Exercises module 6 – Switch statement

6.1 – Integer to color

1. We need to write a program that takes an integer input and prints out one of the following color based on this table:

Input	**Output**
1	Black
2	White
3	Red
4	Green

Use a switch statement and if the input value is not valid, then print out “Invalid color”.

2. Modify the code from the previous step so that it uses a char instead of an integer according to this table:

Input	**Output**
'a'	Black
'b'	White
'c'	Red
'd'	Green

Use a switch statement and if the input value is not valid, then print out “Invalid color”.

6.2- Java switch case fall-through

1. What will the output be from this code when **input='b'**?

```
char input='b';

switch (input)
{
    case 'a':
        System.out.println("Airplane");
        break;
    case 'b':
        System.out.println("Car");
    case 'c':
        System.out.println("Boat");
        break;
}
```

   Try it and understand why, this behaviour is called **fall-through**.

2. Modify the color example from previous step so that its output is based on this table instead:

Input	Output
'a'	Black
'b' 'w' 'W' 'v'	White
'c'	Red
'd'	Green

   Use a switch statement and if the input value is not valid, then print out “Invalid color”.

   You should only use five break statements;

3. Remove the default: part from the switch statement in the previous step.

   What happens when you pass ‘z’ as input to the switch statement? What was the output?

6.3 – Converting If-statements to switch

We are given this program and your assignment is to replace the if-statements with a single switch-statement.

```
Scanner stdin = new Scanner(System.in);
int choice = 0;

System.out.println("Please enter your choice (1-4): ");
choice = stdin.nextInt();

if(choice == 1)
{
    System.out.println("You selected 1.");
}
else if(choice == 2 || choice == 3)
{
    System.out.println("You selected 2 or 3.");
}
else if(choice == 4)
{
    System.out.println("You selected 4.");
}
else
{
    System.out.println("Please enter a choice between 1-4.");
}
```

6.4 – Simple calculator

Your assignment is to write a simple calculator that can be used like this:

```
Please enter the first number:
4
Please enter the second number:
6
Please enter operator + - * /:
+
Result = 10
```

The rules are:

- Should print “Invalid operator” if the operator is not valid
- Should handle Addition, Subtraction, multiplication and Division
- Write a method for each operator that does the actual calculation
- Use a Switch statement and no IF-statements

6.5 – Extract method

We are given this code by our client:

```
char input = 'c'; //try with a, b, c here

switch (input) {
    case 'a':
        System.out.println("You pressed 'a'");

        break;
    case 'b':
        System.out.println("You pressed 'b' and here's 10 numbers");
        for (int i = 0; i < 10; i++) {
            System.out.println(i);
        }
        break;

    case 'c':
        System.out.println("You pressed 'c'");
        System.out.println("The sum of the 10 first numbers are ");
        int sum = 0;
        for (int i = 0; i < 10; i++) {
            sum = sum + i;
        }
        System.out.println("Sum = " + sum);
        break;
}
```

A best practice regarding switch-statements is to keep the code in the case blocks small and compact, something like:

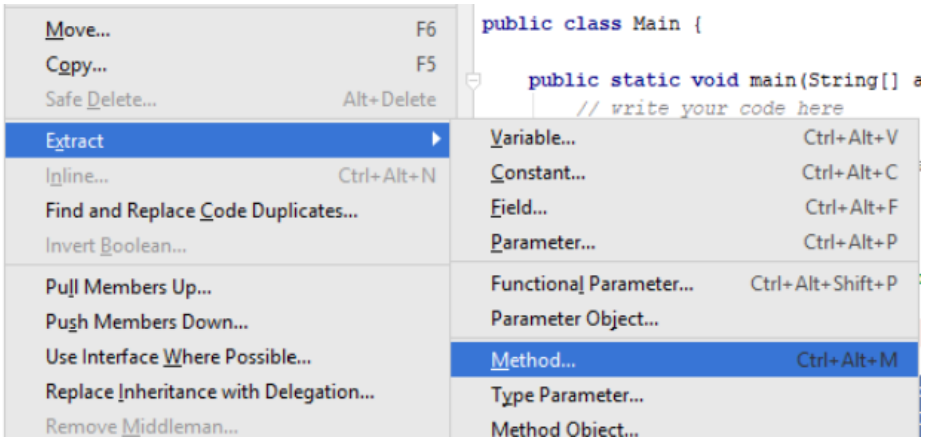
```
switch (input) {
    case 'a':
        handleA();
        break;
    case 'b':
        handleB();
        break;
    case 'c':
        handleC();
        break;
}
```

We could easily have done this **refactoring** (rewrite) manually, but let's be smart here and use the built in refactoring tools in IntelliJ.

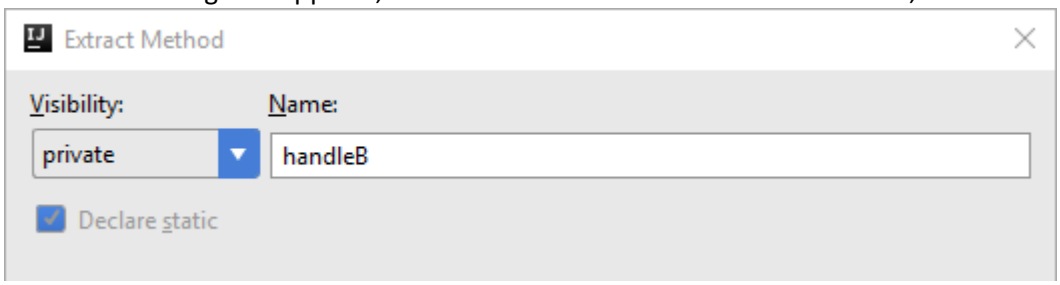
Let's refactor the 'b' case above, first select the text inside the 'b' case block, like this:

```
case 'b':
    System.out.println("You pressed 'b' and here's 10 numbers");
    for (int i = 0; i < 10; i++) {
        System.out.println(i);
    }
    break;
```

Then choose the **Refactor -> Extract -> Method** (Or press **Ctrl + Alt + M**)



Then in the dialog that appears, enter a name for the new method to create, like **handleB**



And then press OK and the code marked in the case statement was extracted into a method by IntelliJ.

Now refactor the two other case statements into methods using the extract method tool so that the switch statement looks like this:

```
switch (input) {
    case 'a':
        handleA();
        break;
    case 'b':
        handleB();
        break;
    case 'c':
        handleC();
        break;
}
```

Questions and concepts to study further on your own:

- Java switch statement and **fall through**
- Converting to and from switch statements
- IntelliJ refactoring tools
  https://www.jetbrains.com/help/idea/2017.1/refactoring-source-code.html
- Java's Switch Statement in Three Minutes
  https://www.sitepoint.com/javas-switch-statement/