HENDRIK PA LENSCH
LUKAS RUPPERT
RAPHAEL BRAUN
HASSAN SHAHMOHAMMADI

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

PRAKTISCHE INFORMATIK 2

SHEET 0

Submission date: 28.04.2022

The first step to get started with java programming language is to equip your operating system (OS) with Java Development Kit (JDK) and the development environment Eclipse (where you write your code). Based on your OS, follow the instruction below to set things up.

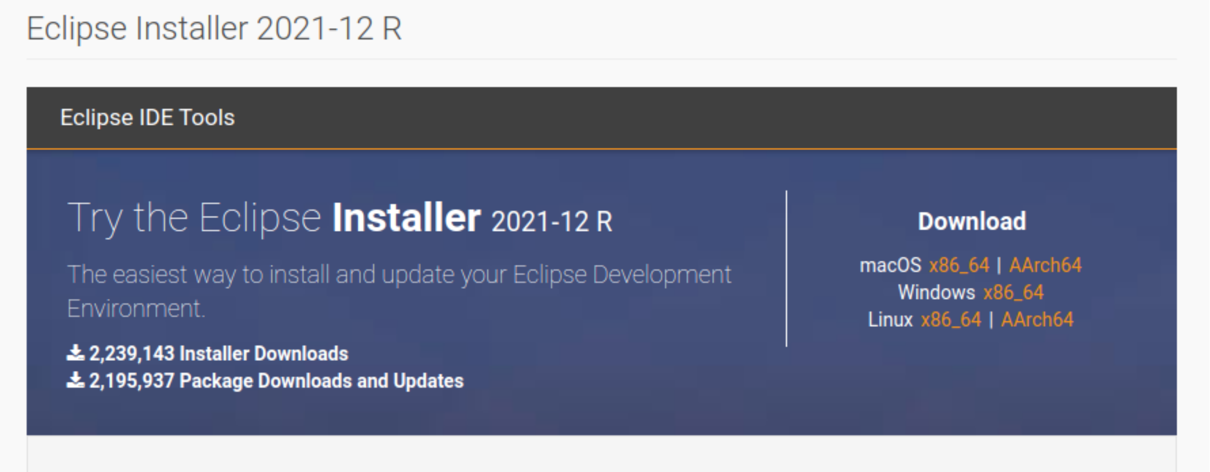
0.1 Installing Java

Follow the links below to download and install JDK17 based on your operating system.
Windows: `https://www.oracle.com/java/technologies/downloads/#jdk17-windows`.
Linux: `https://www.oracle.com/java/technologies/downloads/#jdk17-linux`
mac: `https://www.oracle.com/java/technologies/downloads/#jdk17-mac`

After installing the JDK, you can write your first Java program. However, to make programming easier, we will use another types of software known as integrated development environment (IDE) which helps managing your project files and offers many tools to run and debug (fix errors) your program. We will use the Eclipse IDE for this course.

0.2 Installing Eclipse

Eclipse can be used for other programming languages like C++ as well but in this course we are going to install eclipse for Java. For this, you need to download the eclipse installer based on your operating system at `https://www.eclipse.org/downloads/packages/installer`. as shown below:



Eclipse Installer 2021-12 R

Eclipse IDE Tools

Try the Eclipse **Installer** 2021-12 R

The easiest way to install and update your Eclipse Development Environment.

