

C++ Programming Language

Presented By:

T.A. Asmaa Hamad El-saied

E-mail: eng.asmaa134@gmail.com

Contents



Operators



Decision Making

Operators

❖ Arithmetic Operators

- Assume variable A holds 10 and variable B holds 20, then: [Show Examples](#)

Operator	Description	Example
+	Adds two operands	A + B will give 30
-	Subtracts second operand from the first	A - B will give -10
*	Multiplies both operands	A * B will give 200
/	Divides numerator by de-numerator	B / A will give 2
%	Modulus Operator and remainder of after an integer division	B % A will give 0
++	Increment operator , increases integer value by one	A++ will give 11
--	Decrement operator , decreases integer value by one	A-- will give 9

Operators(cont.)

❖ Relational Operators(Comparison operators)

- Assume variable A holds 10 and variable B holds 20, then: [Show Examples](#)

Operator	Description	Example
==	Checks if the values of two operands are equal or not, if yes then condition becomes true.	(A == B) is not true.
!=	Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(A >= B) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(A <= B) is true.

Operators(cont.)

❖ Logical Operators(Boolean operators)

- Assume variable A holds 1 and variable B holds 0, then:

Show Examples

Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non-zero, then condition becomes true.	(A && B) is false.
	Called Logical OR Operator. If any of the two operands is non-zero, then condition becomes true.	(A B) is true.
!	Called Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true, then Logical NOT operator will make false.	!(A && B) is true.

Operators(cont.)

❖ Assignment Operators

■ Show Examples

Operator	Description	Example
=	Simple assignment operator, Assigns values from right side operands to left side operand	$C = A + B$ will assign value of $A + B$ into C
+=	Add AND assignment operator, It adds right operand to the left operand and assign the result to left operand	$C += A$ is equivalent to $C = C + A$
-=	Subtract AND assignment operator, It subtracts right operand from the left operand and assign the result to left operand	$C -= A$ is equivalent to $C = C - A$
*=	Multiply AND assignment operator, It multiplies right operand with the left operand and assign the result to left operand	$C *= A$ is equivalent to $C = C * A$
/=	Divide AND assignment operator, It divides left operand with the right operand and assign the result to left operand	$C /= A$ is equivalent to $C = C / A$
%=	Modulus AND assignment operator, It takes modulus using two operands and assign the result to left operand	$C \% = A$ is equivalent to $C = C \% A$

Operators(cont.)

❖ Additional Operators

Operator	Description
sizeof	sizeof operator returns the size of a variable. For example, sizeof(a), where a is integer, will return 4.
Condition ? X : Y	Conditional operator . If Condition is true ? then it returns value X : otherwise value Y
,	Comma operator causes a sequence of operations to be performed. The value of the entire comma expression is the value of the last expression of the comma-separated list.
. (dot) and -> (arrow)	Member operators are used to reference individual members of classes, structures, and unions.
Cast	Casting operators convert one data type to another. For example, int(2.2000) would return 2.
&	Pointer operator & returns the address of an variable. For example &a; will give actual address of the variable.
*	Pointer operator * is pointer to a variable. For example *var; will pointer to a variable var.

Operators(cont.)

❖ Additional Operators

- Conditional ? : Operator

- `Exp1 ? Exp2 : Exp3;`

```
#include <iostream>
```

```
using namespace std;
```

```
int main ()
```

```
{ // Local variable declaration:
```

```
    int x, y = 10;
```

```
    x = (y < 10) ? 30 : 40;
```

```
    cout << "value of x: " << x << endl;
```

```
    return 0; }
```

- When the above code is compiled and executed, it produces the following result: **value of x: 40**

Operators(cont.)

