Projektblatt P3 (22 + 3* P)

Abgabe: Freitag 6. Oktober 2023, 12:00h

Entpacken Sie zunächst die Archiv-Datei `vorgaben-p03.zip`, in der sich neben mehreren Rahmendateien für die zu lösenden Aufgaben auch die Hilfsklasse `ImageTools` sowie drei Bilder (in dem Unterverzeichnis `bilder`) befinden. Ergänzen Sie alle Dateien mit Ausnahme der Hilfsklasse und der Bilder 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.P03.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.P03.zip *.java
```

Aufgabe 1: Eindimensionale Felder 12 (3 + 3 + 6)P

Die in der Datei `Aufgabe_3_1` definierte Klasse enthält folgende, z.T. noch leere Methoden:

- Die Methode `randomInt` gibt eine Zufallszahl aus dem Intervall $[a, e]$ zurück, wobei `a` und `e` zwei ganzzahlige Werte sein müssen, für die gilt: $a < e$
- Die Methode `fuellen` füllt ein als Parameter übergebenes Feld mit Zufallszahlen aus dem Intervall $[a, e]$
- Die Methode `ausgeben` gibt den Inhalt eines eindimensionalen Felds von links nach rechts in einer Zeile in der Konsole aus. Nach je 10 ausgegebenen Zahlen wird ein Zeilenumbruch eingefügt.
- Die Methoden `verteilen`, `histogramm` und `sumTwoSmallestPosNums` sind von Ihnen zu implementieren (siehe Aufgabenstellung Teil (a) - (c)).

In der `main`-Methode werden darüberhinaus verschiedene Felder generiert, mit deren Hilfe die von Ihnen zu implementierenden Methoden getestet werden.

(a) Implementieren Sie zunächst die Methode `verteilen`, in der Sie ein neues Feld erzeugen sollen, in dem die Werte aus einem vorgegebenen Feld so eingefügt werden, dass alle negativen Zahlen im linken Bereich und alle postiven Zahlen sowie 0 im rechten Bereich liegen. Gehen Sie hierzu wie folgt vor:

- Initialisieren Sie die vorgegebene Variable `feld2` mit einem Feld, das die gleiche Größe wie das als Parameter übergebene Feld hat.
- Um das neue Feld von links mit negativen bzw. von rechts mit positiven Zahlen und der 0 füllen zu können, definieren Sie zwei Indexvariablen, die Sie mit dem Index für das erste bzw. letzte Element initialisieren.
- Durchlaufen Sie dann das als Parameter übergebene Feld: Fügen Sie jeden negativen Wert an der Stelle in das neue Feld ein, die durch die erste Indexvariable definiert wird,und jeden positiven

Wert sowie 0 an der Stelle, die durch die zweite Indexvariable definiert wird. Aktualisieren Sie die jeweilige Indexvariable so, dass negative Werte von links nach rechts und positive Werte sowie 0 von rechts nach links eingefügt werden

- Die Rückgabe des neuen Felds am Ende der Methode ist bereits vorgegeben.

(b) Implementieren Sie nun die Methode `histogramm`, in der ein einfaches Histogramm[1] in der Konsole "gezeichnet" werden soll, welches die Häufigkeiten von ganzzahligen Werten aus einem Intervall $[min, max]$ in einem vorgegebenen Feld wiedergibt.

Beispiel: Für das Feld mit dem Inhalt $\{3, 2, 1, 4, 4, 4, 1, 5, 2, 5, 4, 5, 4, 2, 5\}$ und die Werte $min = 1$ und $max = 5$ sollte die Ausgabe am Ende so aussehen:

