

Norman May

B6, 29, Raum C0.05
68131 Mannheim
Telefon: (0621) 181-2517
Email: norman@pi3.informatik.uni-mannheim.de

Matthias Brantner

B6, 29, Raum C0.05
68131 Mannheim
Telefon: (0621) 181-2517
Email: msb@pi3.informatik.uni-mannheim.de

Algorithmen und Datenstrukturen
Wintersemester 2004/05

9. Lösungsvorschlag
14. Januar 2005

Aufgabe 1

15 Punkte

Implementieren Sie Rot-Schwarz-Bäume ohne Verwendung des Parent-Zeigers. Wenn Ihnen diese Implementierung zu schwer ist, geben Sie eine Implementierung mit Parent-Zeiger an (- 33%).

Befolgen Sie bei der Implementierung folgende Schnittstelle — Ihr Programm wird gegen diese Schnittstelle getestet. Weitere Hilfsfunktionen können hilfreich sein.

```
public class RBTreeNode {  
  
    /* liefere den Wert, der in dem Knoten gespeichert ist */  
    public Comparable getValue() { }  
  
    /* true, wenn roter Knoten, sonst false */  
    public boolean isRed() { }  
  
    /* liefere linken Kindknoten */  
    public RBTreeNode getLeft() { }  
  
    /* liefere rechten Kindknoten */  
    public RBTreeNode getRight() { }  
  
    /* Textrepräsentation des Knotenwerts und des Rot-Schwarzerts (e.g. 5 (r)) */  
    public String toString() { }  
  
}  
  
public class RedBlackTree {  
  
    /* liefere den NIL-Wert */  
    public static RBTreeNode getNil() { }  
  
    /* liefere den Wurzelknoten des Baums */  
    public RBTreeNode getRoot() { }  
  
}
```

```

/* erzeuge einen leeren red-black tree */
public RedBlackTree() { }

/* loesche Knoten z aus dem Baum */
public RBTreeNode delete(RBTreeNode z) { }

/* fuege value in den Baum ein */
public RBTreeNode insert(Comparable value) { }

/* suche nach einem Knotem mit Wert value */
public RBTreeNode lookup(Comparable value) { }
}

```

Lösung

(siehe BinaryTree.java, RBTreeTest.java, RBTreeNode.java und RedBlackTree.java, RBTree_P.java, RBTreeNode_P.java)

Punkteverteilung: RBTreeNode 1 Punkt, lookup 2 Punkte, delete und insert je 6 Punkte.

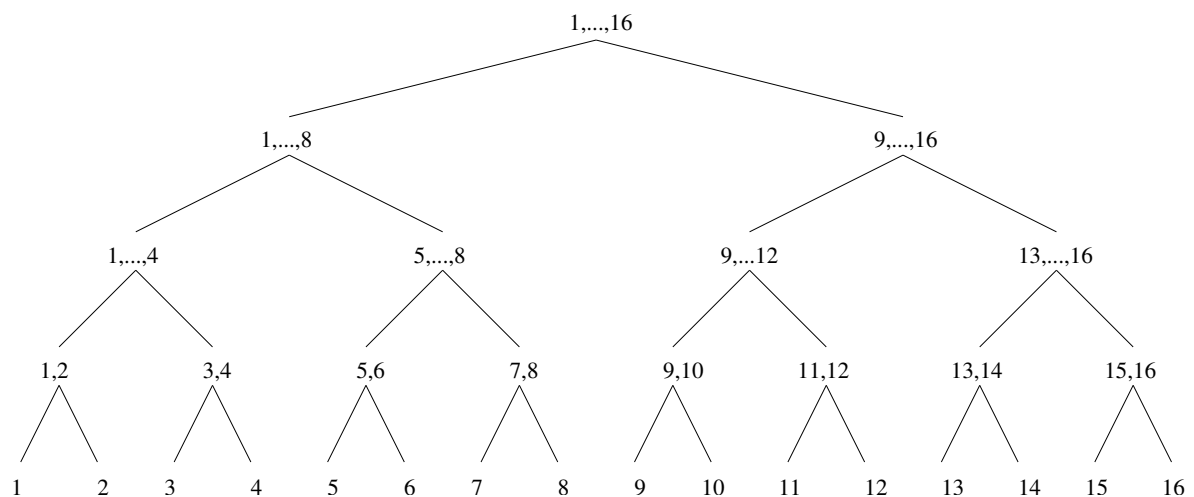
Aufgabe 2

4 Punkte

Zeichnen Sie den Rekursionsbaum von *MERGE-SORT* für ein Feld von 16 Elementen. Erklären Sie, warum *Memoization* zur Steigerung der Effizienz eines guten Teile-und-Herrsche-Algorithmus wie *MERGE-SORT* unwirksam ist.

