Programming in Java - working environment setup

1. [`*nix`] In the terminal/console window, enter the following commands:

   ```
   which java
   which javac
   which javap
   java --version
   javac --version
   javap -version
   ```

   and verify the version of *JDK* available in your system. *JDK 17+* is highly recommended.

2. Start *IntelliJ IDEA* and check its version.

3. [`IntelliJ`] Check if the *JUnit* plugin is enabled, and if not, enable it.

4. In your favourite *source code hosting platform* (e.g., *GitHub*, *Bitbucket*, *GitLab*, ...), create an *empty* (i.e., no `README.md` nor `.gitignore`) repository and name it `programming-in-java` . Set the language to `Java` .

5. [`IntelliJ`] Create a new Java/Gradle project:

   1. Select `File -> New -> Project...`
   2. In the *New Project* window select: `Gradle` , *Project SDK* `17` , and `Java`
   3. Set the project name to `programming-in-java`
   4. Set *GroupId* to `pl.edu.agh.ii` (*hint*: expand *Artifact Coordinates*)
   5. Press `Finish`
   6. Delete `src` directory
   7. Delete `build.gradle` file

6. [`IntelliJ`] Enable version control integration

   1. Select `VCS -> Enable Version Control Integration...`
   2. Select `Git` as the version control system
   3. Press `OK`
   4. Add `.gradle` directory to `.gitignore`

7. [`IntelliJ`] Configure the link to the remote repository

   1. Select `Git -> Manage remotes`
   2. Select `+` and add the *URL* of the just created repository (`programming-in-java`)
   3. Press `OK`

8. [`IntelliJ`] Perform initial commit

   1. Select `Git -> Commit...`
   2. Select all files
   3. As the *Commit Message* enter `Initial commit`
   4. Press `Commit`

9. [`IntelliJ`] Push the changes to the remote repository

   1. Select `Git -> Push...`
   2. Press `Push`

10. [`IntelliJ`] Create a module for the test lab class (`lab00`)

    1. In the `Project` window select `programming-in-java`
    2. Select `File -> New -> Module`
    3. Select `Gradle` , `JDK` (the same as for the project) and `Java`
    4. Press `Next`
    5. As the module name set `lab00` (the parent shoud be `programming-in-java`)
    6. Press `Finish`
    7. Ignore the message "*The IDE modules below were removed by the Gradle project reload:* `programming-in-java` ". *DO NOT restore it*

11. [`IntelliJ`] In `lab00/main/src/java` create package `agh.ii.prinjava.lab00.lst00_01`

12. [`IntelliJ`] In package `agh.ii.prinjava.lab00.lst00_01` create class `Main` . Change the content of `Main.java` to

    ```
    package agh.ii.prinjava.lab00.lst00_01;

    public class Main {
        public static void main(String[] args) {
            System.out.println("add(1,2) = " + Calc.add(1,2));
        }
    }
    ```

13. [`IntelliJ`] In package `agh.ii.prinjava.lab00.lst00_01` create class `Calc` . Change the content of `Calc.java` to

```java
package agh.ii.prinjava.lab00.lst00_01;

public class Calc {
    public static int add(int a, int b) {
        return a + b;
    }
}
```

14. [`IntelliJ`] Double-click on `Calc` class to open the corresponding `.java` file.

15. [`IntelliJ`] Set the cursor somewhere inside the class, then open the pop-up menu (right-click) and select `Generate... -> Test...`

16. [`IntelliJ`] As the testing library select `JUnit5` , check check-boxes `setUp/@Before` and `tearDown/@After` , and `add(a:int, b:int):int` and press `OK` .

17. [`IntelliJ`] Open the generated file (it should be in `lab00/src/test/java/agh.ii.prinjava.lab00.lst00_01`) and change its content to

```java
package agh.ii.prinjava.lab00.lst00_01;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.*;

class CalcTest {

    @BeforeEach
    void setUp() {
        System.out.println("CalcTest.setUp...");
    }

    @AfterEach
    void tearDown() {
        System.out.println("CalcTest.tearDown...");
    }

    @Test
    void onePlusTwoIsThree() {
        // if
        int a = 1, b = 2;

        // then
        assertEquals(3, Calc.add(a,b));
    }
}
```

18. [`IntelliJ`] Run the test (`onePlusTwoIsThree`) by clicking the green triangle on the left panel (with the line numbers).

19. [`IntelliJ`] Commit all the changes (`Git -> Commit...`) and then push the new commit to the remote repository.