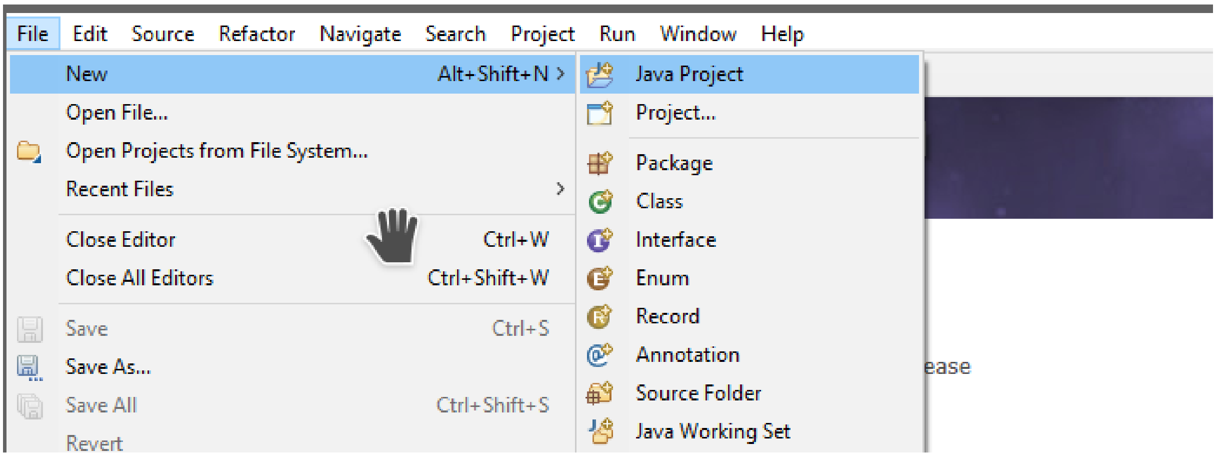
2,239,143 Installer Downloads
2,195,937 Package Downloads and Updates

Download
macOS x86_64 | AArch64
Windows x86_64
Linux x86_64 | AArch64

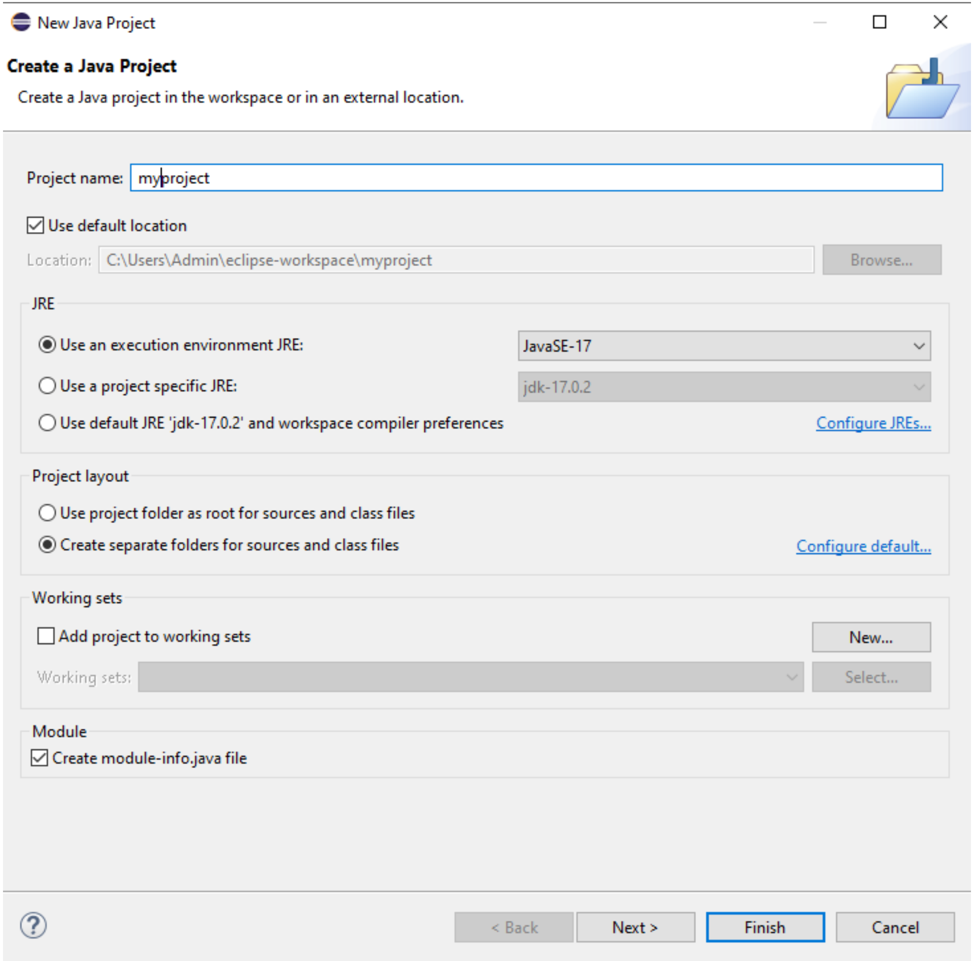
Follow the instructions on that webpage to install eclipse for Java. After the installation process, launch eclipse. It asks you to set the workspace address. This is a place where you would like to save your projects. You can leave it as the default location or set it to your desired location and click launch. Eclipse might detect the JDK installed on your machine and prompt you to choose the JDK you would like to use. You can either choose the installed JDK or use the default development package that comes with eclipse for Java shown as 'JavaSE17'.