Lösung

Rekursionsbaum für *Merge-Sort*:



Das Problem der Sortierung eines Arrays mit 16 Elementen wird durch den *MERGE-SORT*-Algorithmus in nicht überlappende Teilprobleme zerlegt.

Die Fibonacci-Zahlen sind definiert durch folgende Rekurrenz:

$$\begin{aligned}F_0 &= 0 \\F_1 &= 1 \\F_i &= F_{i-1} + F_{i-2}\end{aligned}$$

Aufgabe 3 a)

2 Punkte

Geben Sie am Beispiel der Berechnung von F_{10} an, wieviele Teilberechnungen in diesem Beispiel insgesamt redundant ausgeführt werden?

Lösung

$$\begin{aligned}F_{10} &= F_9 + F_8 \\&= F_8 + F_7 + F_7 + F_6 = F_8 + 2 \cdot F_7 + F_6 \\&= F_7 + F_6 + 2(F_6 + F_5) + F_5 + F_4 = F_7 + 3 \cdot F_6 + 3 \cdot F_5 + F_4 \\&= \dots\end{aligned}$$

Insgesamt 79 duplizierte Rechnungen (siehe Fibonacci.java).

Aufgabe 3 b)

3 Punkte

Implementieren Sie die Fibonacci-Zahlen und verwenden Sie Memoization, um wiederholte Berechnungen von F_i zu vermeiden.

Lösung

(siehe Fibonacci.java)

Aufgabe 3 c)

3 Punkte

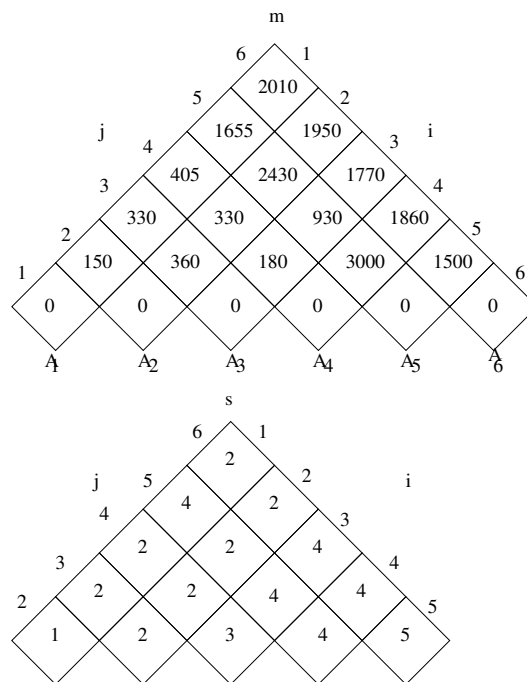
Implementieren Sie die Berechnung der Fibonacci-Zahlen mit dynamischem Programmieren.

Lösung

(siehe Fibonacci.java)
Übrigens gilt für $k \geq 0$: $F_{k+2} = 1 + \sum_{i=0}^k F_i$.

Sei $\langle A_1, A_2, A_3, A_4, A_5, A_6 \rangle$ eine Matrizen­sequenz und $\langle 5, 10, 3, 12, 5, 50, 6 \rangle$ die dazugehörige Dimensionensequenz. Bestimmen Sie ein optimal geklam­mertes Produkt der gegebenen Matrizen­sequenz.

Lösung



Optimale Klammerung:

$$(A_1 A_2)((A_3 A_4)(A_5 A_6))$$

Aufgabe 5

3 Punkte

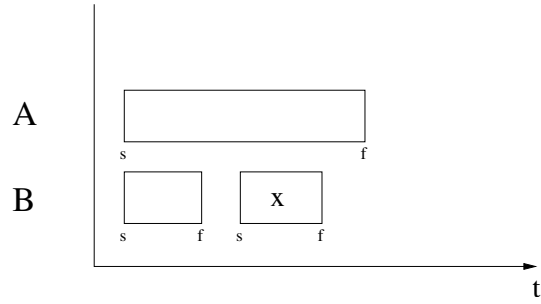
Sei das in der Vorlesung vorgestellte “Aktivitätsauswahlproblem” gegeben. Ist dieses Problem ein Matroid? Beweisen Sie Ihre Aussage.

