technische universität
dortmund

Fakultät für Informatik
Lehrstühle 2, 11, 13

Prof. Dr. C. Sohler
I. Kamehkhosh, N. Kriege, D. Kurz, M. Ludewig

Sommersemester 2017
30. Mai

DAP2 Praktikum – Blatt 8

Abgabe: 12.–16. Juni

Wichtig: Der Quellcode ist natürlich mit sinnvollen Kommentaren zu versehen. Überlegen Sie außerdem, in welchen Bereichen Invarianten gelten müssen, und überprüfen Sie diese ggf. an sinnvollen Stellen mit *Assertions* (siehe Hinweis auf Blatt 2).

Langaufgabe 8.1: Editierdistanz (Dynamische Programmierung) (4 Punkte)

Implementieren Sie einen Algorithmus zur Bestimmung der sogenannten minimalen Editierdistanz zwischen zwei Sequenzen von Buchstaben, die auch als "Levenshtein-Distanz" bekannt ist. Unter der minimalen Editierdistanz zwischen zwei Sequenzen a, b von Buchstaben versteht man die minimal benötigte Anzahl von Einfüge-, Lösch- und Ersetzungs-Operationen einzelner Buchstaben, die benötigt werden, um a in b zu transformieren.

So ist beispielsweise 4 die minimale Editierdistanz zwischen den beiden Sequenzen `baacda` und `abace`, weil sich `baacda` nicht mit drei oder weniger, aber mit Hilfe der folgenden vier Einfüge-, Lösch- und Ersetzungs-Operationen in `abace` transformieren lässt:

Anzuwendende Operation	Sequenz nach Operation
Füge 'a' an Position 1 ein	`abaacda`
Lösche 'a' an Position 4	`abacda`
Ersetze 'd' durch 'e' an Position 5	`abacea`
Lösche 'a' an Position 6	`abace`

Weitere Erläuterungen zum Problem der Bestimmung der minimalen Editierdistanz sowie der zu implementierende, auf dynamischer Programmierung basierende Algorithmus zur Lösung dieses Problems lassen sich in der Literatur oder durch eine Internet-Recherche finden.

Konkret ist Folgendes zu erfüllen:

- Legen Sie eine Klasse `EditDistance` an, die die Methode `int distance(String a, String b)` enthält.
- Implementieren Sie die Methode `int distance(String a, String b)` mit Hilfe des auf dynamischer Programmierung basierenden Algorithmus zur Bestimmung der minimalen Editierdistanz.
- Fügen Sie weiterhin eine Methode ein, die aus einer gegebenen Datei – in der in jeder Zeile genau eine Sequenz von Buchstaben steht – alle Zeilen ausliest und jede Zeile als eine Sequenz von Buchstaben (d.h., als einen `String`) interpretiert.

- Das Programm soll entweder einen oder zwei Parameter übergeben bekommen:
  - Falls ein Parameter übergeben wird, soll dies der Name einer einzulesenden Datei sein, für die für jede Kombination von jeweils zwei verschiedenen Zeilen/Sequenzen aus dieser Datei die minimale Editierdistanz bestimmt werden soll.
    Beispielaufruf: `java EditDistance file.txt`
  - Falls zwei Parameter übergeben werden, sollen diese als Sequenzen interpretiert werden und es soll die minimale Editierdistanz für die Transformation der ersten Sequenz (erster Parameter) in die zweite Sequenz (zweiter Parameter) berechnet werden.
    Beispielaufruf: `java EditDistance informatics interpolation`

Langaufgabe 8.2: Ausgabe der Editieroperationen (4 Punkte)

Das Programm aus Aufgabenteil 1 soll nun so erweitert werden, dass nicht nur die benötigte Anzahl der Operationen, sondern auch die einzelnen Operationen selbst sowie die Zwischenergebnisse der Transformation ausgegeben werden.

- Schreiben Sie eine Methode `printEditOperations`, die schrittweise die konkreten Operationen und deren jeweilige Kosten ausgibt, die für eine optimale Transformation einer Ausgangssequenz in eine Zielsequenz benötigt werden.
- Dabei soll für jeden (Zwischen-)Schritt die Sequenz mit ausgegeben werden, die nach Anwendung der Operation dieses Schrittes entsteht.
- Die Ausgabe soll dabei so wie in dem folgenden Beispiel formatiert sein:

```
Loesung fuer "baacda" --> "abace" mit Gesamtkosten 4:
=====================================================
1) Kosten 1: Fuege a an Position 1 ein --> abaacda
2) Kosten 0: Ersetze b durch b an Position 2 --> abaacda
3) Kosten 0: Ersetze a durch a an Position 3 --> abaacda
4) Kosten 1: Loesche a an Position 4 --> abacda
5) Kosten 0: Ersetze c durch c an Position 4 --> abacda
6) Kosten 1: Ersetze d durch e an Position 5 --> abacea
7) Kosten 1: Loesche a an Position 6 --> abace
```

- Erweitern Sie Ihr Programm derart, dass als jeweils letzter Parameter "-o" übergeben werden kann. In diesem Fall soll an Stelle der einfachen Ausgabe in Form der minimalen Editierdistanz die detaillierte Ausgabe mit Hilfe der Methode `printEditOperations` erfolgen.
  Beispielaufruf: `java EditDistance file.txt -o`
  Beispielaufruf: `java EditDistance informatics interpolation -o`