Programmierung – Nachklausurtutorium

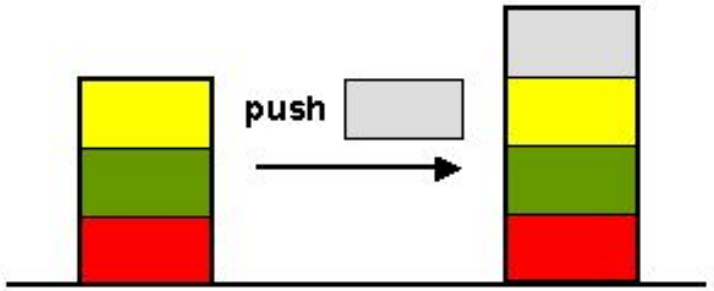
Laryssa Horn, Tim Engelhardt

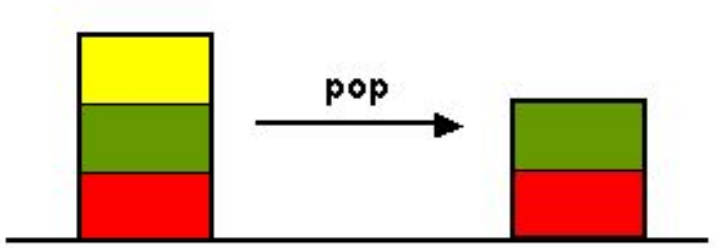
22 März 2018

Datenstrukturen

Stack

- Vorstellung wie ein Stapel Teller
  - LIFO -> Last in First out





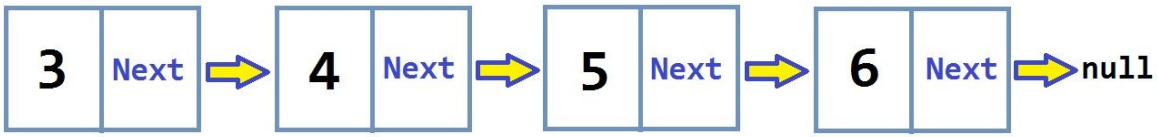
Grundlegende Operationen

- push(Object obj) -> füge ein Element hinzu
- pop() -> entferne das letzte Element
- isEmpty()

Quelle: http://wvsg.schulen2.regensburg.de/joomla/images/Faecher/Informatik/Informatik_11/informatik_11_1_6_Datenstruktur_Stapel.html

