ITS Aufgabe 3

Julian Bertol

October 31, 2024

1 Aufgabe 1

```java
package org.example;

import java.util.Arrays; import java.util.HashSet;
public class VignereChiffreStatic {
    static String[] keywords = { "die", "der", "und", "in", "zu", "den", "das", "nicht", "von", "sie", "ist", "des",
            "sich", "mit", "dem", "dass", "er", "es", "ein", "ich", "auf", "so", "eine", "auch", "als", "an", "nach",
            "wie", "im", "f r", "fuer", "fur" };
    static HashSet<String> germanKeywordsSet = new HashSet<String>();
    static double threshold = 0.3; // Prozentsatz der W rter , die erkannt werden m ssen
    public static void main(String[] args) {
        String s = "Studieren an der HFU ist toll.";
        String key = "TOLL";
        System.out.println("Testen der Verschl sselung mit: " + s + " und Schl ssel: " + key);
        String encrypted = encrypt(s, "TOLL");
        System.out.println(s);
        System.out.println(encrypted + " (verschl sselt)");
        System.out.println(decrypt(encrypted, "TOLL") + " (entschl sselt)");
        System.out.println();
        System.out.println("Ist der Text (vermutlich) lesbar?: " + isProbablyReadable(s));
        System.out.println();
        System.out.println("Nun geht es an die Aufgabe: Versuchen Sie alle m glichen Schl ssel. Erweitern Sie den Code unten");
// Hinweis: Permutationen der L nge zwei aus der Menge {A,B,C}
        String[] buchstaben = { "A", "B", "C" };
        for (String buchstabe1 : buchstaben) {
            for (String buchstabe2 : buchstaben) {
                System.out.println("Hier k nnten Sie mit dem Schl ssel " + buchstabe1 + buchstabe2 + " testen");
                String encrypted_test = encrypt("Dies ist ein test",  buchstabe1+buchstabe2);
                System.out.println(encrypted_test + " (verschl sselt)");

                System.out.println(decrypt(encrypted_test, buchstabe1+buchstabe2) + " (entschl sselt)");
                // Testen Sie: Ist der entschl sselte Text lebar? Wenn ja: Geben Sie Schl ssel und Text aus.
            }
        }

        String encrypt_text = encrypt("Hallo liebe mitstudierende!", "tedi");
        System.out.println(encrypt_text);

        decrypt_bruteforce("dies ist ein test");
    }
    /**
     * String s verschl sseln
     * @param text Nachricht im Klartext
     * @param keyForEncryption Schl ssel
     * @return Verschl sselte Nachricht
```

```
45      */
46     public static String encrypt(String text, String keyForEncryption) {
47         text = text.toUpperCase(); // alles Grossbuchstaben
48         char[] chars = text.toCharArray();
49         char[] key = keyForEncryption.toUpperCase().toCharArray();
50         int keyLength = key.length;
51         for (int i = 0; i < chars.length; i++) {
52             char charToShift = key[i % keyLength];
53             chars[i] = shift(chars[i], charToShift); // Sonderzeichen ignorieren.
                   Schl sselindex weiterz hlen
54         }
55         return String.valueOf(chars); // wieder in String wandeln
56     }
57     /**
58      * String s entschl sseln
59      * @param text Nachricht als Chiffrat
60      * @param keyForDecryption Schl ssel
61      * @return Entschl sselte Nachricht
62      */
63     public static String decrypt(String text, String keyForDecryption) {
64         text = text.toUpperCase(); // alles Grossbuchstaben
65         char[] chars = text.toCharArray();
66         char[] key = keyForDecryption.toUpperCase().toCharArray();
67         int keyLength = key.length;
68         for (int i = 0; i < text.length(); i++) {
69             char charToShift = key[i % keyLength];
70             chars[i] = deShift(chars[i], charToShift);
71         }
72 // wieder in String wandeln
73         return String.valueOf(chars);
74     }
75     private static char shift(char ch, char k) {
76         if (ch >= 'A' && ch <= 'Z') { // Sonderzeichen nicht behandeln
77             return (char) (((ch - 'A') + (k - 'A')) % 26 + 'A');
78         } else {
79             return ch;
80         }
81     }
82     private static char deShift(char ch, char k) {
83         if (ch >= 'A' && ch <= 'Z') { // Sonderzeichen nicht behandeln
84             return (char) (((ch - 'A') - (k - 'A') + 26) % 26 + 'A');
85         } else {
86             return ch;
87         }
88     }
89     /**
90      * Wie h ufig kommt einer der in keyword definierten W rter im Text vor?
91      * @param s Der Text
92      * @return Anzahl der   bereinstimmungen
93      */
94     protected static int occurencesOfKeywords(String s) {
95         if (germanKeywordsSet.isEmpty()) // Array in Set   berfhren
96             germanKeywordsSet.addAll(Arrays.asList(keywords)); // sofern nicht bereits
                    geschehen
97         String[] words = s.toLowerCase().replaceAll("\\.", "").split(" ");
98         int hits = 0;
99         for (String word : words) {
100            if (germanKeywordsSet.contains(word))
101                hits++;
102        }
103        return hits;
104    }
105    /**
106     * Ist der Text lesbar?
107     * @param s Der Text zum Testen
108     * @return wahr, wenn der Text vermutlich lesbar ist
109     */
110    public static boolean isProbablyReadable(String s) {
111        int hits = occurencesOfKeywords(s);
112        int word = s.toLowerCase().replaceAll("\\.", "").split(" ").length;
113        double percentage = ((double) hits) / word;
```

```
114         return percentage >= threshold;
115     }
```

