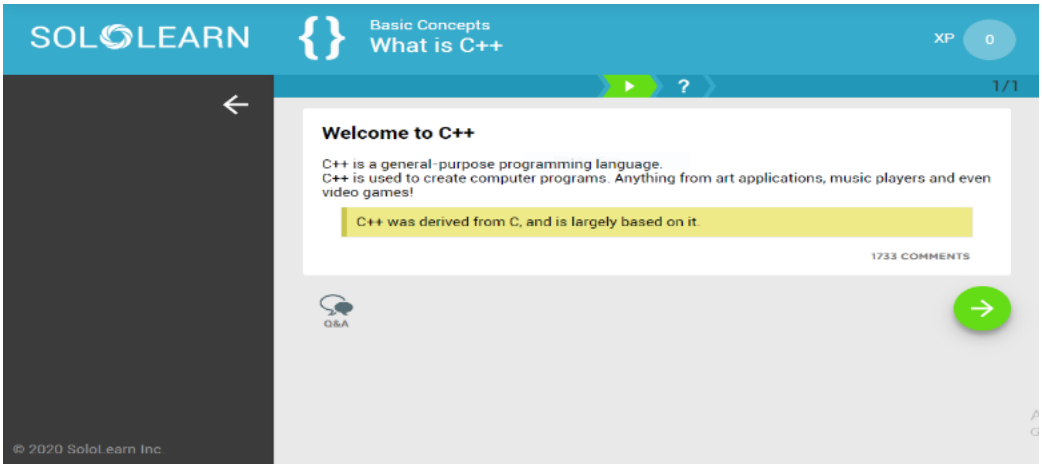
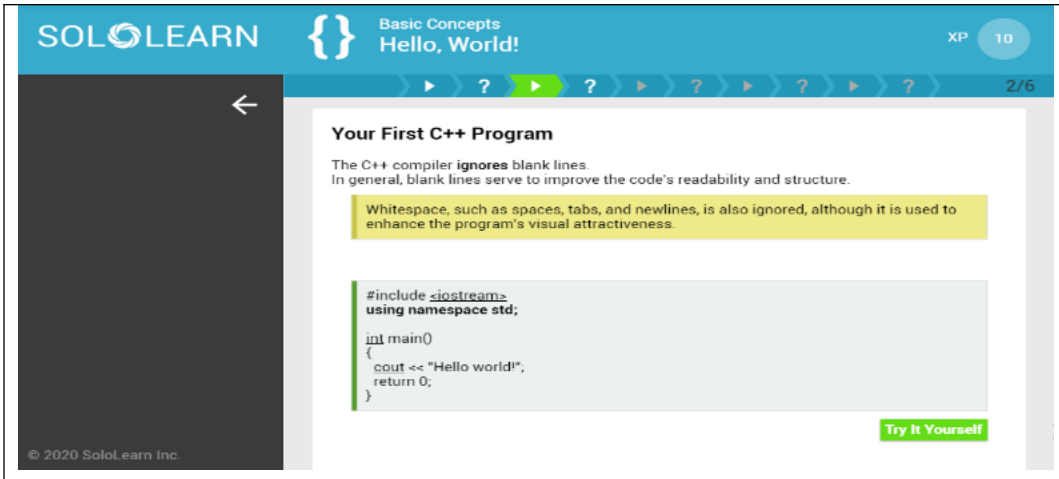
DAILY ASSESSMENT FORMAT

Date:	22-06-2020	Name:	Rajeshwari Gadagi
Course:	C++ Programming	USN:	4AL17EC076
Topic:	Module 1	Semester & Section:	6th SEM & 'B' SEC
Github Repository:	Rajeshwari-gadagi		

FORENOON SESSION DETAILS

Image of session




SOLOLEARN
Basic Concepts
Hello, World!
XP 10
2/6
Your First C++ Program
The C++ compiler ignores blank lines.
In general, blank lines serve to improve the code's readability and structure.
Whitespace, such as spaces, tabs, and newlines, is also ignored, although it is used to enhance the program's visual attractiveness.
#include <iostream>
using namespace std;
int main()
{
cout << "Hello world!";
return 0;
}
Try It Yourself
© 2020 SoloLearn Inc.


