Blatt 4

Aufgabe 2

Bestimmen Sie die Größenordnung (Θ) der Lösungen folgender Rekurrenzen.

$$\text{a)}\ T(n) = 3T(\tfrac{n}{9}) + n^{1/3}$$

$$\begin{gathered} n^{\log_9(3)} = n^{\frac{1}{2}} \\ n^{\frac{1}{3}} \in o(n^{\frac{1}{2}}) \\ \Rightarrow T(n) \in \Theta(n^{\frac{1}{2}}) \end{gathered}$$

$$\text{b)}\ T(n) = 4T(\tfrac{n}{2}) + n\log(n)$$

$$\begin{gathered} n^{\log_2(4)} = n^2 \\ n\log n \in o(n^2) \\ \Rightarrow T(n) \in \Theta(n^2) \end{gathered}$$

$$\text{c)}\ T(n) = T(n-3) + n$$

$$\begin{gathered} T(n) = T(n-3) + n \\ = T(n-6) + n - 3 + n \\ = T(n-9) + 3n - 3 - 6 \\ \dots \\ T(n) = \frac{n}{3} \cdot n - 3 \cdot \frac{\frac{n}{3}(\frac{n}{3}+1)}{2} = \frac{n^2}{3} - \frac{n^2}{6} - \frac{n}{2} \\ \Rightarrow T(n) \in \Theta(n^2) \end{gathered}$$

d) $T(n) = 2T(n-4)$

$$T(n) = 2T(n-4) = 4T(n-8) = 8T(n-12)$$
$$\dots$$
$$= 2^{\frac{n}{4}} T(0)$$
$$\Rightarrow T(n) \in \Theta((\sqrt[4]{2})^n)$$

Aufgabe 3

a) Sei L eine einfach verkettete Liste mit n Elementen und Schlüsseln aus $\mathbb{N}$. Geben Sie Pseudocode für eine Funktion `DeleteSmallerThanLast(`L`)` an, die in $\mathcal{O}(n)$ Zeit alle Elemente aus L entfernt, deren Schlüssel kleiner als der letzte in L vorkommende Schlüssel sind.

Beispiel: Für $L = L.\text{head} \to 19 \to 12 \to 15 \to 17 \to 14 \to L.\text{NIL}$ ist der letzte vorkommende Schlüssel 14 und nach dem Aufruf von `DeleteSmallerThanLast(`L`)` gilt $L = L.\text{head} \to 19 \to 15 \to 17 \to 14 \to L.\text{NIL}$.

```
Algo 1: DeleteSmallerThanLast(L)
Input: Liste L (Zahlen)
Result: void (Liste wird modifiziert)


if L.head=L.NIL do
  return
pointer := L.head
last:=pointer.val
while (pointer != L.NIL) do
  max := pointer.val
  pointer=pointer.head

startVal := L.head.val
while(startVal<last) do
  l.head := l.head.next
  startVal := L.head.val

pointer := L.head
while(pointer.next!=L.NIL) do
  if (pointer.head.val<last) do
```

```
    pointer.next := pointer.next.next
  else do
    pointer=pointer.next

return
```

b) Sei L eine einfach verkettete Liste der Länge n. Weiterhin seien a und b zwei natürliche Zahlen. Geben Sie Pseudocode für eine Funktion `CutSublist(`L`,`a`,`b`)` an, die Folgendes leistet: Entferne alle Elemente vom $a$ten bis zum $b$ten Element aus L. Falls $a > b$ oder $a > n$ soll kein Element entfernt werden, falls $b > n$ sollen alle Elemente ab dem $a$ten Element entfernt werden. Analysieren Sie die Laufzeit Ihres Algorithmus einmal in Abhängigkeit von n und einmal Abhängigkeit von b.

Beispiel: Für $L = L.\text{head} \to 1 \to 5 \to 4 \to 5 \to 7 \to L.\text{NIL}$ gilt nach dem Aufruf von `CutSubList(`L`,2,10)`, dass $L = L.\text{head} \to 1 \to L.\text{NIL}$.

```
Algo 2: CutSublist(L)
Input: Liste L (Zahlen),a ,b
Result: void (Liste wird modifiziert)

if(a>b) do
  return
if(L.head=L.NIL) do
  return

if (a<=1) do
  pointer := L.head
  while (pointer!=L.Nil & b>0) do
    L.head := L.head.next
    b--

else
  pointer := L.head
  pos := 2
  while (pointer.next !=L.NIL) do
    if (pos>=a & pos<=b)do
      pointer.next = pointer.next.next

return
```