❖ Additional Operators

■ Casting Operators

- (type) expression

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
```

```
{   double a = 21.09399;
```

```
    float b = 10.20; int c ;
```

```
    c = (int) a; cout << "Line 1 - Value of (int)a is :" << c << endl;
```

```
    c = (int) b; cout << "Line 2 - Value of (int)b is :" << c << endl ;
```

```
    return 0; }
```

When the above code is compiled and executed, it produces the following result:

Line 1 - Value of (int)a is :21 Line 2 - Value of (int)b is :10

Common Escape Sequences

❖ **Escape sequences** give you the ability to exercise greater control over the way information is output by your program

Escape Sequence	Name	Description
<code>\n</code>	Newline Causes	the cursor to go to the next line for subsequent printing.
<code>\t</code>	Horizontal tab	Causes the cursor to skip over to the next tab stop.
<code>\a</code>	Alarm	Causes the computer to beep.
<code>\b</code>	Backspace	Causes the cursor to back up, or move left one position.
<code>\r</code>	Return	Causes the cursor to go to the beginning of the current line, not the next line.
<code>\\</code>	Backslash	Causes a backslash to be printed.
<code>\'</code>	Single quote	Causes a single quotation mark to be printed.
<code>\"</code>	Double quote	Causes a double quotation mark to be printed.

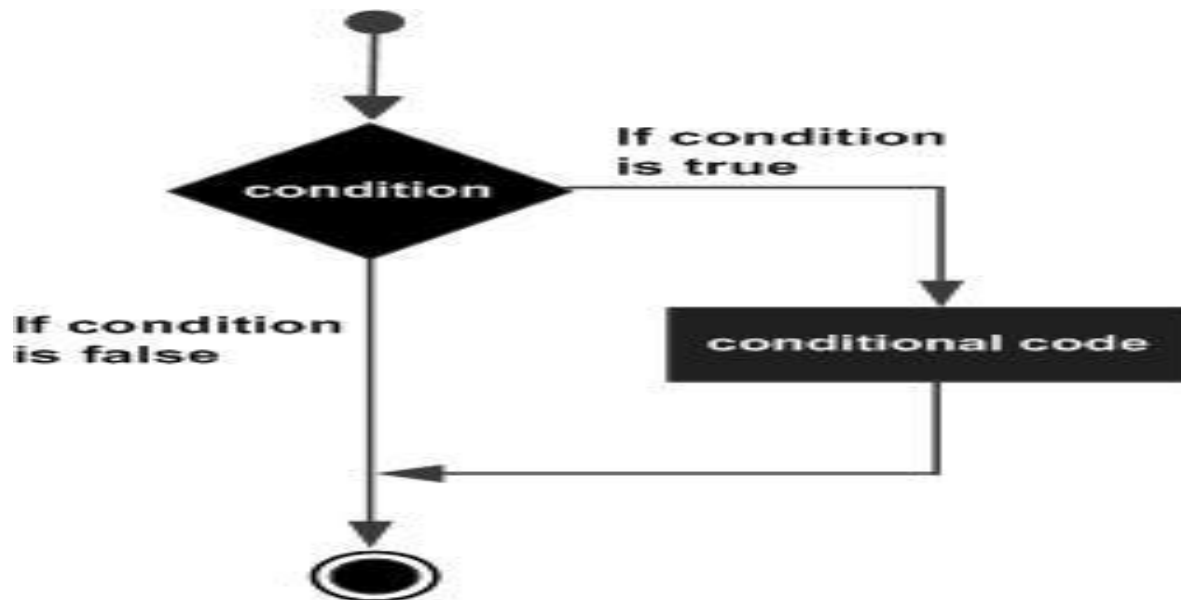
Decision Making

❖ if statement

■ Syntax:

- `if(condition)`

- `{ // statement(s) will execute if the condition is true }`



Decision Making(cont.)

❖ **if statement** :Example

