Universität Karlsruhe (TH)
Forschungsuniversität • gegründet 1825

Institut für Angewandte und Numerische Mathematik

Programmieren: Einstieg in die Informatik mit Java

WS 2006/2007

Dr. G. Bohlender
Dipl.–Math. techn. M. Richter

11.12.2006

Aufgabenblatt 7

Bearbeitungszeitraum: 14.12.2006 – 10.01.2007

Aufgabe 18 (Pflichtaufgabe): *Sortieren*

Schreiben Sie ein Java-Programm mit dem Namen `Sortieren`, welches eine Liste von ganzen Zahlen einliest und diese sortiert wieder ausgibt. Gehen Sie dabei wie folgt vor:

(a) Lesen Sie von der Konsole eine positive, ganze Zahl `N` ein und erzeugen Sie damit ein Feld von `int`-Werten der Länge `N`. Lesen Sie anschließend mit einer `for`-Schleife die einzelnen Feldkomponenten von der Konsole ein.

(b) Erstellen Sie eine statische Methode mit dem Namen `ausgabe`, und definieren Sie für diese Methode ein `int`-Feld mit dem Namen `liste` als formales Argument. Geben Sie in der Methode die Komponenten des Feldes `liste` auf dem Bildschirm aus. Diese Methode soll keinen Rückgabewert besitzen, d.h. sie soll als `void`-Methode definiert werden. Verwenden Sie diese Methode, um das Feld in der `main`-Methode auszugeben.

(c) Erstellen Sie eine statische `void`-Methode mit dem Namen `vertausche`, und definieren Sie ein `int`-Feld `liste`, sowie zwei `int`-Werte `i` und `j` als formale Argumente dieser Methode. Vertauschen Sie in dieser Methode die `i`-te mit der `j`-ten Feldkomponente von `liste`.

(d) Erstellen Sie eine statische `void`-Methode mit dem Namen `sortieren`, welche die Komponenten eines `int`-Feldes in aufsteigender Reihenfolge sortiert. Gehen Sie dabei wie folgt vor: Definieren Sie ein `int`-Feld `liste` als formales Argument der Methode. Definieren Sie außerdem in der Methode eine lokale Variable `sortiert` vom Typ `boolean`. Implementieren Sie anschließend den sogenannten *Bubblesort*-Sortieralgorithmus:

(1) Setzen Sie den Wert der Variable `sortiert` auf `true`.

(2) Durchlaufen Sie das Feld `liste` von Anfang bis Ende und vergleichen Sie jede Komponente mit ihrem Nachfolger. Ist der Nachfolger kleiner als die jeweilige Komponente, so vertauschen Sie beide Komponenten und setzen den Wert von `sortiert` auf `false`.

(3) Wiederholen Sie die obigen Schritte falls `sortiert` den Wert `false` hat.

Verwenden Sie bei der Implementierung eine `do-while`-Schleife sowie die Methode `vertausche`.

(e) Sortieren Sie das `int`-Feld in der `main`-Methode mit der Methode `sortiere` und geben Sie das sortierte Feld auf der Konsole aus.

Aufgabe 19: *Anagramme*

Unter einem Anagramm versteht man eine Folge von Buchstaben, die durch Umstellung der Buchstaben eines Wortes der (z.B. deutschen) Sprache gebildet werden kann. Die Buchstabenfolge MFORNKIITA ist beispielsweise ein Anagramm des Wortes INFORMATIK. In bestimmten Fällen ist ein Anagramm selbst ein Wort der jeweiligen Sprache. Dies gilt zum Beispiel für die Worte NOTEN und TONNE. Prinzipiell können aus einem Wort mit n Buchstaben maximal $n!$ verschiede Anagramme gebildet werden.
Schreiben Sie ein Java-Programm mit dem Namen `Anagramme`, welches alle möglichen Anagramme eines Wortes bildet. Gehen dabei wie folgt vor:

(a) Definieren Sie eine globale Klassenvariable mit dem Namen `anagramm` vom Typ `char[]`. Lesen Sie in der `main`-Methode ein Wort von der Konsole ein, und speichern Sie dieses in einer `String`-Variable. Bestimmen Sie die Länge `N` des Wortes und erzeugen Sie anschließend ein Feld von `char`-Werten mit dem Namen `zeichenkette` und der Länge `N`. Speichern Sie in den Komponenten von `zeichenkette` die einzelnen Buchstaben des eingelesenen Wortes. Initialisieren Sie die globale Klassenvariable `anagramm` als `char`-Feld der Länge `N`.

(b) Erstellen Sie ein statische Methode mit dem Namen `entferne`. Definieren Sie als formale Argumente dieser Methode einen `int`-Wert `index` und ein `char`-Feld `zeichenkette`. Bestimmen Sie in der Methode die Länge `N` des übergebenen `char`-Feldes und erzeugen Sie ein neues `char`-Feld `res` der Länge `N-1`. Speichern Sie in den Komponenten von `res` alle Zeichen von `zeichenkette` bis auf das Zeichen an der Stelle `index`. Geben Sie das neue `char`-Feld zurück.

(c) Erstellen Sie eine statische `void`-Methode mit dem Namen `setze`, und definieren Sie einen `int`-Wert `pos` sowie ein `char`-Feld `zeichenkette` als formale Argumente der Methode. Prüfen Sie zunächst, ob der Wert von `pos` kleiner als die Länge von `anagramm` ist. Falls nicht, geben Sie `anagramm` auf der Konsole aus. Durchlaufen Sie andernfalls das Feld `zeichenkette` Komponentenweise und führen sie für die jeweils `i`-te Komponenten folgende Schritte durch: Speichern Sie den Wert der Komponente `i` von `zeichenkette` in der Komponente `pos` von `anagramm` ab. Erzeugen Sie mit der Methode `entferne` ein neues `char`-Feld `rest`, in der die `i`-te Komponente von `zeichenkette` entfernt wurde. Rufen Sie anschließend die Methode `setze` mit `pos+1` und `rest` rekursiv auf.

(d) Rufen Sie in der `main`-Methode die Methode `setze` mit den aktuellen Argumenten `0` und `zeichenkette` auf. Bilden Sie die Anagramme von TON und HELM. Welches Wort verbirgt sich hinter dem Anagramm KMESA?