

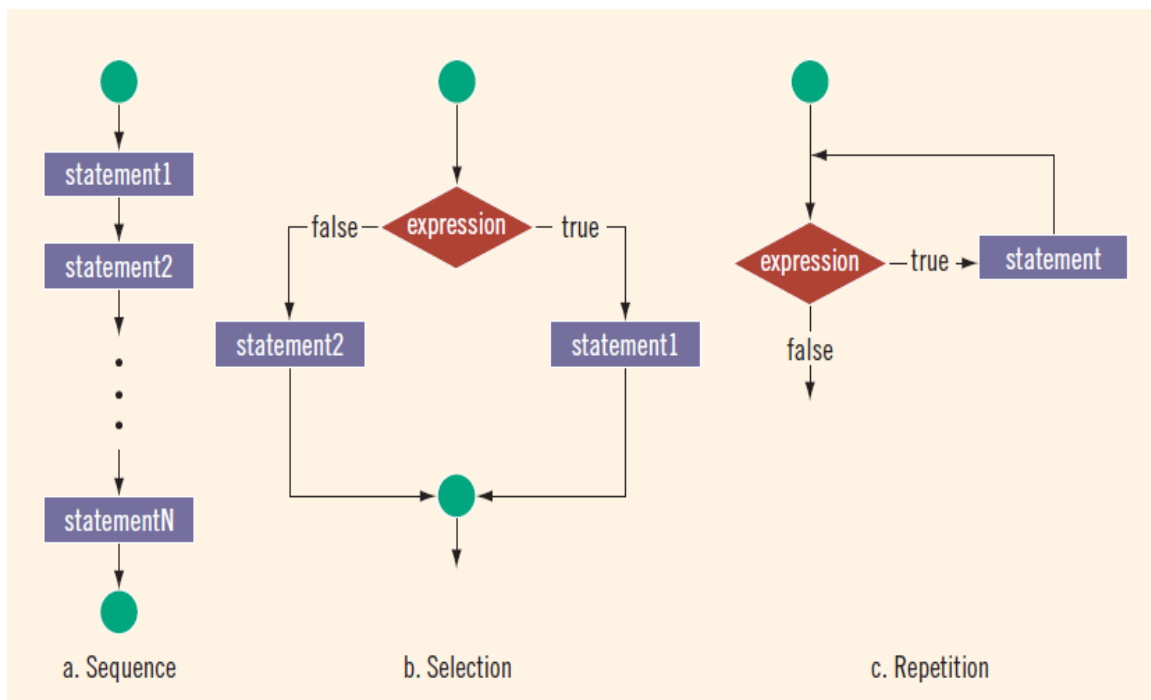


In this lecture you will learn:

- **Control structure.**
- **Relational operator**
- **Logical (Boolean) operator**
- **One-way selection & Two-way selection**
- **Block of statements & Nested if**

Control structure

A computer can process a program in one of the following ways: in sequence; selectively, by making a choice, which is also called a branch; repetitively, by executing a statement over and over, using a structure called a loop; or by calling a function.



- **Relational operator**

C++ includes six relational operators that allow you to state conditions and make comparisons. Table below lists the relational operators.

Operator	Description
==	equal to
!=	not equal to
<	less than
<=	less than or equal to
>	greater than
>=	greater than or equal to

Example 1:

<u>Expression</u>	<u>Meaning</u>	<u>Value</u>
8 < 15	8 is less than 15	true
6 != 6	6 is not equal to 6	false
2.5 > 5.8	2.5 is greater than 5.8	false
5.9 <= 7.5	5.9 is less than or equal to 7.5	true

Logical (Boolean) operator

Logical (Boolean) operators enable you to combine logical expressions. C++ has three logical (Boolean) operators, as shown in Table below:

Operator	Description
!	not
&&	and
	or

Logical operators take only logical values as operands and yield only logical values as results. The operator **!** is unary, so it has only one operand. The operators **&&** and **||** are binary operators.

