Projektblatt P4 (20 P)

Abgabe: Freitag 13. Oktober 2022, 12:00h

Entpacken Sie zunächst die Archiv-Datei `vorgaben-P04.zip`, in der sich neben mehreren Rahmendateien für die zu lösenden Aufgaben auch eine Datei mit der Hilfsklasse `Data` befindet. Ergänzen Sie alle Dateien mit Ausnahme der Hilfsklasse zunächst durch einen Kommentar, der Ihren Namen beinhaltet. Ergänzen Sie die Dateien dann durch Ihre Lösungen gemäß der Aufgabenstellung unten.
Alle Dateien mit der Endung `.java` sollten am Ende fehlerfrei übersetzt werden können.
Verpacken Sie die Dateien für Ihre Abgabe in einem ZIP-Archiv mit dem Namen `IhrNachname.IhrVorname.P04.zip`, die 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.P04.zip *.java
```

Aufgabe 1: Klassenmethoden 11 P

Ergänzen Sie die Klasse `Klassenmethoden` um die im Folgenden beschriebenen Klassenmethoden. Fügen Sie die Methoden jeweils nach dem Kommentar für den zugehörigen Aufgabenteil ein.

(a) Eine narzistische Zahl ist eine Zahl, deren Summe ihrer Ziffern, jeweils potenziert mit der Stellenanzahl der Zahl, wieder die Zahl selbst ergibt.

Beispiel 1: Die dreistellige Zahl 407 ist narzistisch, da gilt:

$$4^3 + 0^3 + 7^3$$

Beispiel 2: Die fünfstellige Zahl 54748 ist narzistisch, da gilt:

$$5^5 + 4^5 + 7^5 + 4^5 + 8^5$$

Sie sollen nun eine Methode schreiben, die prüft, ob eine Zahl $n > 0$ narzistisch ist. Gehen Sie hierzu wie folgt vor:

- Schreiben Sie eine Klassenmethode `anzStellen`, die als Parameter einen ganzzahligen Wert `n` besitzt, und die einen ganzzahligen Wert zurückgibt, der die Anzahl der Ziffern von `n` darstellt. Teilen Sie die Zahl hierzu solange durch 10 (Integer-Division), bis n den Wert 0 hat. Zählen Sie hierbei die Anzahl der benötigten Schleifendurchläufe.

  Sie dürfen davon ausgehen, dass der Methode nur positive Werte übergeben werden.

- Schreiben Sie eine Klassenmethode `istNarzistisch`, die als Parameter einen ganzzahligen Wert `n` besitzt und die einen boole'schen Wert zurückgibt, der angibt, ob die Zahl `n` narzistisch ist oder nicht. Bestimmen Sie zunächst mit Hilfe der Methode `anzStellen` die Anzahl der Stellen. Berechnen Sie dann mit Hilfe eines iterativen Ansatzes die Summe der Ziffern, jeweils potenziert mit der Stellenzahl. Verwenden Sie hierzu die Methode `Math.pow`.

  Wenn die Summe die Zahl `n` ergibt, soll die Methode den Wert `true` zurückgeben, ansonsten den Wert `false`.

(b) Nun sollen Sie eine Methode mit dem Namen `fehlendeZahl` schreiben, die drei Parameter besitzt: ein Feld mit ganzzahligen Werten sowie zwei ganzzahlige Werte a und b (mit $0 < a \leq b$) und einen ganzzahligen Wert zurückgibt, welcher dem kleinsten Wert in dem Intervall $[a, b]$ entspricht, der nicht in dem Feld enthalten ist. Sind alle Werte aus dem Intervall in dem Feld enthalten, soll die Methode den Wert -1 zurückgeben. Prüfen Sie hierzu nacheinander alle Werte aus dem Intervall $[a, b]$, ob diese in dem Feld enthalten sind. Falls Sie einen Wert finden, auf den das nicht zutrifft, geben Sie diesen als Ergebniswert zurück. Falls Sie alle Werte aus dem Intervall finden, sorgen Sie dafür, dass am Ende der Wert -1 zurückgegeben wird.

(c) Schreiben Sie eine Klassenmethode mit dem Namen `filterGerade`, die in einem als Parameter übergebenen Feld alle geraden Zahlen in ein neuen Feld speichert und dieses am Ende zurückgibt. Gehen Sie wie folgt vor:

(1) Legen Sie zunächst ein neues Feld in der gleichen Länge wie das als Parameter übergebene Feld an. Durchlaufen Sie dann das als Parameter übergebene Feld und speichern Sie alle geraden Zahlen von links nach rechts in dem neuen Feld. Hinweis: Verwenden Sie einen eigene Indexvariable für das neue Feld, die Sie nur bei Bedarf (wenn ein Element kopiert wird) erhöhen.

- Definieren Sie ein weiteres, neues Feld, in das genau die geraden Elemente passen und kopieren Sie diese von dem in Schritt (1) generierten und gefüllten Feld in das neue Feld. Geben Sie dieses Feld am Ende zurück.

Testen Sie die von Ihnen entwickelten Methoden jeweils durch den dazugehörigen Code in der `main`-Methode der Klasse. Hierzu müssen Sie die Blockkommentarzeichen um die jeweiligen Aufrufe entfernen.
Die Ausgabe sollte am Ende dann wie folgt aussehen:

```
Aufgabenteil (a)
   153  ist eine narz. Zahl mit 3 Stellen
  1634  ist eine narz. Zahl mit 4 Stellen
 54748  ist eine narz. Zahl mit 5 Stellen
548834  ist eine narz. Zahl mit 6 Stellen
```

```
Aufgabenteil (b)
Kleinste fehlende Zahl in dem Feld {} --> 1
Kleinste fehlende Zahl in dem Feld {1} --> 2
Kleinste fehlende Zahl in dem Feld {1, 3, 2, 5, 6, 7, 1} --> 4
Kleinste fehlende Zahl in dem Feld {4, 2, 1, 5, 6, 8, 2} --> 3
In dem Feld {7, 6, 5, 4, 3, 2, 1} fehlt keine Zahl

Aufgabenteil (c)
{} --> {}
{1} --> {}
{1, 2, 3, 4, 5, 6, 7, 8, 9} --> {2, 4, 6, 8}
```

Aufgabe 2: Klassenmethoden, Überladen 2 P

Die Klasse `Rechnen` definiert eine Methode `max`, die das Maximum zweier ganzzahliger Werte bestimmt und zurückgibt. Darüberhinaus enthält die `main` Methode zwei auskommentierte Anweisungen, in denen die Methode `max` mit drei bzw. vier Parametern aufgerufen wird.
Überladen Sie die vorgegebene Methode `max` in der Klasse mit zwei weiteren Versionen, die jeweils drei bzw. vier Parameter besitzen und das Maximum der drei bzw. vier Werte berechnen. Implementieren Sie diese beiden Methoden so, dass zur Bestimmung des Maximums jeweils nur die bereits vorgegebene Methode `max` verwendet wird, aber keine weiteren Vergleichsoperationen. Hinweis: Nutzen Sie verschachtelte Aufrufe der Methode `max` mit jeweils zwei Parametern.
Entfernen Sie die Zeilenkommentare vor den beiden Anweisungen in der `main`-Methode und testen Sie ihr Programm.

Aufgabe 3: Klassen, Objekte und Methoden 7 P

In dieser Aufgabe sollen Sie einen Datensatz zu allen Flüssen der Welt auswerten, die mindestens 1000 km lang sind. Hierzu sollen Sie zunächst eine Klasse mit dem Namen `Fluss` vervollständigen. Anschließend sollen Sie dann Instanzen dieser Klasse generieren, in einem Feld speichern und dieses Feld dann verwenden, um Informationen daraus zu gewinnen.

Die Klasse `Data` enthält hierzu Folgendes:

- Die Klassenvariable `flussDaten` referenziert ein zweidimensionales Feld, welches Daten zu allen Füssen der Welt beinhaltet, die eine Mindestlänge von 1000 km aufweisen. Jede Zeile in diesem Feld umfasst fünf `String`-Werte, die die Länge, den Namen, den Kontinent, auf dem sich der Fluss befindet, sowie das Quellgebiet und die Mündung des Flusses repräsentieren.
- Die Klassenmethode `toInt` wandelt einen `String` in einen `int`-Wert um und gibt diesen zurück. Repräsentiert der String, der der Methode beim Aufruf übergeben wird, keine ganze Zahl, wird das Programm mit einer Fehlermeldung beendet.

In der Datei `Fluesse.java` sind die folgenden beiden Klassen (teilweise) vorgegeben:

- Die Klasse `Fluss` besitzt fünf Instanzvariablen, welche die Länge des Flusses, den Namen, den Kontinent, das Quellgebiet und die Mündung angeben. Dazu ist eine Methode mit dem Namen `aufKontinent` definiert, die prüft, ob der Fluss sich auf dem Kontinent befindet, dessen Name als Parameter übergeben wird.
- Die Klasse `Fluesse` beinhaltet eine Klassenvariable `fluesse`, in der später Instanzen der Klasse `Fluss` gespeichert werden sollen, sowie eine von Ihnen später zu ergänzende `main`-Methode.

Ergänzen Sie diese beiden Klassen nun wie folgt:

(a) Fügen Sie der Klasse `Fluss` zunächst einen Konstruktor mit einem `int`- und vier `String`-Parametern hinzu. Nutzen Sie die Parameterwerte um die Attribute der Klasse zu initialisieren.

(b) Ergänzen Sie die Klasse `Fluss` durch eine Instanzmethode mit dem Namen `toString`, die einen Parameter vom Typ `int` besitzt. Die Methode soll einen String zurückgeben, der später für die Ausgabe einer `Fluss`-Instanz verwendet werden kann. Implementieren Sie die Methode so, dass ein String konstruiert und zurückgegeben wird, der zu Beginn aus einer vorgegebenen Anzahl von Leerzeichen besteht, die durch den ganzzahligen Parameter der Methode speifiziert wird. Ist der Wert des

Parameters negativ, sollen keine Leerzeichen in den Ergebnisstring eingefügt werden. Ergänzen Sie den Ergebnisstring noch, wie im folgenden Beispiel angegeben, durch den Namen und die Länge des Flusses und geben Sie den String dann als Ergebnis zurück.

