Check und Prepare

Vorbereitende Übungen

TECHNISCHE UNIVERSITÄT DARMSTADT

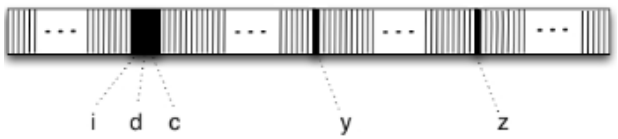
Autoren: Nhan Huynh und Darya Nikitina
Fachbereich: Informatik
Übungsblatt: 02
Version: 20. November 2021
Semesterübergreifend

V1 Referenzen

★★★

Geben Sie in eigenen Worten wieder, was man unter einer Referenz versteht.

Betrachten Sie außerdem folgendes Schaubild und den Codeausschnitt aus der Vorlesung:




```java
public class X {
  int i;
  double d;
  char c;
  ...
}
```

Zeichnen Sie die Referenzpfeile nach den folgenden Aufrufen ein (ergänzen Sie auch die neuen Reservierungen des Speicherplatzes, wenn nötig):

```java
X y = new X();
X z = y;
y = new X();
```

V2 Zuweisen und Kopieren

★★★

Erläutern Sie in Ihren eigenen Worten den Unterschied zwischen Zuweisen und Kopieren. In welchen Fällen sind beide Aktionen synonym zu betrachten?

Wie können Sie eine Zuweisung beziehungsweise eine Kopie in Java umsetzen? Nennen Sie jeweils ein Beispiel.

Übungsblatt 02: Vorbereitende Übungen von Nhan Huynh und Darya Nikitina

V3 Arrays

Welche Aussagen zu einem gegebenen Array `a` sind wahr?

(1) Alle Einträge des Arrays müssen vom selben Typ sein.

(2) Ein Array hat keine feste Größe und es können beliebig viele neue Einträge einem Array hinzugefügt werden.

(3) Um die Anfangsadresse einer Komponente an Index `i` zu bekommen, wird `i`-mal die Größe einer Komponente auf die Anfangsadresse von `a` addiert.

(4) Außer den eigentlichen Komponenten des Arrays enthält das Arrayobjekt nichts weiteres.

(5) Ein Array kann nur primitive Datentypen wie zum Beispiel `int`, `char` oder `double` speichern. Somit ist es insbesondere nicht möglich, Roboterobjekte in einem Array zu speichern.

Schreiben Sie die nötigen Codezeilen (auf Papier), um ein Array `a` der Größe `42` vom Typ `int` anzulegen. Füllen Sie danach das Array mithilfe einer Schleife, sodass an der Stelle `a[i]` der Wert `2i + 1` steht. Nutzen Sie dabei zuerst eine `while`- und danach eine `for`-Schleife.

V4 Wettrennen

Sie haben einen schnellen Roboter `rabbit` erstellt und wollen ihm nun noch ein langsames Gegenstück `turtle` bauen. Beide starten an einem gemeinsamen Punkt, schauen in die gleiche Richtung und besitzen die gleiche Anzahl an Coins. Sie wollen nun schauen, wer in `10` Runden mehr Strecke zurücklegen kann. Jeder der beiden Kontrahenten kommt pro Runde genau einen Schritt voran, der schnelle `rabbit` erhält jedoch in jeder zweiten Runde einen Extraschritt. Betrachten Sie den folgenden Codeausschnitt, der die Situation implementieren möchte:

```
1  Robot rabbit = new Robot(0, 0, RIGHT, 0);
2  Robot turtle = rabbit;
3
4  for (int i = 0; i < 10; i++) {
5    if (i / 2 == 0) {
6      rabbit.move();
7    }
8    rabbit.move();
9    turtle.move();
10 }
```

Führen Sie den Code einmal selbst in Eclipse aus und schauen Sie was passiert! Beheben Sie danach alle vorhandenen Fehler in der Implementierung, um die oben beschriebene Situation exakt umzusetzen.

V5 Roboter miteinander vergleichen

Schreiben Sie eine Methode `int robotsEqual(Robot a, Robot b)`. Diese bekommt zwei Roboter übergeben und soll `2` zurückgeben, wenn die Attribute `x`, `y`, `direction` und `numberOfCoins` bei beiden Robotern die gleichen Werte haben. `1` soll zurückgegeben werden wenn sich beide Roboter nur auf demselben Feld befinden, andernfalls gibt die Methode `0` zurück.

Übungsblatt 02: Vorbereitende Übungen von Nhan Huynh und Darya Nikitina

V6 Primitive Datentypen

Schreiben Sie eine Methode `char smallestPDT(long n)`. Diese bekommt eine ganze Zahl übergeben und soll den primitiven Datentyp zurückgeben, der den wenigsten Speicherplatz verbraucht, aber immer noch die übergebene Zahl speichern kann. Geben sie `"l"` für den Datentyp `long` zurück, `" i"` für `integer` usw.

