Projektblatt P4 (23 P)

Abgabe: Donnerstag 10. Oktober 2024, 12:00h

Entpacken Sie zunächst die Archiv-Datei `vorgaben-p4.zip`, in der sich die Rahmendateien für die Aufgaben befinden. Ergänzen Sie die Dateien zunächst durch Ihren Namen. Ergänzen Sie die Dateien dann durch Ihre Lösungen gemäß der Aufgabenstellung unten. Der hinzuzufügende Java-Sourcecode sollte **syntaktisch richtig** und **vollständig formatiert** sein. Alle Dateien sollten am Ende fehlerfrei übersetzt werden können.
Verpacken Sie die `.java` Dateien für Ihre Abgabe in einem ZIP-Archiv mit dem Namen `IhrNachname.IhrVorname.zip`, welches Sie auf Ilias hochladen. Führen Sie dazu in dem Verzeichnis, in dem Sie die Dateien bearbeitet haben, folgenden Befehl auf der Kommandozeile aus:

```
zip  IhrNachname.IhrVorname.zip  *.java
```

Aufgabe 1: Klassenmethoden 9 P

Die Datei `Rechtecke.java` enthält die folgenden drei Klassen:

- Die Klasse `Punkt` definiert einen Datentyp für einen Punkt in einem zweidimensionalen Raum mit einer x und einer y Koordinate.
- Die Klasse `Rechteck` definiert einen Datentyp für ein Rechteck, welches aus einem Namen (vom Typ `String`), dem linken oberen Eckpunkt (vom Typ `Punkt`) und zwei `int`-Werten für die Breite und die Höhe des Rechtecks besteht.
- Die Klasse `Rechtecke` beinhaltet eine `main`-Methode, in der ein Feld von $N = 10$ Objekten vom Typ `Rechteck` angelegt wird. Anschließend werden mehrere Klassenmethoden aufgerufen, die in der Klasse bereits angelegt, aber noch nicht implementiert sind. Zwei der Methoden geben einen Dummy-Wert zurück, welcher von Ihnen zu ersetzen ist.

Ergänzen Sie die Klasse `Rechtecke` nun wie folgt:

(a) Implementieren Sie die Klassenmethode `fuellen`: In der Methode soll das als Parameter übergebene Feld `rechtecke` in einer Schleife vollständig mit (neu anzulegenden) Instanzen der Klasse `Rechteck` gefüllt werden. Die Namen der Rechtecke sollen fortlaufend mit "R1", "R2", "R3", ... benannt werden. Die x und y Koordinaten für die Rechtecke sollen Zufallswerte aus dem Intervall $[0, 9]$ und die Werte für die Breite und die Höhe der Rechtecke sollen ganzzahlige Zufallswerte aus dem Intervall $[1, 50]$ sein. Verwenden Sie die Methode `Math.random` um passende Zufallswerte zu generieren.

(b) Implementieren Sie die Klassenmethoden `groesse` und `ausgeben`: Die Klassenmethode `groesse` soll den Flächeninhalt eines als Parameter übergebenen Rechtecks berechnen und zurückgeben. In der Methode `ausgeben` soll der Inhalt des als Parameter übergebenen Felds `rechtecke` von links nach rechts und wie in folgendem Beispiel nummeriert von 1 - N (hier $N = 5$) ausgegeben werden. Verwenden Sie hierzu die Attribute der Klasse und berechnen Sie die Fläche mit Hilfe der Methode `groesse`.

```
1. Rechteck
   Ecke:     (7|5)
   Groesse:  48 x 50 = 2400
---------------
2. Rechteck
   Ecke:     (4|0)
   Groesse:  49 x 43 = 2107
---------------
3. Rechteck
   Ecke:     (7|6)
   Groesse:  14 x 47 = 658
---------------
4. Rechteck
   Ecke:     (7|4)
   Groesse:  50 x 32 = 1600
---------------
5. Rechteck
   Ecke:     (7|8)
   Groesse:  24 x 7 = 168
---------------
```

