technische universität
dortmund

Fakultät für Informatik
Lehrstuhl 11

Jonas Ellert
Mart Hagedoorn
Guangping Li

Sommersemester 2022
8. April

DAP2 Praktikum – Blatt H
Hilfestellungen

Abgabe: —

Die Hilfestellungen können sich im Laufe des Semesters ändern. Die aktuelle Version ist immer im Moodle verfügbar!

1 Java in der Kommandozeile

Auch wenn die Verwendung von IDEs nicht explizit verboten ist, empfehlen wir Ihnen, ihre Programme von Hand in der Kommandozeile (Konsole, Terminal) zu kompilieren und auszuführen. Das festigt nicht nur Ihr grundlegenes Verständnis der Anwendungsentwicklung, sondern ist (nach einer kurzen Eingewöhnungsphase) oftmals auch schneller und angenehmer als die Verwendung der IDE.

Befindet man sich im Terminal im selben Verzeichnis wie die zu kompilierende und auszuführende Klasse, so kann man den Quellcode wie folgt in Java-Bytecode übersetzen:

```
javac Klasse.java
```

Der Compiler erstellt dabei die Datei `Klasse.class`. Sofern Ihre Klasse eine `main`-Methode enthält, können Sie nun das Programm ausführen:

```
java Klasse
```

(Beachten Sie, dass die Endung `.class` beim Aufruf entfällt.)

1.1 Übergeben von Command-Line-Argumenten

Beim Aufruf von Java-Programmen können Sie auch Parameter in Form von sogenannten Command-Line-Argumenten übergeben. Das folgende Programm führt beim Aufruf von

```
java Order 17 13
```

zur Ausgabe `"13 is not larger than 17"`. Beachten Sie, dass die Argumente im Programm zunächst nur als Strings zur Verfügung stehen, und daher erst in den gewünschten Datentyp konvertiert werden müssen.

```
class Order {
    public static void main(String[] args) {

        if (args.length != 2) {
            System.out.println("Exactly two arguments are required.");
            return;
        }

        int min = 0;
        int max = 0;

        try {
            min = Integer.parseInt(args[0]);
            max = Integer.parseInt(args[1]);
            if (max < min) {
                int h = min;
                min = max;
                max = h;
            }

        } catch (NumberFormatException e) {
            System.out.println("The arguments must be integers.");
            return;
        }

        System.out.println(min + " is not larger than " + max);
    }
}
```

1.2 Lesen des Standard-Input-Streams

Command-Line-Argumente eignen sich nur für simple Parameter ihres Programms, z.B. für einzelne Ganzzahlen. Oftmals wollen Sie allerdings gleich eine größere Menge von Eingabedaten verarbeiten. Dazu bietet sich der Standard-Input-Stream (kurz Standard-In) an. Mit diesem können Sie beispielsweise eine Liste von Werten als Eingabe lesen, wobei die Werte durch einen Zeilenumbruch getrennt sind.

Ein Beispiel-Programm dazu sehen Sie unten. Mit einem `Scanner` werden solange Eingabezeilen gelesen, bis das Ende der Eingabe erreicht wird. Auch hier liegen die Werte zunächst als Strings vor, und müssen noch in den gewünschten Typ überführt werden.

```java
import java.util.Scanner;

class Maximizer {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        int max = 0;

        while (input.hasNextLine()) {
            try {
                int value = Integer.parseInt(input.nextLine());
                max = (value > max) ? value : max;
            } catch (NumberFormatException e) {
                System.err.println("Input list contains a non-number.");
                return;
            }
        }

        System.out.println("The largest number was " + max + ".");
    }
}
```

Ein Vorteil von Standard-In ist es, dass Sie mit einer einzigen Implementierung gleich eine Vielzahl von Anwendungsfällen bedienen. So können Sie das Beispiel-Programm `Maximizer` auf mehrere Arten ausführen:

- Sie verwenden im Terminal den Operator `<`, um den Inhalt einer Datei als Eingabe zu verwenden. Im folgenden Aufruf ist `/datasets/primes.txt` der Pfad zu einer Datei, die in jeder Zeile genau eine Ganzzahl enthält.

  ```
  java Maximizer < /datasets/primes.txt
  ```

- Sie verwenden im Terminal den Operator `|` („Pipe"), um die Ausgabe eines anderen Programms als Eingabe zu verwenden. Der Aufruf `seq 100` erzeugt eine Liste der positiven Ganzzahlen von 1 bis 100. Diese können Sie direkt als Eingabe für Ihr Programm verwenden:

  ```
  seq 100 | java Maximizer
  ```

- Sie können mit `|` auch mehrere Programme „verketten". Beispielsweise können Sie die Liste der positiven Ganzzahlen von 1 bis 100 mit dem Befehl `shuf` erst noch zufällig permutieren, und erst danach als Eingabe für Ihr Programm nutzen:

  ```
  seq 100 | shuf | java Maximizer
  ```

- Natürlich können Sie die Eingabewerte für Standard-In auch einfach per Tastatur eingeben, was besonderns hilfreich ist, um schnelle Tests während der Implementierung auszuführen. Dazu rufen Sie das Programm erst ganz normal auf, und geben dann die Werte ein. Nach jedem Wert bestätigen Sie mit der `Enter`-Taste. Wenn Sie keine weiteren Werte mehr eingeben möchten, drücken Sie `Strg+D` (je nach Betriebssystem muss eventuell eine andere Tastenkombination gedrückt werden).

  ```
  java Maximizer
  ```

  ```
  13 Enter 5 Enter 17 Enter 1 Enter 42 Enter 0 Enter Strg+D
  ```