SOLOLEARN
Basic Concepts
Hello, World!
XP 10
3/6
Main
Program execution begins with the main function, int main().
#include <iostream>
using namespace std;
int main()
{
cout << "Hello world!";
return 0;
}
Try It Yourself
Curly brackets { } indicate the beginning and end of a function, which can also be called the function's body. The information inside the brackets indicates what the function does when executed.
The entry point of every C++ program is main(), irrespective of what the program does.
© 2020 SoloLearn Inc.

C++ Programming

Monday: 22/06/2020

Module 1: Basic Concepts

> What is C++?

C++ is a general purpose programming language. C++ is used to create computer programs. Anything from word applications, music players & even useful games!

> Hello World!

- Your first C++ is a collection of commands or statements.

- The C++ compiler ignores blank lines. In general, blank lines are used to improve the code's readability & structure.

- Main

Program execution begins with the main function, int main()

- The next line, cout << "Hello World!"; results in the display of "Hello World" to the screen

- Statements:

A block is a set of logically connected statements surrounded by opening & closing curly braces.

- Return

The last instruction in the program is the return statement. The line return 0; terminates the main() function & causes it to return the value 0 to the calling process. A non-zero value signals abnormal termination.

> Getting the tools

You can write, save & share your C++ codes on our code playground without installing any additional software.

> Printing a Text

- Your 1st C++ Program

You can add multiple instructions of printing after cout.

- New line

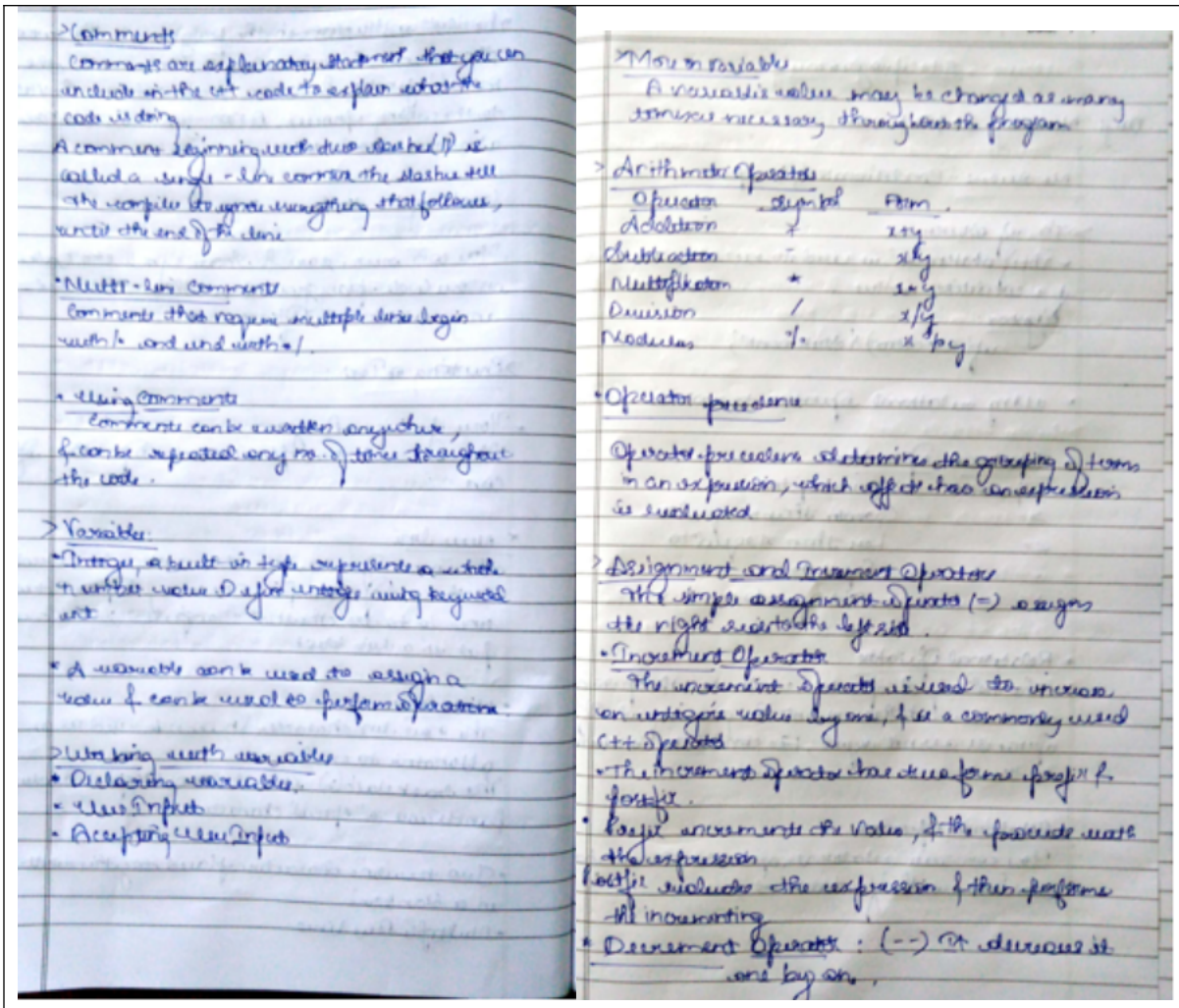
The cout sequence does not insert a line break at the end of the output. One way to print multiple lines is to use the endl manipulator, which will put in a line break.

