INSTITUT FÜR INFORMATIK V
UNIVERSITÄT BONN

Prof. Dr. N. Blum
H.-H. Leinen
M. Löhnertz

53117 BONN
Römerstraße 164

14. Oktober 2002

Übungen zur Informatik III im WS 02/03

Blatt 1

Abgabe: 22.10.2002

Aufgabe 1. (4 Punkte)

a) Gegeben sei das Feld A[1:20] mit A[i] := i. Suchen Sie mithilfe des Algorithmus "Binäre Suche" die folgenden Zahlen: 8, 20, 100. Geben Sie hierbei für jeden Schritt des Algorithmus die Inhalte von p und q an. Geben Sie außerdem an, wie viele Schritte der Algorithmus jeweils für den Schleifendurchlauf benötigt, sowie die Ausgabe. In welchen Fällen sind $\lceil \log 20 \rceil$ Schleifendurchläufe erforderlich?

Beispiel: "Binäre Suche" mit Eingabe A und x := 5 liefert die folgenden Schritte:

1. (0,20)
2. (0,10)
3. (10,10)

Die Ausgabe ist "5", die Anzahl der Schritte im Schleifendurchlauf beträgt 2.

b) Der Algorithmus "Binäre Suche" sei nun auf Mengen von Zeichenketten definiert; Eingabe x ist eine solche Zeichenkette. Die vorgegebene Ordnung sei die lexikographische Ordnung[1] $\tilde{<}_{lex}$. Das zugrundeliegende Alphabet Σ enthalte mindestens zwei Zeichen a und b.

Das Feld A[1:20] sei definiert durch:
A[1] := (a,a)
A[2] := (a,a,a)
$\vdots$
A[20] := $(\underbrace{a, a, \ldots, a}_{21})$

Suchen Sie die folgenden Elemente: (a), (a, a, a, a), (b). Geben Sie dabei wieder die Schritte des Algorithmus sowie dessen Ausgabe an.

*Für die in den folgenden Aufgaben zu erstellenden Befehlsfolgen dürfen Sie die gleichen Konstrukte verwenden, die auch der Algorithmus "Binäre Suche" aus der Vorlesung verwendet (**for**, **while**,...).*

Aufgabe 2. (5 Punkte)

a) Gegeben sei ein nichtleeres Alphabet Σ und ein Zeichen #, so dass $\Sigma \cap \{\#\} = \emptyset$. Die Schlange S enthalte eine beliebige Folge w von Zeichen aus Σ. Wir können w als n-Tupel schreiben, wenn w n Zeichen enthält ($w \in \Sigma^n$). Schreiben Sie eine Befehlsfolge, die als Eingabe einen leeren Keller K und die Schlange S erhält, und die die Zeichenfolge $w\#w^R$ in die Schlange schreibt. Hierbei ist w^R das Tupel $(a_n, a_{n-1}, \ldots, a_1)$, falls w = $(a_1, a_2, \ldots, a_n)$, $a_i \in \Sigma$ ($i \in \{1, \ldots, n\}$).

Die Befehlsfolge darf nur die beiden Datenstrukturen S und K benutzen. Um auf die Datenstrukturen zuzugreifen, schreiben Sie:

- PUSH$_S(x)$, um ein Element in die Schlange einzufügen
- x := POP$_S$, um der Schlange ein Element zu entnehmen

[1] Sie finden hierzu am Ende dieses Übungszettels eine entsprechende Definition.

- $\mathrm{PUSH}_K(x)$, um dem Keller ein Element hinzuzufügen
- $x := \mathrm{POP}_K$, um ein Element aus dem Keller zu entfernen

Beachten Sie, dass Sie jederzeit auf das oberste Element des Kellers mittels K[TOP] zugreifen dürfen.

b) Schreiben Sie eine Befehlsfolge, die als Eingabe eine Schlange S und einen leeren Keller K bekommt – wobei S eine beliebige Folge natürlicher Zahlen enthält – und die diese Zahlen in aufsteigend sortierter Reihenfolge in die Schlange schreibt. Wir nehmen dabei an, dass sowohl Schlange als auch Keller beliebige ganze Zahlen aufnehmen können.

Ihre Befehlsfolge darf neben S und K nur eine einzige Variable x benutzen! Die Zugriffsoperationen für Keller und Schlange sind die gleichen wie in Aufgabe 2 a).

Aufgabe 3. (2 Punkte)

Geben Sie eine Befehlsfolge an, die eine neue Zahl a derart in eine aufsteigend sortierte verkettete Liste von Zahlen einfügt, dass diese sortiert bleibt.

Aufgabe 4. (5 Punkte)

Sie haben in der Vorlesung die Datenstruktur "verkettete Liste"[2] kennengelernt. Eine "doppelt verkettete Liste" ist eine verkettete Liste, bei der in jedem Listenelement zusätzlich ein Zeiger auf das *vorhergehende* Listenelement gespeichert wird.

Erweitern Sie die Datenstruktur "verkettete Liste" zu einer doppelt verketteten Liste. Passen Sie die Befehlsfolgen "Einfügen" und "Streichen" entsprechend an. Realisieren Sie zusätzlich die Operationen "Einfügen vor(x, I)" und "Streichen(x)", denen als Parameter lediglich das einzufügende bzw. zu streichende Element x, sowie im Falle des Einfügens noch ein Index I ($I \geqslant 1$) übergeben wird. Hierbei soll x in der Liste vor dem Element eingefügt werden, das unter dem Index I aufzufinden ist.

Welche Vorteile haben doppelt verkettete Listen gegenüber einfach verketteten Listen? Welche Nachteile haben sie?

Aufgabe 1 b) auf diesem Blatt benötigt den Begriff der *lexikographischen Ordnung*, den wir Ihnen im folgenden vorstellen:

Alphabet Sei Σ eine endliche, nichtleere Menge von Zeichen. Dies können sowohl Ziffern als auch Buchstaben, aber auch beliebige andere Symbole sein. Wir nennen Σ ein *Alphabet*.

Zeichenkette Sei Σ ein Alphabet. Sei $w \in \Sigma^n$, d.h. w ist das n-Tupel $(a_1, a_2, \ldots, a_n)$, wobei $a_i \in \Sigma$ $(i \in \{1, 2, \ldots, n\})$. Wir nennen w eine *Zeichenkette* aus Σ.

Lexikographische Ordnung Auf dem Alphabet Σ sei eine lineare Ordnung $<_{lex}$ vorgegeben. Seien $w = (a_1, a_2, \ldots, a_m)$ und $v = (b_1, b_2, \ldots, b_n)$ zwei beliebige Zeichenketten aus Σ. Die Relation $\tilde{<}_{lex}$ sei wie folgt definiert:

$w \tilde{<}_{lex} v \iff$ genau eine der beiden folgenden Bedingungen trifft zu:

1. Für ein $i \leqslant m$ ist $a_j = b_j$ für $1 \leqslant j < i$, $a_i \neq b_i$ und $a_i <_{lex} b_i$
2. $m \leqslant n$ und $a_i = b_i$ für alle $i, 1 \leqslant i \leqslant m$.

Wir nennen $\tilde{<}_{lex}$ eine *lexikographische Ordnung*. (Die Reihenfolge der Einträge in einem Telefonbuch ist beispielsweise eine lexikographische Ordnung.)

□

[2] Um diese Art der Verkettung zu betonen, spricht man manchmal auch von "*einfach* verketteten Listen".