Linked-List

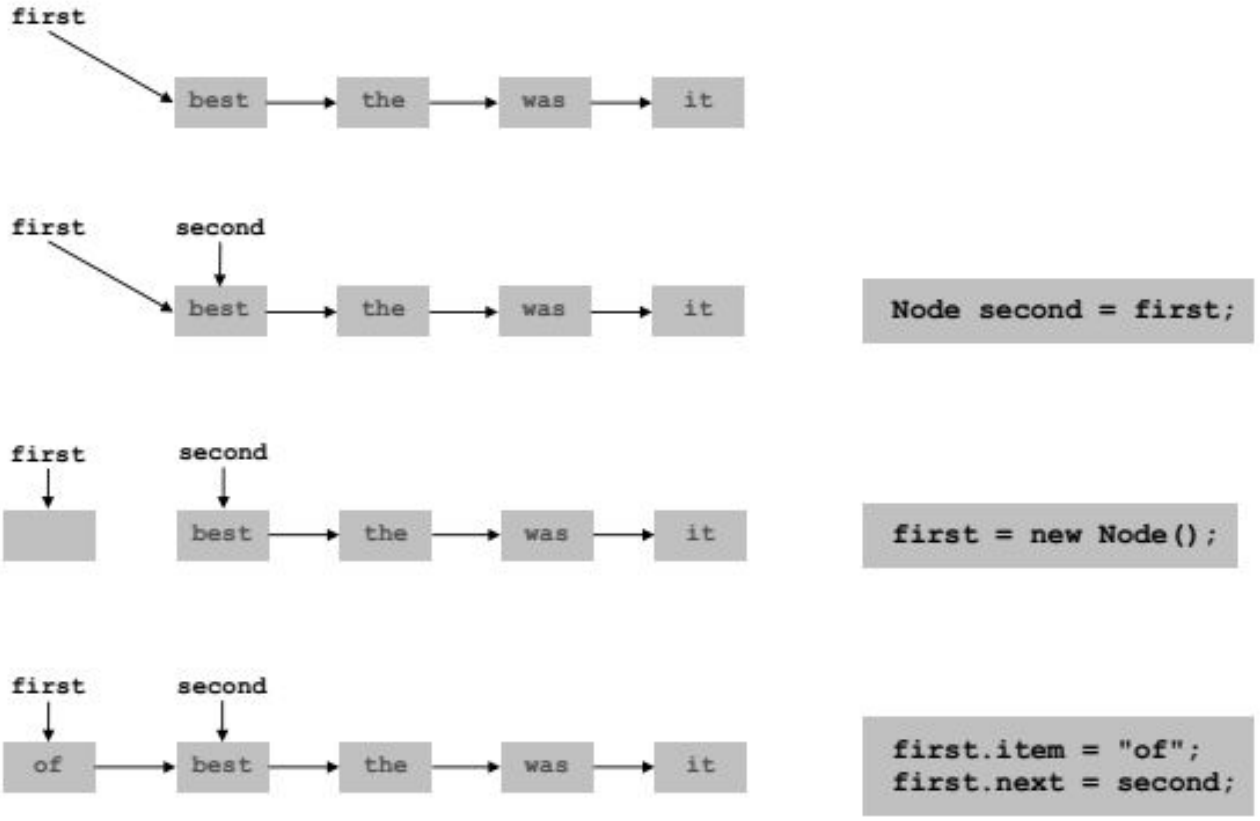
- Rekursive Datenstruktur
- Ein Element der List enthält einen Wert und einen Pointer auf das nächste Element



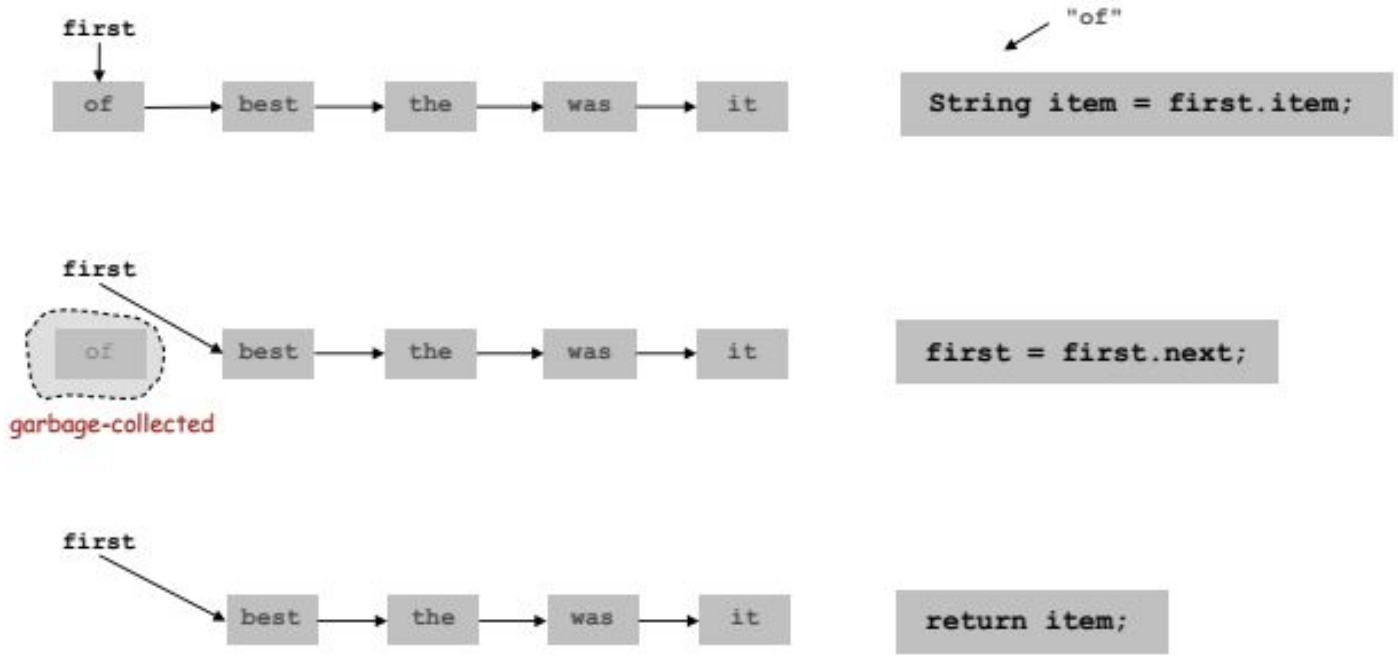

```
public class Node {
  private String item;
  private Node next;
}
```