- New line

The new line character \n can be used as an alternative to endl.

The backslash(\) is called an escape character, & indicates a "special" character.

- Two newline characters placed together result in a blank line.
- Multiple line using



> Comments

Comments are explanatory statement that you can include in the C++ code to explain what the code is doing.

A comment beginning with two slashes (//) is called a single-line comment. The compiler ignores everything that follows, until the end of the line.

• Multi-line Comments

Comments that require multiple lines begin with /* and end with */

• Using Comments

Comments can be written anywhere, & can be repeated any no. of times throughout the code.

> Variables

• Integer, a built in type, represents a whole number value. Define integer using keyword int

• A variable can be used to assign a value & can be used to perform operations.

> Working with variables

• Declaring variables
• User Input
• Accepting User Input

> Main variable

A variable's value may be changed as many times necessary, throughout the program.

> Arithmetic Operators

Operation	Symbol	Form
Addition	+	x+y
Subtraction	-	x-y
Multiplication	*	x*y
Division	/	x/y
Modulus	%	x%y

• Operator precedence

Operator precedence determines the grouping of terms in an expression, which affects how an expression is evaluated.

> Assignment and Increment Operators

The simple assignment operator (=) assigns the right operand to the left side.

• Increment Operator

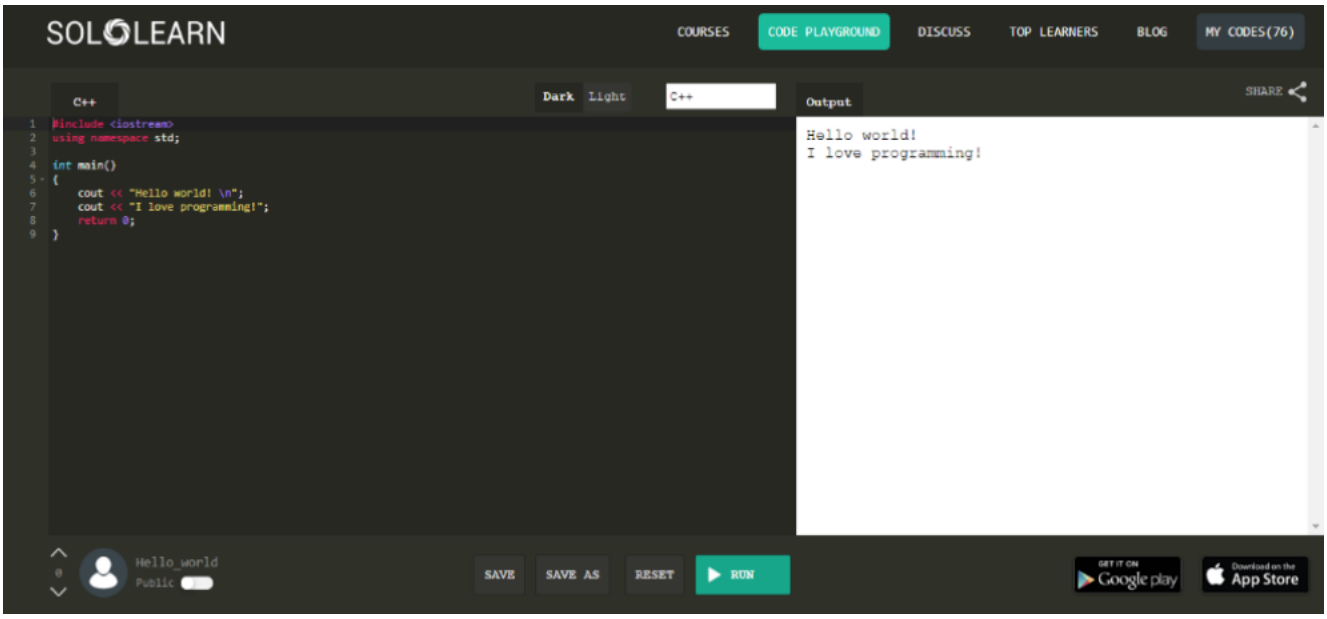
The increment operator is used to increase an integer value by one, & is commonly used in C++ operators.

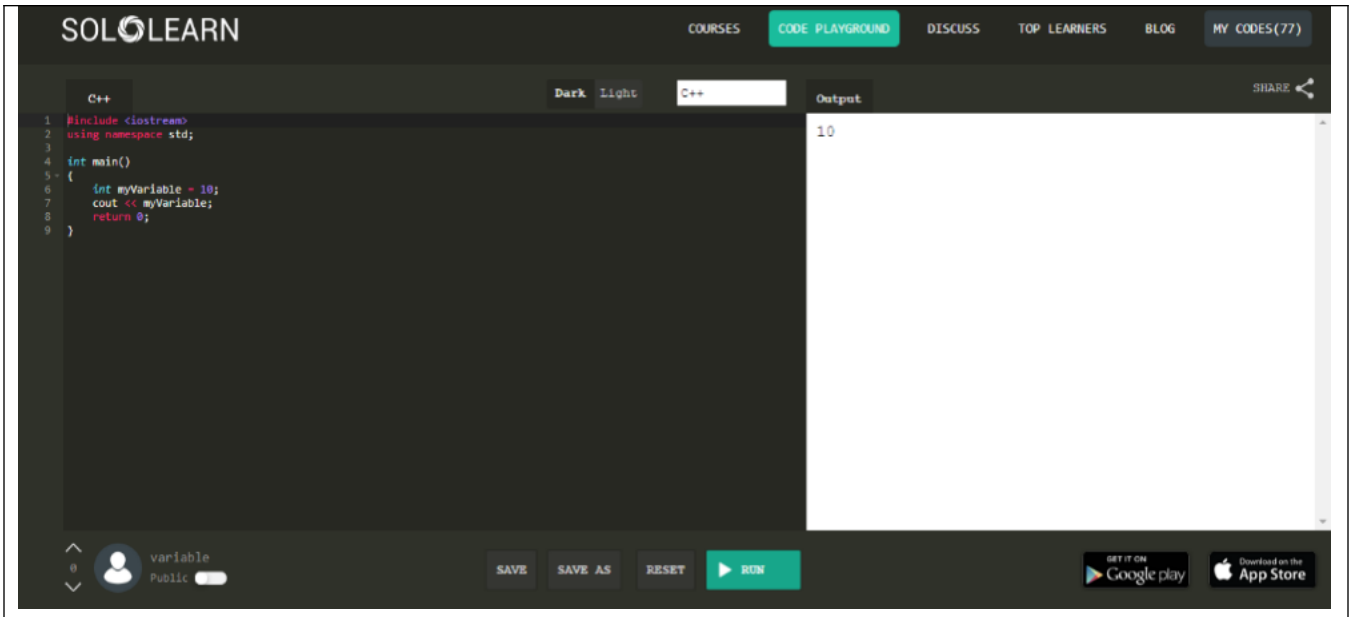
• The increment operator has two forms: prefix & postfix.

