Übung 9: Hashing

Abgabetermin: 20.05.2014

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 linearem Probieren (24 Punkte)

Implementieren Sie ein assoziatives Datenfeld (Map) für beliebige Objekte mit Hilfe einer Hashtabelle. Verwenden Sie dabei Strings als Schlüssel (Key), um die enthaltenen Elemente (Value) zu adressieren. Bei Kollisionen verwenden Sie lineares Probieren als Kollisionsstrategie. Die Schnittstelle ist durch die abstrakte Klasse *Map* gegeben (für Methodenbeschreibungen siehe Java-Dokumentation in der Vorgabedatei).

```
package at.jku.ssw;
public abstract class Map {
  public abstract boolean containsKey(String key);
  public abstract Object get(String key);
  public abstract void put(String key, Object value);
  public abstract Object remove(String key);
}
```

Implementieren Sie die Klasse *HashMap* im Paket *at.jku.students* wie folgt mit der Vorgabeklasse *Element* aus dem Paket *at.jku.ssw*.

```
class HashMap extends Map {
  Element[] table;
  HashMap(float fillFactor) {  ... }
  String makeDot() {  ... }
  String makeDot(float width) {  ... }
  ...
}
class Element {
  final String key;
  final Object value;
  boolean isDeleted;
  Element(String key, Object value) {...}
}
```

```
Map m = new HashMap(0.9f);
m.put("Flughafen Wien", "01 7007 0");
m.put("BAWAG", "05 99 05");
m.put("Hypo Alpe Adria", "05 02 02-0");
m.put("BUWOG", "01 878 28 1130");
m.put("Meinl Bank AG", "01 53 18 80");
Out.println(m.get("Hypo Alpe Adria"));
// Ausgabe: 05 02 02-0
Out.open("hashtable.dot");
Out.println(((HashMap)m).makeDot());
Out.close();
```

Implementierungshinweise:

a) Initialisieren Sie die Hashtabelle mit Länge 1 und vergrößern Sie automatisch, wenn der im Konstruktor definierte maximale Füllfaktor überschritten wird. Vergrößern Sie um den Faktor 2, setzen Sie die Länge auf die nächstgrößere Primzahl und führen Sie ein *rehashing* durch. Der maximale Füllfaktor muss größer 0.0 und kleiner oder gleich 1.0 sein.

b) Berechnen Sie den Hashwert der Schlüssel mit der Hashfunktion *hashCode*() in der Klasse *String*.

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

d) Definieren Sie für alle Klassen, Methoden und Felder die geeignete Sichtbarkeit (private, protected, package, public).

e) Verwenden Sie die Methoden *DotMaker.makeDotForHashtable(Element[] table)* und *DotMaker.makeDotForHashtable(Element[] table, float width)*, um GraphViz-Bilder Ihrer Streuwerttabelle zu erzeugen.

Abzugeben ist: Java-Programm, Testergebnisse