Technische Universität München
Institut für Informatik
Prof. Dr. Ernst W. Mayr

WS 1993/94
Übungsblatt 1
3. Nov. 1993

Einführung in die Informatik III

(Abgabetermin: 10. Nov. 1993, 13:00Uhr)

Aufgabe 1

Beweisen Sie: Seien f und g zwei Funktionen von $\mathbb{N}$ nach $\mathbb{N}$. Dann gilt $f = O(g)$ oder $f = \omega(g)$, aber nicht beides.

Aufgabe 2

Vergleichen Sie das Wachstumsverhalten der folgenden Funktionen (von $\mathbb{N} \to \mathbb{N}$):

1. n
2. $\lceil 1/e \rceil$
3. $\sin(n\pi)$
4. $n \sin(n\pi)$
5. $n|\sin(\frac{n\pi}{2})|$
6. $\lceil \log n \rceil$
7. $\lceil 2^{\sqrt{\log n}} \rceil$
8. $\lceil n^{\frac{1}{n}} \rceil$
9. $\lceil (n!)^{\frac{1}{n}} \rceil$

Aufgabe 3

Schreiben Sie ein Programm (in C, Modula oder Pascal), das mit Hilfe eines in der Programmiersprache verfügbaren Pseudozufallszahlengenerators zufällige, einfach verkettete Listen erzeugt. Die Verkettungsinformation für die jeweils erzeugte Liste soll in einer Reihung l vom Typ **vector**=ARRAY[0..$n-1$] OF INTEGER (wo n eine Programmkonstante ist) in der Form abgelegt werden, daß die Zeigerkette der Liste durch

$$0 \to l[0] \to l[l[0]] \to l[l[l[0]]] \to \dots$$

gegeben ist. Achten Sie bei der Implementierung darauf, daß alle solche Listen mit gleicher Wahrscheinlichkeit erzeugt werden, sofern der Pseudozufallszahlengenerator gleichverteilte Zahlen liefert. Führen Sie für drei verschiedene Werte von n (zwischen 128 und 256) jeweils drei Läufe des Programms durch und geben Sie die erzeugten Listen aus.

Aufgabe 4

Das unten abgedruckte Programm ist ein Assemblerprogramm für die (abstrakte) Maschine MI. Versuchen Sie, seine Funktion zu beschreiben! Begründen Sie, wieso das Register

R1 um drei Positionen nach rechts und dann wieder um zwei Positionen nach links geshiftet wird. Welche Bedingung muß die Eingabereihung *array* erfüllen, damit das Programm funktioniert?

Lese-Aufgabe 5

Lesen Sie in einem Buch über die Programmiersprache C (z.B. Kernighan/Ritchie) die Kapitel über Typdeklarationen, Operatoren, Kontrollfluß, Prozeduren, Blockstruktur, und die Präprozessoranweisungen.

MI-Programm:

```
whatsit:
        SEG
        MOVE    W       I H'10000',SP   -- Vorbesetzung Stackpointer
        JUMP            start
array:  DD      W       23,35,37,96,245,346,347
endarr:
x:      DD      W       345             -- ein Wort (W) umfaßt 4 Bytes
a:      DD      W       array
b:      DD      W       endarr
i:      RES             4               -- reserviere Speicherplatz fuer ein W
res:    RES             4               -- reserviere Speicherplatz fuer ein W
start:
        MOVE    W       a,R2
loop:   SUB     W       R2,b,R1         -- R1:=b-R2
        SH              I -3,R1,R1      -- shifte drei Positionen nach rechts
        SH              I 2,R1,R1
        ADD     W       R1,R2,R1
        CMP     W       R1,R2
        JEQ             done
        CMP     W       x,!R1           -- vergleiche den Inhalt von x mit dem
                                        -- Inhalt der Speicherzelle, deren
                                        -- Adresse in R1 steht (indirekte
                                        -- Adressierung
        JGE             greater
        MOVE    W       R1,b
        JUMP            loop
greater:
        MOVE    W       R1,R2
        JUMP            loop
done:   CMP     W       x,!R2
        JNE             end1
        MOVE    W       !R2,res
        HALT                            -- erfolgreiches Ende

end1:   HALT                            -- erfolgloses Ende
        END
```