0.3 Your first project in Eclipse

Now lets create your first Java project. When you first launch eclipse, you will see the welcome page, you can either click on create new project or right click on *file>New>Java Project*. This should be on the left top corner of the screen. If *Java Project* does not show up for you, you can find it under *file>New>Project...>Java>Java Project*.



you will be shown a new page where you are asked to set a name for your project. Shown in the following picture, write an arbitrary name (usually in lowercase) for your project.


New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: myproject

☑ Use default location

Location: C:\Users\Admin\eclipse-workspace\myproject

JRE

● Use an execution environment JRE: JavaSE-17

○ Use a project specific JRE: jdk-17.0.2

○ Use default JRE 'jdk-17.0.2' and workspace compiler preferences

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files

Working sets

☐ Add project to working sets

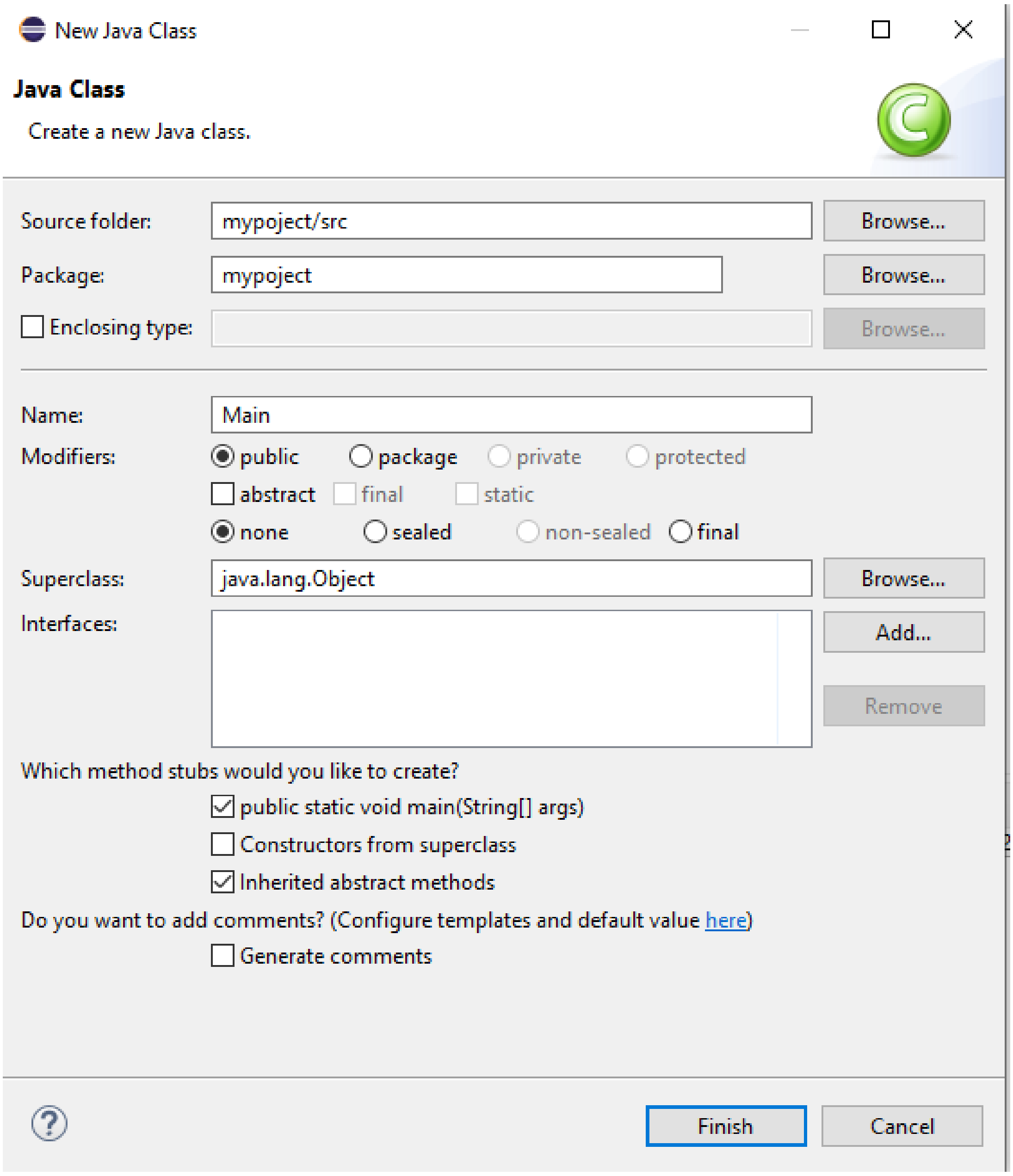
Working sets:

