Prof. Dr. A. Schwill, Dr. Peter-Uwe Zettiér, Marco Thomas
Institut für Informatik 19.11.97

Übungen zur Vorlesung
Algorithmen, Daten, Programme I
Blatt 6

Abgabe: 25.11.97, 13.00 Uhr im Postfach von Herrn Zettiér bzw. Herrn Thomas in Raum 1.2.2.08

Aufgabe 1: (10 Punkte)

a) Schreiben Sie ein Programm in PRO, das eine Zahlenfolge einliest und diese absteigend sortiert wieder ausgibt. Das Programm soll folgende Strategie verwenden: Die Folge wird logisch in zwei Teilfolgen f1 und f2 zerlegt. f1 besteht zunächst aus dem ersten, f2 aus den restlichen Elementen der Ausgangsfolge. Nun wird fortlaufend das erste Element der Folge f2 entfernt und an die bezüglich der Sortierung richtige Stelle in die Folge f1 eingefügt. Zum Schluß ist f2 leer und f1 enthält das gesuchte Ergebnis. Diese Vorgehensweise nennt man *Sortieren durch Einfügen*. (sort() aus dem Skript steht nicht zur Verfügung!).

b) Ermitteln Sie die Worst-Case-Komplexität ihres PRO-Programms.

Aufgabe 2: (3 Punkte)
Geben Sie für jede der drei Wertzuweisungen

$y1 \leftarrow (a+b)*(c+d)-(e+f)$, $y2 \leftarrow (a*b*(c+d*(c-d)))$, $y3 \leftarrow a+(b+c)*12-a$

eine Folge von Wertzuweisungen mit möglichst wenigen Hilfsbezeichnern an, die jeweils nur einen Operator enthalten und die diese Wertzuweisungen realisieren.

Aufgabe 3: (5 Punkte)

Schreiben Sie ein ASS-Programm, das den Inhalt zweier Speicherzellen vertauscht. Die Adressen der beiden Speicherzellen werden vorher eingelesen.

Aufgabe 4: (8 Punkte)

a) Bestimmen Sie die Semantik des nebenstehenden Programms, indem Sie exakt die berechnete Funktion angeben.

b) Zeigen Sie die Korrektheit des Programms.

```
def z,q: Zahl;
lies(z);
wenn z<0 dann
        z← -z
sonst
ende;
q←1;
solange z≥1 tue
        q←q*z;
        z←z-1
ende;
zeige(q).
```

Aufgabe 5: (6 Punkte)

Verwenden Sie die PCs mit dem Betriebssystem LINUX (oder melden sie sich von einer anderen Maschine aus auf den LINUX-Rechnern an; z. B. mit: `rlogin ati03` oder `rlogin ati02`).
Kopieren Sie sich die Dateien `aufg5.c` und `ld_Script` aus dem Verzeichnis `/home/SUNOS/ADP`.

a) Mit dem C-Compiler `cc -S -O aufg5.c` übersetzen Sie das Programm aufg5 von Ebene 5 auf Ebene 4 der Rechnerarchitektur. Drucken Sie das enstandene Assembler-Programm in `aufg5.s` aus und markieren Sie den Programmcode, der der while-Schleife des C-Programms in `aufg5.c` entspricht.

b) Assemblieren Sie mit `as aufg5.s -o aufg.o` und übersetzen Sie anschließend den Assemblercode in ausführbaren Maschinencode (Ebene 2) mit `ld_Script` (ruft den Binder mit versch. Parametern auf). Den Maschinencode können Sie sich mit `ob -b a.out` ansehen. Starten Sie nun das erzeugte Programm a.out und leiten Sie die Ausgabe auf den Drucker um.

Anm. Dateien können Sie sich u.a. mit `cat dateiname` anschauen.