```
1:**
2:***
3:*
4:*****
5:****
```

Gehen Sie wie folgt vor:

- Legen Sie für das Histogramm ein Feld von ganzen Zahlen an, welches für alle Werte in dem Intervall $[min, max]$ die Häufigkeiten der Werte in dem als Parameter übergebenen Feld (mit dem Namen `feld`) speichern soll. Hinweis: Das Feld muss hierzu $max - min + 1$ Elemente haben.
- Durchlaufen Sie nun das Feld `feld`. Erhöhen Sie für jeden Wert in dem Feld den zugehörigen Eintrag in dem Feld für das Histogramm um 1. Berechnen Sie jeweils den korrekten Indexwert für den Eintrag: Hierbei soll das erste Element in dem Feld (Index 0) die Häufigkeit des Werts `min` angeben, das zweite Element die Häufigkeit des Werts `min+1` usw.
- Zeichnen Sie nun das Histogramm in der Konsole: Durchlaufen Sie hierzu das Feld für das Histogramm und geben Sie für jeden

[1] siehe z.B. https://de.wikipedia.org/wiki/Histogramm

Eintrag in dem Feld zunächst den zugehörigen Wert aus. Für den Häufigkeitswert geben Sie eine entsprechende Anzahl von '*' Zeichen aus.

(c) Nun sollen Sie die Summe der beiden kleinsten positiven Zahlen in einem vorgegebenen Feld bestimmen. Sie dürfen hierbei davon ausgehen, dass dieses Feld mindestens zwei positive Zahlen enthält. Ergänzen Sie hierzu die Methode `sumTwoSmallestPosNums` wie im Folgenden beschrieben:

- Legen Sie zwei ganzahlige Variablen `v1` und `v2` an, denen Sie die ersten beiden positiven Werte in dem als Parameter übergebenen Feld (mit dem Namen `feld`) zuweisen. Deklarieren Sie hierzu eine Indexvariable, die Sie mit 0 initialisieren und die Sie dann solange inkrementieren, bis Sie das erste positive Feldelement gefunden haben, welches Sie der Variablen `v1` zuweisen. Inkrementieren Sie die Indexvariable dann weiter solange, bis die das zweite positive Feldelement gefunden haben, welches Sie der Variablen `v2` zuweisen.
- Weisen Sie den beiden vorgegebenen Variablen `m1` und `m2` das Minimum bzw. Maximum von `v1` und `v2` zu.
- Durchlaufen Sie dann das vorgegebene Feld, beginnend an der Position direkt nach der Position des zweiten positiven Elements in dem Feld. Vergleichen Sie jedes Element dann zunächst mit `m1`: Wenn das Vergleichselement kleiner ist als `m1`, weisen Sie den Wert von `m1` der Variablen `m2` zu. Danach sollte `m1` der Wert des Vergleichselements zugewiesen werden. Ist das Vergleichselement dagegen nicht kleiner als `m1`, vergleichen Sie es mit `m2`. Falls der Wert dieses Elements kleiner als `m2` ist, ersetzen Sie den Wert von `m2` durch den Wert des aktuell betrachteten Feldelements.

Die Ausgabe nach einem Programmlauf sollte dann wie in folgendem Beispiel aussehen Durch die Zufallswerte in den Feldern für die Aufgabenteile (a) und (b) sieht die Ausgabe natürlich bei jedem Mal etwas anders aus.

```
Aufgabe 1a:

 39 -34  76 -84 -79 -61 -59 -50  96  55
-71 -51 -89  77  64  93 -20 -86  43  33

-34 -84 -79 -61 -59 -50 -71 -51 -89 -20
-86  33  43  93  64  77  55  96  76  39


Aufgabe 1b:

  2   3   4   9   2   9   8   8   4   4
  5   8   2   7   3   9   7   4   3   3

1:
2:***
3:****
4:****
5:*
6:
7:**
8:***
9:***

Aufgabe 1c:

