Department Mathematik/Informatik, Abteilung Informatik
Software & System Engineering
Weiterführende Konzepte der Programmierung, SS 2025

Niklas Bockholt, Chiara Hafner, Alicja Bezpalko,
Jiyan Alis, Dilara Günay,
Simon Wolf, Adrian Bajraktari, Dr. Mersedeh Sadeghi

Homework 6. Threads, Testing, File Handling

Registration Deadline: 18.05.2025, 23:59
Hand-in Deadline: 21.05.2025, 23:59

Exercise 1. Matrix Multipication

Create class `DotProductThread` with variables `int v1, v2, dot` which extends Thread. Create the following:

- Constructor, which assigns `v1, v2`.
- Implement the run(); method. It should compute the dot product of `v1` and `v2` and assign it to `dot`.

Create class `MatrixMultMain` with:

- A main method
- A method which computes the product of two matrices using `DotProductThread`.

Try using `run()` and `start()` for starting threads. Which one is the better choice? Explain your observations. Is this a good application to use threads? Explain why or why not.

Exercise 2. Installing JUnit5

Here you will find a description how to set up JUnit5 in your Project, this task will not be graded and you can't present it in class for the bonus. We recommend you use Maven.

JUnit in maven

Maven is a open-source build and project management tool primarily used for Java projects. Maven essentially provides a package structure and allows access to a variety of libraries, including JUnit. For more information on Maven's features see `https://maven.apache.org/what-is-maven.html`.

Most IDE's have the option to start a Maven project:

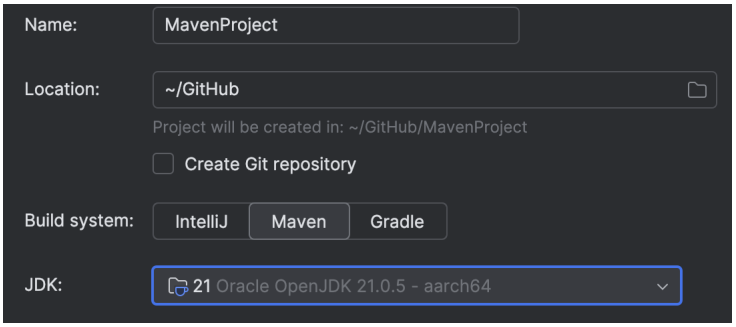

Figure 1: *IntelliJ:* New Project -> Build System -> Maven

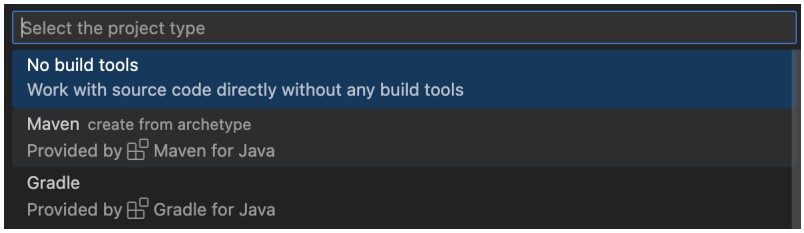

Figure 2: *VSCode:* cmd+shift+p -> Java: Create new Java project -> Select project type Maven

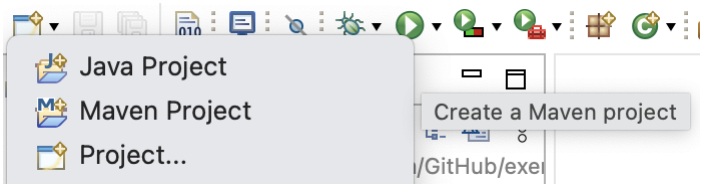

Figure 3: *Eclipse:* new -> Maven project

In a Maven project you will find an automatically generated file *pom.xml* which contains information about your project and configurations. You need to modify this file in order to use JUnit5. Generally you need to add a dependency for JUnit5 which can look something like this:

```xml
<dependencies>
    <!-- JUnit 5 -->
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-api</artifactId>
        <version>5.10.0</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-engine</artifactId>
        <version>5.10.0</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Additionally you will want to add the Maven surefire plugin which is needed to run the tests of an application. In order to do so you have to add the following to your pom.xml file:

```xml
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-surefire-plugin</artifactId>
            <version>3.1.2</version>
        </plugin>
    </plugins>
</build>
```

If you have problems with the JUnit5 dependency see https://maven.apache.org/surefire/maven-surefire-plugin/examples/junit-platform.html.

If you want to learn more about dependency management in Maven see https://maven.apache.org/guides/introduction/introduction-to-dependency-mechanism.html#Dependency_Scope.

Now you should be able to write JUnit5 test classes in the automatically generated `test` package.

JUnit without build tool

If you want to use JUnit without a build tool such as Maven you need to download the following JARs manually:

After downloading you need to put the JARs into a separate folder in your project (e.g. `MyProject/libraries/`) and add them to the classpath. You can find the JARs here https://repo1.maven.org/maven2/org/junit/.

Exercise 3. JUnit testing

The following code is not entirely correct. We need your help to solve it! Please copy the class into your repository and create a separate class, in which you use JUnit tests to find out what the issues are.

```
package ex2;

public class Fraction{
    private int zaehler;
    private int nenner;

    //icomplete
    public Fraction(int zaehler, int nenner){
        this.zaehler = zaehler;
        this.nenner = nenner;
    }

    //wrong
    public void Kehrwert(){
        this.setNenner(this.zaehler);
        this.setZaehler(this.nenner);
    }

    //wrong
    public void multiplikation(Fraction a){
        this.setNenner(this.nenner * nenner);
        this.setZaehler(this.zaehler * zaehler);
    }

    //wrong
    public void addition(Fraction a){
        if (a.getNenner() != this.getNenner()){
            int neuerZaehler;
            this.setZaehler(this.getZaehler()*a.getNenner());
```

```
            neuerZaehler = a.getZaehler()*this.getNenner();
            this.setZaehler(this.getZaehler()+neuerZaehler);
        } else {
            this.setZaehler(this.getZaehler() + a.getZaehler());
        }
    }

    //wrong
    public void kuerzen(){
        int ggt = ggt(Math.abs(zaehler), Math.abs(nenner));
        this.zaehler *= ggt;
        this.nenner *= ggt;
    }

    //helper method Euklidian algorithm,
    // here are no mistakes
    private int ggt(int a, int b) {
        while (b != 0) {
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }

    @Override
    public String toString(){
        return this.getZaehler()+"/"+this.getNenner();
    }

    public int getZaehler() {
        return zaehler;
    }

    public void setZaehler(int zaehler) {
        this.zaehler = zaehler;
    }

    public int getNenner() {
        return nenner;
    }

    public void setNenner(int nenner) {
        this.nenner = nenner;
    }
}
```