- Ich habe den Satz ”hallo liebe mitstudierende!” mit dem key ”tedi” verschlüsselt

2 Aufgabe 2

```
1   public static void decrypt_bruteforce(String text) {
2       text = text.toUpperCase(); // alles Grossbuchstaben
3       char[] alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ".toCharArray();
4       int alphabetlength = alphabet.length;
5       int combinationlength = 4;
6
7       for (int i = 0; i < Math.pow(alphabetlength, combinationlength); i++) {
8
9           StringBuilder combination = new StringBuilder();
10
11          // Berechne die Zeichen f r jede Position in der Kombination
12          int temp = i;
13          for (int j = 0; j < combinationlength; j++) {
14              int index = temp % alphabetlength; // Index im Alphabet
15              combination.insert(0, alphabet[index]); // Zeichen an der Position
                    einf gen
16              temp /= alphabetlength;
17
18          }
19          //System.out.println(combination.toString());
20          if (isProbablyReadable(decrypt("LHFOBSCPG LY RPC VQF WDE HZWE",
                combination.toString()))) {
21              System.out.println(decrypt("LHFOBSCPG LY RPC VQF WDE HZWE",
                    combination.toString()));
22          }
23      }
```

2.1 Bruteforce funktion erklärung:

- Es wird jede Permutation des Alphabets genommen und der verschlüsselte string wird 1 mal mit jeder kombination der decrypt funktion übergeben. Danach mithilfe der "isProbablyReadable" funktion geprüft ob es sich möglicherweise um einen Deutschen Text handelt. Nun muss man händisch noch den Satz nehmen der einen Sinn ergibt, da es mehrere möglichkeiten zur auswahl gibt.

2.2 Wie viele der Texte Ihrer Kommilitonen können Sie entschlüsseln?

- Einen da nur einer hochgeladen.
- Phileas Schwarz Chiffre: BH DBQSPKVPTM TDM OPK MPLHP LHFOBSYRTBR Klartext = IT SICHERHEIT IST DER BESTE STUDIENGANG

3 Wieso war es Ihnen in diesem Beispiel möglich die verschlüsselten Texte zu entschlüsseln?

- Da wir nur einen schlüssel von 4 zeichen haben dadurch ergeben sich $26^4 möglichkeiten$.

3.1 Wie groß ist der Schlüsselraum? Das bedeutet wie viele möglich Schlüssel gibt es?

  - Genau 4.

- Dadurch gibt es $26^4 möglichkeiten = 456976$

3.2 Was wäre, wenn der Schlüssel doppelt so viele Stellen hätte, also acht Stellen?

  - Dann würde die berechnen um einiges länger dauern.

- Dann gibt es $26^8 möglichkeiten = 208827064576$

3.3 Könnten Sie einen achtstelligen Schlüssel immer noch brechen?

  - Man könnte diesen immer noch berechnen nur es dauert etwas länger.
  - Durchschnittliche Systeme berechnen 1 000 000 kombinationen pro Sekunde
  - Berechnen wir also 208827064576 / 100000 kommen wir auf 208,827 Sekunden was ungefähr 57 Stunden wären.

3.4 Wie groß wäre der Schlüsselraum bei einem vierstelligen Schlüssel, der nicht nur aus A-Z, sondern aus a-z, A-Z besteht, also aus doppelt so vielen Zeichen?

  - Dann gibt es doppelt so viel zeichen 26 * 2 = 52
  - Bei einem key von 4 Zeichen sind es
$$52^4 = 7311616 \tag{1}$$
    Kombinationen.
  - Bei einem key von 8 Zeichen sind es
$$52^8 = 53459728531456 \tag{2}$$
    Kombinationen.

3.5 Argumentieren Sie, was die Sicherheit eines Schlüssels mehr verbessern würde: Ein größeres Alphabet (d.h. mehr Möglichkeiten pro Stelle) oder mehr Stellen. Wie berechnen Sie die Anzahl möglicher Permutationen? Die Änderung welches Parameters lässt die Anzahl der Permutationen schneller steigen?

- Ein Schlüssel mit einer größeren länge würde die Sicherheit erhöhen
- Ein Schlüssel mit klein,Groß,Zahlen und Sonderzeichen würde die Sicherheit erhöhen, wenn diese auch bei der verschlüsselung beachtet und nicht ignorieren würden wie in unserem Fall.
- Das erhöhen der Schlüssellänge ist die bessere wahl um schnell eine bessere Sicherheit zu erhalten.
- Beispiel:
  - erweiterung des Alphabets auf 52
    $$52^4 = 7311616 \tag{3}$$
  - erweiterung des Schlüssels auf 5
    $$26^5 = 11881376 \tag{4}$$

3.6 Wieso kann eine EC-Karte dennoch (einigermaßen) sicher angesehen werden, obwohl einePIN lediglich vierstellig und aus den Zeichen 0-9 aufgebaut ist?

- Da das Konto nach 3 Fehlgeschlagenen versuchen gesperrt wird.