Information Hiding
Binärbäume

Dr. Reinhard Wolfinger

Email: reinhard.wolfinger@jku.at
Telefon: 0732/2468-7134
Sprechstunde: Dienstag, 14:00-15:00

Institut für Systemsoftware
Hochschulfondgebäude
3. Stock, Raum HF301

Häufige Fehler (1)

bei Übung 1: Zahlenmenge als verkettete Liste

```
public boolean get(int val) {
  Node p = head;
  while(p != null) {
    if(p.val == val)  break;
    p = p.next;
  }
  if(p != null)
    return true;
  else
    return false;
}
```

```
public boolean get(int val) {
  Node p = head;
  while(p != null) {
    if(p.val == val)  break;
    p = p.next;
  }
  return (p != null);
}
```

```
public boolean get(int val) {
  Node p = head;
  while(p != null) {
    if(p.val == val)  return true;
    p = p.next;
  }
  return false;
}
```

Häufige Fehler (2)

```
public boolean get(int val) {
  boolean found = false;
  if(size() != 0) {
    Node p = head;
    while(p != null && p.val != val) {
      p = p.next;
    }
    if(p != null)
      found = true;
  }
  return found;
}
```

```
found = (p != null);
```

```
public boolean get(int val) {
  Node p = head;
  while(p != null && p.val != val) {
    p = p.next;
  }
  return (p != null);
}
```

```
public boolean get(int val) {
  for(Node p = head; p != null; p = p.next) {
    if(p.val == val)  return true;
  }
  return false;
}
```

Häufige Fehler (3)

```
void set(int val) {
   if(get(val) == false) {
     Node q = new Node(val);
     q.next = head;
     head = q;
   }
}
```

```
void set(int val) {
   if(!get(val)) {
     Node q = new Node(val);
     q.next = head;
     head = q;
   }
}
```

```
void set(int val) {
   if(get(val)) return;
   Node q = new Node(val);
   q.next = head;
   head = q;
}
```

Häufige Fehler (4)

```
public void set(int val) {
   boolean truth = false;
   Node p = head;
   while(p != null) {
      if(p.val == val) {
        truth = true;
      }
      p = p.next;
   }
   if(truth == false) {
      insert(val);
    }
}
```

```
void set(int val) {
   if(!get(val))
    insert(val);
}
```

```
public void insert(int val) {
  Node q = new Node(val);
  q.next = head;
  head = q;
}
```

```
void set(int val) {
  if(get(val)) return;
  Node q = new Node(val);
  q.next = head;
  head = q;
}
```

Information Hiding

```
public class Tree {
  Node head;
  public Tree() { … }
  public void insert(String name) { … }
  public boolean search(String name) { … }
  public boolean remove(String name) { … }
}
public class Node {
  String name;
  Node left, right;
  Node(String name) { … }
}
```

- Ist "Information Hiding" verletzt?
- Können Variablen oder Methoden weniger restriktiv deklariert werden, ohne das "Information Hiding" aufzuweichen?

Lineare Liste vs. Binärer Suchbaum

Eine sortierte verkette lineare Liste und ein vollständiger binärer Suchbaum enthalten die gleiche Anzahl von n String-Elementen.

- Wie viele Stringvergleiche sind jeweils minimal, durchschnittlich, maximal nötig um ein Element zu finden?
- Angenommen der binäre Suchbaum ist maximal entartet? Wie ändern sich die Zahlen (minimal, durchschnittlich, maximal)?
- Welche Tiefe hat der binäre Suchbaum im entarteten Fall?

Einfügen in binären Suchbaum

Binärer Suchbaum für Integer-Werte
Suchbaum-Ordnung:
  Elemente linken Unterbaumr < Wurzelelement < Elemente rechter Unterbaum

- Einfügen 6, 12, 4, 13, 9, 7, 8, 5, 3, 1, 20
- Welche max. Tief hat der Baum?
- Fügen sie Element 7 erneut ein? geht das? Welche Möglichkeiten gibt es?

Binärer Suchbaum

Welche der folgenden Aussagen stimmen (nicht)? Und warum?

- Der Grad eines Knoten ist die Anzahl der Knoten, die auf ihn verweisen.
- Die Anzahl der Blätter eines vollständigen Baumes wächst exponentiell mit dessen Höhe.
- Die Höhe eines beliebigen Baumes mit n Knoten ist ungefähr ld(n).
- In einem Binärbaum hat jeder Knoten Grad 2.
- Binäre Suchbäume sind entweder aufsteigend oder absteigend sortiert.
- Von der Wurzel bis zu einem beliebigen Blatt gibt es immer genau einen Pfad.