Javakurs SS03 - Übung 5

1 Schatzsuche

Gegeben ist eine Folge von Anweisungen mit Schrittzahlen und Drehungen um 90 Grad nach links oder rechts, z.B. „2 3 L 1 L L 2“. Am Anfang steht auf der Koordinate $(0,0)$ mit Blickrichtung und Schrittweite derart, daß ein Schritt einen auf die Koordinate $(0,1)$ bringt. Schreibe ein Programm, daß für die auf der Kommandozeile gegebenen Anweisungen die Endkoordinate ausgibt, also etwa für das Beispiel $(1,5)$.

Hinweis:

`String`s `s1`, `s2` kannst Du mit `s1.equals(s2)` vergleichen. `s1 == s2` funktioniert nicht - es vergleicht nur die Referenzen, d.h. ob es die gleichen Objekte sind, nicht ob sie den gleichen Inhalt haben.

2 Uhrwerk

Vervollständige die folgende Klasse, die ein Uhrwerk darstellen sollen. (Stunden bis maximal 12 Uhr). Zum Testen kannst Du die Klasse Clock von der Kurswebseite verwenden, siehe auch
http://www.inf.fu-berlin.de/lehre/SS03/javakurs/src/Clock.java .

Die Klasse muß folgendermaßen aussehen:

```
public class Clockwork {

  // Objektvariablen

  Clockwork(int h, int m, int s) {
    // your task
  }

  // ggf. Hilfsmethoden
```

```
  void stepSecond() {
    // your task
  }

  int getSeconds() {
    // your task
  }

  int getMinutes() {
    // your task
  }

  int getHours() {
    // your task
  }
}
```

3 Verschlüsselung mit Caesarchiffre

Bei der Caesarchiffre, einem einfachen Verfahren zur Verschlüsselung von Nachrichten, ersetzt man die Buchstaben mit dem *n*. Nachfolger im Alphabet (zyklisch), also z.B. für $n = 3$ wird a zu d, b zu e, c zu f, ..., w zu z, x zu a, y zu b und z zu c. Aus *eswarschondunkel* wird dann *hvzduvfkrqgxqnho*. (Traditionell verwendet man bei den einfachen Verschlüsselungen keine Groß-/Kleinschreibung, Interpunktion und Leerzeichen - das „Knacken“ des Codes wäre sonst zu einfach.)

Schreibe ein Programm, das ein *n* und einen String als Kommandozeilenparameter nimmt und dafür die verschlüsselte Zeichenkette ausgibt. Für einen String `s` gibt der Aufruf `s.charAt(i)` den `i`. Buchstaben von `s` zurück. Alternativ kannst Du auch `s.toCharArray()` verwenden, was den gesamten Stringinhalt als `char[]` zurückliefert.

Historischer Hintergrund:
Der Name der Verschlüsselung kommt daher, daß Caesar tatsächlich diese Verschlüsselung für *n* gleich drei verwendet hat (Suetonius). Noch 1915 hat die russische Armee das Verfahren verwendet, weil man meinte, den Stäben nichts komplizierteres zumuten zu können.

In den 90ern war das Verfahren für *n* gleich 13 (da ist der Verschlüsselungsschritt gleich der Entschlüsselung) unter dem Namen *ROT13* im Newsnet recht

populär, um sog. Spoiler, z.B. Lösungen von Rätseln, Überraschungen in Filmen oder Büchern, vor versehentlichen Lesen zu schützen. Viele Newsreader hatten den Befehl für *ROT13* gleich eingebaut. Nebenbei ist auch der Name des Computers Hal aus dem Film *2001: A Space Odyssey* das Ergebnis einer Caesar-Chiffrierung.

Etwas allgemeiner ist das Verfahren, bei dem man eine allgemeine Vertauschung der Buchstaben verwendet (eine sog. monoalphabetische Substitution), von Hobbychiffrieren oft auch mit anderen Alphabetzeichen verwendet. Karl der Große soll eine solche Verschlüsselung verwendet haben. Ein anderes Beispiel ist die sog. Freimaurerchiffre. Von Edgar Allen Poe gibt es sogar eine Kurzgeschichte, *The Gold-Bug*, in der eine solchermaßen chiffrierte Nachricht den Weg zu einem Piratenschatz weisen soll und eine der Figuren erläutert, auf welche Weise er die Nachricht entschlüsselt hat.

4 Verschlüsselung mit Spaltentransposition

Ein weiteres einfaches Verschlüsselungsverfahren ist die Spaltentransposition. Dabei wird als Parameter neben der zu Verschlüsselnden Zeichenkette eine Permutation genommen, also Zahlen von 1 bis n (oder von 0 bis $n-1$) in einer vertauschten Reihenfolge. Man schreibt dann die Nachricht in eine Matrix mit n Spalten und liesst dann spaltenweise gemäß der Permutation aus. Als Beispiel für die Nachricht *eswarschondunkel* und die Permutation (2143) ergibt sich über

```
2 1 4 3
e s w a
r s c h
o n d u
n k e l
```

die verschlüsselte Nachricht *ssnkeronahulwcde*.

Schreibe eine Ver- und Entschlüsselung mit Spaltentransposition.

Historischer Hintergrund:
Die doppelte Spaltentransposition (zwei Transpositionen hintereinandergeschaltet) gehörte schon zu den stärkeren Verschlüsselungen, die im 1. Weltkrieg verwendet wurden. Ohne Rechnerunterstützung ist das Knacken recht schwierig. (Die Franzosen es damals schafften aber, die doppelte Transposition der deutschen Armee zu brechen.) Nach modernen Maßstäben handelt es sich aber nicht um eine ernstzunehmende Chiffrierung. Zum Brechen solcher Codes verwendet man statistische Analysen über Häufigkeiten von Buchstabenpaaren (den sog. Bigrammen) und eventuell auch längerer Buchstabenketten (Trigramme etc.).