Order of Precedence

Complex logical expressions can be difficult to evaluate. Consider the following logical expression:

11 > 5 || 6 < 15 && 7 >= 8

This logical expression yields different results, depending on whether **||** or **&&** is evaluated first. If **||** is evaluated first, the expression evaluates to false. If **&&** is evaluated first, the expression evaluates to true. An expression might contain arithmetic, relational, and logical operators, as in the expression:

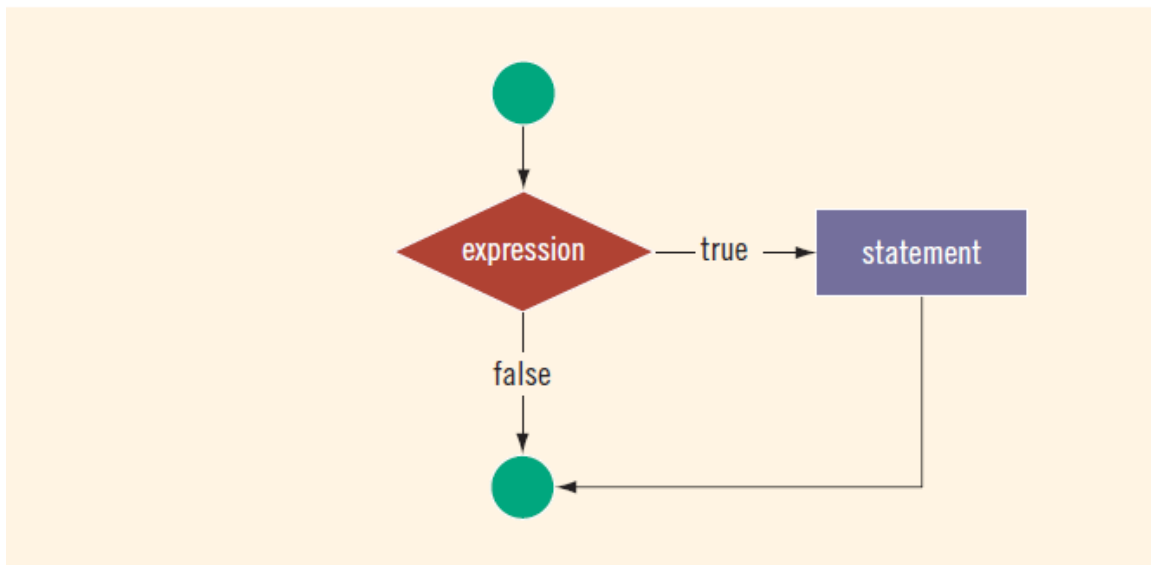
5 + 3 <= 9 && 2 > 3

To work with complex logical expressions, there must be some priority scheme for evaluating operators. Table below shows the order of precedence of some C++ operators:

Operators	Precedence
!, +, - (unary operators)	first
*, /, %	second
+, -	third
<, <=, >=, >	fourth
==, !=	fifth
&&	sixth
	seventh
= (assignment operator)	last

One-Way Selection

Figure below shows the flow of execution of the if statement (one-way selection).



The syntax of one-way selection is:

```
if (expression)  
statement
```

Example 2:

// demonstrates IF statement

#include <iostream>

using namespace std;

int main()

{

int x;

cout << "Enter a number: ";

