ADF 2*x* & PRO 2*x* **Übungen zu Fortgeschrittenen Algorithmen & Datenstrukturen und OOP** **SS 17, Übung 1**

Abgabetermin: Mi in der KW 13

☐ **Gr. 1, Dr. G. Kronberger** **Name** _____ **Aufwand in h** _____
☐ **Gr. 2, Dr. H. Gruber**
☐ **Gr. 3, Dr. D. Auer** **Punkte** _____ **Kurzzeichen Tutor / Übungsleiter** _____ / _____

1. Index-Generator (18 Punkte)

Gesucht ist ein Pascal-Programm *IndexGen*, das für einen gegebenen Text (in einer Textdatei) einen *Index* erzeugt. Ein Index ist die lexikographisch sortierte Liste aller Wörter des Texts, wobei für jedes Wort in aufsteigend sortierter Reihenfolge die Nummern all jener Zeilen angegeben ist, in denen das Wort im Text vorkommt. Dabei ist zwischen Groß- und Kleinschreibung nicht zu unterscheiden, alle Wörter können deshalb z. B. in Kleinbuchstaben umgesetzt werden.

Beispiel:

Text:
```
Ach wie gut, dass niemand weiß,
dass ich Rumpelstilzchen heiß.
```

Index (nur auszugweise dargestellt):
```
ach       1
...
dass      1, 2
...
wie       1
```

Ihr Programm muss mit

```
IndexGen  InputFileName.txt
```

aufgerufen werden können (der Name der Textdatei wird also in Form eines Kommandozeilen-Parameters übergeben) und muss den Index auf die Standardausgabe schreiben. Der Index kann dann bei Bedarf mit Hilfe von Ausgabeumleitung auch in eine Datei umgeleitet werden, z. B. mit

```
IndexGen  InputFileName.txt > IndexFileName.txt
```

Verwenden Sie eine Hashtabelle zur Verwaltung der Einträge (= Wort mit seinen Zeilennummern). Vor Ausgabe des Ergebnisses sind die Wörter im Index mit Quicksort zu sortieren. Testen Sie Ihre Lösungen ausführlich, indem Sie für das Fachhochschul-Studiengesetz (in der Datei *FHStG2011.txt* im moodle-Kurs) einen Index generieren und vergleichen Sie Ihre Lösung auch mit jener von KollegInnen um festzustellen, welche effizienter ist. Dazu können Sie das Modul *Timer* (im moodle-Kurs) verwenden. Um längere Laufzeiten zu erhalten, sollten Sie sich auch (wieder einmal?) mit Franz Kafka beschäftigen (siehe *Kafka.txt* moodle-Kurs).

Bemerkungen: Da das Thema Dateibearbeitung noch nicht besprochen wurde, finden Sie im moodle-Kurs in *IndexGen.pas* eine Vorlage für das zu erstellende Programm.

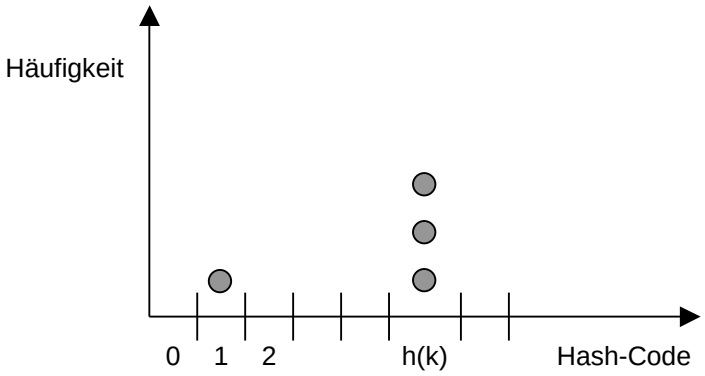
2. Güte von Hash-Funktionen (6 Punkte)

Betrachtet werden Hash-Funktionen *h*, die Wörter (Schlüssel, engl. *keys*) *k* auf *n* positive ganze Zahlen (engl. *hash codes*) *hc* = *h*(*k*) im Bereich von 0 bis *n* - 1 abbilden. Diese Hash-Codes können zum Indizieren von Hash-Tabellen verwendet werden. Die Güte einer Hash-Funktion wird neben ihrer Effizienz (geringer Aufwand zur Berechnung) vor allem dadurch bestimmt, wie gut sie die Schlüsselmenge (den Wertebereich) auf den Bereich der Hash-Codes (den Bildbereich) abbildet. Dabei ist eine Gleichverteilung anzustreben.

Ende des Wintersemesters wurde ein Pascal-Programm zur einfachen Erstellung von Graphiken unter Windows (*WinGraph.pas* mit dem Testprogramm *WG_Test.pas*) vorgestellt und dazu verwendet, um die Güte von Zufallszahlengeneratoren zu visualisieren (z. B. in Form des Himmelstests). Benutzen Sie dieses System (im moodle-Kurs) nun, um die Güte von *mindestens drei unterschiedlichen* Hash-Funktionen zu visualisieren.

Zu Testzwecken finden Sie in der Datei *KafkaWords.txt* über 10.000 unterschiedliche Wörter (aus „Das Schloss“ von Franz Kafka). Ändern Sie die Prozedur *Redraw*-Prozedur so ab, dass die Wörter aus der Wortdatei gelesen werden, für jedes Wort der Hash-Code mittels einer Hash-Funktion berechnet wird und die Häufigkeit der einzelnen Hash-Codes ermittelt und visualisiert wird.

Dazu werden in einem zweidimensionalen Koordinatensystem horizontal die Hash-Codes von 0 bis n - 1 aufgetragen und vertikal jeweils ein Punkt dargestellt, wenn der entsprechende Hash-Code ermittelt wurde. Folgende Darstellung zeigt einen Zustand, bei dem z. B. der Hash-Code $h(k)$ bereits dreimal ermittelt wurde (es also schon zu zwei Kollisionen gekommen ist):



Die beiden Abbildungen unten zeigen zwei mögliche Ergebnisse für zwei unterschiedliche Hash-Funktionen mit jeweils n = 211 und den Wörtern aus der Datei *KafkaWords.txt*: