ADF 2*x* & PRO 2*x* **Übungen zu Fortgeschrittenen Algorithmen & Datenstrukturen und OOP** **SS 23, Übung 1**

Abgabetermin: Mi, 15. 03. 2023

☒ **Gr. 1,** Dr. S. Wagner **Name** Wurm Elias **Aufwand in h** 3

☐ **Gr. 2,** Dr. D. Auer

☐ **Gr. 3,** Dr. G. Kronberger **Punkte** _____ **Kurzzeichen Tutor / Übungsleiter*in** _____ / _____

1. Eindeutige Teilketten (8 Punkte)

Entwickeln Sie ein Pascal-Programm `UniqueSubstr`, das einen Text zeichenweise von der Standardeingabe einliest und die Anzahl der darin enthaltenen eindeutigen Teilketten (engl. *unique substrings*) der Länge *L* berechnet. Speichern Sie dazu die eindeutigen Teilketten in einer Hashtabelle.

Beispiel: Für die Eingabe `ABABABB` und die Länge $L = 3$ gibt es 3 eindeutige Teilketten `ABA`, `BAB` und `ABB`.

Testen Sie Ihr Programm auch mit längeren Texten, z.B. mit den Ziffern von π. Im elearning-Kurs finden Sie die Datei *pi-1million.txt*. Den Inhalt der Datei können Sie einfach auf die Standardeingabe umleiten. Beispiel:

```
UniqueSubstr.exe < pi-1million.txt
```

2. Index-Generator (16 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.

Eingabe:

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

Index (Auszug):

```
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 zu sortieren. Testen Sie Ihre Lösungen ausführlich, indem Sie für verschiedene Textdateien (z.B. Datei Kafka.txt im elearning-Kurs) einen Index generieren.

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

Hinweise:

1. Geben Sie für alle Ihre Lösungen immer eine „Lösungsidee“ an.
2. Dokumentieren und kommentieren Sie Ihre Algorithmen.
3. Bei Programmen: Geben Sie immer auch Testfälle ab, an denen man erkennen kann, dass Ihr Programm funktioniert, und dass es auch in Fehlersituation entsprechend reagiert.

Algorithm UniqueSubstr: take a user input string and a length L, and then count the number of unique substrings of length L in the input string using a hash table. Then iterate over all possible substrings of length L in the input string, and inserting each unique substring into the hash table. If a substring already exists in the hash table, it is not counted as a new unique substring. The algorithm uses a hash function to convert each substring into an index in the hash table, and then uses chaining to handle collisions.

Algorithm IndexGen: read a text file and extracts all the words. The words are hashed using a hash function, and then inserted into the hash table. Each entry in the hash table consists of a word and a list of line numbers where the word occurs. if a word is already in the hash table add the line number of the occurance to the list of occurances. Then sort the list with bubble sort descending by word and extract everything into an output file.