**Software Entwicklung I**

# Übung 01

**Verschlüsselung**

11. Juni 2016

FH JOANNEUM
University of Applied Sciences

Internet Technologien & Anwendungen - SWD15

# 1. Einleitung

Diese Übung ist in zwei Unterübungen aufgeteilt, in *Teil A* geht es darum verschiedene simple Verschlüsselungsalgorithmen zu implementieren und diese unter einen gemeinsamen Interface zu vereinen. Durch die Verwenden von I

# Teil A

Teil A beschränkt sich ausschließlich auf die Implementierung der Verschlüsselungsalgorithmen. Die Algorithmen sollen das unten definierte Interface implementieren. Die Funktionsweise soll mittels *Unit Test* überprüft werden. Was ihre JUnit Testfälle genau abdecken bleibt grundsätzlich Ihnen überlassen. Denken Sie jedoch daran dass die Testfälle möglichst alle Use-Cases abdecken.

## 1.1. Interface

Das Interface bietet grundsätzliche zwei Methoden an um Texte (Strings) mit einen Schlüssel (Passwort) zu Ver- und Entschlüsseln. Zusätzlich habe ich mir für die Kommende GUI Applikation noch die Freiheit genommen das Interface zu erweitern. Die Datei mit dem vollen Interface (`Cipher.java`) ist der Angabe beigelegt. Das Interface darf unter keinen Umständen verändert werden!

Im folgenden Abschnitt werden die Interface Methoden erläutert welche nicht im Rahmen des Unterrichts

```
String getCipherName()
```

Liefert den Namen des Algorithmus zurück z.B. zur Anzeige auf einer graphischen Oberfläche.

```
Sboolean isValidKey(String key)
```

Prüft ob der Schlüssel `key` von den Algorithmus verwenden werden. Diese Methode soll verwenden werden um im Vorhinein zu Prüfen ab der Schlüssel gültig ist bevor noch mit der Ver- Entschlüsslungen begonnen wird.

## 1.2. Caesar Verschlüsslung

Die klassisches Caesar Verschlüsselung bei welcher ein Buchstabe um einen gegebenen Wert im Alphabet verschoben wird. Andere Zeichen wie Zahlen, Umlaute und Sonderzeichen bleiben unverändert. Die genaue Erklärung des Cäsar-Verschlüsselung entnehmen Sie bitte der Angabe des Abschlussbeispiels des 3. Semesters.

## 1.3. Caesar Verschlüsslung (mit Zahlen)

Gleich wie die reguläre Caesar Verschlüsselung allerdings werden auch Zahlen (0-9) verschlüsselt.

## 1.4. Vigenere Verschlüsselung

Die Vigenere Verschlüsselung ist eine Erweiterung der Cäsear Verschlüsselung, im Gegensatz zur Cäsarverschlüsselung wird nicht nur ein einziger nummerischer Wert als Schlüssel verwendet sondern mehrere Zeichen (meist in Form eines Schlüsseltexts).

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Plain Text:** | K | A | P | F | E | B | E | R | G |
| **P:** | 10 | 0 | 15 | 5 | 4 | 1 | 4 | 17 | 6 |
| **Key** | S | W | D | S | W | D | S | W | D |
| **K:** | 18 | 22 | 3 | 18 | 22 | 3 | 18 | 22 | 3 |
| **C: *(P+K) % 26*** | 2 | 22 | 18 | 23 | 0 | 4 | 22 | 13 | 9 |
| **Cipher Text:** | C | W | S | X | A | E | W | N | J |
| **P: *(C-K) % 26*** | 10 | 0 | 15 | 5 | 4 | 1 | 4 | 17 | 6 |
| **Plain Text:** | K | A | P | F | E | B | E | R | G |

Abbildung 1: Vigenere-Verschlüsselung des Wortes *"KAPFENBERG"* mit den Schlüssel *S̈WD"*

Jeder Buchstabe des Klartext wird mittels der Cäsar-Verschlüsselung verschlüsselt. Als Schlüssel dient der korrespondierte Buchstabe aus den Schlüssel.

**Beispiel:** Der erste Buchstabe des Klartextes ist $K$ (10) und der korrespondierte Buchstabe des Schlüssels ist $S(18)$. Somit wird folgende Verschlüsselung angewandt:

$$(10 + 18) \mod 26 = 28 \mod 26 = 2 = C$$

Die Entschlüsselung funktioniert nach den gleichen Prinzip, wiederum wird der Cäsar anwendet. Zum entschlüsseln wird der vorher verschlüsselte Buchstabe mit wiederum mit den korrespondierenden Buchstaben des Schlüsseltexts entschlüsselt

**Beispiel:** Der verschlüsselte Buchstabe $C$ (2) wird mit dem korrespondierte Buchstabe des Schlüssels $S(18)$ entschlüsselt:

$$(2 - 18) \mod 26 = -16 \mod 26 = 10 = K$$

Ist der Klartext (Plaintext) länger als der Schlüsseltext so wird der Schlüsseltext einfach wiederholt um ihn auf die Länge des Klartextes zu bringen.

# Teil B

In diesem Teil soll eine Swing GUI Applikation entwickelt werden, welche das Ver- und Entschlüsseln von kurzen Texten ermöglichen soll. Die Graphische Oberfläche soll in etwas wie folgt aussehen:

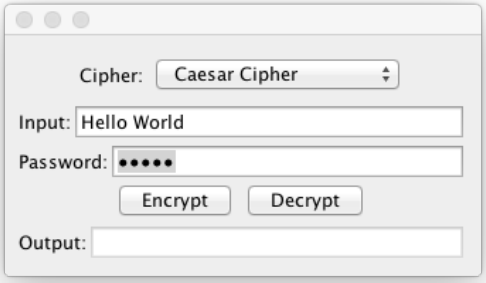


Abbildung 2: Vigenere-Verschlüsselung des Wortes *"KAPFENBERG"* mit den Schlüssel *S̈WD"*

Die genaue Gestaltung bleibt grundsätzlich Ihnen überlassen solange die Funktionalität gegeben ist. Wichtig ist, dass die GUI Anwendungen das Interface bzw. die Implementierung aus den Teil A übernimmt.

## 2. Funktionalität

Der Benutzer soll den Verschlüsselungsalgorithmus auswählen können, den Eingabe Text und den Schlüssel eingeben können. Optional soll die Eingabe des Schlüssels nicht über einen `JTextTextField` sondern über eine `JPasswordField` erfolgen, damit der Schlüssel während der Eingabe nicht sichtbar ist.

Je nach verwenden Algorithmus können unterschiedliche Schlüssel eingegeben werden ist der eingegebene Schlüssel für den auswählten Algorithmus nicht zulässig soll, so soll der *Encrypt* und *Decrypt* Button deaktiviert (grau) werden.

**Hinweis:** Änderungen in `JTextField` bzw. `JPasswordField` lässen sich über `DocumentListener` ermitteln. (`getDocument().addDocumentListener(...)` )