Hochschule Bonn-Rhein-Sieg
Fachbereich Informatik
M.Sc. Andre Kless
Dipl. Inf. Sigrid Weil

Übung 7

Einführung in die
Programmierung
SoSe 2023

Themen:
Definition und Verwendung von Arrays, Zählschleifen mit `for`

- Verwenden Sie **nicht** die Klassen `ArrayList` oder `List`!
- Verwenden Sie **nicht** die `for-each`-Schleife!
- Verwenden Sie **keine** der sortier-, min- oder max-Methoden aus der Java-API!

Aufgabe 7.1
Welche Ergebnisse liefern die beiden Test-Methoden? Machen Sie durch ein Speicherbild klar, wie es zu den Ergebnissen kommt.

Dies ist *keine* Programmieraufgabe, sondern eine *Theorie-Aufgabe*! **Zeichnen Sie!**

```
public static boolean test1() {
     int[] a = {2, 4, 6};
     int[] b = {2, 4, 6};
     return (a == b);
}

public static boolean test2() {
     int[] a = {2, 4, 6};
     int[] b = a;
     b[0] = 5;
     return (a == b);
}
```

(weitere Aufgaben auf den folgenden Seiten)

Aufgabe 7.2
Schreiben Sie eine Java-Klasse `IntArrayUtil`, die die folgenden (statischen) Methoden zum Umgang mit int-Arrays bereitstellt: (Die mit (*)] gekennzeichneten Aufgabeteile sind optional.)

a) `toString()`: liefert einen String, der das übergebene Array in „lesbarer“ Form wiedergibt.

b) `areEqual()`: erhält zwei int-Arrays als Eingabe und prüft, ob diese Arrays gleiche Länge und (an allen Positionen) gleichen Inhalt haben.

c) `shuffle()`: mischt den Inhalt des übergebenen Arrays zufällig. Gehen Sie zum Mischen eines Arrays der Länge n wie folgt vor:
Lassen Sie eine Schleife $(n-1)$-mal durchlaufen. Innerhalb der Schleife lassen Sie im i-ten Durchlauf einen zufälligen Index aus dem Bereich (i-1) bis (letzte Position im Array) ziehen und tauschen Sie die Inhalte des Arrays an dieser Position mit dem Inhalt an der Position (i-1).
In jedem der Schleifendurchgänge kann es passieren, dass die gezogene Zahl x = i-1 ist und daher eigentlich kein Tausch nötig ist.

d) `createSequence()`: erhält eine natürliche Zahl n als Eingabe und erzeugt ein Array, das mit den Werten $0, 1, 2, \ldots (n-1)$ belegt ist (in dieser Reihenfolge). Falls die übergebene Zahl ≤ 0 ist, soll ein Array der Länge 0 erzeugt werden.

(*) `contains()`: erhält ein Array und einen int-Wert als Eingabe und prüft, ob der Wert in dem Array vorkommt.

(*) `containsMultiples()`: prüft, ob das Array „Duplikate“ enthält, also ob es (mindestens) einen Wert gibt, der mehrfach im Array vorkommt.

(*) `pos()`: erhält ein Array und einen int-Wert als Eingabe und gibt die Position des Wertes im Array zurück, falls der Wert im Array vorkommt. Falls der Wert mehrmals vorkommt, wird die Position des ersten Auftretens zurückgegeben, falls der Wert gar nicht vorkommt, wird -1 zurückgegeben.

(*) `putRandom()`: erhält ein Array und zwei int-Werte (`von` und `bis`) als Eingabe und und belegt dieses Array mit zufälligen int-Werten zwischen `von` (einschließlich) und `bis` (ausschließlich).

(*) `createRandom()` erhält eine natürliche Zahl als Eingabe und erzeugt ein Array von zufälliger Länge (maximal der übergebenen Zahl) und belegt es mit zufälligem Inhalt. Falls die übergebene Zahl ≤ 0 ist, soll ein Array der Länge 0 erzeugt werden.

(*) `max(), min(), sum()`: bestimmen den größten bzw. kleinsten der Werte bzw die Summe aller Werte des übergebenen Arrays.

(*) `isSorted()`: prüft, ob das übergebene Array aufsteigend sortiert ist (Duplikate sind erlaubt).

