Conditional Statements

if/else, switch/case, (and a bonus conditional expression: ternary)

Learning objectives

- The if statement (if, else, else if)
- The switch statement (switch, case, break, default)
- The ternary operator
- Operators in conditions (relational and logical)

The if statement

- The if statement specifies a block of code to be executed if a condition is true

- The condition is a statement that is always evaluated to true or false

```
if (condition) {
    System.out.println("The condition is true");
}
```

The else statement

- The else statement specifies a block of code to be executed if the condition in the if statement is false

- The else statement must follow after the if statement

```
if (condition) {
    System.out.println("The condition is true");
}
else {
    System.out.println("The condition is false");
}
```

The else if statement

- The else if statement specifies a new condition that will be evaluated if the first condition is false

- The else if statement must follow after the if statement

```
if (condition1) {
    System.out.println("condition1 is true");
}
else if (condition2) {
    System.out.println("condition1 is false and condition2 is true");
}
else {
    System.out.println("both conditions are false");
}
```

Demo 1 - if/else/else if

- The if statement
- The if else statement
- The else statement

The switch statement

- The switch statement evaluates the value of an expression
- Then it selects a code block to be executed depending on if the value of the expression matches the value of the case statement

```
switch (expression) {
    case x:
        System.out.println("The value of expression is x");
        break;
    case y:
        System.out.println("The value of expression is y");
}
```

The break keyword

- The break keyword breaks out of the switch statement
- Without the break keyword the execution would continue with the next case code block, a so called fall-through in a switch statement

```
switch (expression) {
    case x:
        System.out.println("The value of expression is x");
        break;
    case y:
        System.out.println("The value of expression is y");
}
```

The default keyword

- The default keyword specifies code to be executed if there is no case match
- It is similar to the else statement in an if-else statement

```
switch (expression) {
    case "x":
        System.out.println("The value of expression is x");
        break;
    case "y":
        System.out.println("The value of expression is y");
        break;
    default:
        System.out.println("The value of expression is neither x or y");
}
```

Demo 2 - switch statement

- The switch statement
- The break keyword
- The default keyword

The ternary operator

- The ternary operator is like a short-hand if/else

- It consists of three operands. One condition that will be evaluated to be true or false, then an expression that will be executed if the condition is true and then an expression that will be executed if the condition is false

```
variable = (condition) ? expressionIfTrue : expressionIfFalse;
```

Demo 3 - ternary operator

- The ternary operator
- Comparing to using if/else

Relational operators

- The condition in if/else or ternary statements often uses relational operators

Operator	Meaning	Example int a = 10; int b = 5;
==	Equal to	a == b is false
!=	Not equal to	a != b is true
>	Greater than	a > b is true
<	Less than	a < b is false
>=	Greater than or equal to	a >= b is true
<=	Less than or equal to	a <= b is false

Logical operators

- More complex conditions can be created by combining several relational statements with logical operators

- Short circuit logic - statements will only be evaluated as long as it matters

Operator	Meaning	Example boolean a = true; boolean b = false;
&&	AND	a && b is false
||	OR	a || b is true
!	NOT	!(a && b) is true

Demo 4 - Operators in conditions

- Relational operators
- Logical operators

Exercise 1 - Create a Calculator

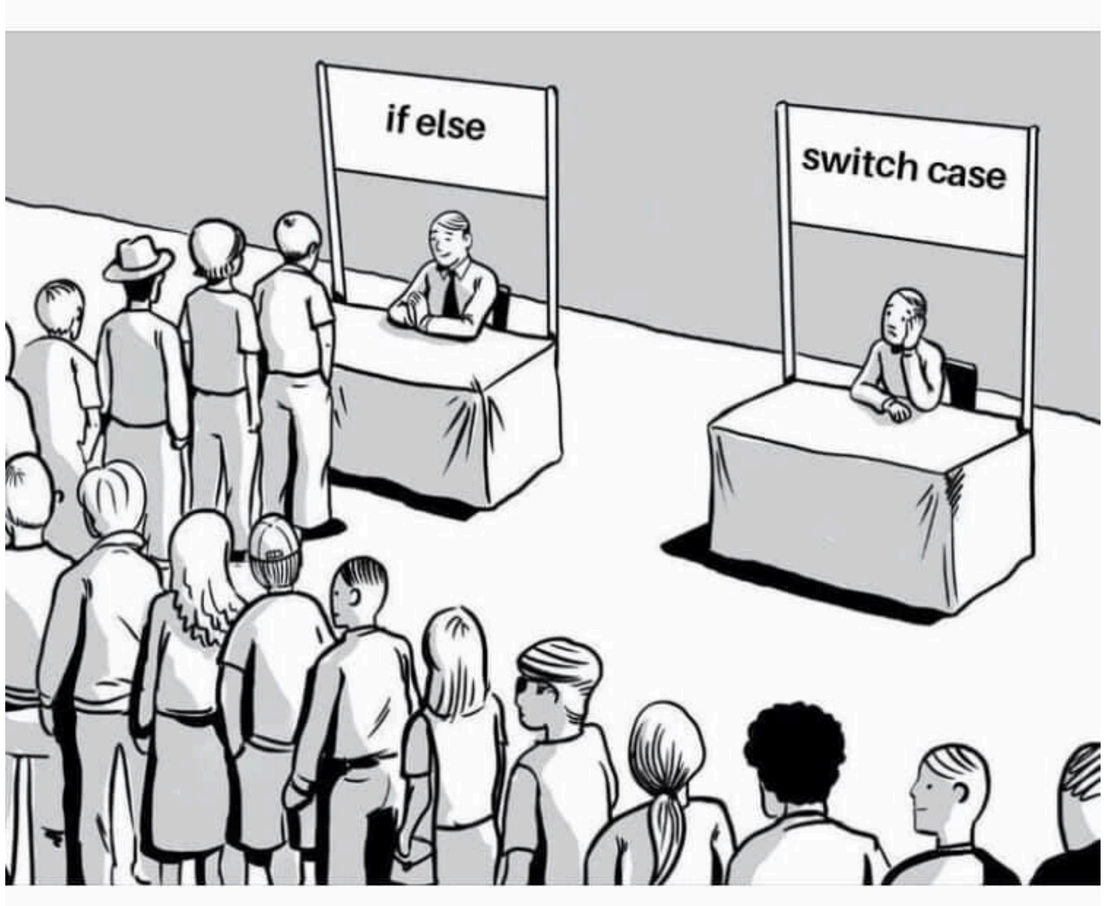
- The program should ask the user to enter two numbers and an operator
- The operator should be any of these: + or - or * or /
- Then print the result of using the operator on the two numbers
- Create one version using if/else and another version using switch/case
- Hint: read the operator as a String and use the equals method on the String to compare to different values. For example: if (operator.equals(“+”))

Exercise 2 - Ticket price calculator

- Create an int variable for age and a boolean variable for being a member or not
- Then use a if/else statement with relational and logical operators to print out the correct price, according to the rules, and try out different values in the age and member variables:
- 1. Tickets are free for children younger than 5 years
- 2. Tickets cost 6 euros for children that are between 5 and 10 years old
- 3. Tickets cost 10 euros for children that are between 11 and 18 years old
- 4. Tickets cost 14 euros for adults (from 18 years old)
- 5. Members get a discount of 2 euros
- Why should you use the if/else statement and not the switch statement for this exercise?

Summary

Do you have a favorite?



ACTLEA
Active Learning

Learning objectives

- The if statement (if, else, else if)
- The switch statement (switch, case, break, default)
- The ternary operator
- Operators in conditions (relational and logical)