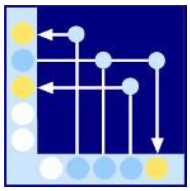
Fakultät Elektronik und Informatik
Studiengang Informatik



Algorithmen und Datenstrukturen 2

Vorlesung im Wintersemester 2017/2018
Prof. Dr. habil. Christian Heinlein

1. Praktikumsaufgabe (6. – 27. November 2017)

Aufgabe 1: Streuwerttabellen

Auf der Vorlesungswebseite befindet sich eine Datei `hashing.java` mit folgenden – teilweise unvollständigen – Schnittstellen und Klassen:

- `interface HashFunction`
  Schnittstelle einer Streuwertfunktion mit Einschränkung des Wertebereichs, die zu jedem Schlüssel einen gültigen Streuwert/Index liefert
- `abstract class AbstractHashFunction implements HashFunction`
  Abstrakte Oberklasse für Implementierungen von Streuwertfunktionen
- `class DivisionMethod extends AbstractHashFunction`
  Streuwertfunktion gemäß Divisionsrestmethode
- `class MultiplicationMethod extends AbstractHashFunction`
  Streuwertfunktion gemäß Multiplikationsmethode (Implementierung mit 32-Bit-Ganzzahlarithmetik)
- `interface HashSequence`
  Schnittstelle einer Sondierungsfunktion für offene Adressierung, die zu jedem Schlüssel eine gültige Sondierungssequenz liefert
- `abstract class AbstractHashSequence implements HashSequence`
  Abstrakte Oberklasse für Implementierungen von Sondierungsfunktionen
- `class LinearProbing extends AbstractHashSequence`
  Sondierungssequenz gemäß linearer Sondierung
- `class QuadraticProbing extends AbstractHashSequence`
  Sondierungssequenz gemäß quadratischer Sondierung
- `class DoubleHashing extends AbstractHashSequence`
  Sondierungssequenz gemäß doppelter Streuung
- `interface HashTable`
  Schnittstelle einer Streuwerttabelle
- `class HashTableChaining implements HashTable`
  Implementierung von Streuwerttabellen mit Verkettung
- `class HashTableOpenAddressing implements HashTable`
  Implementierung von Streuwerttabellen mit offener Adressierung
- `class WordCount`
  Einfaches Testprogramm

Implementieren Sie alle leeren und fehlenden Methoden (sowie die hierfür erforderlichen Datenstrukturen), das heißt:

- die Methode `compute` der Klassen `DivisionMethod` und `MultiplicationMethod`
  (Der Streuwert eines Schlüssels `key` soll jeweils aus dem Resultatwert des Methodenaufrufs `key.hashCode()` bestimmt werden, indem dieser geeignet in den Wertebereich $\{ 0, \ldots, N - 1 \}$ abgebildet wird.
  Achtung: Der von `hashCode` gelieferte Wert kann negativ sein!)
- die Methoden `first` und `next` der Klassen `LinearProbing` , `QuadraticProbing` und `DoubleHashing`
- die Methoden `put`, `get`, `remove` und `dump` sowie den Konstruktor der Klassen `HashTableChaining` und `HashTableOpenAddressing`

Beachten Sie die in der Datei enthaltenen Kopfkommentare der Methoden, die ihr Verhalten spezifizieren!

Um automatisierte Tests aller Implementierungen zu ermöglichen, dürfen die vorgegebenen Namen der Schnittstellen, Klassen und Methoden nicht verändert werden und es dürfen keine `package`-Deklarationen verwendet werden. Außerdem sollten keine Diagnoseausgaben produziert werden.

Es dürfen keine Klassen aus dem Paket `java.util` verwendet werden!

Abzugeben ist entweder eine einzige Java-Datei, die alle Schnittstellen und Klassen enthält, oder eine Zip-Datei mit mehreren Java-Dateien auf oberster Ebene (d. h. keine Unterordner!).

Einfaches Testbeispiel

Wenn die Eingabedatei `words` aus den Wörtern `eins`, `zwei`, `drei`, `zwei`, `eins`, `eins` (in dieser Reihenfolge) besteht, muss das Testprogramm z. B. folgende Ausgaben produzieren:

```
$ java WordCount c d 4 < words
1 zwei 2
1 eins 3
2 drei 1

$ java WordCount l d 4 < words
1 eins 3
2 zwei 2
3 drei 1

$ java WordCount q d 4 < words
1 eins 3
2 zwei 2
3 drei 1

$ java WordCount d d 4 < words
1 eins 3
2 zwei 2
3 drei 1
```

```
$ java WordCount c m 2 3 < words
0 drei 1
0 zwei 2
0 eins 3

$ java WordCount l m 2 3 < words
0 eins 3
1 zwei 2
2 drei 1

$ java WordCount q m 2 3 < words
0 eins 3
1 zwei 2
3 drei 1

$ java WordCount d m 2 3 < words
0 eins 3
1 zwei 2
2 drei 1
```

Um die Korrektheit einer Implementierung umfassend zu testen, sind aber zahlreiche weitere Tests – auch mit anderen Typen als `String` und `Integer` – erforderlich!

Organisatorische Hinweise

Beachten Sie hierzu das Dokument „Wichtige Hinweise zur Prüfung“ auf der Vorlesungswebseite!