 19   5  42   2  77
   ---> Summe: 7
  2   9   6  -1
   ---> Summe: 8
 -1  -1   1   1
   ---> Summe: 2
3683 2902 3951 -475 1617 -2385
   ---> Summe: 4519
```

Aufgabe 2: Zweidimensionale Felder 3 + 3* P

In dieser Aufgabe sollen Sie Unterschiede zwischen je zwei Grauwertbildern bestimmen und diese dann nutzen um herauszufinden, wo sich ein bewegtes Objekt in den einzelnen Bildern befindet. Hierzu müssen Sie jeweils die Bildpunktmatrizen dieser Bilder analysieren und die Ergebnisse in neuen Bildern speichern. Für das Laden der Bilder und das Speichern der Ergebnisbilder werden die Methoden `getImageDataFromFile` und `writeImageDataToFile` aus der vorgegebenen Klasse `ImageTools` (in der Datei `ImageTools.java`) verwendet. Diese Methoden liefern entweder eine Bildpunktmatrix als Ergebnis oder verwenden eine solche, um diese in eine Bildstruktur einzubetten, welche dann in eine Datei geschrieben wird. Die erforderlichen Aufrufe sind in der Klasse `Aufgabe_3_2` bereits vorgegeben. Sie müssen hier die beiden Methoden `diff` und `detectMovingObject` vervollständigen.

(a) Zunächst sollen Sie die Methode `diff` implementieren, der die Bildpunktmatrizen `img1` und `img2` von zwei Bildern übergeben werden. Sie sollen daraus die Bildpunktmatrix eines neuen Bilds erstellen, welches alle signifikanten Unterschiede zwischen den beiden Bildern enthält. Sie dürfen hierbei davon ausgehen, dass die Bildpunktmatrizen `img1` und `img2` die gleiche Breite und Höhe haben und alle Zeilen die gleiche Anzahl Spalten besitzen.

Initialisieren Sie zunächst die beiden vorgegebenen Variablen `w` und `h` mit der Breite und der Höhe des ersten Bilds, indem Sie die Werte aus der Anzahl der Zeilen der ersten Bildpunktmatrix sowie der Länge der ersten Zeile in dieser Matrix ablesen. Initialisieren Sie damit die Bildpunktmatrix des vorgegebenen Ergebnisbilds `diffImg`, die in der Vorgabe noch mit null initialisiert wird. Alle Werte in der Bildpunktmatrix sollten dann zunächst den Wert 0 haben.

Bestimmen Sie dann für alle Zeilen und Spalten den Absolutbetrag der Differenz der beiden jeweiligen Bildpunkte aus `img1` und `img2`. Hierzu können Sie die Methode `Math.abs` verwenden. Wenn dieser Wert größer ist als 50, setzen Sie den Wert des Bildpunkts in der Matrix des Ergebnisbilds auf den Differenzbetrag.

(b) Möchte man in mehreren Bilder einer Bildfolge, in denen ein unbewegter Hintergrund und ein (gleichmäßig) bewegtes Objekt zu sehen

sind, die Bewegung dieses Objekts bestimmen, muss man das bewegte Objekt in jedem Einzelbild identifizieren.

Eine sehr einfache Methode (die auch nur unter bestimmten Randbedingungen funktioniert), besteht darin, Differenzbilder zu berechnen, in denen nur die Regionen markiert sind, in denen Unterschiede zwischen zwei Bildern detektiert wurden.

Allerdings ist in den einzelnen Differenzbildern nicht erkennbar, welche Regionen das bewegte Objekt kennzeichnen und welche Regionen nur freigewordenen Hintergrund darstellen.

Hierzu muss man mehr als zwei Bilder berücksichtigen. Betrachtet man beispielsweise drei Bilder b1, b2 und b3, in denen sich das Objekt jeweils signifikant bewegt und berechnet die Differenzen zwischen b1 und b2 sowie b1 und b3, kann man über eine logische Kombination der Ergebnisse auf die Regionen zurückschließen, die das Objekt beinhalten. Betrachten Sie hierzu die drei folgenden Bilder aus einem Labor, in denen auf der rechten Seite eine Armbewegung zu sehen ist.

