hochschule mannheim

FAKULTÄT FÜR INFORMATIK

Prof. Dr. Peter Knauber

PR1 – Programmieren 1 Bachelor-Studiengang Informatik WS2017/2018

Übungsblatt 4: Methoden

Ausgabe: 23.10.17
Abgabe: 6.11.17 12:00 Uhr elektronisch mittels Git

Aufgabe 1

Das Spiel TicTacToe spielt man auf einem 3x3 Felder großen Feld. Es spielen zwei Spieler gegeneinander, wobei sie immer abwechselnd ein leeres Feld mit ihrem Zeichen markieren. Der eine Spieler benutzt Kreuze als Markierung und der andere Spieler benutzt Kreise. Der erste Spieler, der es schafft drei seiner Symbole ohne Unterbrechung in einer Reihe zu haben, gewinnt das Spiel. Als Reihe gelten waagerechte, senkrechte oder diagonale Reihen.
Im Wiki finden Sie die Java-Klasse `TicTacToe`, die ein fast vollständiges Spielprogramm enthält. Es fehlt lediglich die Implementierung von zwei Methoden:

- Die Methode `success` soll genau dann `true` liefern, wenn einer der beiden Spieler eine Dreierreihe auf dem Spielfeld erreicht hat, sonst `false`.
- Die Methode `draw` soll genau dann `true` liefern, wenn das Spielfeld vollständig mit Symbolen belegt ist, ohne dass einer der Spieler gewonnen hat, sonst `false`.

Implementieren Sie diese beiden Methoden, ohne deren Parameter oder den Programmcode außerhalb der Methoden zu verändern. Wenn Sie die Methoden korrekt implementiert haben, können Sie TicTacToe spielen.

Aufgabe 2

Erstellen Sie ein Java-Programm (Klasse "DividersArrayResult"), das alle Teiler einer *natürlichen Zahl* berechnet.
In der `main`-Methode soll der Benutzer aufgefordert werden, eine ganze Zahl einzugeben. Nachdem er seine Zahl eingetippt hat, wird eine Methode `calculateDividers` aufgerufen, welche die eingegebene Zahl als Parameter erhält. Diese Methode soll alle Teiler des übergebenen Parameters in ein Array von ganzen Zahlen der Länge 500 schreiben. Dieses Array wird wie folgt vereinbart:

```
int[] dividers = new int[500];
```

Die erste gültige Indexposition ist die Position 0, mit

```
dividers[0] = ersterGefundenerTeiler;
```

wird darauf zugegriffen.

Nachdem alle Teiler in das Array geschrieben wurden, liefert `calculateDividers` das Array als Ergebnis an `main` zurück. In `main` soll der sinnvolle Inhalt des Arrays (d.h. nur die gefundenen Teiler) in einer Zeile ausgegeben werden. Dabei werden die enthaltenen Zahlen jeweils durch ein Leerzeichen getrennt.
Falsche Eingaben sollen in `main` durch eine Fehlermeldung "Eingabe ungültig" signalisiert werden. Das Programm endet nach Ausgabe der Teiler bzw. der Fehlermeldung.

Sie können sich darauf verlassen, dass nur ganze Zahlen eingegeben werden.
Wählen Sie vernünftige Namen für Ihre Variablen!

Aufgabe 3

Erstellen Sie ein Java-Programm (Klasse "SearchInRandomNumbers"), dessen `main`-Methode zwei ganze Zahlen `numberCount` und `numberToSearch` abfragt.
Klarstellung: Sie müssen die Eingaben nicht überprüfen.

Die Zahl `numberCount` wird an eine Methode `generate` übergeben, die ein Array für ganze Zahlen mit `numberCount` Elementen anlegt, dieses Array mit Zufallszahlen zwischen 1 und 1000 (beide inklusive) füllt und das befüllte Array als Ergebnis an `main` zurückliefert.
Klarstellung: Ist die Zahl `numberCount` kleiner als 0, wird in der Methode `generate` eine `PRException` ausgelöst. Halten Sie sich für die Erzeugung der Zufallszahlen an das Muster mit den Lottozahlen aus der Vorlesung.

Die `main`-Methode übergibt das erzeugte Array und die zweite eingegebene Zahl `numberToSearch` an eine zweite Methode `searchAll`. Diese Methode sucht *alle* Positionen von `numberToSearch` im übergebenen Zufallsarray, legt ein neues Array an, dessen Länge genau der Anzahl der gefundenen Positionen entspricht, befüllt dieses Array mit den gefundenen Positionen und liefert es als Ergebnis zurück. Die `main`-Methode gibt dann alle Positionen auf der Konsole aus.
Klarstellung: Falls die Zahl `numberToSearch` nicht gefunden wird, wird ein Positionsarray der Länge 0 erzeugt und nichts ausgegeben.

Danach übergibt `main` das Array mit den Zufallszahlen und die Zahl `numberToSearch` an eine dritte Methode `searchLast`. Diese sucht die *letzte* Position von `numberToSearch` und liefert diese Position als Ergebnis zurück. In der `main`-Methode wird diese Position auf der Konsole ausgegeben.
Klarstellung: Falls die Zahl `numberToSearch` nicht gefunden wird, wird in der Methode `searchLast` eine `PRException` ausgelöst.

Allgemeines

- Die Aufgaben sind in Eclipse zu bearbeiten und beim Testat vorzuführen.
- Legen Sie für die Bearbeitung dieses Übungsblattes ein Paket namens `pr1.uebung04` an, in dem Sie Ihre Klassen anlegen.
- Erlaubt sind `MakeItSimple`-Funktionen (keine nicht besprochene Funktionalität aus der Java-Standard-Bibliothek) und das bisher erworbene Wissen aus den PR1-Vorlesungen. Sie müssen alle(!) von Ihnen verwendeten Konstrukte der Sprache sowie alle verwendeten Methoden, die nicht aus der Hilfsbibliothek `MakeItSimple` stammen, gut erklären und nötigenfalls im Testat selbst programmieren können!
- Sie geben ab, indem Sie vor Ende der Abgabefrist Ihr Projekt mit den lauffähigen Programmen des Übungsblattes in das Repository auf *ovid* pushen. Achten Sie darauf, ob Sie die Kontroll-E-Mail bekommen! Die letzte hochgeladene Version Ihres Projekts wird gewertet. Andere Abgaben, ob elektronisch oder auf Papier, zählen als **nicht abgegeben**!