Übung 8: Hashing

Abgabetermin: 24.05.2011

Name: _____ **Matrikelnummer:** _____

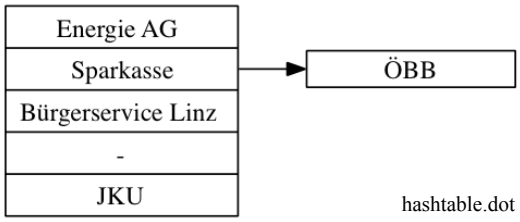
Gruppe: ☐ **G1** Di 10:15 ☐ **G2** Di 11:00 ☐ **G3** Di 12:45

Aufgabe	Punkte	gelöst	abzugeben schriftlich	abzugeben elektronisch	Korr.	Punkte
Aufgabe 1	24	☐	Java-Programm, Testfälle und Ergebnisse	Java-Programm	☐	

Aufgabe 1: Hashtabelle mit Überlauflisten

Implementieren Sie eine Hashtabelle (hash map) für beliebige Objekte. Bei Kollisionen verwenden Sie Überlauflisten. Die Hashtabelle soll mit Länge 1 initialisiert werden und beim Einfügen automatisch wachsen, wenn der im Konstruktor definierte maximale Füllfaktor überschritten wird. Vergrößern Sie um den Faktor 2 und führen Sie ein "rehashing" durch. Setzen Sie die neue Länge der Hashtabelle nach dem Vergrößern auf die nächstgrößere Primzahl. Doppelte Schlüssel sind nicht erlaubt, d.h. wenn die *put*-Methode wiederholt mit dem selben Schlüssel aufgerufen wird, soll der Wert ersetzt werden.

```
HashMap m = new HashMap(0.9f);
m.put("JKU", "0732/2468");
m.put("Bürgerservice Linz", "0732/7070");
m.put("ÖBB", "05-1717");
m.put("Energie AG", "0800-81 8000");
m.put("Sparkasse", "50-100-10100");
m.makeDot("hashtable.dot");
Out.println(m.get("ÖBB"));
// Ausgabe: 05-1717;
```



hashtable.dot

Implementieren Sie die Klassen *HashMap* und *Node* mit folgenden Schnittstellen (alles *public*):

```
class HashMap {
  class Node {
    Object key;
    Object value;
    Node next;
    Node(Object key, Object value) {...}
  }
  Node tab[];

  HashMap(float maxFillFactor) { ... }
  void put(Object key, Object value) { ... }
  Object get(Object key) { ... }
  Object remove(Object key) { ... }
  void makeDot(String filename) { ... }
}
```

Implementierungshinweise:

a) Die Methode *makeDot(String filename)* erstellt eine Datei zur Anzeige in GraphViz (http://graphviz.org). Verwenden Sie dazu *DotMakerHashtable.makeDot(HashMap m)* aus der Vorgabe. Achtung, die Vorgabe setzt die Schnittstelle von *HashMap* und *Node* wie oben spezifziert voraus.

b) Die Methode *PrimeNumbers.nextPrime(int n)* aus der Vorgabe liefert zu einer gegebenen Zahl die nächstgrößere Primzahl.

Abzugeben ist: Java-Programm, Testfälle und Ergebnisse