• Prefix increments the value, & the operand was the expression.

Postfix includes the expression & then performs the incrementing.

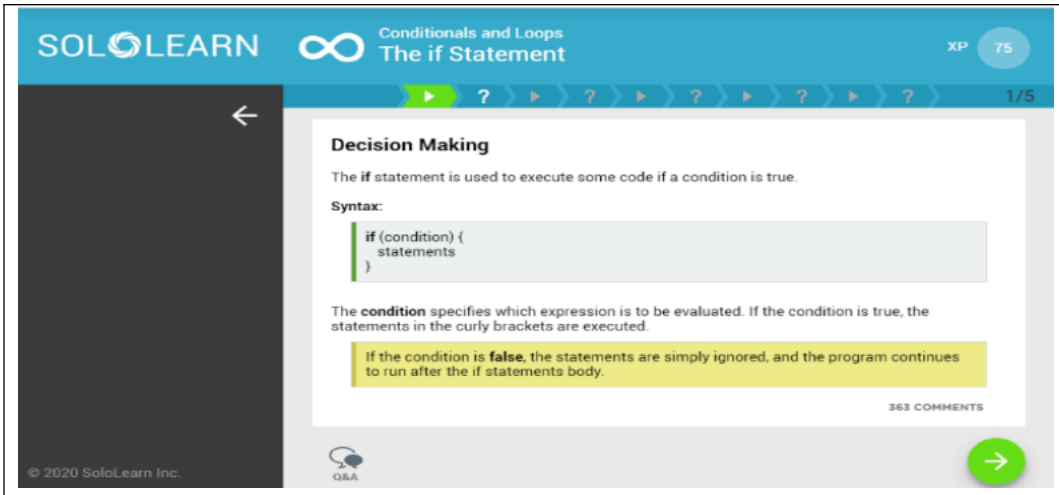
• Decrement Operator: (--) It decreases it by one.

