Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Einführung in die Informatik
Sommersemester 2012

Prof. Dr. Wolfram Burgard
PD Dr. Cyrill Stachniss
Dipl. Ing. Jörg Röwekämper
Dipl. Inf. Markus Kuderer

Übungsblatt 10

Abgabe bis Freitag, 13.07.2012, 12:00 Uhr

Hinweis:

Aufgaben immer per E-Mail (eine E-Mail pro Blatt und Gruppe) an den zuständigen Tutor schicken (Bei Programmieraufgaben Java Quellcode und eventuell benötigte Datendateien).

Aufgabe 10.1

Schreiben Sie eine Klasse `Sort`, die Methoden zum Sortieren von Arrays bereitstellt.

1. Implementieren Sie eine Methode
   `static int[] union(int[] a1, int[] a2)`,
   der zwei **sortierte** Folgen von Integer-Zahlen übergeben werden. Diese Methode soll eine sortierte Folge zurückgeben, die alle Elemente aus `a1` und `a2` enthält.

2. Bestimmen Sie die Komplexität der Methode `union` in Abhängigkeit der Länge der Arrays `a1` und `a2`.

3. Implementieren Sie eine **rekursive** Methode
   `static int[] sort(int[] array)`,
   die eine Folge von Integer-Zahlen in sortierter Reihenfolge zurückgibt. Gehen Sie dabei wie folgt vor:

   (a) Unterteilen Sie die Folge in zwei möglichst gleich große Teilfolgen.

   (b) Sortieren Sie beide Teilfolgen.

   (c) Fügen Sie die beiden sortierten Teilfolgen zusammen.

4. Bestimmen Sie die Komplexität der Methode `sort` in Abhängigkeit der Länge des Arrays `array`.

Aufgabe 10.2

Betrachten Sie den folgenden Algorithmus:

```
static int f(int x, int y) {

  if (y > x) {
    return f(y, x);   // Zeile 4
  }
  int z = x % y;

  if (z == 0) {
    return y;         // Zeile 9
  } else {
    return f(y, z);   // Zeile 11
  }
}
```

Zeichnen Sie die Activation Records für den Aufruf $f(12, 21)$ zu dem Zeitpunkt, an dem die maximale Rekursionstiefe erreicht ist.

Aufgabe 10.3

Ab diesem Übungsblatt wird es jeweils Zusatzaufgaben geben, die aufeinander aufbauen, mit dem Ziel, bis zum Ende des Semesters das bekannte Spiel Pac-Man[1] zu programmieren. In diesem Übungsblatt beginnen wir mit der Entwicklung des Spielfeldes und der Figuren. Auf den nächsten Übungsblättern werden wir diese in einer grafischen Oberfläche darstellen und uns danach mit der Spiellogik beschäftigen. Zu jeder dieser Aufgaben finden Sie JUnit-Tests auf der Vorlesungsseite, mit denen Sie ihre Implementierung testen können.

1. (a) Schreiben Sie eine Klasse `Map` mit einer Member-Variable `int[][] mapCells`, die das Spielfeld repräsentiert. Hierbei soll der erste Index des Arrays für die x- und der zweite für die y-Koordinate stehen.

   (b) Schreiben Sie die Methode `static Map getInstance(String filename)`, die eine Datei einliest und die Variable `mapCells` initialisiert.

   Eine Leveldatei hat das folgende Format:

   - Zeile 1: `size{width:height}`
   - Zeilen `2,3,...,height+1` enthälten die gleiche Anzahl an Zeichen (`0 bis 6`) wie die Spielfeldbreite (`width`):
     0: Mauer
     1: frei begehbar
     2: Pille
     3: Kraftpille
     4: Mauer, die nur für Monster passierbar ist
     5: Startpunkt der Monster
     6: Startpunkt des PacMan

   Auf der Vorlesungsseite finden Sie eine Beispieldatei.

2. (a) Schreiben Sie eine Klasse `Creature` mit den folgenden Member-Variablen:

   - `Vector2D direction` – Die Richtung, in die sich die Kreatur bewegt.
   - `double speed` – die Geschwindigkeit, mit der sich die Kreatur bewegt. Einheit: Pixel/Sekunden
   - `Circle boundingCircle` – ein Kreis, der Position und Hülle der Kreatur repräsentiert (siehe Übungsblatt 5).

   Sie finden die Klassen `Circle` und `Vector2D` auf der Vorlesungsseite.

   (b) Schreiben Sie eine Methode `void move(double elapsedTime)`, die die Position der Kreatur aktualisiert. Das Argument `elapsedTime` soll die vergangene Zeit in Sekunden seit dem letzten Aufruf sein. Sie müssen bei der Implementierung dieser Methode **nicht** darauf achten, dass einige Felder eventuell nicht passiert werden können bzw. die Kreatur das Spielfeld verlässt.

   (c) Schreiben Sie `get`- und `set`-Methoden für die Member-Variablen `direction` und `speed`.

[1] `https://de.wikipedia.org/wiki/Pac-Man`