```
#include <iostream>
using namespace std;
int main ()
{
    int a = 10;
    if( a < 20 ) // if condition is true then print the following
    { cout << "a is less than 20;" << endl; }
    cout << "value of a is : " << a << endl;
    return 0;
}
```

Output: a is less than 20; value of a is : 10

Decision Making(cont.)

❖ if...else statement

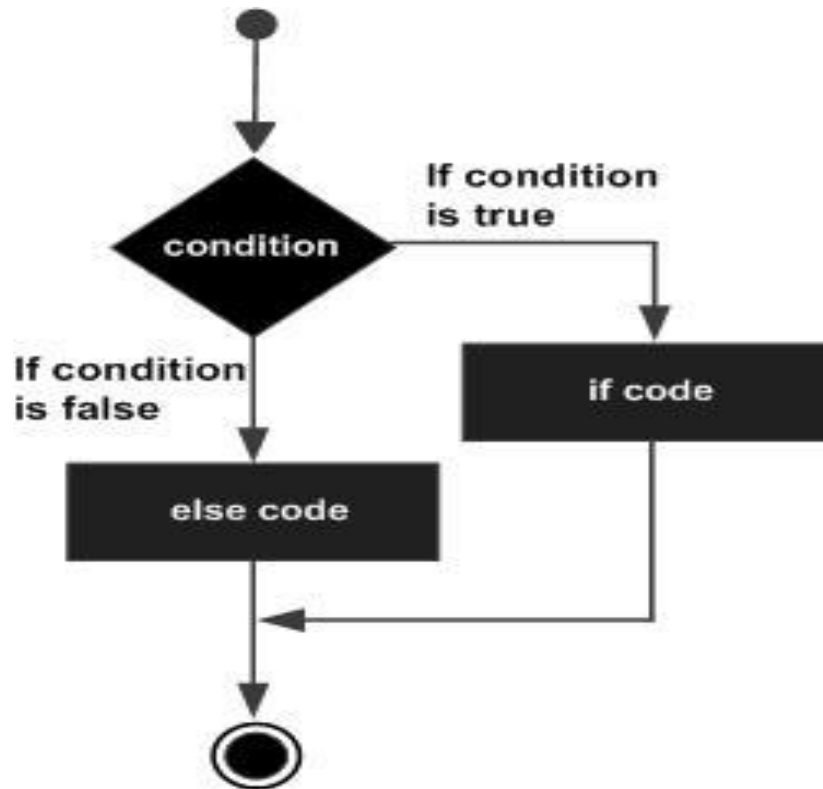
- Syntax:

if (condition)

```
{  
    S1;  
}
```

else

```
{  
    S2;  
}
```



Decision Making(cont.)

❖ **if...else statement:** Example

```
#include <iostream>
using namespace std;
int main ()
{   int a = 100;
    if( a < 20 ) // if condition is true then print the following
    {
        cout << "a is less than 20;" << endl; }
    else // if condition is false then print the following
    {
        cout << "a is not less than 20;" << endl;}
    cout << "value of a is : " << a << endl; return 0; }
```

Output: a is not less than 20; value of a is : 100

Decision Making(cont.)

❖ **if...else if...else Statement**

■ Syntax:

```
if(Condition 1)
{ // Executes when the Condition 1 is true }
else if(Condition 2)
{ // Executes when the Condition 2 is true }
else if(Condition 3)
{ // Executes when the Condition 3 is true }
else { // executes when the none of the above
Condition is true. }
```

Decision Making(cont.)

❖ if...else if...else Statement:Example

```
#include <iostream>
using namespace std;
int main ()
{   int a = 100;
    if( a == 10 ) // if condition is true then print the following
    {cout << "Value of a is 10" << endl; }
    else if( a == 20 ) // if else if condition is true
    {cout << "Value of a is 20" << endl; }
    else if( a == 30 ) // if else if condition is true
    {cout << "Value of a is 30" << endl; }
    else // if none of the conditions is true
    {cout << "Value of a is not matching" << endl; }
    cout << "Exact value of a is : " << a << endl; return 0; }
```

Output: Value of a is not matching

Exact value of a is : 100

Decision Making(cont.)

❖ switch statement

■ Syntax:

```
switch(expression)
{
    case constant-expression : statement(s);
        break; //optional
    case constant-expression : statement(s);
        break; //optional
    // you can have any number of case statements.
    default : //Optional
        statement(s);
}
```

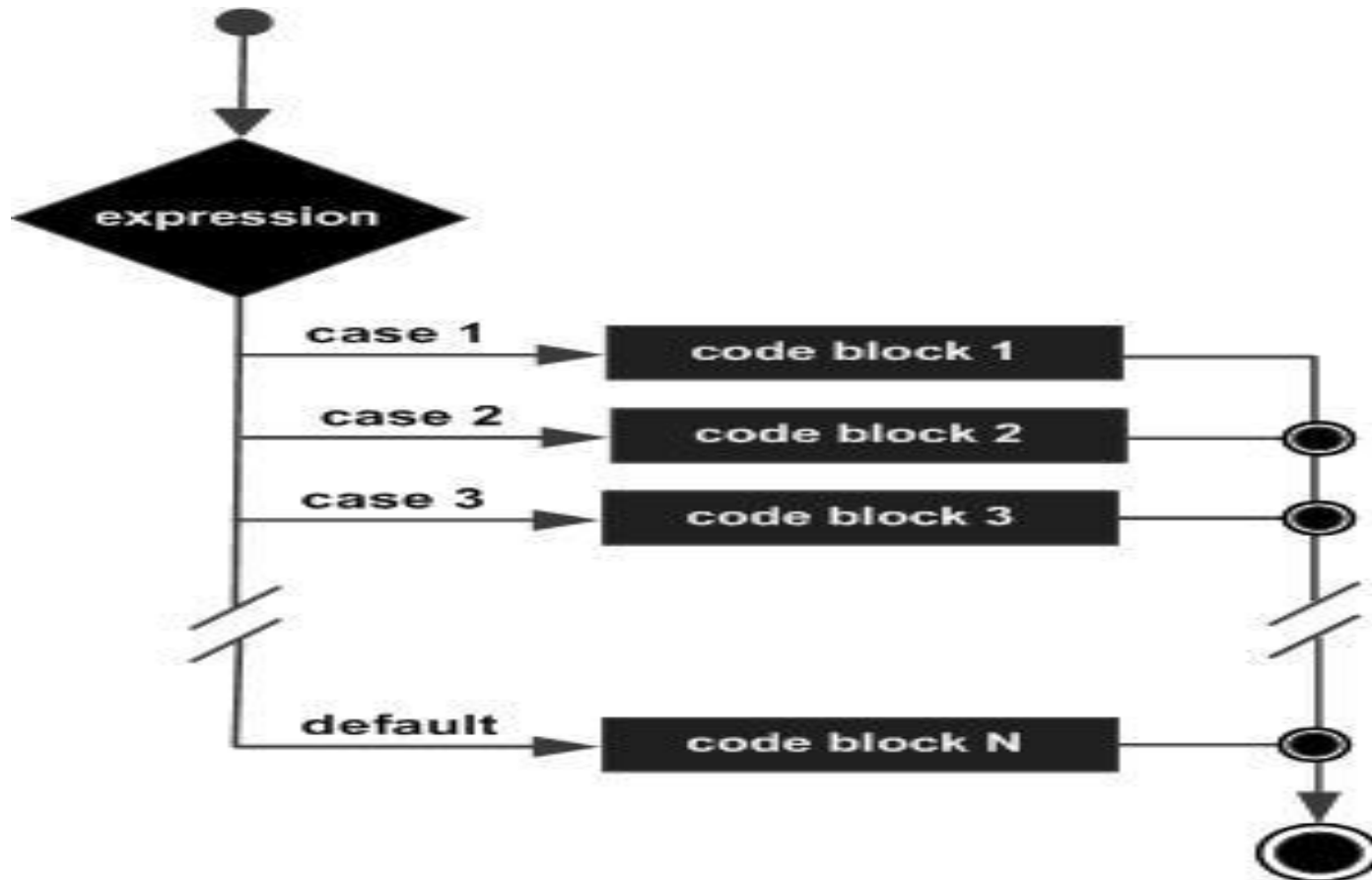
Decision Making(cont.)

❖ switch statement

- The expression used in a switch statement must have an integral or enumerated type, or be of a class type in which the class has a single conversion function to an integral or enumerated type.
- You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.
- The constant-expression for a case must be the same data type as the variable in the switch, and it must be a constant or a literal.
- When the variable being switched on is equal to a case, the statements following that case will execute until a break statement is reached.
- When a break statement is reached, the switch terminates, and the flow of control jumps to the next line following the switch statement.
- Not every case needs to contain a break. If **no break appears**, the flow of control will *fall through* to subsequent cases until a break is reached.
- A switch statement can have an optional default case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases is true. No break is needed in the default case.

Decision Making(cont.)

❖ switch statement



Decision Making(cont.)

❖ switch statement: Example

