Programmierpraktikum

Wintersemester 2021/22

Prof. Dr. rer. nat. habil. Petra Hofstedt
Ilja Becker M. Sc., Sven Löffler M. Sc.
Sonja Breuß B. Sc., Carlo Bückert B. Sc., Endris Wenzke

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Zusatzblatt

Abgabedatum: 02.01.2022

Die Abgabe Ihrer Lösungen erfolgt vor Ablauf der Abgabefrist digital über die Moodle-Plattform. Erstellen Sie dazu ein PDF-Dokument, das die Lösungen Ihrer schriftlichen Aufgaben enthält. Laden Sie dieses PDF-Dokument und den erarbeiteten Java-Code (.java-Dateien) mit den in den Aufgaben vorgegebenen Namen bei Moodle hoch. Bitte laden Sie die Dateien einzeln hoch, Dateiarchive (z.B. .zip-Dateien) werden nicht akzeptiert.

***** Halten Sie sich an die Java Code Conventions! *****

Sie können maximal **(7 Punkte)** mit diesem Übungsblatt erreichen.

Aufgabe 1 (Kleine Berechnungen.)

1 Punkt

1. **Wochentage**

   Schreiben Sie ein Programm `Wochentag.java`, welches nach einander den Tag, den Monat und das Jahr einliest und für die Eingaben Daten den jeweiligen Wochentag ausgibt.

   Lesen Sie sich dazu zunächst den Folgenden Wikipedia-Artickel `https://de.wikipedia.org/wiki/Wochentagsberechnung` (vor allem den Abschnitt Algorithmus) zu diesem Thema durch.

   Ein mögliches durchlaufen des Programmes könnte z.B. wie Folgt aussehen:

   ```
   >> Geben Sie den Tag ein.
   << 06
   >> Geben Sie den Monat ein.
   << 06
   >> Geben Sie das Jahr ein.
   << 1944
   >> Dienstag
   ```

   Zeilen die mit einem » losegehn, sind dabei Ausgaben und Zeilen die mit einem « losgehen, sind dabei die User-Eingaben.

   (0,5 Pkt.)

2. **Fakultät**

   Implementieren Sie eine Methode `multipliziere`, die mittels Additionsoperationen und einer Schleife zwei Variablen miteinander multipliziert und das Ergebnis zurückgibt. Schreiben Sie eine Methode `fakultaet`, die eine Zahl n als Eingabe erhält und unter Nutzung der `multipliziere`-Methode und einer Schleife die Faklutät von n ($n!$) zurückgibt.

   Verwenden Sie für die `Methoden` `multipliziere` und `fakultaet` jeweils eine andere Schleifenart.

Schreiben Sie eine `main`-Methode in der von einem Scanner eine Zahl n eingelesen werden soll, die `fakultaet`-Methode aufgerufen und deren Ergebnis in der Konsole ausgegeben werden soll.

(0,5 Pkt.)

Aufgabe 2 (Leise rieselt der Schnee) 2 Punkte

Implementieren Sie ein Programm `Schnee.java`. Unter Verwendung der Klasse `Console.java` soll rieselnder Schnee im simulierten Textmodus dargestellt werden. Orientieren Sie Sich dabei an dem Video `Schnee-Beispielvideo.mp4` aus der Vorgabe.

Solange der Schnee fällt soll er sternförmig (dargestellt als Sternchen-Zeichen '*') sein. Ist der Schnee auf dem Boden oder auf einen Schneehaufen angekommen, soll er das entsprechende Feld ausfüllen ('□' bzw. ein weißes Kästchen).

Sobald eine liegenbleibende Schneeflocke die 20. Zeile erreicht sollen die untersten 10 Zeilen der Konsole geleert werden. Alle darüber liegenden Schneeflocken sollen dann um 10 Felder nach unten rutschen. Das Schneetreiben endet somit niemals.

Aufgabe 3 (Vier gewinnt) 4 Punkte

Vier gewinnt ist ein Zweipersonenstrategiespiel mit dem Ziel, als Erster vier der eigenen Spielsteine in eine Linie zu bringen (siehe `https://de.wikipedia.org/wiki/Vier_gewinnt`). Das Standardspielfeld hat dabei ein Größe von 7x6.

Programmieren Sie eine Konsolenversion von Vier gewinnt (`VierGewinnt.java`) so, dass zwei Spieler an einem Computer das Spiel gegeneinander spielen können. Ein Spieler soll dabei jeweils durch die Eingabe einer Zahl zwischen Null und Sechs angeben können, wo ein Stein fallengelassen werden soll. Der Stein soll dann automatsch soweit „nach unten fallen" bis unter ihm das Feld endet oder ein anderer Stein liegt. Vor jedem Setzen eines Steines soll in der Konsole ausgegeben werden welcher Spieler am Zug ist. Die Information, welcher Spieler am Zug ist kann z. B. in einem `boolean`-Wert gespeichert werden.

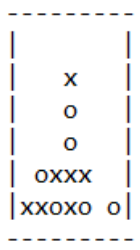
Programmieren Sie für eine bessere Überschaubarkeit des Programms zunächst eine Methode `setzeStein`, die, wenn Spieler1 an der Reihe ist ein 'x' und ansonsten ein 'o' an die entsprechende Position (`position`) eines Araays (`field`) setzt, beachten Sie dabei das „Herunterfallen" des Steines. Nutzen Sie für die Methode die folgende Signatur:

```
private static void setzeStein(boolean spieler1, int position, char[][] field)
```

Schreiben Sie weiterhin eine Methode `maleFeld`, die das Spielfeld auf der Konsole ausgibt und folgende Signatur besitzt:

```
private static void maleFeld(char[][] field) {
```

Das Spielfeld könnte nach ein paar Runden z. B. wie folgt aussehen.



Das Programm muss nicht überprüfen ob ein Spieler gewonnen hat. Dafür sind die Spieler selbst verantwortlich.

Die Spieler sollen solange nach einer Eingabe gefragt werden (und diese soll auch verarbeitet werden) bis ein Spieler keine Zahl sondern eine andere Zeichenkette eingibt. Dann soll das Programm ohne Fehlermeldung beendet werden.

Für eine eingegebene Spaltennummer soll zunächst geprüft werden, ob diese im `Array` liegt (Wert zwischen 0 und 6) bzw. ob die Spalte schon voll ist. Tritt einer der beiden Fälle auf, so soll das dem Spieler mitgeteilt werden und er soll erneut eine Eingabe tätigen können.

(1,5 Pkt.)

Weitere Aufgaben

1. Schreiben Sie eine Methode `hatGewonnen`, die nach jedem Zug überprüft, ob ein Spieler das Spiel gewonnen hat. Bauen Sie die Methode so in Ihr Programm ein, dass das Spiel automatisch beendet wird, wenn ein Spieler gewonnen hat.
   Sollte das Spielfeld voll sein, soll das Spiel ebenfalls beendet werden.

   (1 Pkt.)

2. Schreiben Sie einen Computergegner, der immer, wenn er am Zug ist, einen Stein an einen zufälligen gültigen Platz setzt.

   (0,5 Pkt.)

3. Erweitern Sie den Computergegner wie Folgt. Wenn der menschliche Spieler drei Steine in einer Reihe hat, soll der Computergegner dies erkennen und nach Möglichkeit verhindern, dass der menschliche Spieler das Spiel gewinnt.

   (1 Pkt.)