

Übungsblatt 4: Software-Entwicklung 1 (WS 2017/18)

Ausgabe: 13.11.17
Abgabe: 20.11.17

Aufgabe 1 Arrays und Prozeduren (11 Punkte)

Laden Sie sich für diese Aufgabe die Vorlage `ArraysAndProcedures.java` herunter. Diese enthält bereits Vorlagen für die zu implementierenden Prozeduren.

Zum Testen können Sie die `main`-Prozedur anpassen und mit der Bibliotheksfunktion `Arrays.toString(ar)` die Ergebnisse ausgeben. Verwenden Sie für diese Aufgabe sonst keine weiteren Bibliotheksfunktionen.

- a) Schreiben Sie eine Prozedur `replaceAll`, welche zwei `int` Werte `x` und `y` sowie ein Array von `int`-Werten nimmt. Die Prozedur soll dann alle Vorkommen von `x` im gegebenen Array durch `y` ersetzen.

Beispiel:

```
int[] ar = {1, 2, 3, 2, 4};
replaceAll(2, 7, ar);
System.out.println(Arrays.toString(ar)); // Gibt "[1, 7, 3, 7, 4]" aus
```

- b) Schreiben Sie zwei weitere Prozeduren wie `replaceAll`. Eine Variante `replaceFirst`, die nur das erste Vorkommen im Array ersetzt und eine andere `replaceLast`, die nur das letzte Vorkommen im Array ersetzt.
- c) Schreiben Sie eine Prozedur `substAll`, welche wie `replaceAll` funktioniert, jedoch das Eingabe-Array nicht verändert, sondern ein neues Array mit dem Ergebnis zurückliefert.

Beispiel:

```
int[] ar = {1, 2, 3, 2, 4};
int[] ar2 = substAll(2, 7, ar);
System.out.println(Arrays.toString(ar)); // Gibt "[1, 2, 3, 2, 4]" aus
System.out.println(Arrays.toString(ar2)); // Gibt "[1, 7, 3, 7, 4]" aus
```

- d) Schreiben Sie eine Prozedur `onlyEven`, welche ein Array von `int`-Werten nimmt und ein neues Array zurückgibt, in dem sich nur die geraden Zahlen aus dem ursprünglichen Array befinden.

Beispiel:

```
int[] a1 = {1, 3, 2, 4, 7, 2};
int[] a2 = onlyEven(a1);
System.out.println(Arrays.toString(a1)); // Gibt "[1, 3, 2, 4, 7, 2]" aus
System.out.println(Arrays.toString(a2)); // Gibt "[2, 4, 2]" aus
```

- e) Schreiben Sie eine Prozedur `allHaveZero`, welche ein Array von `int`-Arrays nimmt und prüft, ob alle Arrays die Zahl 0 enthalten.

Beispiel:

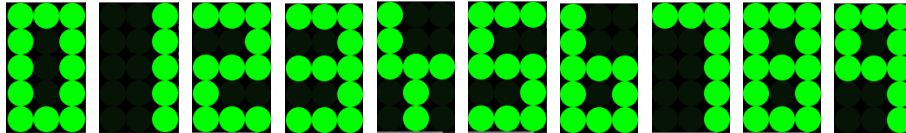
```
int[][] a1 = {{1,2,0}, {0,7}, {7,0,7}};
int[][] a2 = {{1,2,0}, {4,5,6}};
System.out.println(allHaveZero(a1)); // Gibt "true" aus
System.out.println(allHaveZero(a2)); // Gibt "false" aus
```

Aufgabe 2 2D-Arrays (11 Punkte)

In dieser Aufgabe soll ein Programm entwickelt werden, welches Zahlen mit bis zu drei Stellen in ein geeignetes Format für eine LED-Matrix-Anzeige umwandelt.

Bei einer LED-Matrix-Anzeige werden Ziffern durch eine Anzahl an Leuchtdioden angezeigt. Diese sind in einem Rechteck angeordnet. Durch das aktivieren und deaktivieren bestimmter LEDs kann das angezeigte Muster verändert werden. Wir gehen in dieser Aufgabe von einer Matrix von drei LEDs in der Breite und fünf LEDs in der Höhe für jede Ziffer aus.

Die Ziffern sollen auf der Anzeige wie folgt angezeigt werden:



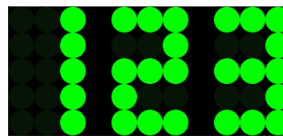
Die Matrixrepräsentation einer Zahl besteht aus einer Matrix mit 5 Zeilen und jeweils 9 Spalten aus 0 und 1. Ist eine Stelle gleich 1, so leuchtet die LED an dieser Stelle, ist die Stelle gleich 0 so ist die LED aus. Zur Anzeige stehen drei Ziffern zur Verfügung, sodass Zahlen zwischen 0 und 999 angezeigt werden können. Bei einstelligen und zweistelligen Zahlen sollen die ersten Ziffern dunkel bleiben.

Aufgabe ist es, ein Programm `ToMatrix` zu schreiben, welches Zahlen von der Standardeingabe liest bis diese leer ist und für jede dieser Zahlen die Matrixrepräsentation auf die Standardausgabe ausgibt. Die Matrix der nächsten Zahl folgt direkt darunter (ohne Trennung).