Quelle: Vorlesung Prof. Harmeling 17/18
http://cforbeginners.com/linkedlists.html

Linked-List Element einfügen



Element löschen



Generics <...>

- Zum Zeitpunkt der Klassenimplementierung ist die erforderte Klasse noch unbekannt

->Verwende Typ-Variable / Generic

Bsp.: Mit welchem Objekt befülle ich meine Liste?

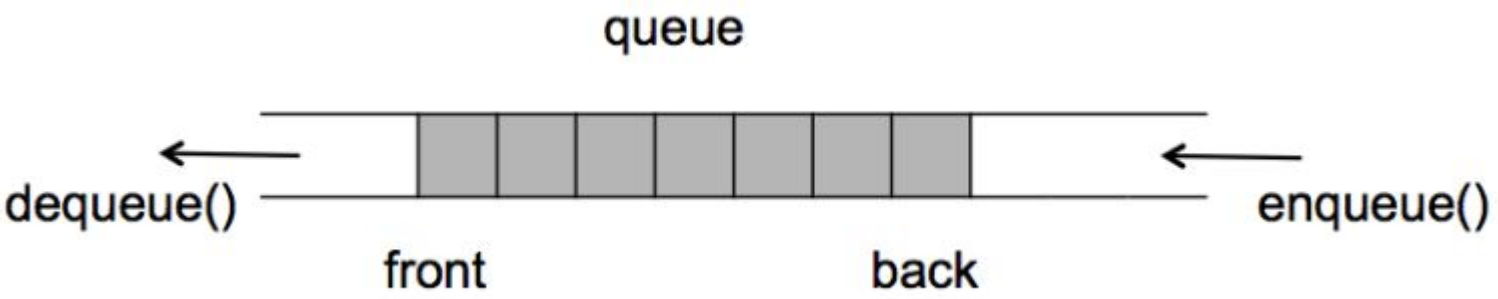
LinkedList<object> liste = new LinkedList <object>

(hier wird eine Liste mit Strings erstellt)
LinkedList<String> liste = new LinkedList <String>

Queue

FIFO -> First in First out

enqueue() -> hängt ein Element an das Ende der Liste
dequeue() -> entfernt das
length() -> gibt die Länge der Liste zurück



Quelle: https://www.5balloons.info/implementation-of-queue-data-structure-in-java/

Symbol Tabellen (Map,…)

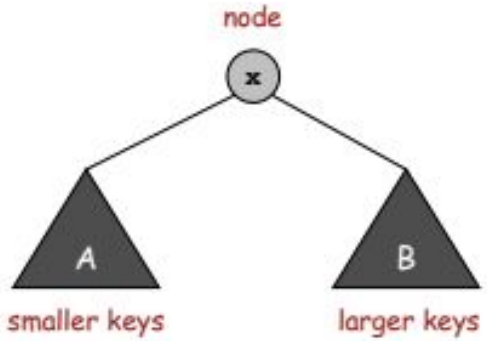
- jedes Element einer Map besteht aus einem Schlüssel (key) und dem zugehörigen Wert (value)
- Jeder Schlüssel darf in einer Map nur genau einmal vorhanden sein
- Es können beliebige Objekte hinzugefügt oder entfernt werden

