Prof. Dr. Gerd Stumme, Folke Eisterlehner, Dominik Benz
Fachgebiet Wissensverarbeitung

04.05.2009

3. Hausübung „Algorithmen und Datenstrukturen“

Sommersemester 2009

Abgabetermin: Montag, 11.05.2009, 10:00 Uhr

1 Theorie

1.1 Komplexität.

Die folgende Java-Methode `sum42` ermittelt alle Möglichkeiten, den Wert `42` als Summe dreier Elemente des Arrays `a` darzustellen. Bestimmen Sie die worst-case Laufzeit der Methode in Abhangigkeit von `n:= a.length` in O-Notation und begründen Sie Ihre Antwort (`println` hat konstanten Zeitaufwand).

```
static void sum42(int[] a) {
  // Es wird vorausgesetzt, dass a nur nichtnegative Zahlen enthaelt.
  for (int i = 0; i < a.length; ++i) {
    if (a[i] <= 42) {
      for (int j = i + 1; j < a.length; ++j) {
        if (a[i] + a[j] <= 42) {
          for (int k = 0; k < a.length; k++) {
            if (a[i] + a[j] + a[k] == 42) {
              System.out.println(42 + " = " + a[i] + " + "
                  + a[j] + " + " + a[k]);
            }
          }
        }
      }
    }
  }
}
```

(10 Punkte)

2 Programmierung

Sequenzielle Suche: Schreiben Sie eine Java-Klasse **`LinearSearch`** zur sequenziellen Suche, welche das Interface **`DataSearch`** von der Webseite zur Vorlesung implementiert. Das heißt, eine Methode „**`doSearch`**“ implementiert den in der Vorlesung vorgestellten Algorithmus zur sequenziellen Suche und liefert bei Eingabe eines Arrays **`data`** von ganzen Zahlen (**`int`**) und eines Schlüsselwerts **`key`** (**`int`**) den Wahrheitswert **`true`** zurück, falls **`key`** in **`data`** enthalten und **`false`** sonst. (5 Punkte)

Binäre Suche: Implementieren Sie den Algorithmus **BinarySearch** aus der Vorlesung in Java. Schreiben Sie hierzu eine Klasse **BinarySearch**, welche das Interface **DataSearch** von der Webseite zur Vorlesung implementiert. (5 Punkte)

Laufzeitvergleich: Schreiben Sie eine Java-Klasse **StopWatch** zum Laufzeitvergleich der binären und sequenziellen Suche. Dabei sollen die Klassen **LinearSearch** und **BinarySearch** zur Suche der Zahlen 3, 2147483145, 1074047991 und 42 in Feldern der Größen 10^4, 10^7 und 10^8 verwendet und die jeweiligen Laufzeiten in Millisekunden ausgegeben werden. Verwenden Sie als Datenquelle die Klasse **DataSource** von der Webseite zur Vorlesung. Mittels des Aufrufs `DataSource.getSortedData(100)` erhalten Sie beispielsweise ein Array mit 100 sortierten Zufallszahlen. Beachten Sie, dass Ihr Java-Programm viel Speicher benötigt und stellen Sie Ihrer JVM ausreichend zur Verfügung (Aufruf z.B. mittels „**java -Xmx512m StopWatch**“).

Die Ausgabe der **main**-Methode von **StopWatch** soll wie folgt aussehen (wobei $< X_i >$ jeweils für die Laufzeit der Suche nach dem Schlüssel *i* steht): (10 Punkte)

```
Laufzeitvergleich der linearen und binaeren Suche in Daten der Groessen n:
Sequenzielle Suche (n=1000):
    3: <X1> ms
    42: <X2> ms
    1074047991: <X3> ms
    2147483145: <X4> ms
Sequenzielle Suche (n=1000000):
    3: <X1> ms
    42: <X2> ms
    1074047991: <X3> ms
    2147483145: <X4> ms
Sequenzielle Suche (n=10000000):
    3: <X1> ms
    42: <X2> ms
    1074047991: <X3> ms
    2147483145: <X4> ms
Binaere Suche (n=1000):
    3: <X1> ms
    42: <X2> ms
    1074047991: <X3> ms
    2147483145: <X4> ms
Binaere Suche (n=1000000):
    3: <X1> ms
    42: <X2> ms
    1074047991: <X3> ms
    2147483145: <X4> ms
Binaere Suche (n=10000000):
    3: <X1> ms
    42: <X2> ms
    1074047991: <X3> ms
```

```
2147483145: <X4> ms
```

Viel Spaß und viel Erfolg!