Beispiel:

Die Matrixrepräsentation der Zahl 123 ist auf der linken Seite dargestellt. Diese wird wie rechts zu sehen auf der LED-Matrix angezeigt:

```
001111111
001001001
001111111
001100001
001111111
```



Hinweise:

- Zum Aufteilen der eingelesenen Zahl in Ziffern sind die Modulo Operation (%) und die Integer-Division (/) hilfreich.
- Überlegen Sie sich eine geeignete Einteilung in Prozeduren. Die Aufgabe, eine Ziffer an eine bestimmte Stelle in der Matrix zu schreiben kommt mehrfach mit verschiedenen Stellen vor.
- Zum Einlesen der Zahlen von der Standardeingabe können Sie die Bibliothek `StdIn` verwenden.
- Sie können das Programm `ShowMatrix` von der Vorlesungsseite verwenden, um die Anzeige zu kontrollieren. Dazu leiten Sie die Ausgabe Ihrer Implementierung von `ToMatrix` an das Programm `ShowMatrix` weiter durch `java ToMatrix | java ShowMatrix`.

Aufgabe 3 StdIn und StdOut (4 Punkte)

Laden Sie sich für diese Aufgabe die Bibliothek `StdIn.java` von der Vorlesungsseite herunter. Diese Bibliotheken unterstützen das Lesen von der Standardeingabe und das Schreiben von Ausgaben. Die Prozeduren, die Sie für diese Übung brauchen, finden Sie im Script (Kapitel 5, Abschnitt 5 “Eingabe und Ausgabe”).

Schreiben Sie eine Prozedur `static String[] readStrings()` in einer Datei `readStrings.java`, welche mit `StdIn.readString()` `String`-Werte von der Standardeingabe liest, bis die Standardeingabe leer ist und die Werte in einem Array speichert.

Hinweis: Arrays können in Java nach ihrer Erstellung nicht vergrößert oder verkleinert werden. Legen Sie deshalb zu Beginn der Prozedur ein Array der Größe 5 an und erstellen Sie ein neues Array der doppelten Größe, sobald das aktuelle Array zu klein wird. Kopieren Sie dann am Ende der Prozedur die Werte in ein Array der passenden Größe.

Aufgabe 4 Formatierte Ausgabe (3 Punkte)

Die Prozedur `StdOut.printf` kann zur formatierten Ausgabe verwendet werden. Als ersten Parameter nimmt die Funktion einen String mit Platzhaltern für Werte. Als weitere Parameter folgen dann die Werte, die für die Platzhalter eingesetzt werden. Platzhalter beginnen mit einem Prozentzeichen und enden mit einem Buchstaben, welcher beschreibt, wie der Wert umgewandelt werden soll. So steht zum Beispiel `%d` für die Dezimaldarstellung eines `int`-Wertes, `%f` für die Fließkommadarstellung einer Zahl, `%e` für die Darstellung einer Zahl in wissenschaftlicher Darstellung mit Exponent und `%s` für die String-Darstellung eines Wertes.

Beispiel:

```
StdOut.printf("In %s ist es %d Grad warm.", "Kaiserslautern", 3);
```

Bei Gleitkomma-Werten kann die Präzision festgelegt werden. Dies geschieht nach einem Punkt durch eine weitere Zahl, welche zum Beispiel die Anzahl der Nachkommastellen festlegt.

Es gibt noch einige weitere Funktionen für die formatierte Ausgabe `printf`. Die offizielle Dokumentation finden Sie unter:

<http://docs.oracle.com/javase/8/docs/api/java/util/Formatter.html#syntax>.

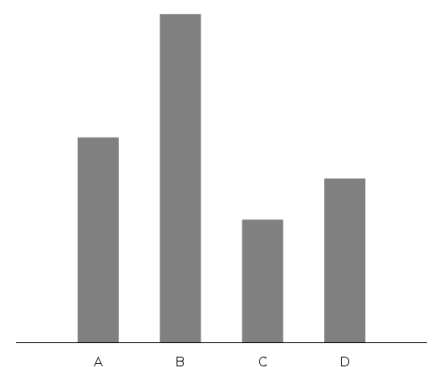
Schreiben Sie ein Programm `SqrtTable`, welches `double`-Werte von der Standardeingabe liest und nach jeder gelesenen Zahl die Zahl selbst und die Wurzel der Zahl, jeweils durch Komma getrennt in einer Zeile ausgibt. Die Zahlen sollen dabei mit zwei Nachkommastellen ausgegeben werden. Verwenden Sie hierzu die Prozeduren `StdIn.isEmpty`, `StdIn.readDouble`, `StdOut.printf`, und `Math.sqrt`.

Aufgabe 5 StdDraw (11 Punkte)

Geben Sie für diese Aufgabe eine einzelne Java-Datei `BarChart.java` ab. Verwenden Sie wenn möglich Prozeduren aus vorherigen Aufgabenteilen um Redundanz zu vermeiden.

