**Lesson: №8.**

**Module: .NET/C# Syntax.**

**Lesson Content:**

* *Boolean operators.*
  + *Ternary operator.*
  + *If – operator.*
  + *Switch – operator.*
  + *Switch – expression.*

**Theme: Boolean operators.**

C# supports the usual logical conditions from mathematics:

* Less than: a < b
* Less than or equal to: a <= b
* Greater than: a > b
* Greater than or equal to: a >= b
* Equal to a == b
* Not Equal to: a != b

Logical operators also supports by C#, and can be used to determine the logic between variables or values:

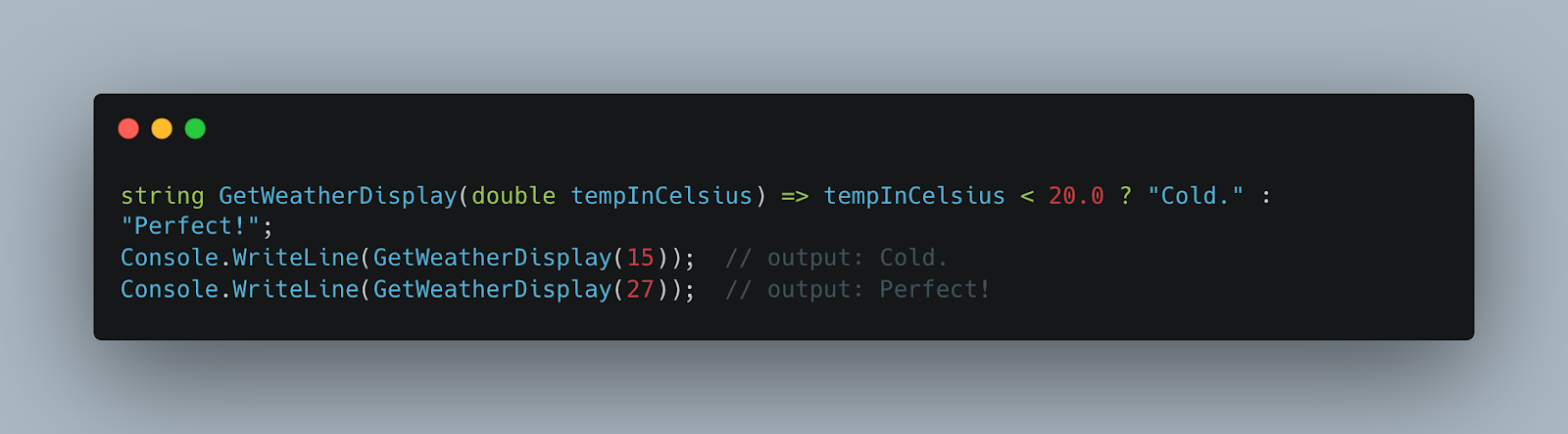
* Logical Conditional *and* && - Returns True if both statements are true.
* Logical Conditional *or* || - Returns True if one of the statements is true.
* Logical Conditional *not* ! - Reverse the result, returns False if the result is true.
* Logical *and* & - Returns True if both statements are true.
* Logical or | - Returns True if one of the statements is true.
* Logical exclusive *or* ^ - The result of x ^ y is true if x evaluates to true and y evaluates to false, or x evaluates to false and y evaluates to true. Otherwise, the result is false.

You can use these conditions to perform different actions for different decisions.

C# has the following conditional statements:

* Use if to specify a block of code to be executed, if a specified condition is true
* Use else to specify a block of code to be executed, if the same condition is false
* Use else if to specify a new condition to test, if the first condition is false
* Use switch to specify many alternative blocks of code to be executed

