Week 1 - A

Exercise Description

Installation of Java software, study of any Integrated development environment, Use Netbeans IntelliJ platform and acquaint with the various menus.

Create a test project, add a test class and run it.

See how you can use auto suggestions, auto fill.

Try code formatter and code refactoring like renaming variables, methods and classes.

Try debug step by step with java program to find prime numbers between 1 to n.

Step 1 - Installation and IDE

Download + Install IntelliJ Community

IntelliJ Download

Create a new project

Create Project

1. Chose Java and JDK 11 and press 'next'
2. Create a project from a template → Command Line App and press 'next'.
3. Give it a name: `MyFirstJavaProgram`
4. Chose a project location - best in a sub folder of your user's home folder. Base package: e.g.: `sseptp.org.first`

Test run

Add `System.out.println("Hello World");` and use auto complete to show how the IDE supports typing. Use `Ctrl+Space` - see more here

Step 2 - Code Formatting and Refactoring in IntelliJ

Code Formatting

Show students code formatting in the menu "Code" → "Reformat Code". Press `Ctrl+Z` and `Ctrl+Alt+L to visualize the changes.

Renaming

Ask students or point out wrong formatting:

1. Classes should be capitalized
2. Methods should start with lowercase
3. Variables should start with lowercase

Rename the Class, "main" Method and the variable "message" to fit the style norms. Use "Right Click" → "Refactor" → "Rename..." and illustrate how the variable, and the class file name changes as well.

Run the program to see if everything still works.

Step 3 - Debugging

The class `DebuggingPrimeNumbers` has a bug on line 37. It contains the solution to find all prime numbers between 0 and an upper boundary.

Discussion

Talk with the students and discuss what are prime numbers. How are they defined?

> Prime number is a number that is divisible only by itself and 1

Run the program in the class `DebuggingPrimeNumbers` and show them the output. Ask if the output is correct. Are these numbers really the prime numbers from 0 to 100?

Answer: **no, there must be a bug.**

Debugging Mechanics

Discuss standard debugging mechanics:

1. Thinking and reflecting about the code to find the error
2. Use System.out.println() functions everywhere to find the error
3. Or use a debugger

Debugging

Change the `searchUntil` to 10 to make the debugging a bit easier.

1. Show how to add breakpoints and add one in a loop.
2. Show how to run a program in debug mode and that the program stops at the break point.
3. Explain the debug window. Show where the variables, and their values can be found.
4. Explain "Step Over", "Step Into", "Step Out", "Resume Program" and "Stop".
5. Show how to remove a breakpoint and how to deactivate all break points.
6. Show the breakpoints window ("Run" → "View Breakpoints...") to see all breakpoints of a project.

Debug the project

1. Debug the project with the students to find the bug and fix it.
2. Put a breakpoint into `if (counter == 3)` and see with the students if it is correct.
3. Ask them what is wrong and let them find out on how to fix it.

Further Reading

Here are some links to learn more about these topics:

- Java Syntax: https://www.w3schools.com/java/java_syntax.asp
- IntelliJ Documentation: https://www.jetbrains.com/help/idea/getting-started.html