Key	Value
1	„Max Mustermann“
2	„Hans Otto“

Binärer Suchbaum

- Symmetrische Anordnung

der Werte (kleinere Werte entsprechen dem linken, größere Werte dem rechten Teilbaum)



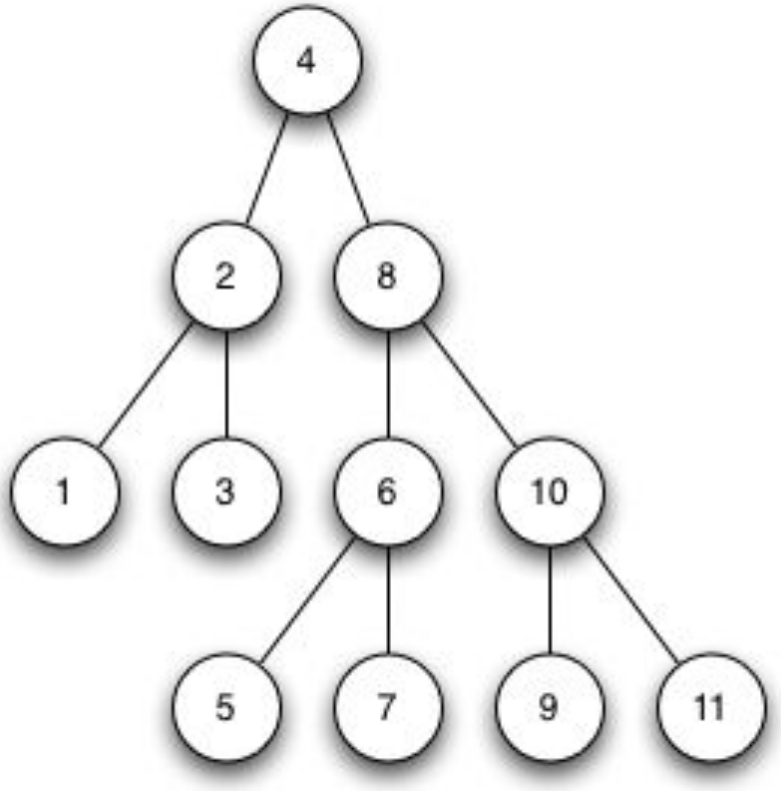
- Ähnlich wie die LinkedList besitzen die einzelnen Knoten Referenzen auf den nächsten Knoten

```
public class LinkedListNode {
  private String item;
  private LinkedListNode next;
}
```

Unterschied hier: Es werden auf 2 Knoten gezeigt

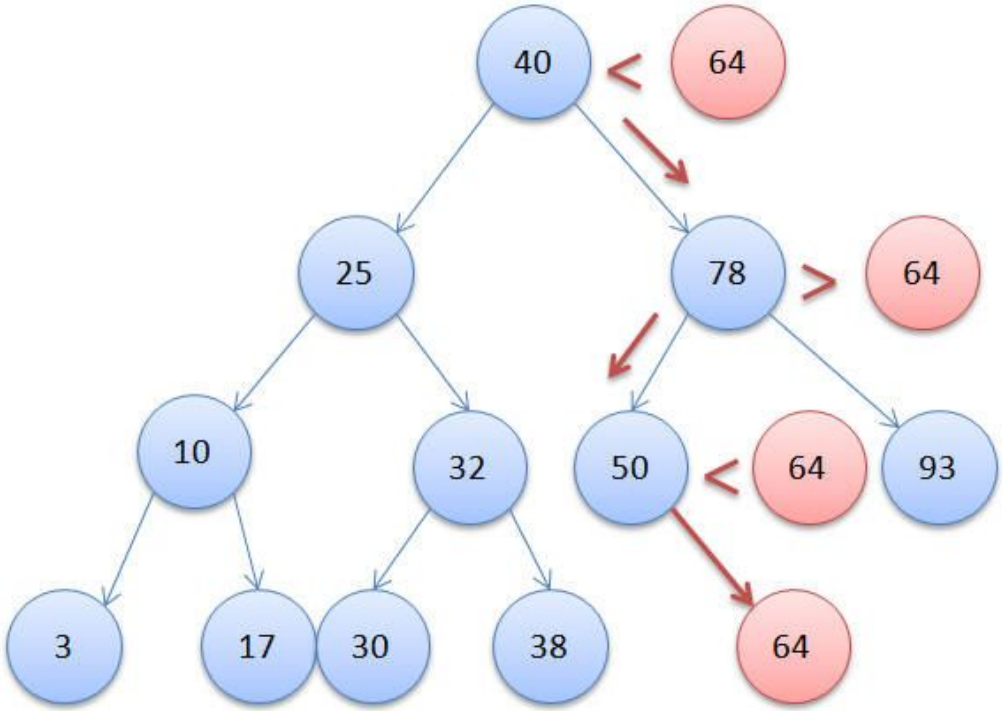
```
private class BTreeNode {
   private Key key;
   private Value val;
   private BTreeNode left;
   private BTreeNode right;
}
```