```
#include <iostream>
using namespace std;
int main ()
{   char grade = 'D';
    switch(grade)
    {   case 'A' : cout << "Excellent!" << endl;
        break;
        case 'B' : case 'C' : cout << "Well done" << endl;
        break;
        case 'D' : cout << "You passed" << endl;
        break;
        case 'F' : cout << "Better try again" << endl;
        break;
        default : cout << "Invalid grade" << endl; }
    cout << "Your grade is " << grade << endl; return 0; }
```

Decision Making(cont.)

❖ **switch statement:** Example

Output:

You passed

Your grade is D

Decision Making(cont.)

❖ nested if statements

- Syntax:

```
if( Condition1)
```

```
{
```

```
// Executes when the Condition1 is true
```

```
    if(Condition 2)
```

```
    {
```

```
        // Executes when the Condition2 is true
```

```
    }
```

```
}
```

Decision Making(cont.)

❖ nested if statements Example:

```
#include <iostream>
using namespace std;
int main ()
{ int a = 100, b = 200;
  if( a == 100 )
  {
    if( b == 200 )
    { cout << "Value of a is 100 and b is 200" << endl; }
  }
  cout << "Exact value of a is : " << a << endl;
  cout << "Exact value of b is : " << b << endl;
  return 0; }
```

Decision Making(cont.)

nested if statements Example:

Output

Value of a is 100 and b is 200

Exact value of a is : 100

Exact value of b is : 200

Decision Making(cont.)

❖ nested switch statements

- Syntax:

```
switch(ch1)
{
    case 'A':
        cout << "This A is part of outer switch";
        switch(ch2)
        {
            case 'A': cout << "This A is part of inner switch"; break; case
            'B': // ...
        }
        break;
    case 'B': // ...
}
```

Decision Making(cont.)

❖ nested switch statements Example:

```
#include <iostream>
using namespace std;
int main ()
{
    int a = 100; int b = 200;
    switch(a)
    { case 100:
        cout << "This is part of outer switch" << endl;
        switch(b)
        {
            case 200: cout << "This is part of inner switch" << endl;
        }
    }
    cout << "Exact value of a is : " << a << endl;
    cout << "Exact value of b is : " << b << endl; return 0; }
```

Decision Making(cont.)

❖ **nested switch statements** Example:

Output:

This is part of outer switch

This is part of inner switch

Exact value of a is : 100

Exact value of b is : 200

some Examples

Examples

❖ Example 1

- Write a program which input three numbers and display the largest number using ternary operator.

❖ Example 2

- What is the output of following program?

```
int result = 4 + 5 * 6 + 2;  
cout<<result;
```

```
int a = 5 + 7 % 2;  
cout<<a;
```

Examples

❖ Example 3

- What is the output of following program?

<pre>int x = 10,y; y = x++; cout<<y;</pre>	<pre>int x = 10,y; y = x++; cout<<x;</pre>	<pre>int x = 10; x++; cout<<x;</pre>
<pre>int x = 10,y; y = ++x; cout<<y;</pre>	<pre>int x = 10; cout<<++x;</pre>	<pre>int x = 10; cout<<x++;</pre>

Examples

❖ Example 4

- What is the output of following program?

<pre>int x = 10; cout<<x<<x++;</pre>	<pre>int x = 10; cout<<++x<<x++<<x;</pre>	<pre>int x = 10; cout<<x++<<x<<++x;</pre>
<pre>int x = 10,y; y = x + x++; cout<<y;</pre>	<pre>int x = 10,y; y = ++x + x++ + x; cout<<y;</pre>	<pre>int x = 10,y; y = x++ + x + ++x; cout<<y;</pre>



Thanks