```
Wolga (3530)
```

(c) Ergänzen Sie die `main`-Methode der Klasse `Fluesse` durch eine Schleife, in der das Feld `fluesse` mit Instanzen der Klasse `Fluss` gefüllt wird. Verwenden Sie hierzu das Feld `flussDaten` aus der Klasse `Data`, welches selbst eine Klassenvariable mit öffentlichem Zugriff darstellt, sowie den Konstruktor der Klasse `Fluss` aus Aufgabenteil (b).

(d) Ergänzen Sie nun die `main`-Methode der Klasse `Fluesse` noch wie folgt:

- Legen Sie ein Feld von `String` Werten an, welche Sie mit den Namen der Kontinente (*Afrika, Asien, Australien, Europa, Nordamerika* und*Südamerika*) füllen.
- Sie sollen nun für jeden Kontinent die drei längsten Flüsse in der Konsole ausgeben. Da die Einträge in dem Feld `fluesse` absteigend nach der Länge sortiert sind, entspricht dies den ersten drei Flüssen eines Kontinents, die in dem Feld zu finden sind, wenn Sie dieses von links nach rechts durchlaufen.

  Durchlaufen Sie daher für alle Kontinente das Feld `fluesse` und geben Sie den Namen des jeweiligen Kontinents sowie der ersten drei Flüsse auf diesem Kontinent so formatiert aus, dass beim Ausführen des Programms folgende Ausgabe erzeugt wird:

```
Afrika:
    Nil mit Kagera-Nil(6852 km)
    Kongo mit Luvua, Luapula und Chambeshi(4835 km)
    Niger(4184 km)
Asien:
    Jangtsekiang mit Tongtian He(6380 km)
    Jenissei mit Angara, Selenga und Ider(5540 km)
    Ob mit Irtysch(5410 km)
Australien:
    Murray River mit Darling River, Culgoa, Balonne und Condamine(3672 km
```

```
    Darling River(2844 km)
    Murray River(2508 km)
Europa:
    Wolga(3530 km)
    Donau mit Breg(2857 km)
    Dnepr(2285 km)
Nordamerika:
    Mississippi mit Missouri, Jefferson und Red Rock River(6051 km)
    Mackenzie mit Slave, Peace und Finlay(4260 km)
    Missouri mit Jefferson und Red Rock River(4130 km)
Südamerika:
    Amazonas mit Ucayali, Tambo, Ene und Apurimac(6448 km)
    Parana mit Rio Grande(3998 km)
    Rio Madeira mit Rio Mamora und Rio Grande(3380 km)
```

Verwenden Sie hierbei die Instanzmethode `toString` um die Ausgabe für einen einzelnen Fluss zu generieren. Es sollen dabei jedem Fluss jeweils 4 Leerzeichen vorangestellt werden. Sie dürfen nicht voraussetzen, dass für jeden Kontinent mindestesns drei Flüsse in dem Feld `fluesse` vorhanden sind.

Hinweis: Alle Methoden, die keinen expliziten Ergebniswert zurückgeben, müssen den formalen Rückgabetyp `void` haben.