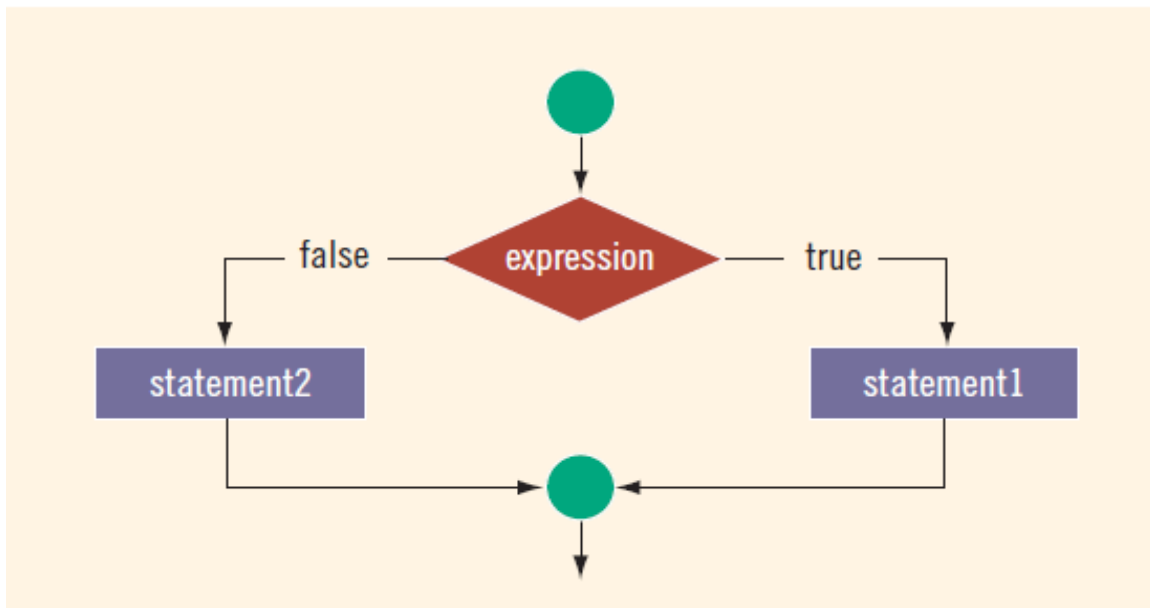
cin >> x;

if(x > 100)

```
cout << "That number is greater than 100\n";  
return 0;  
}
```

Two-Way Selection

Figure below shows the flow of execution of the if. .else statement (two-way selection).



Two-way selection uses the following syntax:

```
if (expression)  
statement1  
else  
statement2
```

Example 3:

```
// demonstrates IF...ELSE statement  
  
#include <iostream>
```

```
using namespace std;

int main()

{

int x;

cout << "\nEnter a number: ";

cin >> x;

if( x > 100 )

cout << "That number is greater than 100\n";

else

cout << "That number is not greater than 100\n";

return 0;

}
```

Block of statements

Example 4:

```
// demonstrates IF with multiline body

#include <iostream>

using namespace std;

int main()

{
```

```
int x;

cout << "Enter a number: ";

cin >> x;

if( x > 100 )

{

    cout << "The number " << x;

    cout << " is greater than 100\n";

}

return 0;

}
```

Nested if

Example 5:

```
#include <iostream>

using namespace std;

int main()

{

    int x, y, z;

    cout<<" enter the three numbers :\n";

    cin >> x >> y >> z;
```

```
if (x < y && x < z)

    cout << x << "\n" y << "\n" << z << endl;

else

    cout << x << "\n" << z << "\n" << y << endl;

else if (y < x && y < z)

    if (x < z)

        cout << y << "\n" << x << "\n" << z << endl;

    else

        cout << y << "\n" << z << "\n" << x << endl;

else if (z < y && z < x)

    if (x < y)

        cout << z << "\n" << x << "\n" << y << endl;

    else

        cout << z << "\n" << y << "\n" << x << endl;

return 0;

}
```


Programming Exercises

- Write a program that prompts the user to input a number. The program should then output the number and a message saying whether the number is positive, negative, or zero.
- Write a program that prompts the user to input three numbers. The program should then output the numbers in ascending order.
- In a right triangle, the square of the length of one side is equal to the sum of the squares of the lengths of the other two sides. Write a program that prompts the user to enter the lengths of three sides of a triangle and then outputs a message indicating whether the triangle is a right triangle.
- Write a program that mimics a calculator. The program should take as input two integers and the operation to be performed. It should then output the numbers, the operator, and the result. (For division, if the denominator is zero, output an appropriate message.) Some sample outputs follow:

$$3 + 4 = 7$$

$$13 * 5 = 65$$

- (Calculating a Circle's Diameter, Circumference and Area) Write a program that reads the radius of a circle (as a double value) and computes and prints the diameter, the circumference and the area. Use the value 3.14159 for π .
- (Odd or Even) Write a program that reads an integer and determines and prints whether it's odd or even. [Hint: Use the modulus operator. An even number is a multiple of two. Any multiple of two leaves a remainder of zero when divided by 2.]

References:

- [1] Deitel, P. & Deitel, R., "C++ How to Program", Eighth Edition, Pearson Education Inc., 2012.
- [2] Stefan B., " C++ Primer Plus ", Sixth Edition, Pearson Education Inc., 2012.