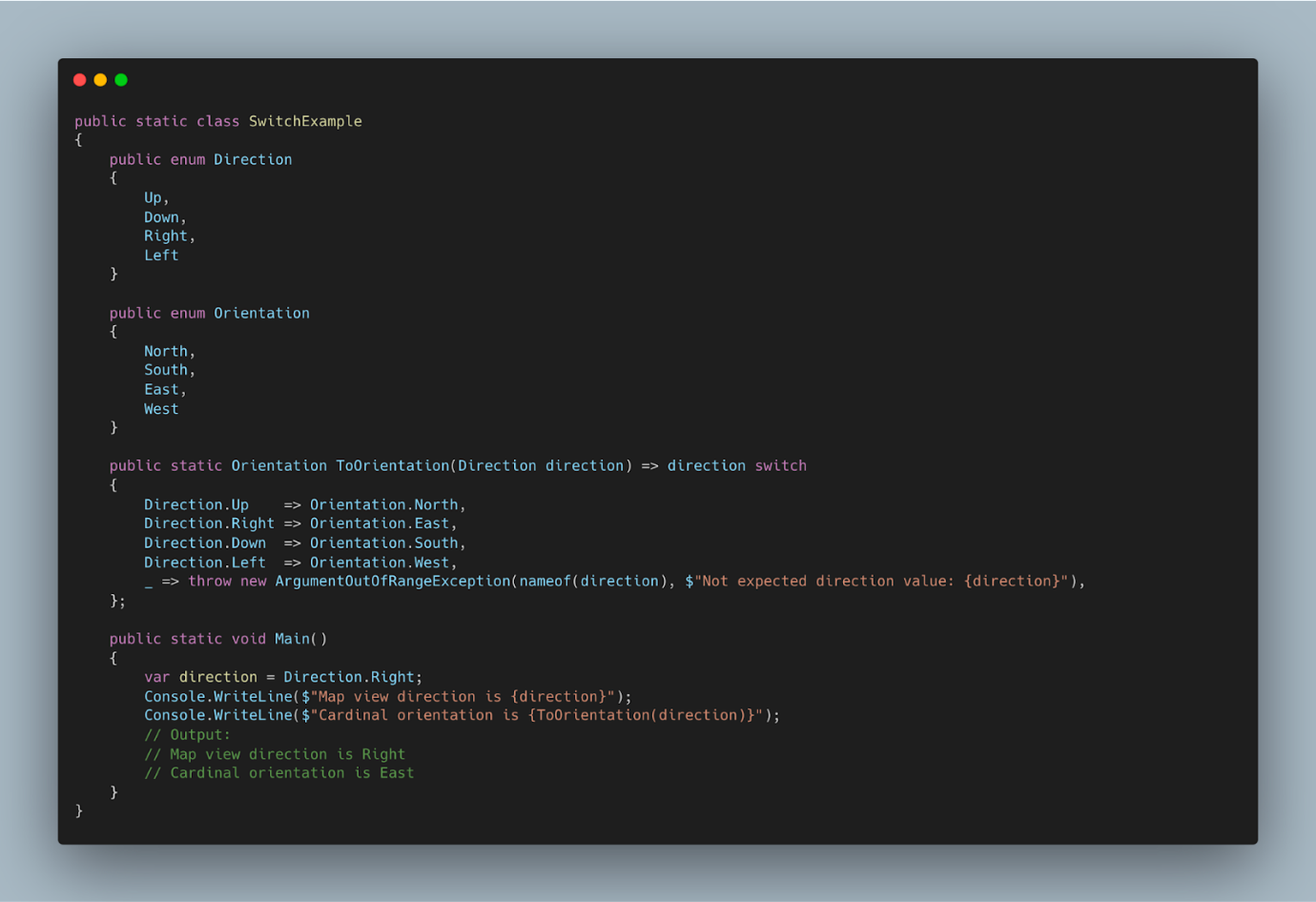
The conditional operator ?:, also known as the ternary conditional operator, evaluates a Boolean expression and returns the result of one of the two expressions, depending on whether the Boolean expression evaluates to true or false, as the following example shows:

****

**Sub-Theme: Switch Expression.**

You use the switch expression to evaluate a single expression from a list of candidate expressions based on a pattern match with an input expression. For information about the switch statement that supports switch-like semantics in a statement context, see the switch statement section of the Selection statements article.

The following example demonstrates a switch expression, which converts values of an enum representing visual directions in an online map to the corresponding cardinal directions:



The preceding example shows the basic elements of a switch expression:

* An expression followed by the switch keyword. In the preceding example, it's the direction method parameter.
* The *switch expression arms*, separated by commas. Each switch expression arm contains a *pattern*, an optional *case guard*, the => token, and an *expression*.

At the preceding example, a switch expression uses the following patterns:

* A constant pattern: to handle the defined values of the Direction enumeration.
* A discard pattern: to handle any integer value that doesn't have the corresponding member of the Direction enumeration (for example, (Direction)10). That makes the switch expression exhaustive.

**Homework:**

1. Explain the logical conditional operators AND (&&) and OR (||) in C# and when they return true.
2. What is the difference between the logical AND (&) and OR (|) operators and their conditional counterparts (&& and ||)?
3. What is the role of the "else" operator and when is it executed in relation to the "if" operator?
4. What is the purpose of the "switch" operator in C#, and how does it differ from the "if-else" structure?
5. Describe the components of a switch expression, including the expression, shoulders of a switch expression, and patterns.
6. Give an example of a switch expression in C# that demonstrates how it converts values based on pattern matching.