Lösung

Sei E die Menge der Aktivitäten und S die Menge der kompatiblen Aktivitäten.

Beweis durch Gegenbeispiel:

Sei $A \in S$ und $B \in S$ mit $|B| < |A|$. Dann erfüllt das Element x nicht die Austausch­eigenschaft, da $A' = A \cup \{x\} \notin S$. Die Elemente von A' sind nicht mehr kompatibel.

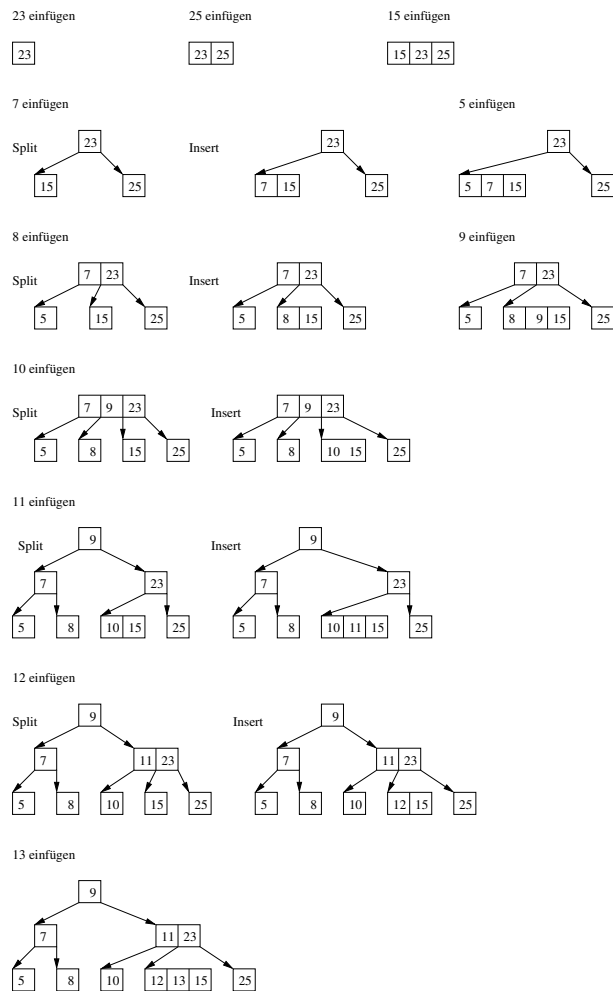


Aufgabe 6

5 Punkte

Die Schlüssel 23, 25, 15, 7, 5, 8, 9, 10, 11, 12, 13 sollen in einen B-Baum mit $t = 2$ eingefügt werden. Geben Sie den B-Baum an, der durch das Einfügen der Schlüssel in einen leeren B-Baum entsteht.

Lösung



Aufgabe 7

9 Punkte

Zeigen oder widerlegen Sie folgende Aussagen.

Aufgabe 7 a)

3 Punkte

$$f(n) = \Theta(f(n/2))$$

Lösung

Gegenbeispiel: $f(n) = 2^n$.

$$\begin{aligned} 0 &\leq 2^n \leq c2^{n/2} \\ \Leftrightarrow 0 &\leq 2^{n/2}2^{n/2} \leq c2^{n/2} \\ \Leftrightarrow 0 &\leq 2^{n/2} \leq c \end{aligned}$$

Was ein Widerspruch zur Unbeschränktheit der Natürlichen Zahlen ist.

Aufgabe 7 b)

3 Punkte

$$f(n) = O(g(n)) \Rightarrow g(n) = \Omega(f(n))$$

Lösung

$$\begin{aligned} f(n) &= O(g(n)) \\ \Rightarrow 0 &\leq f(n) \leq cg(n) \\ \Rightarrow 0 &\leq 1/cf(n) \leq g(n) \\ \Rightarrow g(n) &= \Omega(f(n)) \end{aligned}$$

Aufgabe 7 c)

3 Punkte

$$o(f(n)) + \omega(f(n)) = \Theta(f(n))$$

Lösung

Gegenbeispiel: $f(n) = n^2$.

Für $f(n) = n^2$ gilt $n^3 = \omega(f(n))$ und $n = o(f(n))$.

Da $o(f(n)) + \omega(f(n)) = \Theta(\max(o(f(n)), \omega(f(n))))$ gilt, und $n^3 \neq \Theta(n)$, ist die Aussage falsch.