(c) Nun sollen die Rechtecke noch aufsteigend nach ihrer Größe in dem Feld sortiert werden. Hierbei sollen Sie folgendem, einfachen Sortieralgorithmus implementieren:

Bei einem Feld der Länge N wird N-1 Mal das kleinste Element in einem (schrumpfenden) Teilbereich des Felds gesucht. Dieses Element wird dann jeweils mit dem ersten Element in diesem Bereich vertauscht. Beim ersten Durchlauf umfasst der Teilbereich das ganze Feld, im zweiten Durchlauf wird der Bereich vom 2. Element bis zum Ende betrachtet, im dritten Durchlauf der Bereich vom 3. Element bis zum Ende usw. Im letzten Durchlauf werden nur noch die beiden letzten Elemente betrachtet.

Gehen Sie wie folgt vor:

- Implementieren Sie die Methode `findeIndexVonMin`. Diese Methode soll den Index des kleinsten Rechtecks in dem als Parameter übergebenen Feld bestimmen und zurückgeben. Hierbei sollen

aber nur die Rechtecke in einem Teilbereich des Felds berücksichtigt werden, welcher on einem Anfangs-Index bis zum Ende des Felds reicht. Der Wert dieses Index ist durch den Methoden-Parameter `vonIndx` gegeben. Verwenden Sie die Methode `groesse` um jeweils die Größen der Rechtecke zu berechnen.

- Implementieren Sie nun die Methode `sortieren`. Diese soll mit Hilfe der Methode `findeIndexVonMin`in einer Schleife im i-ten Durchlauf jeweils die Position (den Index) des kleinsten Rechtecks in dem Teilbereich des Felds bestimmen, der vom i-ten Element bis zum Ende des Felds reicht. Falls der Index nicht identisch zu dem Index des ersten Elements in dem jeweiligen Bereich ist, soll das Rechteck an dieser Position mit dem ersten Rechteck in dem Bereich vertauscht werden.

Die Ausgabe für das sortierte Feld sollte dann für das Beispiel oben wie folgt aussehen:

```
1. Rechteck
   Ecke:     (7|8)
   Groesse:  24 x 7 = 168
----------------
2. Rechteck
   Ecke:     (7|6)
   Groesse:  14 x 47 = 658
----------------
3. Rechteck
   Ecke:     (7|4)
   Groesse:  50 x 32 = 1600
----------------
4. Rechteck
   Ecke:     (4|0)
   Groesse:  49 x 43 = 2107
----------------
5. Rechteck
   Ecke:     (7|5)
   Groesse:  48 x 50 = 2400
----------------
```

Aufgabe 2: Klassenmethoden, Überladen 4 P

In dieser Aufgabe sollen Sie eine vorgegebene Methode zum Aneinanderhängen von Strings durch zwei weitere Methoden überladen und die resultierenden Methoden dann verwenden um einige vorgegebene Zeichenketten zu konstruieren.
Hierzu ist in der Klasse `Overload` Folgendes vorgegeben:

(1) Die Methode `concat` hängt die beiden als Parameter übergebenen Strings aneinander und gibt den resultierenden String als Ergebnis zurück.

(2) In der `main`-Methode sind drei `String`-Variablen sowie eine `char`-Variable vorgegeben, mit deren Hilfe Sie später die vorgegebenen Zeichenketten generieren sollen.

Ergänzen Sie nun die Klasse `Overload` wie im Folgenden beschrieben:

(a) Überladen Sie die vorgegebene Methode `concat` mit einer Methode, die zwei Parameter vom Typ `int` und `String` hat. Die Methode soll einen String zurückgeben, der aus der n-fachen Wiederholung des als Parameter übergebenen Strings besteht. Dabei ist n der Wert des `int` Parameters. Sie dürfen davon ausgehen, dass dieser Wert größer oder gleich 0 ist. Verwenden Sie hier den + Operator für Strings, um den Ergebnis-String zu erstellen, aber keine Methoden aus der Klasse `String`. Beispiel: Für den Wert `n = 3` und den String "ab" soll die Methode den String "ababab" zurückgeben.

