator) two integers if they are equal or not.

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int a = 10;
6      int b = 10;
7
8      if (a == b)
9      {
10         printf("Numbers are equal.");
11     }
12 }
```

Fig. 20 (if Statement)

The console will display “**Numbers are equal.**” as the condition is true. If the condition is false then nothing will be displayed on the console.



```
Microsoft Visual Studio Debug Console
Numbers are equal.
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 7040) exited with code 0.
Press any key to close this window . . .
```

Fig. 21 (if Statement)

Multiple if statement:

Figure below shows an example code for multiple if Statement.

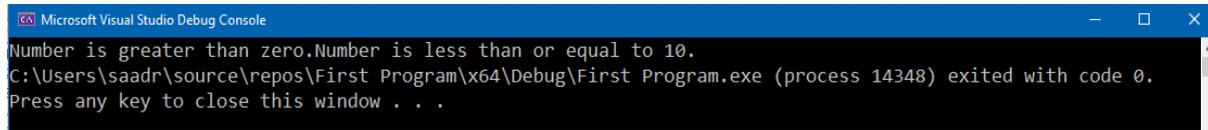
```
1  #include <stdio.h>
2
3  int main()
4  {
5      int a = 10;
6
7      if (a > 0)
8      {
9          printf("Number is greater than zero.");
10
11         if (a > 10)
12         {
13             printf("Number is greater than 10.");
14         }
15
16         if (a <= 10)
17         {
18             printf("Number is less than or equal to 10.");
19         }
20     }
21 }
```

Fig. 22 (if Statement)

The execution of above program is as follows:

- At line 5, integer “**a**” is assigned value 10
- At line 7, it checks if condition, as it is true i.e., a is bigger than 0, so “**Number is greater than 0**” is displayed on the console

- At line 11, it checks if condition, as it is false, as a is not greater than 10, so program shifts to line 15
- At line 16, it checks if condition, as it is true i.e., a is equal to 0, so “**Number is less than or equal to 10**” is displayed on the console



```

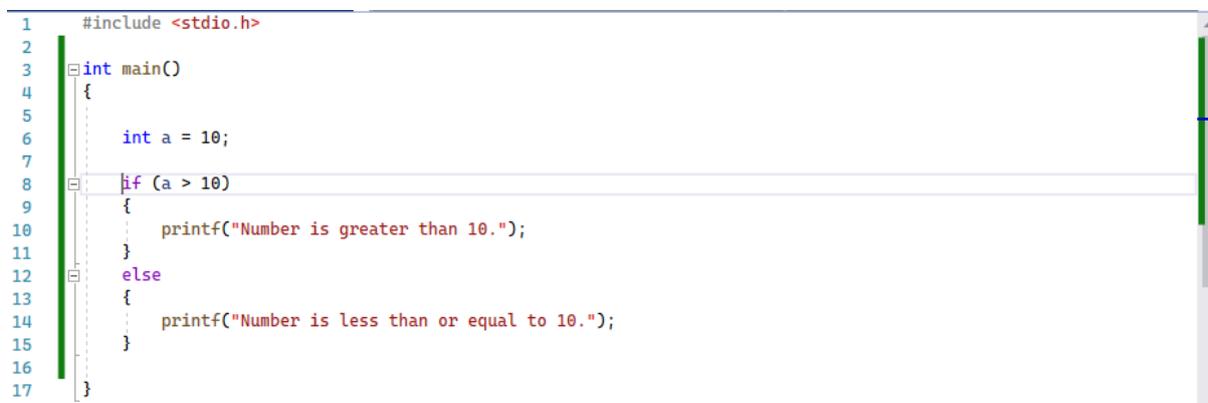
Microsoft Visual Studio Debug Console
Number is greater than zero.Number is less than or equal to 10.
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 14348) exited with code 0.
Press any key to close this window . . .

```

Fig. 23 (if Statement)

if...else Statement:

The if...else Statement runs a specific block of code depending if a condition is true or false. Its syntax is “**if (condition) {...} else {...}**”. Figure below shows an example code for if...else Statement.



