Lehrstuhl für Informatik 3
Rheinisch-Westfälische Technische Hochschule
Aachen
Univ.-Prof. Dr.-Ing. M. Nagl

Aachen, 25. 05. 2007

Nagl, Ranger, Wörzberger

Übungen zur Vorlesung „Grundgebiete der Informatik 2: Algorithmen und Programmiertechniken“

— Blatt 7 —

14. Aufgabe *Sortiertes Histogramm*: *(7 Punkte)*

Die Ausgabe des Histogramm-Programms ist noch recht unordentlich. Das Programm soll jetzt die Zähler absteigend nach Häufigkeit sortiert ausgeben. Dazu darf die Zählerliste umgeordnet werden. Implementieren Sie also eine Funktion zur folgenden Deklaration und rufen Sie sie an der passenden Stelle in der Funktion *main* auf.

```
/* Sortiere die Eintraege im Histogramm in absteigender Reihenfolge.
   Vorbedingung : dasHistogramm ist ein gueltiges Histogramm
   Eingabe      : Referenz auf dasHistogramm --- das zu sortierende Histogramm
   Ausgabe      : keine
   Nachbedingung: Fuer alle Eintraege e aus dasHistogramm gilt:
                       e->naechster==NULL    oder
                       e->wert >= e->naechster->wert.
*/
void sortiereHistogramm(HistogrammT &dasHistogramm);
```

Sie können sich den Sortieralgorithmus selber aussuchen, sollten dabei jedoch zuerst auf Verständlichkeit achten. Bei der Korrektur wird bewertet, inwiefern der Code plausibel ist! Denken Sie daran Ihren Code ausführlich zu kommentieren!

15. Aufgabe *Sortierverfahren*: *(8 Punkte)*

In der Vorlesung wurde neben Quicksort ein weiterer Sortieralgorithmus erwähnt, aber nicht näher vorgestellt: *MergeSort*. Während die vorgestellten Sortieralgorithmen nur auf Daten im Hauptspeicher arbeiten können, kann *MergeSort* auch dazu verwendet werden, Daten innerhalb einer Datei zu sortieren.

MergeSort arbeitet nach folgendem Algorithmus:

1. Teile das Feld in zwei gleich große Teile, wenn möglich bzw. nötig[1].
2. Sortiere diese beiden Teile mit *MergeSort* (rekursiv, also beginne wieder mit Schritt 1).
3. Vereinige die beiden sortierten Felder so, dass das Ergebnis der Vereinigung wiederum sortiert ist.

(a) Implementieren Sie *MergeSort* für ein Feld. Stellen sich Sie hierbei die Frage, wann die Rekursion abgebrochen werden kann, da eine weitere Sortierung des Feldes nicht notwendig ist. Bei der Vereinigung

[1] Zur Erinnerung: Felder der Größe 1 sind trivialerweise sortiert.

der beiden sortierten Felder (auch Mischphase genannt) reicht es, beide Felder parallel zu durchlaufen und das jeweils kleinste Element in das Ergebnisfeld zu kopieren. Am Ende befinden sich in einem der beiden Felder noch Elemente, die nun einfach an das Ende des Ergebnisfeldes kopiert werden können, da sie alle größer sind. *(6 Punkte)*

(b) Wieviel Speicher benötigt Ihre Lösung zusätzlich zu dem Feld, das sortiert werden soll? *(1 Punkt)*

(c) Erläutern Sie aus welchen Grund der Algorithmus für Dateien besonders geeignet ist. *(1 Punkt)*

16. Aufgabe *Testen*: *(5 Punkte)*

(a) Geben Sie zur Funktion `ggT` einen Flussgraphen für einen White-Box-Test an. Ermitteln Sie alle möglichen Programmpfade zu dieser Funktion und geben Sie die entsprechenden Klassen von Eingabedaten an. Welche Fälle sollten von der Funktion noch ausdrücklich behandelt werden? *(3 Punkte)*

```
int ggT(int a, int b)
{
   int m, n, hilf;

   if (a < b) { m = b;   n = a; }
   else       { m = a;   n = b; };

   while (m != n) {
      m = m-n;
      if (m < n) { hilf = m;   m = n;   n = hilf; };
   };

   return n;
}
```

(b) Für eine Funktion, die das kleinste gemeinsame Vielfache zweier Zahlen berechnet, soll ein Black-Box-Test durchgeführt werden. Geben Sie hierfür Testdaten an, die die Funktion möglichst vollständig testen. *(2 Punkte)*

Abgabe: 08. 06. 2007 vor der Vorlesung.