(b) Überladen Sie die `concat` Methode mit einer weiteren Methode, die ein Feld von Strings sowie eine Variable vom Typ `char` als Parameter besitzt. Die Methode soll einen String zurückgeben, der aus allen Elementen des Felds besteht, die von links nach rechts aneinandergehängt wurden, wobei zwischen je zwei aufeinanderfolgenden String-Elementen jeweils das als Parameter übergebene Zeichen stehen soll. Sie dürfen davon ausgehen, dass das als Parameter an die Methode übergebene Feld mindestens ein Element enthält.

Beispiel: Für das Feld {"ab","bc", "ca"} und das Zeichen 'X' sollte der String "abXbcXca" zurückgegeben werden.

(c) Ergänzen Sie die drei `System.out.println` Anweisungen in der `main`-Methode jeweils durch einen Ausdruck, dessen Ergebnis der String ist, welcher über der jeweiligen Anweisung angegeben ist. Sie dürfen hierzu in den drei Ausdrücken nur die Variablen `s1, s2, s3` und `c` sowie (verschachtelte) Aufrufe der überladenen `concat` Methode verwenden. Im dritten Ausdruck dürfen Sie zusätzlich ein Feld mit der der Anweisung `new String[] {...}`
erzeugen, um die in Teil (b) entwickelte Methode nutzen zu können. In den geschweiften Klammen müssen Sie (anstelle der Punkte) die entsprechenden Argumente eintragen. Hinweis: Diese dürfen wiederum Ausdrücke sein.

Sie dürfen hier auch nicht den + Operator für Strings verwenden.

Aufgabe 3: Klassenmethoden 6 P

Für diese Klasse sollen Sie zwei statische Methoden entwickeln und zu jeder dieser Methoden noch eine dazugehörige Testmethode, in der diese mit verschiedenen Daten getestet werden soll.
Ergänzen Sie die Klasse `Methoden` wie im folgenden beschrieben:

(a) Schreiben Sie eine Methode `mehrheit`, die drei boole'sche Variablen als Parameter besitzt, welche drei Stimmenabgaben (z.B. bei einer Wahl) repräsentieren sollen. In der Methode soll der boole'sche Wert ermittelt und zurückgegeben werden, der die mehrheitliche Meinung repräsentiert. Das bedeutet, dass die Methode den Wert `true` zurückgeben soll, wenn mindestens zwei der Stimmen den Wert `true` haben und ansonsten `false`.

Implementieren Sie die Methode `testMehrheit`, in der die Methode `mehrheit` getestet werden soll. Testen Sie die Methode `mehrheit` hierbei mit allen möglichen Kombinationen für die drei boole'schen Parameter der Methode. Legen Sie dazu ein Feld an, welches die Werte `true` und `false` beinhaltet und verwenden Sie eine Dreifachschleife, um alle möglichen Parameterkombinationen für die drei Stimmen zu durchlaufen. Geben Sie dabei alle Parameterkombinationen an, für die die Methode `mehrheit` `true` zurückliefert.

Die Ausführung der Test-Methode sollte somit folgende Ausgabe produzieren:

```
v1 = true  v2 = true  v3 = true
v1 = true  v2 = true  v3 = false
v1 = true  v2 = false  v3 = true
v1 = false  v2 = true  v3 = true
```

(b) Entwickeln Sie eine Methode `monoton`, der als einziger Parameter ein Feld von ganzen Zahlen übergeben wird. Die Methode soll überprüfen, ob die Folge der Zahlen, die sich ergibt, wenn man das Feld von links nach rechts ausliest, monoton steigend ist. Das bedeutet, dass jede Zahl kleiner oder gleich der nachfolgenden Zahl ist. Ist dies der Fall, soll die Methode den Wert `true` zurückgeben, ansonsten aber den Wert `false`.