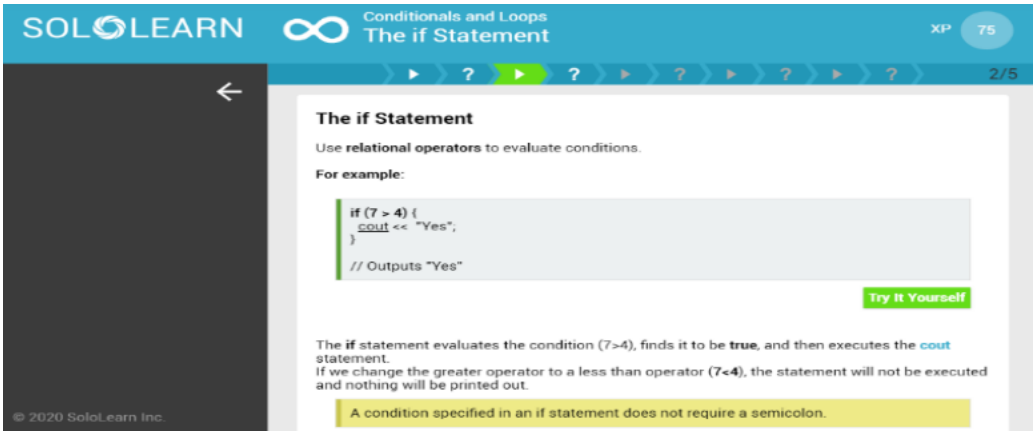


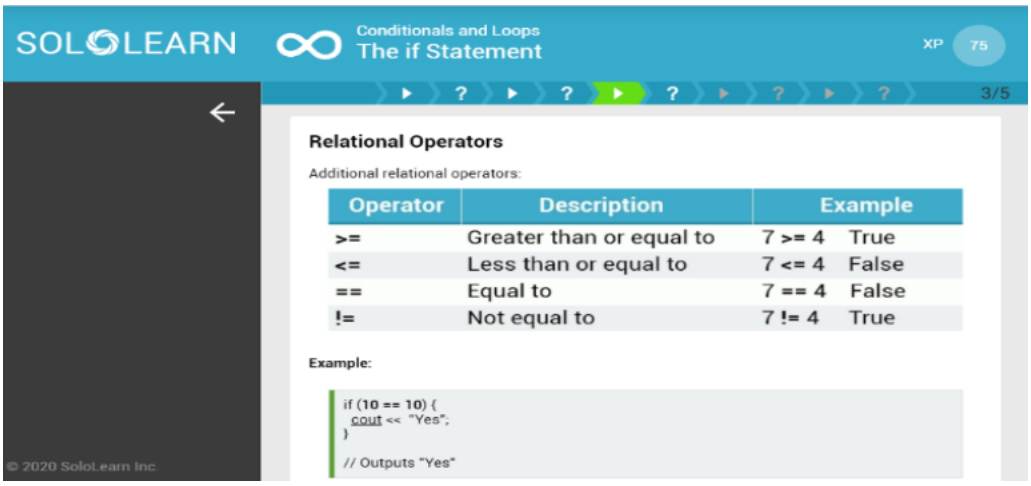


Date:	22-06-2020	Name:	Rajeshwari Gadagi
Course:	C++ Programming	USN:	4AL17EC076
Topic:	Module 2	Semester & Section:	6th SEM & 'B' SEC
Github Repository:	Rajeshwari-gadagi		

AFTERNOON SESSION DETAILS
Image of session


SOLOLEARN
Conditionals and Loops
The if Statement
XP 75
1/5
Decision Making
The if statement is used to execute some code if a condition is true.
Syntax:
if (condition) {
statements
}
The condition specifies which expression is to be evaluated. If the condition is true, the statements in the curly brackets are executed.
If the condition is false, the statements are simply ignored, and the program continues to run after the if statements body.
363 COMMENTS
Q&A
© 2020 SoloLearn Inc.


SOLOLEARN
Conditionals and Loops
The if Statement
XP 75
2/5
The if Statement
Use relational operators to evaluate conditions.
For example:
if (7 > 4) {
cout << "Yes";
}
// Outputs "Yes"
Try It Yourself
The if statement evaluates the condition (7>4), finds it to be true, and then executes the cout statement.
If we change the greater operator to a less than operator (7<4), the statement will not be executed and nothing will be printed out.
A condition specified in an if statement does not require a semicolon.
© 2020 SoloLearn Inc.


SOLOLEARN
Conditionals and Loops
The if Statement
XP 75
3/5
Relational Operators
Additional relational operators:
Operator | Description | Example
>= | Greater than or equal to | 7 >= 4 True
<= | Less than or equal to | 7 <= 4 False
== | Equal to | 7 == 4 False
!= | Not equal to | 7 != 4 True
Example:
if (10 == 10) {
cout << "Yes";
}
// Outputs "Yes"
© 2020 SoloLearn Inc.

C++ programming

Day: Monday, 22/06/2020

Module: Conditionals & Loops

> The if statement

- This statement is used to execute some code if a condition is true.

