| **CL-1002 Programming Fundamentals** | **LAB - 03**  **Introduction to IDE and Basic Programming Constructs** | |
| --- | --- | --- |
| **NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES**  **Fall 2022** | |  |

**Scratch and C side by side**

| **Scratch** | **C Code** |
| --- | --- |
| CodeGraphical user interface, text  Description automatically generated  Graphical user interface, application  Description automatically generated  Output | Code  Output |

| **Scratch** | **C Code** |
| --- | --- |
| CodeGraphical user interface  Description automatically generated  OutputA child standing in front of a black screen  Description automatically generated with medium confidence | Code  Output |

| **Scratch** | **C Code** |
| --- | --- |
| Code  Graphical user interface, text  Description automatically generated  OutputGraphical user interface, application, PowerPoint  Description automatically generated | Code  Output |

**INTRODUCTION TO THE IDE**

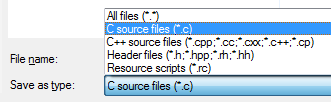
* IDE stands for Integrated Development Environment. It is a software application that provides facilities for developing software.
* It consists of tools such as source code editor, automation tools, and debugger.
* For C language programming we will be using Dev-C++.
* Dev-C++ is a full-featured Integrated Development Environment (IDE) for the C/C++ programming language.

**HELLO WORLD PROGRAM**

* Launch Dev-C++ and use **Ctrl + n** to create a new source file.
* Write the following code snippet:



* Press F11 to Compile & Run the Program. You will be prompted to save the file to some location. Save it as a “.c” file as shown below:



**Understanding HELLO WORLD Program**

* This program uses (that is, it ‘includes’) code from the C-language ‘standard input/output library, stdio, using this statement:

**#include <stdio.h>**

* The code that starts with the name main is the ‘main function’ – in other words, it is the first bit of code that runs when the program runs. The function name is followed by a pair of parentheses. The code to be run is enclosed between a pair of curly brackets:

**main() {  
   
}**

* In this case, the code calls the C printf function to print the string (the piece of text) between double-quotes.

**printf("hello world");**

**DATATYPES IN C**

* In C programming, data types are declarations for variables. This determines the type and size of data associated with variables.

Here's a table containing commonly used types in C programming for quick access.

| **Data Type** | **Description** | **Size(bytes)** |
| --- | --- | --- |
| int | Integers are whole numbers that can have both zero, positive and negative values but no decimal values.  It can take 232 distinct states from -2147483648 to 2147483647. | 4 |
| float | Floating type variables can hold real numbers precision of 6 digits | 4 |
| double | Floating type variables can hold real numbers with precision of 14 digits | 8 |
| char | Character data type allows a variable to store only one character. | 1 |

**VARIABLES IN C**

* In programming, a variable is a container (storage area) to hold data.
* To indicate the storage area, each variable should be given a unique name (identifier). Variable names are just the symbolic representation of a memory location.

**Syntax**:

A variable can be declared and initialized using following syntax:

**datatype variable\_name = value ;**

**Example:**

**int playerScore = 95;**

Here, **playerScore** is a variable of **int** type. **95** is stored into the variable**.**

**Rules for naming a variable**

* A variable name can only have letters (both uppercase and lowercase letters), digits and underscore.
* The first letter of a variable should be either a letter or an underscore.
* There is no rule on how long a variable name (identifier) can be. However, you may run into problems in some compilers if the variable name is longer than 31 characters.

Note: You should always try to give meaningful names to variables. For example: firstName is a better variable name than fn.

* C is a strongly typed language. This means that the variable type cannot be changed once it is declared.

**FORMAT SPECIFIERS**

* Format Specifiers are strings used in the formatted input and output.
* The format specifiers are used in C to determine the format of input and output.
* Using this concept the compiler can understand that what type of data is in a variable while taking input using the scanf() function and printing using printf() function.

There are many different format specifiers, but some of the commonly used ones are given below:

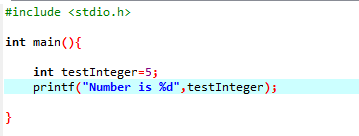
| Format specifier | Description |
| --- | --- |
| %d | Used for integers (int) |
| %c | Used for characters (char) |
| %f | Used for floats (float) |
| %s | Used for string (array of chars) |