(*) `swap()`: erhält ein Array und zwei Positionsnummern als Eingabe und tauscht die Inhalte des Arrays an den angegebenen Positionen. (Falls die Positionsnummern „ungültige“ Positionen darstellen, macht die Methode nichts.)

(*) `merge()`: erhält zwei sortierte Arrays als Eingabe und erzeugt ein neues Array, das die Werte beider übergebenen Arrays in aufsteigend sortierter Reihenfolge enthält.

Sie können die Methoden dieser Klasse bei folgenden Aufgaben nutzen und auch um beliebige zusätzliche Methoden erweitern, wenn es Ihnen opportun erscheint.
Nutzen Sie ein statisches Random-Objekt zur Erzeugung von Zufallszahlen.
Hinweis:
Keine der Methoden erzeugt eine Bildschirm-Ausgabe.

Zulassungsaufgabe

Bitte laden Sie Ihre Lösung nur dann in LEA hoch, wenn Sie wirklich auf die Zulassung angewiesen sind oder vor dem Drittversuch stehen! Wir kommen sonst mit der Korrektur nicht hinterher. Nach Ende der Abgabefrist können wir die Aufgabe in den Übungen oder der Vorlesung besprechen.

Aufgabe 7.3
In einem Computerspiel können pro Spiel zwischen 0 und 99 Punkten erreicht werden.
An dem Spiel können mehrere Spieler teilnehmen; es wird in Runden gespielt, in jeder Runde kommt jeder Teilnehmer einmal dran.
Im Highscore werden die 10 besten Ergebnisse in absteigend sortierter Reihenfolge gespeichert.

Im Download-Bereich stehen die Klassen `Spieler, Ergebnis` und eine Klasse `TestHighscore` zur Verfügung, die Sie nutzen sollen. An den beiden Klassen `Spieler` und `Ergebnis` brauchen (und sollen!) Sie nichts ändern. Die Testklasse können Sie natürlich um weitere Tests ergänzen.

Ihre Aufgabe:
Erstellen Sie eine Klasse `Spiel` mit folgenden Eigenschaften:

a) Ein Spiel enthält als Instanzattribut ein Array von Spielern (das Teilnehmerfeld), das dem Konstruktor der Klasse übergeben wird.
Außerdem gibt es ein Array von Ergebnissen, in das die 10 besten Ergebnisse eingetragen werden (den Highscore). Dieses Array speichert den „ewigen“ Highscore, also die besten Ergebnisse über mehrere Spiele hinweg.

b) Jedes Spiel-Objekt verfügt über eine Methode `spielen()`. Der Methode wird die Anzahl der zu spielenden Runden als Parameter übergeben. Die Methode startet entsprechend viele Spielrunden.

c) In einer Methode `spielrunde` soll jeder Spieler des Teilnehmerfeldes einmal spielen.
Das erzielte Ergebnis wird auf dem Monitor angezeigt und - falls damit die TopTen erreicht wurden - in den Highscore eingetragen.

d) Der Methode `eintragen` wird ein Spiel-Ergebnis als Parameter übergeben. Falls das Ergebnis zu den bisher ereichten TopTen gehört, wird es in den Highscore an die richtige Position eingetragen. Alle schlechteren Ergebnisse rutschen um eine Position nach hinten bzw. fallen ganz aus dem Highscore heraus.

e) Eine Methode `ausgabe` gibt den aktuellen Highscore aus. Falls noch nicht mindestens 10 Spiele gespielt wurden, werden nur die bis dahin erreichten Ergebnisse ausgegeben.

Tipps und Hinweise:

- Laut Aufgabenstellung liegt es nahe, das Array für den Highscore als Array der Länge 10 anzulegen, da ja die jeweils 10 besten Ergebnisse gespeichert werden sollen. Es könnte aber nützlich sein, das Array um einen Eintrag zu verlängern.
- Es ist sinnvoll, das Highscore-Array von hinten nach vorne zu durchlaufen, um die richtige Position für das neue Ergebnis zu finden.
- Der Konstruktor und die Methoden `spielen()` und `ausgabe()` sind die einzigen Elemente, die nach aussen sichtbar sein müssen. Alle anderen Attribute und Methoden können/sollten Sie als private deklarieren.
- In der Datei `HighscoreTestErgebnis.txt` ist die Ausgabe eines Test-Durchlaufs angegeben.