Syntax:

```
if(condition) { statements; }
```

- Using relational operators to evaluate conditions

- Relational Operators

Operator	Description
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
!=	Not Equal to

- Relational Operators

The not equal operator evaluates the operands determines whether or not they are equal. If the operands are not equal, the condition is evaluated to true

- Relational Operators

You can use relational operators to compare variables in the if statement.

> The else statement:

And if statement can be followed by an optional else statement, which executes when the condition is false.

Syntax:

```
if(condition) { //statements
}
else
{
//statements
}
```

- The else statement

- Nested if statements

If statements within another if statement

- Nested if else statements

C++ provides the option of testing an unlimited number of if/else statements

- The if else statement

The if else statement, a single statement can be executed without enclosing it into curly braces

> Loops

A loop repeatedly executes a set of statements until a specified condition is satisfied.

A while loop statement executes a target statement as long as a given condition remains true

The while loop

> using a while loop.

- using increment or decrement

The increment or decrement operator can be used to change value in the loop.

> The for loop.

The for loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

Syntax: for(init ; condition ; increment){
    statement(s);
}

> The do while loop

do{ statement(s);
} while(condition);

- while vs. do while

If the condition is evaluated to false, the statements in the do loop will run once.

> The switch statement

- Multiple conditions

Sometimes there is need to test a variable for equality against multiple values.

