Algorithmen und Datenstrukturen
Tutorium III

Michael R. Jung

04. - 07. 05. 2015

Aufgabe 1

Implementieren Sie einen Stack und eine Queue auf Grundlage einer einfach verketteten Liste!

Folgende Funktionen seien für eine Liste L in natürlicher Weise implementiert:

- `int L.length();`
- `boolean L.isEmpty();`
- `void L.add(real value, int position);`
- `void L.addfirst(real value);`
- `void L.delete(int position);`
- `real L.getValue(int position);`

```
function getM(A: array_of_int){
n:=|A|
if (n=1) then
  return A[1];
endif
m:= ⌊ n/2 ⌋;
a:=getM(A[1...m]);
b:=getM(A[m+1...n]);
if (a>b)return a;
 else
  return b;
endif;
}
```

Aufgabe 2

Was berechnet `getM(A: array_of_int)`? Beweisen sie ihre Behauptung! Welche Laufzeit benötigt dieser Algorithmus?

Lösung: `getM(A: array_of_int)` berechnet $\max\{x \in A\}$.

Beweis via Induktion über $n := |A|$.

- Induktionsanfang n=1:

  `getM` gibt das einzige Element aus, welches Maximum ist.

- Induktionsvoraussetzung:

  `getM` berechnet $\max\{x \in A\}$ für alle A mit $|A| < n$.

- Induktionsbehauptung:

  `getM` berechnet $\max\{x \in A\}$ für alle A mit $|A| = n$.

- Induktionsschluss:

  Betrachte $A[1 \ldots n]$ für $n > 1$. Dann wird für $m := \lfloor n/2 \rfloor$ zunächst $\texttt{getM}(A[1 \ldots m])$ und $\texttt{getM}(A[m+1 \ldots n])$ berechnet. Nach Induktionsvoraussetzung sind also

  $a = \max\{x | x \in A[1 \ldots m]\}$ und

  $b = \max\{x | x \in A[m+1 \ldots n]\}$.

  1.Fall: $a > b$.

  Es gilt $\forall c \in A[1 \ldots m] : a \geq c$ sowie

  $\forall c \in A[m+1 \ldots n] : a > b \geq c$ und somit

  $\forall c \in A[1 \ldots n] : a \geq c$. Dies ist auch der Rückgabewert.

  2.Fall: $a \leq b$.

  Analog. $\square$