Die folgenden beiden Bilder zeigen die Differenzbilder, die aus dem Vergleich des ersten und zweiten bzw. ersten und dritten Bilds mit Hilfe der Methode `diff` entstanden sind.

Implementieren Sie nun die Methode `detectMovingObject`, welche die Bildpunktmatrizen von drei Bildern einer Bildfolge als Parameter hat

und in der drei Ergebnisbilder generiert und mit Hilfe der Methode `writeImageDataToFile` in drei Dateien gespeichert werden sollen, in denen jeweils die Region weiß (Bildpunkte mit dem Wert 255) markiert ist, in der sich das bewegte Objekt befindet.

Gehen Sie hierzu wie folgt vor:

- Initialisieren Sie zunächst die beiden vorgegebenen Variablen `w` und `h` mit der Breite und der Höhe des ersten Bilds. (siehe Aufgabenteil (a)). Legen Sie dann die Bildpunktmatrizen `resImg1`, `resImg2` und `resImg3` für die Ergebnisbilder an.
- Generieren Sie nun zeilen- und spaltenweise die Ergebnisbilder, indem Sie diejenigen Bildpunkte in den Bildpunktmatrizen `resImg1`, `resImg2` und `resImg3` auf 255 setzen, die das sich bewegende Objekt (den Arm) im ersten, zweiten bzw. dritten Bild kennzeichnen.
  Hinweis: Kombinieren Sie die Informationen aus den Differenzbildern.
  (1) Nur die Bildpunkte, die in beiden Differenzbildern einen Wert größer 0 haben, kennzeichnen das Objekt im ersten Bild
  (2) Nur die Bildpunkte, die im ersten (bzw. zweiten) Differenzbild einen Wert ungleich 0 haben und im zweiten (bzw. ersten) Differenzbild einen Wert gleich Null, kennzeichnen das Objekt im zweiten (bzw. dritten) Bild.

Die Ergebnisbilder sollten dann wie folgt aussehen:

Aufgabe 3: 7 (2+2+3) P

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

- Die Klasse `Punkt` definiert einen Datentyp für einen Punkt in einem zweidimensionalen Raum.
- Die Klasse `Kreis` definiert einen Datentyp für einen Kreis, der aus einem Mittelpunkt (vom Typ `Punkt`) und einem Radius besteht.
- Die Klasse `Kreise` definiert eine Methode, mit der ganzzahlige Zufallszahlen aus dem Intervall $[1, n]$ mit $(0 < n)$ generiert werden können. In der `main`-Methode ist eine ganzzahlige Konstante $N = 10$ definiert sowie eine Variable `kreise`, die ein Feld von Kreis-Objekten referenziert.

Ergänzen Sie die Klasse `Kreise` wie folgt:

(a) Ergänzen Sie zunächst die `main`-Methode der Klasse, indem Sie die Variable `kreise` anstatt mit dem Wert `null` mit einem Feld von N Objekten vom Typ `Kreis` initialisieren. Generieren Sie dann in einer Schleife N Kreise, mit denen Sie das Feld füllen. Initialisieren Sie dabei die (numerischen) Attribute der Kreise jeweils mit Hilfe der Methode `randomInt`. Die x- und die y-Koordinate des Mttelpunkts sollten dabei im Intervall $[1, 50]$ liegen und der Radius im Intervall $[1, 20]$.

(b) Vervollständigen Sie die Methode `ausgeben`, indem Sie den Inhalt des Felds `kreise`, nummeriert von 1 - N, wie in folgendem Beispiel (im Beispiel: $N = 3$) in der Konsole ausgeben:

```
1. Kreis:
   Mittelpunkt: (21|35)
   Radius: 7
2. Kreis:
   Mittelpunkt: (35|43)
   Radius: 12
3. Kreis:
   Mittelpunkt: (13|20)
   Radius: 4
```

(c) Nun sollen die Kreise in dem Feld noch aufsteigend nach dem Radius sortiert und anschließend nochmal ausgegeben werden.

Implementieren Sie hierzu in der Methode `sortieren` folgenden einfachen Sortieralgorihmus:

Bei n zu sortierenden Elementen muss der folgende Schritt $n-1$ Mal ausgeführt werden:
Beim ersten Feldelement beginnend, werden je zwei aufeinanderfolgende Elemente i und $i+1$ verglichen. Ist das Element an der Stelle i größer als das Element an der Stelle $i+1$, werden die Inhalte der Feldelemente i und $i+1$ vertauscht.