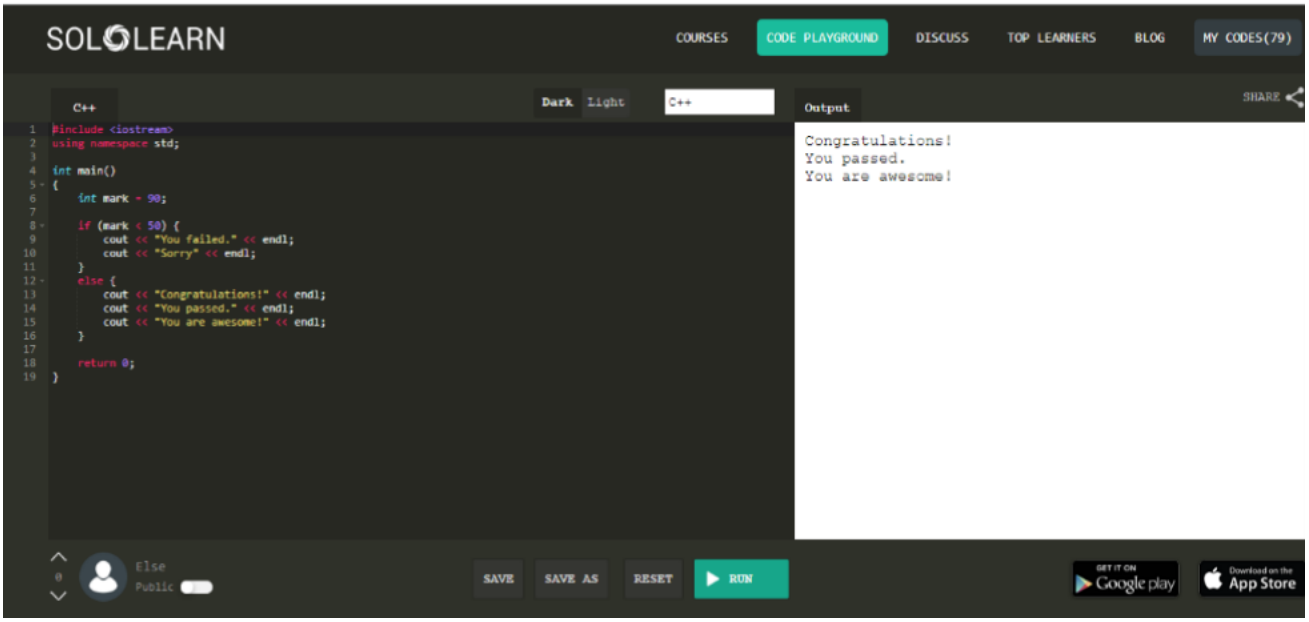
> Logical Operators

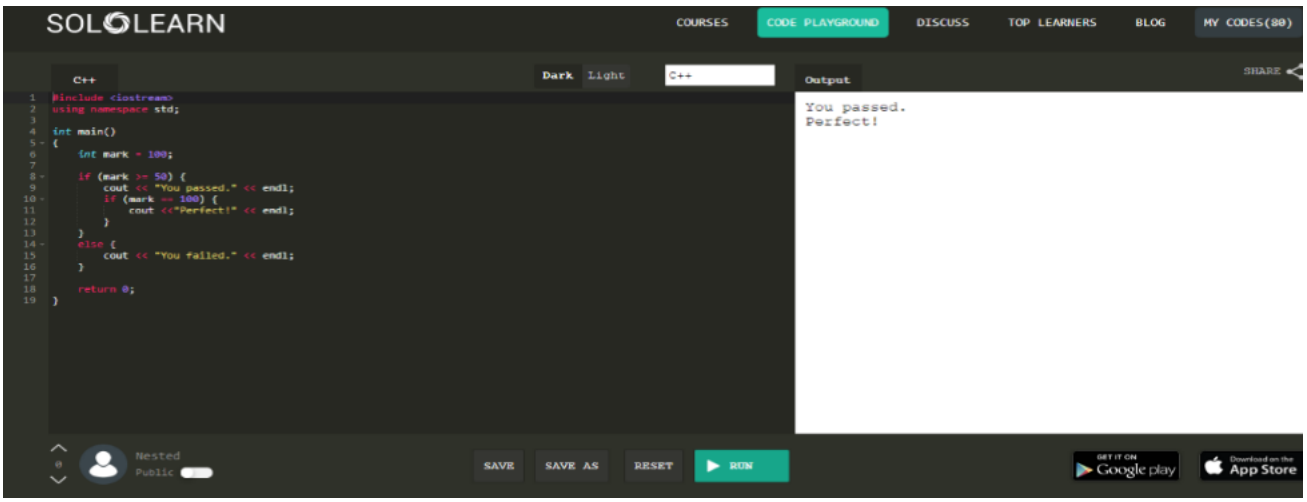
The logical operators to combine conditional statements & return true or false.

Operator	Name of the operator	Form
&&	AND Operator	x && y
||	OR Operator	x || y
!	NOT Operator	!x

Logical NOT

The logical NOT (!) operator is used with just a single operand, returning its logical value. Thus, if a condition is true, the NOT operator makes it false, & vice versa.


SOLOLEARN
COURSES
CODE PLAYGROUND
DISCUSS
TOP LEARNERS
BLOG
MY CODES(79)
C++
Dark
Light
C++
Output
SHARE
#include <iostream>
using namespace std;
int main()
{
int mark = 90;
if (mark < 50) {
cout << "You failed." << endl;
cout << "Sorry" << endl;
}
else {
cout << "Congratulations!" << endl;
cout << "You passed." << endl;
cout << "You are awesome!" << endl;
}
return 0;
}
Congratulations!
You passed.
You are awesome!
Else
Public
SAVE
SAVE AS
RESET
RUN
GET IT ON Google play
Download on the App Store


SOLOLEARN
COURSES
CODE PLAYGROUND
DISCUSS
TOP LEARNERS
BLOG
MY CODES(80)
C++
Dark
Light
C++
Output
SHARE
#include <iostream>
using namespace std;
int main()
{
int mark = 100;
if (mark >= 50) {
cout << "You passed." << endl;
if (mark >= 100) {
cout <<"Perfect!" << endl;
}
}
else {
cout << "You failed." << endl;
}
return 0;
}
You passed.
Perfect!
Nested
Public
SAVE
SAVE AS
RESET
RUN
GET IT ON Google play
Download on the App Store