```

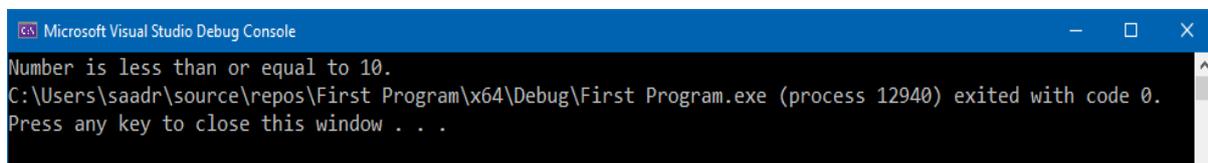
1  #include <stdio.h>
2
3  int main()
4  {
5
6      int a = 10;
7
8      if (a > 10)
9      {
10         printf("Number is greater than 10.");
11     }
12     else
13     {
14         printf("Number is less than or equal to 10.");
15     }
16
17 }

```

Fig. 24 (if...else Statement)

The execution of above program is as follows:

- At line 6, integer “**a**” is assigned value 10
- At line 8, it checks if condition, as it is false, as a is not greater than 10, so program shifts to line 12
- At line 12, it enters the else condition and “**Number is less than or equal to 10**” is displayed on the console



```

Microsoft Visual Studio Debug Console
Number is less than or equal to 10.
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 12940) exited with code 0.
Press any key to close this window . . .

```

Fig. 25 (if...else Statement)

if...else if Statement:

The if...else if Statement runs a specific block of code depending if a any condition is true out of multiple condition or all conditions are false. Its syntax is “**if (condition) {...} else if(condition){...} else {...}**”. Figure below shows an example code for if...else if Statement.

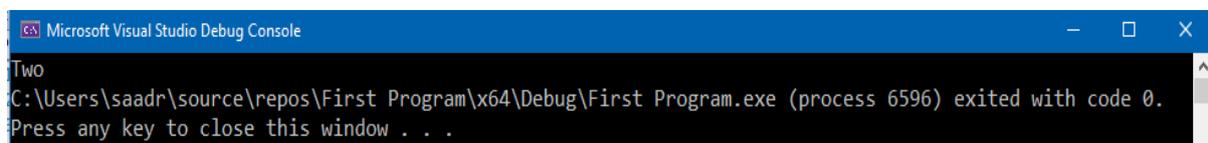
```
1  #include <stdio.h>
2  int main()
3  {
4      int a = 2;
5      if (a == 1)
6      {
7          printf("One");
8      }
9      else if (a==2)
10     {
11         printf("Two");
12     }
13     else if(a == 3)
14     {
15         printf("Three");
16     }
17     else
18     {
19         printf("Number is not one, two or three.");
20     }
21 }
```

Fig. 26 (if...else if Statement)

The execution of above program is as follows:

- At line 4, integer “a” is assigned value 2
- At line 5, it checks if condition, as it is false, as a is not equal to 1, so program shifts to line 9
- At line 9, it checks if condition, as it is true, i.e., a is equal to 2 then “Two” is displayed on the console

If the value of a was “5”, then else portion will be executed as all conditions becomes false.



```
Microsoft Visual Studio Debug Console
Two
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 6596) exited with code 0.
Press any key to close this window . . .
```

Fig. 27 (if...else if Statement)

Increment and Decrement Operators:

Increment Operator increase the value of integer by 1 while decrement operator decreases the value of integer by 1.

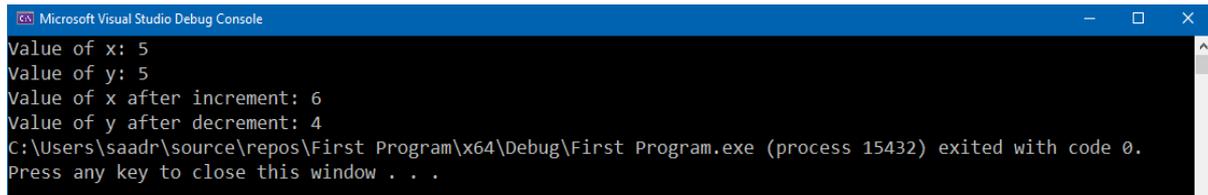
A coding example is shown below in figure:

```
1  #include <stdio.h>
2  int main()
3  {
4      int x = 5;
5      int y = 5;
6
7      printf("Value of x: %d", x);
8      printf("\nValue of y: %d", y);
9
10     x++;
11     y--;
12
13     printf("\nValue of x after increment: %d", x);
14     printf("\nValue of y after decrement: %d", y);
15 }
```

Fig. 28 (Increment and Decrement Operators)

The execution of above program is as follows:

- At line 4, integer “**x**” is assigned value 5
- At line 5, integer “**y**” is assigned value 5
- At line 7, value of x is displayed on Console
- At line 8, value of y is displayed on Console
- At line 10, value of x increased by 1 i.e., value of x becomes “**6**”
- At line 11, value of y decreased by 1 i.e., value of y becomes “**4**”
- At line 13, value of x is displayed on Console
- At line 14, value of y is displayed on Console



```
Microsoft Visual Studio Debug Console
Value of x: 5
Value of y: 5
Value of x after increment: 6
Value of y after decrement: 4
C:\Users\saadr\source\repos\First Program\x64\Debug\First Program.exe (process 15432) exited with code 0.
Press any key to close this window . . .
```

Fig. 29 (Increment and Decrement Operators)

Task 01: Find Largest Number

[40 minutes / 25 marks]

Write a C program which: [Hint: Use Multiple if Statements]

- Reads three numbers as an input from user
- Find the largest number
- Displays “**Largest Number**” on the Console

Input	Output
2 3 4	Largest number is 4
89 33 56	Largest number is 89
10 33 33	Largest number is 33.

Task 02: Fill the blank area

[25 minutes / 25 marks]

The following program:

- Take a “**obtained marks**” as an input from the user
- Displays grade corresponding to marks using following table:

Criterion	Grade
Greater than equal to 80	A
Greater than equal to 70	B
Greater than equal to 60	C

And “**F**” otherwise.

```

1  #include <stdio.h>
2  int main(){
3      int marks;
4      scanf("%d", &marks);
5      if (/*code here*/)
6      {
7          printf(/*code here*/);
8      }
9      else if (/*code here*/)
10     {
11         printf(/*code here*/);
12     }
13     else if (/*code here*/)
14     {
15         printf(/*code here*/);
16     }
17     else
18     {
19         printf(/*code here*/);
20     }
21 }
    
```

Fig. 30 (In-Lab Task)

Input	Output
88	A
67	B-

Task 03: Dry Run Program.

[30 minutes / 30 marks]

Dry Run the following program and fill up the trace table:

```

1  #include <stdio.h>
2
3  int main()
4  {
5      int num;
6      scanf_s("%d", &num);
7      num++;
8      if (num < 0)
9          num = 0;
10     else if (num % 2 == 0)
11         num--;
12     else
13         num = num * 2;
14     num--;
15     num = num + 5;
16     printf("%d", num);
17
18 }
```

Fig. 31 (In-Lab Task)

Input	num at line 7	num at line 14	Output
1	2	0	5
23			
6			
72			
-12			
0			
99			

Task 04: Find and resolve the errors.

[25 minutes / 20 marks]

(a):

```

1  int main()
2  {
3      int num = 10;
4
5      num = num * 2;
6
7      printf(num);
8
9  }
```

Fig. 32 (In-Lab Task)

(b):

