RWTH Aachen **SS 2011**
Lehrgebiet Theoretische Informatik Blatt 1
Emmes—Langer—Reidl—Rossmanith 18.4.2011

Übung zur Vorlesung Datenstrukturen und Algorithmen

Eigenständige Präsenzübung (Gruppe A)

Name: _____ Matrikelnummer: _____

Alle Antworten sind zu beweisen!

Aufgabe 1 (10 Punkte)

Gegeben sei ein Array $a[n]$ mit n Elementen $a[0]$ bis $a[n-1]$. Der folgende Algorithmus soll testen, ob ein Element x im Array enthalten ist oder nicht.

function *find3*(**int** x) **boolean** :
$temp := a[n-1]$;
$a[n-1] := x$;
$i := 0$;
while $a[i] \neq x$ **do** $i := i+1$ **od**;
$a[n-1] := temp$;
return $i < n-1$

Leider liefert dieser Algorithmus manchmal das falsche Ergebnis. Erklären Sie, wann und warum dieser Fehler auftritt, und wie man ihn beheben kann.

RWTH Aachen **SS 2011**
Lehrgebiet Theoretische Informatik Blatt 1
Emmes—Langer—Reidl—Rossmanith 19.4.2011

Übung zur Vorlesung Datenstrukturen und Algorithmen

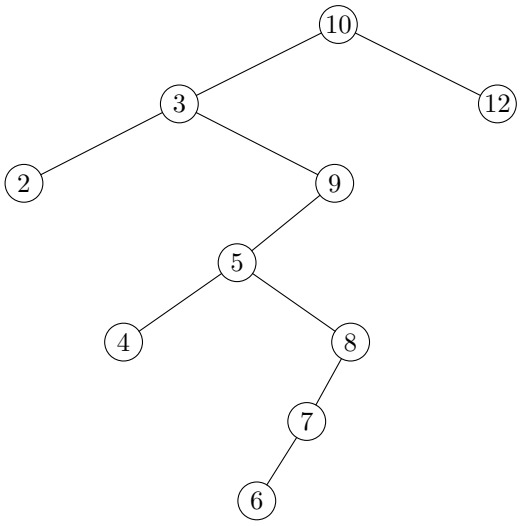
Eigenständige Präsenzübung (Gruppe B)

Name: _____ Matrikelnummer: _____

Alle Antworten sind zu beweisen!

Aufgabe 1 (10 Punkte)

Gegeben sei der folgende binäre Suchbaum:




```
void delete() {
  if(left ≡ null && right ≡ null) {
    if(parent.left ≡ this) parent.left = null;
    else parent.right = null; }
  else if(left ≡ null) {
    if(parent.left ≡ this) parent.left = right;
    else parent.right = right;
    right.parent = parent; }
  else {
    Searchtreenode⟨K, D⟩ max = left;
    while(max.right ≠ null) max = max.right;
    copy(max); max.delete();
  }
}
```

Konstruieren Sie den binären Suchbaum, welcher sich ergibt, wenn das Element 10 gelöscht wird. Als mögliche Hilfe ist die relevante Methode der Klasse *Searchtreenode*$\langle K, D\rangle$ angegeben. Erklären Sie, was hier beim Löschen passiert.

RWTH Aachen **SS 2011**
Lehrgebiet Theoretische Informatik Blatt 1
Emmes—Langer—Reidl—Rossmanith 20.4.2011

Übung zur Vorlesung Datenstrukturen und Algorithmen

Eigenständige Präsenzübung (Gruppe C)

Name: _____ Matrikelnummer: _____

Alle Antworten sind zu beweisen!

Aufgabe 1 (10 Punkte)

Gegeben sei ein sortieres Array $a[n]$ mit n Elementen $a[0]$ bis $a[n-1]$, die in aufsteigender Reihenfolge sortiert sind. Betrachten Sie den folgenden Algorithmus, der mit Hilfe binärer Suche testet, ob das Element x im Array enthalten ist.

function *binsearch*(**int** x) **boolean** :
$l := 0; r := n - 1;$
while $l \leq r$ **do**
  $m := \lfloor (l + r)/2 \rfloor;$
  if $a[m] < x$ **then** $l := m + 1$ **fi**;
  if $a[m] > x$ **then** $r := m - 1$ **fi**;
  if $a[m] = x$ **then return** *true* **fi**
od;
return *false*

Beweisen Sie, daß der Algorithmus terminiert. Hinweis: Wie verhält sich $r - l$?