Module

☑ Create module-info.java file

in the JRE section, you can choose between the installed JDK-17 or the JavaSE17. As long as you use the same version (17), it should not matter. click finish to create the project. Eclipse might ask you whether you would like to create a Jave module after you click finish, for now, you dont need a module, you can click *dont create* if it prompts you.

You can see the project files on the left of your screen. The *JRE System Library* is imported automatically to your projects, it contains the necessary libraries to run your Java progarms. The *src* folder contains the source files (codes) for you project. For now there is no file here, so lets create a file here. Right click on the *src* folder then select *New > Class*. You will learn later in this course what a class is (nothing fancy!). You will be prompted to set the class properties.



New Java Class

Java Class

Create a new Java class.

Source folder: mypoject/src

Package: mypoject

☐ Enclosing type:

Name: Main

Modifiers: ● public ○ package ○ private ○ protected

☐ abstract ☐ final ☐ static

● none ○ sealed ○ non-sealed ○ final

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☑ public static void main(String[] args)

☐ Constructors from superclass

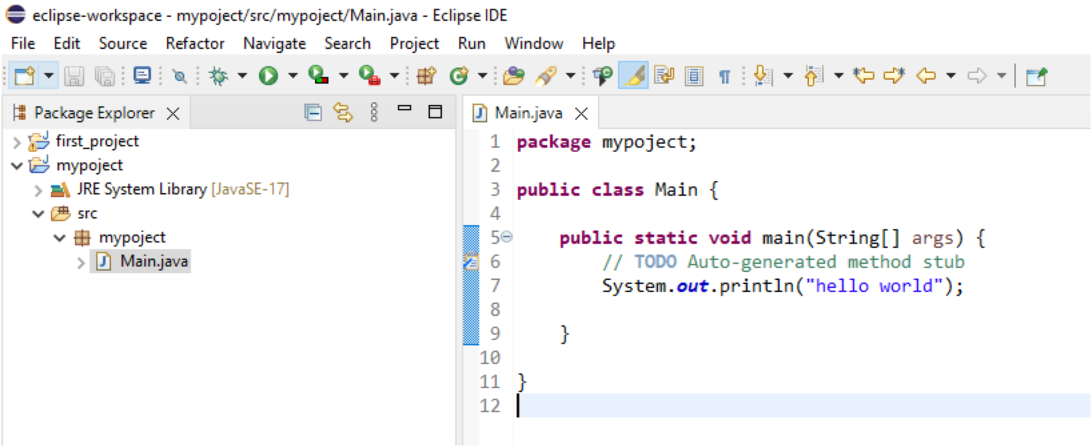
☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

☐ Generate comments

Write your class name in front of the *Name* textbox (usually start with Capital letter) and make sure `public static void main(String[] args)` is checked. This will enable you to run this class and see the results of your codes (more about this later in the course). Then click finish. You will then see the content of your class.

Our first program is very simple, we will tell Java to print a massage and that is all. To do this, we will use the `System.out.println("your message");` command. Write this line of code as shown below:

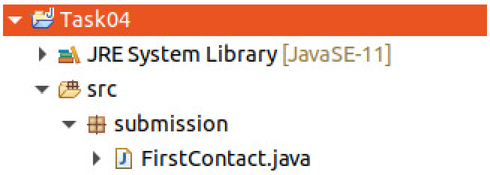


```
package mypoject;

public class Main {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("hello world");

    }

}
```

To run your program, you can simply click on the green circle with a triangle inside under the *Navigate* tab or right click inside the *Main.java* file then *Run As > Java Application*. You should be able to see your message in the *Console* section on the bottom of your screen. If so, congratulations, you just wrote your first Java program!

