ulm university universität ulm

Einführung in die Informatik

Institut für Eingebettete Systeme/Echtzeitsysteme | Wintersemester 2020/21
Valentina Richthammer, Michael Glaß

Übungsblatt 4: Schleifen

Abgabetermin: 06.12.2020, 23:59 Uhr

Geben Sie **Programmieraufgaben als Java Code (`*.java` Dateien)** ab. Alle anderen Aufgaben, die Text oder Grafiken erfordern, geben Sie **als PDF Dateien** ab. PDFs können Sie beispielsweise mit dem kostenlosen Programm *LibreOffice* erstellen. Alternativ können Sie etwas mehr Zeit investieren und LaTeX lernen, was Sie im späteren Studium immer wieder brauchen werden.

Wenn Sie **mehrere Dateien** abgeben wollen, dann fassen Sie diese zu **einem ZIP File** zusammen.

Präsenzaufgaben werden direkt im Tutorium bearbeitet, werden nicht bepunktet und müssen nicht abgegeben werden.

Präsenzaufgabe

Drücken Sie die folgenden Anweisungen jeweils durch äquivalente Schleifen der angegebenen Art aus!

a) Mittels einer Do-While-Schleife:

```
import java.util.Scanner;

public class MyLoop{
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        System.out.print("Insert a number: ");
        int k = scan.nextInt();

        for (int i=1; i<k; ++i) {
            System.out.println(i*i);
        }
    }
}
```

b) Mittels einer For-Schleife:

```
int k = ...;
int x = 0, i = 0;
while (i < k) {
    x += k * ++i;
}
System.out.println("x: " + x);
```

c) Mittels einer For-Schleife:

```
int k = ...;
if(k > 0) {
    int i = 1, m = 0;
    while(i < k) {
        if (k*i > m)
            m = k + i;
        ++i;
    }
    System.out.println("m: " + m);
}
```

Aufgabe 1: Schwobifying Light (5)

Der UNiMUT Schwobifying Proxy[1] war ein Programm zur automatisierten "Übersetzung"von Websites in's Schwäbische.

Schreiben Sie einen Algorithmus `Schwobifying Light` in Java, der zu Beginn einen Satz einliest und ihn zusammen mit seiner Schwäbischem Übersetzung ausgibt. Der Algorithmus soll dabei jedes Vorkommen der Buchstaben `st` im Satz durch `scht` ersetzen. Außerdem soll dem Satz die Endung `, woischt.` angehängt werden.
Die Ausgabe soll beispielsweise wie folgt aussehen:

```
Hier ist mein allerbester Satz
Hier ischt mein allerbeschter Satz, woischt.
```

Hinweis: Verwenden Sie ausschließlich die `charAt()` Funktion der Strings, um den String zu traversieren und die "Übersetzung" zu erstellen. Methoden wie String.replace(), etc. dürfen **nicht** verwendet werden. Daten einlesen können Sie mit der Scanner[2] Klasse (siehe Beispiel in der Präsenzaufgabe). Ein Scanner kann nicht direkt `char` einlesen. Sie können sich z.B. so behelfen: `char c = scan.nextLine().charAt(0);`

Aufgabe 2: `Zufallsmuster` (5)

Implementieren Sie in Java einen Generator für Zufallsmuster, die sich aus mehreren Zeilen, bestehend aus `0`en und `1`en, zusammengesetzen. Dabei sollen Zeilen nach folgender Vorschrift erstellt werden: Das erste Element jeder Zeile ist `0`, das mit Wahrscheinlichkeit 1.0 erzeugt wird. Für jedes nachfolgende Element halbiert sich die Wahrscheinlichkeit für eine `0`. Sobald eine `1` ausgegeben wird, ist die Zeile beendet.

Es sollen so lange Zeilen nach diesem Muster generiert werden, bis eine vorgegebene Anzahl an `0`en erzeugt wurde. Ihr Programm soll als Parameter die gewünschte Anzahl `0`en einlesen. Einen zufälligen Doublewert im Bereich $[0, 1[$ erhalten Sie mit `Math.random()`.

Hier ein Beispiel-Muster für $n = 11$

```
0 0 0 0 1
0 0 0 1
0 1
0 0 0
```

Terminiert der Algorithmus? Begründen Sie Ihre Antwort.

Aufgabe 3: Verschlüsselung (7+5)

a) Schreiben Sie ein Javaprogramm *Decrypt*, das einen String entschlüsseln soll. Der String wurde mit einem Rotationsverfahren verschlüsselt. Dabei wird jedem Zeichen ein neues Zeichen zugeordnet, das um x Positionen

[1] `http://unimut.fsk.uni-heidelberg.de/schwob.html`
[2] `https://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html`

im Alphabet verschoben wird. Der bekannteste Fall dieses Algorithmus heißt ROT13. Hierbei wird bei der Verschlüsselung aus einem *a* ein *n* (eine Verschiebung um 13 Stellen). Gehen Sie davon aus, dass die ASCII-Zeichen 32 bis 125 Ihr Alphabet bilden. Die Rotation findet also nur in diesem Bereich statt und es wird über den kompletten Bereich rotiert. Mit ROT13 wird demnach aus einem # eine 0.
Ihr Programm soll in der Lage sein jeden String (auch mit Zahlen oder Sonderzeichen) zu entschlüsseln und ihn auf der Konsole auszugeben. Da Sie die Verschiebung der Verschlüsselung nicht kennen, sollen alle möglichen Verschiebungen getestet und ausgegeben werden. Aus der Liste der Ergebnisse können Sie dann manuell den entschlüsselten Text heraussuchen.

Als Testfälle können Sie folgende Sätze benutzen (bitte Übersetzung mit abgeben):
`|}}n-tn{t{nz-"#(yr`
`hyr'1u"v&1'yv1w"+1&r,P` (Achtung, die Ticks sind einfache Anführungszeichen!)
`Yzjgnsljs%|fwzr%gnxy%iz%xt%mzjljqnlD`
`PV0Grv#Pdqiuhg#xqg#0rhwnroehq#0xgzlj`

b) Für die Entschlüsselung einer unbekannten Verschiebung kann auch zusätzliches Wissen über die Sprache genutzt werden. So kommt beispielsweise in deutschen Texten der Buchstabe e im Durchschnitt mit 17,4% am Häufigsten vor. Analysiert man die Häufigkeit der einzelnen Buchstaben in einem verschlüsselten Text, kann man auf potentielle Verschlüsselungen von e an Hand der bekannten Häufigkeit von 17,4% schließen und dadurch die Verschiebung bestimmen.

Erweitern Sie den Algorithmus aus Teilaufgabe a) so, dass Häufigkeitsanalyse genutzt wird, um vielversprechende Kandidaten für den unbekannten Schlüssel zu finden. Die Eingabe soll dabei aus einem verschlüsselten Satz, einem Buchstaben und einem Prozentwert bestehen, der eine untere Schranke für die Häufigkeit des Buchstaben in den Entschlüsselungen repräsentiert (z.B. sollen für eine Schranke von 15% und Buchstabe e alle Entschlüsselungen, in denen e mit einer relativen Häufigkeit $\geq 15\%$ vorkommt, ausgewählt werden).

Geben Sie für folgende Testfälle die möglichen Entschlüsselungen für die zugehörigen gegebenen Buchstaben und Schranken an. