Albert-Ludwigs-Universität Freiburg
Informatik 1
Wintersemester 2011/12

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

Übungsblatt 7

Abgabe bis Donnerstag, 17.12.11, 12:00 Uhr

Hinweis: Kompilieren und testen Sie Ihre Programme mit Hilfe des `ant`-Buildsystemes. Schicken Sie Ihrem Tutor eine Kopie des kompletten Projektordners, der alle benötigten Dateien enthält.

Aufgabe 7.1

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 7.2

Erweitern Sie die Klasse `Matrix` aus der Vorlesung um folgende Methoden.

1. Implementieren Sie die Methode `double trace()`, die die Spur (die Summe der Diagonalenelemente) der Matrix berechnet.

2. Implementieren Sie die Methode `boolean add(Matrix m)`, so dass bei dem Aufruf `m1.add(m2)` die Matrix `m2` zur Matrix `m1` addiert wird. Die Methode soll `false` zurückgeben, wenn die Matrizen unterschiedlicher Größe sind. In diesem Fall soll die Matrix `m1` unverändert bleiben.

3. Implementieren Sie die Methode `boolean subtract(Matrix m)`, so dass bei dem Aufruf `m1.subtract(m2)` die Matrix `m2` von Matrix `m1` subtrahiert wird. Die Methode soll `false` zurückgeben, wenn die Matrizen unterschiedlicher Größe sind. In diesem Fall soll die Matrix `m1` unverändert bleiben.

4. Implementieren Sie die Methode `boolean multiply(Matrix m)`, so dass bei dem Aufruf `m1.multiply(m2)` Matrix `m1` mit Matrix `m2` multipliziert wird. Die Methode soll `false` zurückgeben, wenn die Größe der Matrizen nicht kompatibel ist. In diesem Fall soll die Matrix `m1` unverändert bleiben.

5. Führen Sie eine Komplexitätsanalyse für Ihre Methoden durch.

6. Implementieren Sie einen `JUnit`-Test für Ihre Methoden.

Aufgabe 7.3

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.