0.4 Importing project in Eclipse

To work on the exercises, you first need to import them into Eclipse. To do that, you first need to create a new project - for this task call it 'Task04' (see Task 0.3 for instructions). Then, you will have to import the source files into the `src` folder. To do that, you can simply copy the `submission` folder which we provide in `student_template_04` and paste it into the `src` folder by *right click on src > paste.*

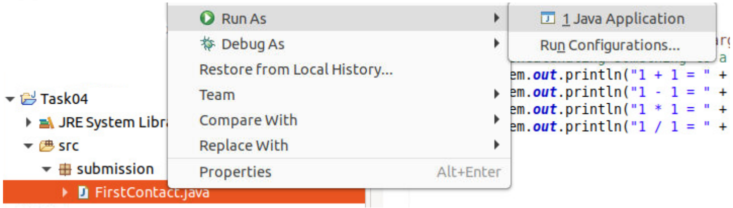


This procedure can be used for all tasks in this course. Note that a single sheet can consist of multiple tasks, thus you might have to create multiple projects in a single exercise.
If the import was successful you will be able to see the `.java` file(s) inside the `submission` package as shown below.
By clicking on the `.java` file, you will see its content on the screen. Then the floor is yours to show your programming skills!

Running the program

After opening the `FirstContact.java` file as described above, you can run the program in two different ways:

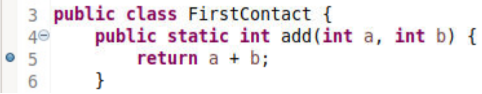
- Press the 'Run' button in the top tool bar.
- Right-click the `FirstContact.java` file in the project manager and select: `Run As -> Java Application`.



```
em.out.println("1 + 1 = " +
em.out.println("1 - 1 = " +
em.out.println("1 * 1 = " +
em.out.println("1 / 1 = " +
```

Debugging the program

Similarly, you can start your program in 'Debug Mode'. In debug mode the program can be interrupted at *breakpoints*. This allows you to inspect the state of each variable in the program during execution.
You can set breakpoints in any line of your code by **double-clicking** the **space left of the line number**.
A break point is indicated as blue dot like for example here in line 5:



```
  3 public class FirstContact {
  4⊖    public static int add(int a, int b) {
● 5         return a + b;
  6     }
```

If you run your program in debug mode it will interrupt the execution when it reaches a line with a breakpoint.
You can get rid of a break point with another double-click.
To run the program in debug mode you have again the same two options:

- Press the 'Debug' button in the top tool bar.
- Right-click the `FirstContact.java` file in the project manager and select: `Debug As -> Java Application`.

You can switch between *Debug View* and regular *Java View* by clicking the corresponding buttons at the very right end of the top tool bar.

0.5 Register on InfoMark

The exercise submissions will be managed via the InfoMark platform at `https://infomark.informatik.uni-tuebingen.de`. InfoMark is a submission management system, that automatically provides feedback from a set of unit-tests. So whenever you upload a solution it will be tested for correctness and you will immediately see the results. This can help you identify obvious mistakes in your code.
Please be aware that you'll only see the test results of some very simple sanity checks. The system will also run more thorough tests in the background, whose results are only shown to the tutors.
Create an account on InfoMark by registering on InfoMark. Please stick to the following rules:

- If possible, register with your **student email address**. Your tutor will see this email address. The student email address should be something like this: `first_name.surname@student.uni-tuebingen.de`.
- As study subject use the English version. We need this in order to send your grade to the correct examination office in the end:

Informatik	Computer Science
Medien Informatik	Media Informatics
Bio Informatik	Bio Informatics
Kognitionswissenschaften	Cognitive Science
Medizin Informatik	Medical Informatics
Informatik Lehramt	Computer Science Teaching Degree
Physik	Physics
Mathematik	Math
Other	Use official English term, all words starting with a capital letter. Avoid typos.

After registration you will receive an email. You have to **confirm your account** before you can use it.

0.6 Enroll for the Informatik 2 course

Please enroll for our Informatik 2 course. This is the only course there currently is, so it should be straight forward. After enrolling, you can show the content of the course.

Please specify your preferences for exercise groups. You will automatically be assigned to an exercise group based on your preferences. Please be honest and don't just specify a single group, to prevent completely random assignments if the desired group is full.

You will find the lecture-slides, additional material and all exercise sheets here as soon as they are made available.

0.7 Submit Something

In this task you will familiarize yourself with InfoMark - the submission system used to test and grade your submissions.

For this task we provided the `student_template_07`. It is very similar to `student_template_04`, except that there is an error in the code. You don't have to do anything with the code - it is just there for you to upload and observe the feedback yourself.

Prepare submission

Before uploading anything you have to zip your submission. The zip file only has to contain the `submission` package directory, which in turn contains all the `.java` files.

When solving an exercise, you typically start from such a `student_template` that contains the `submission` directory that you import to eclipse. You change or add some `.java` files in the `submission` directory. Then you zip the modified `submission` directory from your eclipse workspace and upload your submission.

When you upload a submission, it is automatically tested by InfoMark. You will see feedback from some of those tests, which allows you to go back to your code and fix issues that were detected by the tests. You can upload as often as you want until the deadline. Only the last submission will be used for grading.

Test Results

- Please zip the `submission` directory in `student_template_04` and upload it for **Task07: First-Contact** on InfoMark.

You will immediately get the feedback:

```
submission received and will be tested
```

After 20 - 60 seconds the message will update to:

```
openjdk 17 2021-09-14
OpenJDK Runtime Environment (build 17+35-2724)
OpenJDK 64-Bit Server VM (build 17+35-2724, mixed mode, sharing)
javac 17
[   OK   ] addValues()
[   OK   ] subtractValues()
[   OK   ] multiplyValues()
[   OK   ] divideValues()
[   OK   ] classStructureTest()
```

You can see that 5 tests have been run, and all five have passed successfully, which is indicated by the OK.

Tests can also fail. In that case, InfoMark will provide more detailed information about what exactly went wrong. There is a detailed guide on how to interpret everything that can be reported by InfoMark.

You find this guide as “InfoMark Guide” on InfoMark. This guide is somewhat technical and requires some knowledge that you will learn in the first few weeks of the course, so don’t worry if you don’t understand it entirely right now.

Compiler Errors

Running tests is impossible if you upload a submission that does not contain valid Java code. We prepared such a submission in `student_template_07`. In this submission, we removed a semicolon ’`;`’ symbol at the end of an instruction. You will learn that this is not allowed - in the Java syntax, each instruction must end with a semicolon. Statements without semicolons are therefore *syntax errors*, which make it impossible to compile and therefore run the program. Your grades for the exercises is determined by the test results. If you upload broken code which cannot be tested, you will get **0 Points** for your submission. It is therefore very important to pay attention to the feedback from InfoMark.

Luckily, Eclipse automatically detects all kinds of syntax errors on the fly. So it should be very easy to spot them before the submission.

Let’s now upload `student_template_07` to see how InfoMark handles such problems. You should get feedback that looks like this:

```
1 openjdk 17 2021-09-14
2 OpenJDK Runtime Environment (build 17+35-2724)
3 OpenJDK 64-Bit Server VM (build 17+35-2724, mixed mode, sharing)
4 javac 17
5 [ERROR] Failed to compile the submission. Use the following information to
      resolve the issue:
6 submission/FirstContact.java:5: error: ';' expected
7         return a + b
8                     ^
9 1 error
```

The important bit is *Line 5*. Here it says “Failed to compile the submission.” This automatically means that you would get **0 Points** if you didn’t fix that issue.

In *Line 6*, you see that it even tells you that a ’`;`’ seems to be missing, which is exactly what we would expect from that submission.

If you want more information on how tests can fail and what to do about it please refer to the aforementioned "InfoMark Guide".

Public and Private tests

You will only get the feedback of some simple tests, which we call ’public tests’. The feedback of most tests, which check all the weird corner cases and problematic inputs, is only shown to the tutors. Those are the ’private tests’. This mechanism prevents you from *overfitting* the tests. For an example refer to the "InfoMark Guide"

Just keep in mind that passing all public tests does **not** mean that your submission is correct! It just means that it is not completely wrong.