Binärer Suchbaum Beispiel



Quelle: http://www.scalingbits.com/book/export/html/664

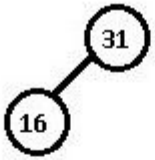
Binärer Suchbaum Suche



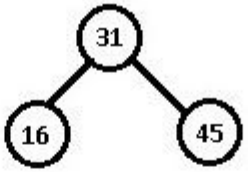
- Vergleiche den gesuchten Wert mit dem Wert der Wurzel
- Ist der gesuchte Wert kleiner, vergleiche den nächsten Wert des linken Teilbaums
- Ist der gesuchte Wert größer, vergleiche den nächsten Wert des rechten Teilbaums

Binärer Suchbaum Einfügen

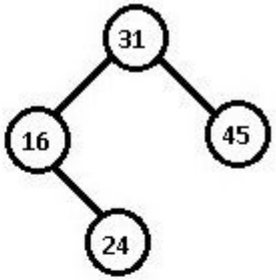
Insert 31



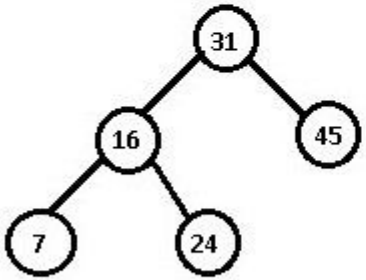
Insert 16



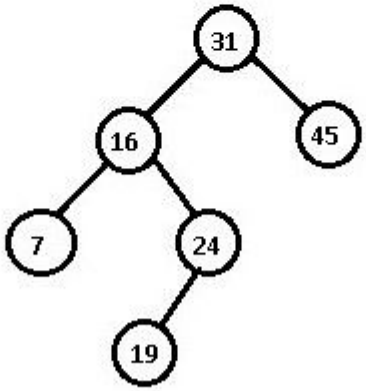
Insert 45



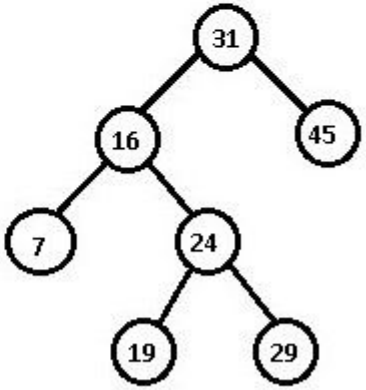
Insert 24



Insert 7



Insert 19



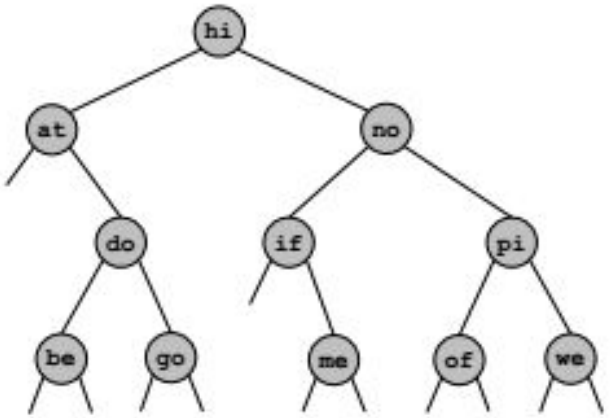
Insert 29

Binärer Suchbaum Ausgabeformen

L -> Links
R -> Rechts