Schreiben Sie nun eine Methode `char[] smallestPDTs(long[] a)`. Diese bekommt ein Array von ganzen Zahlen übergeben und soll ein Array zurückgeben, dass die Methode `smallestPDT(long n)`, in der gleichen Reihenfolge wie in `a`, auf jede Zahl in `a` anwendet.

V7 Aufsummieren

Schreiben Sie eine Methode `void sumUp(int[] a)`, die ein Array `a` von Typ`int` erhält.

An Index $i \in \{0, \ldots,$ a.length-i$\}$ in `a` soll nun der Wert a$[0]+\cdots+$a[i] geschrieben werden. Dabei bezeichnen a$[0]+\cdots+$a[i] die Werte in `a` unmittelbar vor dem Aufruf der Methode.

Übergeben wir der Funktion das folgende Array `a` $= [3, 4, 1, 9, -5, 4]$, so wird das Array folgendermaßen modifiziert:

$$\begin{aligned}
&\to [3, 3+4, 3+4+1, 3+4+1+9, 3+4+1+9+(-5), 3+4+1+9+(-5)+4] \\
&\to [3, 7, 8, 17, 12, 16]
\end{aligned}$$

V8 Liste von Positionen ★★★

In dieser Aufgabe werden wir ein wenig kreativ und zeichnen die Initialen der FOP mit dem FOP-Bot. Dazu müssen wir zunächst sicherstellen, dass unser Roboter eine beliebige gegebene Position automatisiert ansteuern kann.

Um uns die Arbeit zu vereinfachen, verwenden wir die Klasse `Point` der Java-Standardbibliothek, deren Instanzen Punkte im zweidimensionalen Raum repräsentieren. Ein `Point`-Objekt kann mittels des Konstruktors `Point(int x, int y)` erzeugt werden. Die Abfrage der Werte ist dann über die Objekt-Attribute `x` und `y` möglich.

V8.1 Setzen der Blickrichtung

Als Erstes soll die `public`-Methode `void setDirection(Robot robot, Direction direction)` implementiert werden: Diese bekommt ein `Robot`-Objekt sowie eine `Direction`-Konstante übergeben. Nach Aufruf der Methode soll der Roboter in die gewünschte Richtung blicken.

V8.2 Bewegen zu einer Position

Implementieren Sie nun die `public`-Methode `void moveToPoint(Robot robot, Point point)`: Mit dem Aufruf der Methode soll der gegebenen Roboter mittels der soeben von Ihnen implementierten Methoden `setDirection` und der Ihnen bereits bekannten Methode `move` an die gegebene Position bewegen werden. Sie können davon ausgehen, dass sich auf dem Weg zu dieser Position keine Hindernisse befinden. Dabei ist nicht wichtig, dass der Roboter den kürzesten Weg findet.

V8.3 Coin Patterns

Nun implementieren Sie die `public`-Methode `void putCoins(Robot robot, Point[] points)`: Der gegebene Roboter soll an jeder der im gegeben Array enthaltenen Position eine Münze ablegen, sich anschließend in die Mitte der Welt bewegen und nach oben blicken.

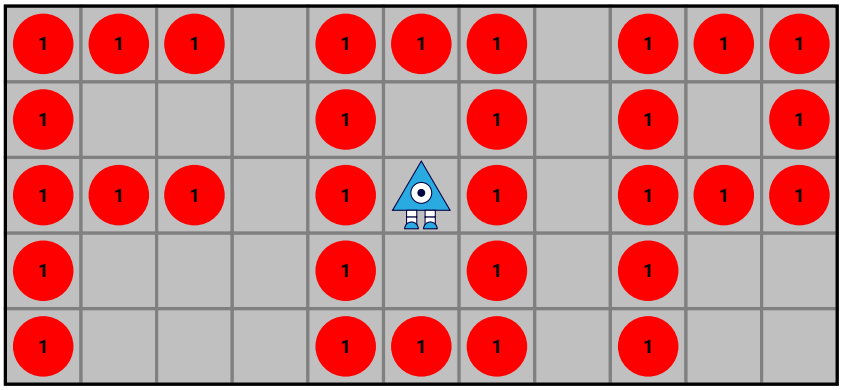


Abbildung 1: Gewünschtes Ergebnis

Verbindliche Anforderung: Die einzelnen Positionen des `Point`-Arrays müssen in einer einzigen `for`-Schleife abgelaufen werden. Die Objektmethode `putCoin` der Klasse `Robot` darf nur innerhalb dieser `for`-Schleife aufgerufen werden.