Hinweis zur Bearbeitung: Es handelt sich bei dieser Aufgabe um ein etwas größeres Programm. Warten Sie nicht bis zum Ende um Ihr Programm zu testen. Sie sollten einzelne Prozeduren direkt nach dem Schreiben durch Aufrufe in der `main`-Prozedur testen.

Laden Sie sich für diese Aufgabe die Bibliothek `StdDraw.java` von der Vorlesungs-Seite herunter. Diese Bibliothek bietet Prozeduren zum Zeichnen von Bildern.



Das Zeichnen von Bildern basiert auf einem Koordinaten-System, wobei die Werte der x-Achse und y-Achse die Werte 0 bis 1 annehmen können. Der Punkt (0,0) ist die Ecke links unten und der Punkt (1,1) ist die Ecke rechts oben auf der Zeichenfläche.

Für diese Aufgabe sollen Sie die folgenden Prozeduren verwenden:

```
void text(double x, double y, String text)
void line(double x0, double y0, double x1, double y1)
void filledRectangle(double x, double y, double halfWidth, double halfHeight)
```

Um ein Gefühl für das Koordinaten-System zu bekommen, können Sie zuerst das folgende Beispiel verwenden und mit den Werten experimentieren:

```

public class GraphBeispiel {
    public static void main(String[] args) {
        StdDraw.text(0.1, 0.1, "0.1, 0.1");
        StdDraw.text(0.1, 0.9, "0.1, 0.9");
        StdDraw.text(0.9, 0.1, "0.9, 0.1");
        StdDraw.text(0.9, 0.9, "0.9, 0.9");
        StdDraw.line(0.3, 0.8, 0.9, 0.3);
        StdDraw.filledRectangle(0.4, 0.3, 0.3, 0.1);
    }
}

```

Legen Sie dazu die heruntergeladene Datei `StdDraw.java` in den gleichen Ordner und übersetzen Sie dann das Programm mit dem Java-Übersetzer. Anschließend können Sie das Programm ausführen und es sollte ein Fenster mit den gezeichneten Elementen angezeigt werden.

- a) Ändern Sie das Beispiel-Programm so ab, dass ein Balken-Diagramm gezeichnet wird, das etwa wie das oben gezeigte aussieht.

Die Lösung zu Teilaufgabe a) müssen Sie nicht abgeben. Geben Sie am Ende nur eine Datei mit dem kompletten Programm ab, wie es in Aufgabe d) gefordert ist.

- b) Schreiben Sie eine Prozedur `void drawBar(double x, double width, String text, double height)`, welche einen einzelnen Balken des Balken-Diagramms zeichnet. Dabei ist der Parameter `text` der Text, der unter dem Balken angezeigt wird, `x` ist die `x`-Koordinate für die Mitte des Balkens und `width` und `height` geben die Breite und Höhe des Balkens an. Schreiben Sie Ihr Programm so um, dass diese Prozedur zum Zeichnen des Diagramms verwendet wird.

- c) Schreiben Sie eine Prozedur `void drawBars(String[] labels, double[] heights, double width, double gap)`, welche ein Array von `String`-Werten und ein Array von `double`-Werten nimmt, die jeweils die Beschriftung und Höhe eines Balkens angeben. Außerdem nimmt die Prozedur die Breite der Balken `width` und den Abstand zwischen den einzelnen Balken `gap`. Sie können annehmen, dass es für jede Höhe ein Label gibt.

Die Prozedur soll das Diagramm mit den gegebenen Balken und Beschriftungen zeichnen, so wie es im obigen Bild zu sehen ist.

Verwenden Sie zur Implementierung die Prozedur `drawBar`.

- d) Schreiben Sie eine `main`-Prozedur für Ihr Programm.

Das Programm soll 2 Programm-Parameter nehmen:

1. Die Breite der Balken
2. Der Abstand zwischen den Balken

Die Werte und Beschriftungen für das Diagramm sollen von der Standardeingabe gelesen werden (bis die Eingabe leer ist). Dabei soll immer zuerst ein `String` gelesen werden, der die Beschriftung angibt und dann ein dazugehöriger `double`-Wert.

Das gezeichnete Diagramm soll entsprechend den gegebenen Parametern die Prozedur `drawBars` verwenden, um ein Balkendiagramm zu zeichnen.

Hinweis: Zum Einlesen der Daten können Sie die Prozedur `readStrings` aus [Aufgabe 3](#) verwenden. Zum Testen können Sie sich die Datei `wahlen.txt` von der Vorlesungsseite laden und Ihr Programm mit dem folgenden Befehl starten:

```
java BarChart 0.1 0.03 < wahlen.txt
```

- e) (Zusatzaufgabe) Verbessern Sie Ihr Programm so, dass die Breite der Balken und der Abstand zwischen den Balken automatisch sinnvoll berechnet werden, wenn sie nicht als Programm-Parameter angegeben werden.
- f) (Zusatzaufgabe) Normalisieren Sie die Eingabewerte so, dass der größte Balken eine Höhe von 80% des Bildschirms hat. Beim Normalisieren sollen alle Werte mit dem gleichen Faktor multipliziert werden.