**OUTPUT IN C**

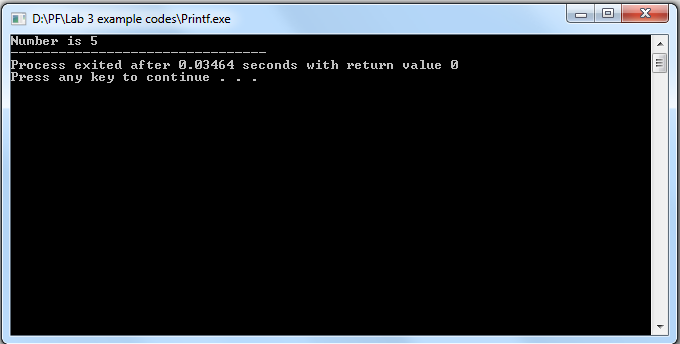
* In C programming, printf() is one of the main output function. The function sends formatted output to the screen.
* The syntax of printf() function is given below:

**printf("format string",variable\_name);**

**Example # 2:**



**Output**



We use %d format specifier to print int types. Here, the %d inside the quotations will be replaced by the value of testInteger

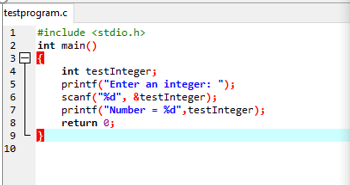
**INPUT IN C**

* In C programming, scanf() is one of the commonly used function to take input from the user.
* The scanf() function reads formatted input from the standard input such as keyboards.
* The syntax of printf() function is given below:

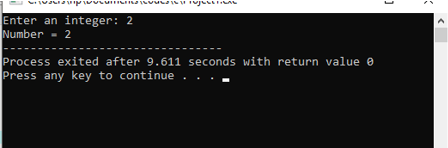
**scanf("format string",&variable\_name);**

* The format string will contain the format specifier, which will give information about the data type to be input.
* The variable name must be preceded by “&” sign, “&” sign is the address operator in C language. This is because scanf() function needs to know memory address of location to store the variable.

**Example:**



**Output**



**ESCAPE SEQUENCE**

* Escape sequences are used to enter some special characters into the program.
* When a character is preceded by a backslash (\), it is called an escape sequence and it has a special meaning to the compiler. For example, \n in the following statement is a valid character and it is called a new line character.

| Escape Sequence & Description |
| --- |
| \t Inserts a tab in the text. |
| \b Inserts a backspace in the text. |
| \n Inserts a newline in the text. |
| \r Inserts a carriage return in the text. |
| \f Inserts a form feed in the text. |
| \’ Inserts a single quote character in the text. |
| \” Inserts a double quote character in the text. |
| \\ Inserts a backslash character in the text. |
| \?  Inserts a question mark in the text. |
| \a  Play beep or Alarm |

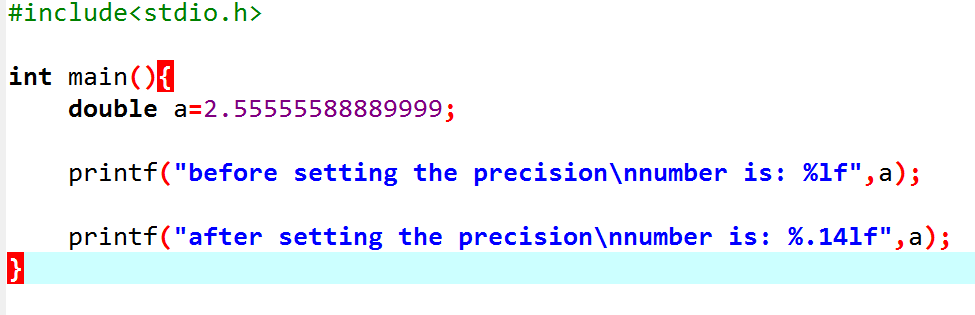
**PRECISION SETTING IN C**

Precision is specified by the number of digits after the decimal point for the outputs for float as well as double numbers.

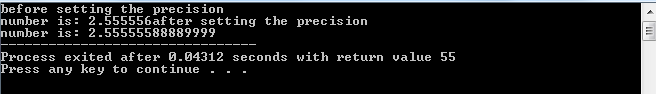
If the precision is not specified, it would be according to the default setting in the computer, which is generally 6 digits.

The precision can be specified in the format specifiers place by a period (.) followed by a positive number equal to the number of digits desired.

**Example**:



**Output**:



The above will display 14 digits after the decimal point after setting the precision otherwise it will display only 6 characters even though double datatype precision is 14 digit as seen in datatypes section.