W -> Wurzel

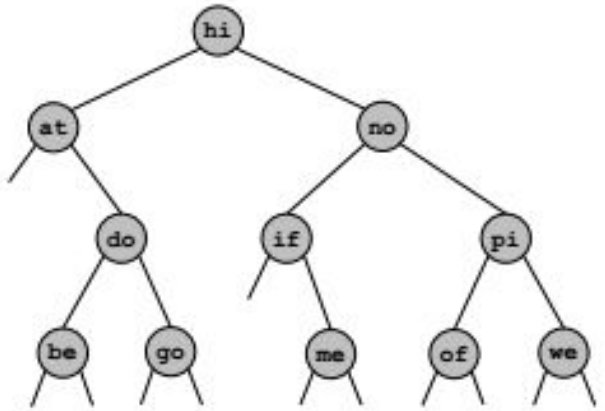
- Inorder -> L W R



inorder: at be do go hi if me no of pi we

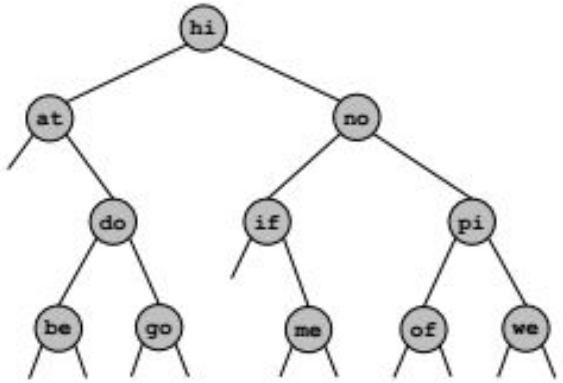
Binärer Suchbaum Ausgabeformen

- Preorder -> W L R



preorder: hi at do be go no if me pi of we

- Postorder -> L R W



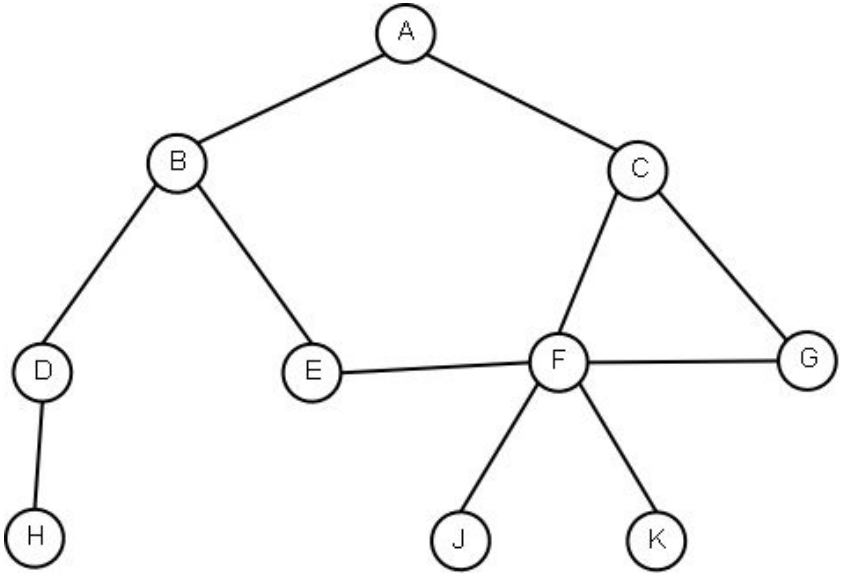
postorder: be go do at me if of we pi no hi

Set

- Ungeordnete Sammlung/ Menge an Elementen
- Jedes Element darf nur ein Mal in einem Set vorkommen
- Vergleichbar mit der mathematischen Menge

Graph

- Repräsentiert Verbindungen (Kanten/ Edge) zwischen verschiedenen Elementen (Knoten/ Vertex)



- Graphen werden verwendet, um verschiedenste Anwendungen übersichtlich darzustellen (Bsp. Straßenbahnnetz,…)

Beispiel-Operationen:

addEdge(Element e1, Element e2)