```

1  #include <stdio.h>
2
3  int main()
4  {
5      int a = 10;
6      scanf("%s", a);
7      if (a = 10)
8      {
9          printf(a);
10     }
11     else if (a > 10)
12         scanf(a);
13     a--
```

Fig. 33 (In-Lab Task)

Bonus Task: Add Digits of a Number

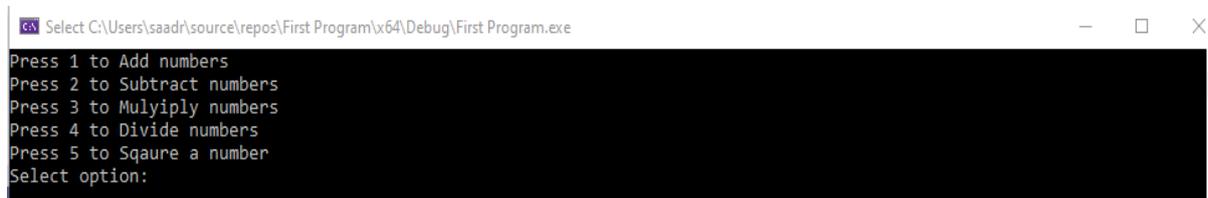
[10 marks]

Read a 3-digit number from user. Add each digit and display the sum of digits at output screen.

Post-Lab Activities:**Task 01: Simple Calculator****[Estimated 45 minutes / 30 marks]**

Create a Console based calculator which:

- Displays the menu on Console and takes an input from user as follows:



```
Select C:\Users\saad\source\repos\First Program\x64\Debug\First Program.exe
Press 1 to Add numbers
Press 2 to Subtract numbers
Press 3 to Mulyiply numbers
Press 4 to Divide numbers
Press 5 to Sqaure a number
Select option:
```

Fig. 34 (Post-Lab Task)

- If user enters “1” as an input then the program takes two numbers as input and display their addition on Console
- If user enters “2” as an input then the program takes two numbers as input and display their subtraction on Console
- If user enters “3” as an input then the program takes two numbers as input and display their multiplication on Console
- If user enters “4” as an input then the program takes two numbers as input and display their division on Console
- If user enters “5” as an input then the program takes a number as input and display its square on Console [Hint: Use Multiplication]
- If user enter any other number, then display “**Incorrect Option**”

Submissions:

- For Pre-Lab Activity:
 - Submit the .c file on Google Classroom and name it with your roll no.
- For In-Lab Activity:
 - Save the files on your PC.
 - TA's will evaluate the tasks offline.
- For Post-Lab Activity:
 - Submit the .c file on Google Classroom and name it with your roll no.

Evaluations Metric:

- All the lab tasks will be evaluated offline by TA's
- **Division of Pre-Lab marks:** [20 marks]
 - Add Two Numbers [10 marks]
 - Arithmetic Expression [10 marks]
- **Division of In-Lab marks:** [100 marks]
 - Task 01: Find Largest Number [25 marks]
 - Task 02: Fill in the blank area [25 marks]
 - Task 03: Dry Run Program [30 marks]
 - Task 04: Find and Resolve Errors [20 marks]
- **Division of Post-Lab marks:** [30 marks]
 - Task 01: Simple Calculator [30 marks]

References and Additional Material:

- MinGW Installation
<https://sourceforge.net/projects/mingw/>
- **if...else** Statement in C
<https://www.programiz.com/c-programming/c-if-else-statement>
- Operators in C
<https://www.programiz.com/c-programming/c-operators>

Lab Time Activity Simulation Log:

- Slot – 01 – 00:00 – 00:15: Class Settlement
- Slot – 02 – 00:15 – 00:30: Execute C on Command Prompt
- Slot – 03 – 00:30 – 00:45: Execute C on Command Prompt
- Slot – 04 – 00:45 – 01:00: In-Lab Task
- Slot – 05 – 01:00 – 01:15: In-Lab Task
- Slot – 06 – 01:15 – 01:30: In-Lab Task
- Slot – 07 – 01:30 – 01:45: In-Lab Task
- Slot – 08 – 01:45 – 02:00: In-Lab Task
- Slot – 09 – 02:00 – 02:15: In-Lab Task
- Slot – 10 – 02:15 – 02:30: In-Lab Task
- Slot – 11 – 02:30 – 02:45: In-Lab Task
- Slot – 12 – 02:45 – 03:00: Discussion on Post-Lab Task