OPERATORS:

**C Arithmetic Operators**

There are many operators in C for manipulating data which include arithmetic Operators, Relational Operators, Logical operators and many more which will be discussed   
accordingly. Some of the fundamental operators are:

| **Operator** | **Description** | **Example** |
| --- | --- | --- |
| + | Adds two operands. | A + B = 30 |
| − | Subtracts second operand from the first. | A − B = -10 |
| \* | Multiplies both operands. | A \* B = 200 |
| / | Divides 1st operand by 2nd operand. | B / A = 2 |
| % | Modulus Operator and remainder of after an integer division. | B % A = 0 |
| ++ | Increment operator increases the integer value by one. | A++ = 11 |
| -- | Decrement operator decreases the integer value by one. | A-- = 9 |

**C Relational Operators**

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

Relational operators are used in decision making and loops.

| Operator | Meaning of Operator | Example |
| --- | --- | --- |
| == | Is Equal to | 5 == 3 returns 0 |
| > | Is Greater than | 5 > 3 returns 1 |
| < | Is Less than | 5 < 3 returns 0 |
| != | Is Not equal to | 5 != 3 returns 1 |
| >= | Is Greater than or equal to | 5 >= 3 returns 1 |
| <= | Is Less than or equal to | 5 <= 3 returns 0 |

**C Logical Operators**

An expression containing a logical operator returns either 0 (false) or 1 (true) depending upon whether the 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 (false) | If c = 5 then, expression !(c==5) equals to 0. |

**Bitwise Operators C**

The computer runs on binary, thus on the lowest level all operations are performed on bits. In C language, you can perform these operations using bitwise operators.

| Operator | Meaning of operators |
| --- | --- |
| & | Bitwise AND |
| | | Bitwise OR |
| ^ | Bitwise exclusive OR |
| ~ | Bitwise complement |
| << | Shift left |
| >> | Shift right |

**Comma Operator**

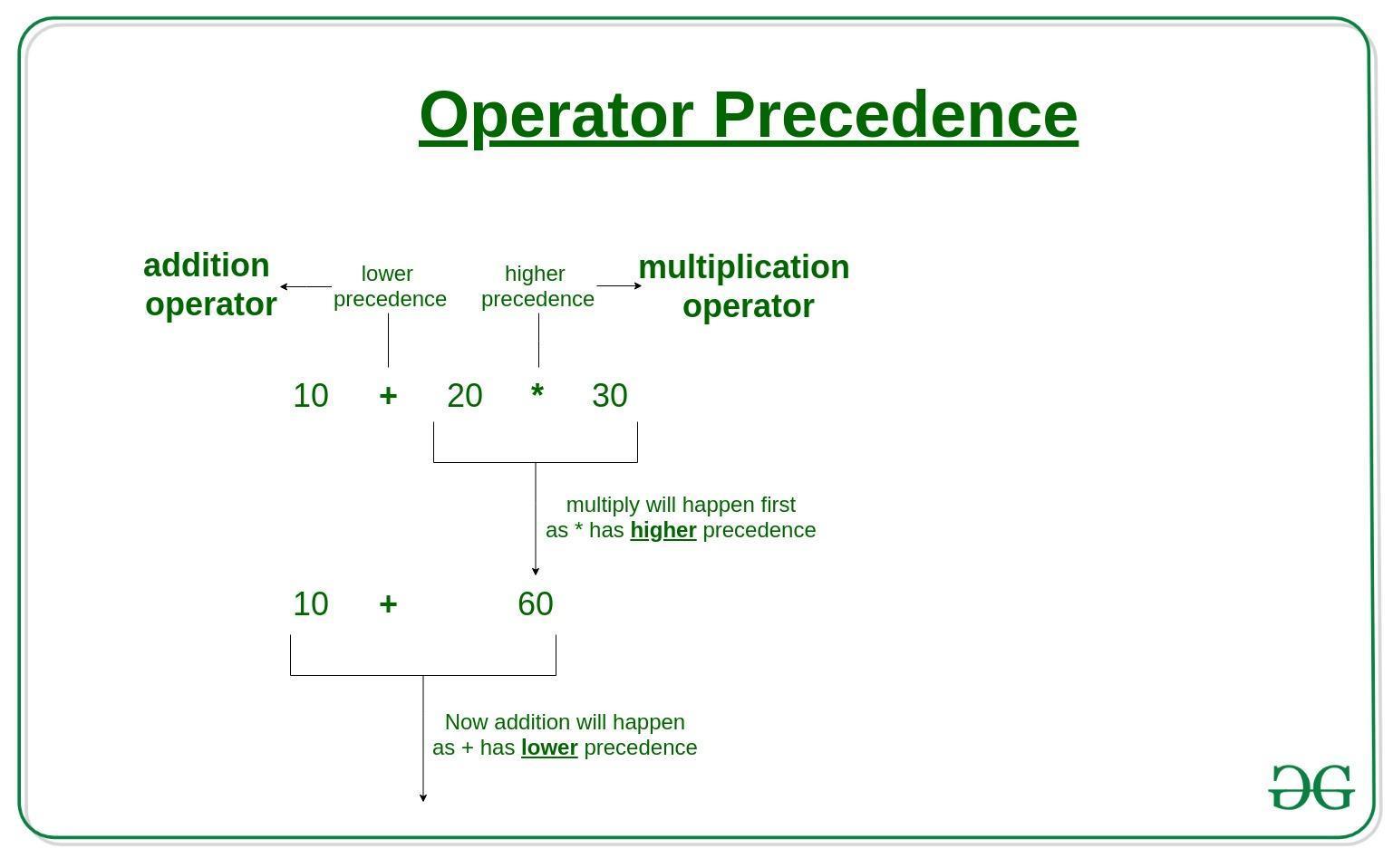
Comma operators are used to link related expressions together. For example:

**int a, c = 5, d;**

**The sizeof operator**

The sizeof is a unary operator that returns the size of data (constants, variables, array, structure, etc).

**Operators Precedence in C**



Operator precedence determines the grouping of terms in an expression and decides how an expression is evaluated. Certain operators have higher precedence than others; for example, the multiplication operator has a higher precedence than the addition operator.

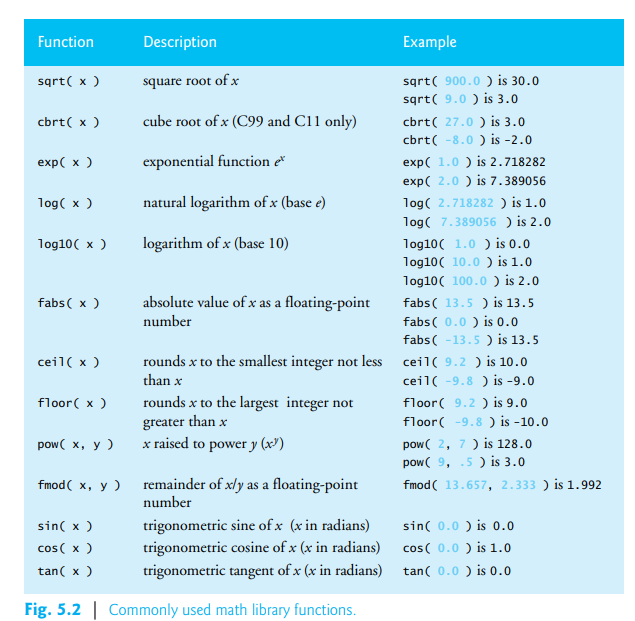
For example, x = 7 + 3 \* 2; here, x is assigned 13, not 20 because operator \* has a higher precedence than +, so it first gets multiplied with 3\*2 and then adds into 7.

In the table below, operators with the highest precedence appear at the top of the table, those with the lowest appear at the bottom. Within an expression, higher precedence operators will be evaluated first.

| **Category** | **Operator** | **Associativity** |
| --- | --- | --- |
| Postfix | () [] -> . ++ - - | Left to right |
| Unary | + - ! ~ ++ - - (type)\* & sizeof | Right to left |
| Multiplicative | \* / % | Left to right |
| Additive | + - | Left to right |
| Shift | << >> | Left to right |
| Relational | < <= > >= | Left to right |
| Equality | == != | Left to right |
| Bitwise AND | & | Left to right |
| Bitwise XOR | ^ | Left to right |
| Bitwise OR | | | Left to right |
| Logical AND | && | Left to right |
| Logical OR | || | Left to right |
| Conditional | ?: | Right to left |
| Assignment | = += -= \*= /= %=>>= <<= &= ^= |= | Right to left |
| Comma | , | Left to right |

**Math library functions**

*#include <math.h>*

Math library functions allow you to perform certain common mathematical calculation

**LAB#03 EXERCISES**

| **QUESTION#1** A car traveled for some hours. Take the number of Hours travelled as input on runtime, and then take input for the Distance Travelled during this time, Calculate Average Speed and display it on the screen.  **QUESTION#2**  Write a C program to print the following pattern using escape sequences.  escape sequence  **QUESTION#3** Write a C program that takes two integer values as input from the user. Then swap the values taken from the user and display the output of the variables. (Value of num1 should be stored in num2 and vice versa).  **QUESTION#4** A customer asks the IT firm to develop a program in C language, which can take tax rate and salary from the user on runtime and then calculate the tax, the user has to pay and the salary he/she will have after paying the tax. Display the net income after tax deduction.  **QUESTION#5** Calculate the volume of a cone, using FLOAT data type for all values and output the final answer up to 3 decimal places exactly.  **QUESTION#6** Find both roots of the quadratic equation using the formula, after taking a, b, c as inputs: |
| --- |
|  |