Vervollständigen Sie die Methode `testMonoton`, in der die Methode `monoton` mit mehreren Folgen unterschiedlicher Länge getestet werden

soll. Hierzu sind in der Methode bereits vier eindimensionale Felder definiert. Definieren Sie ein zweidimensionales Feld vom Typ `int` und initialisieren Sie dieses mit den Feldern `a,b,c,d` und `e`. Durchlaufen sie das resultierende Feld in einer Schleife und rufen Sie die Methode `monoton` mit jedem einzelnen, darin enthaltenen Feld auf. Geben Sie das jeweilige Feld und das Ergebnis dazu jeweils aus. Für die in der Methode `test_b` vorgegebenen Felder sollte die Ausgabe dann wie folgt lauten:

```
Die Folge -1 0 1  steigt monoton.
Die Folge 2 2 2 2  steigt monoton.
Die Folge -1 -2 -3  steigt nicht monoton.
Die Folge 3 2  steigt nicht monoton.
Die Folge 0  steigt monoton.
```

Aufgabe 4: Instanzmethoden 4 P

In dieser Aufgabe sollen Sie eine Klasse vervollständigen, die einen einfachen Würfel repräsentiert und diese dann mit Hilfe einer weiteren Klasse testen.

(a) In der Klasse `Wuerfel` sind drei Attribute definiert: eine Klassenvariable, die maximale Augenzahl für den Würfel angibt und zwei Instanzvariablen, die dazu dienen, einem Würfel-Objekt eine (fortlaufende) Nummer und eine (Augen-) Zahl zuzuordnen. Ergänzen Sie die Klasse `Wuerfel` nun wie folgt:

- Schreiben Sie einen Konstruktor, der den (Augen-)Wert des Würfels mit dem Wert 1 initialisert und der dem Attribut `nr` eine (während eines Programmlaufs) eindeutige, fortlaufende Nummer zuordnet. Hierzu dürfen Sie eine Klassenvariable in der Klasse anlegen.
- Schreiben Sie eine Methode `print`, die eine Ausgabe für den Würfel generiert, welche die Nummer und den aktuellen Wert des Würfels umfasst und welche wie in folgendem Beispiel aussehen sollte:

  ```
  Wuerfel Nr. 1 -> 1
  ```

- Schreiben sie eine Methode `werfen`, die das Würfeln dadurch simuliert, dass der Wert des Würfels auf einen Zufallswert aus der Menge $\{1, 2, 3, \ldots, \texttt{max}\}$ gesetzt wird. Der Wert von `max` ist hierbei durch die Klassenvariable gegeben.

(b) Ergänzen Sie nun die Klasse `WuerfelTest` so, dass das vorgegebene Feld von Würfeln in einer Schleife mit n neuen Objekten der Klasse `Wuerfel` gefüllt wird (n ist dabei als lokale Variable vorgegeben). Geben Sie die Objekte dabei mit Hilfe der Methode `print` aus. Durchlaufen Sie dieses Feld nun dreimal. Werfen Sie alle Würfel in jedem Durchlauf einmal und geben Sie sie anschließend aus.

Bei einem Programmlauf sollte die Ausgabe dann wie in folgendem Beispiel aussehen.

```
Wuerfel Nr. 1 -> 1
Wuerfel Nr. 2 -> 1
Wuerfel Nr. 3 -> 1
---------------
```

```
1. Wuerfeln:
Wuerfel Nr. 1 -> 6
Wuerfel Nr. 2 -> 1
Wuerfel Nr. 3 -> 5
2. Wuerfeln:
Wuerfel Nr. 1 -> 4
Wuerfel Nr. 2 -> 5
Wuerfel Nr. 3 -> 2
3. Wuerfeln:
Wuerfel Nr. 1 -> 6
Wuerfel Nr. 2 -> 6
Wuerfel Nr. 3 -> 1
```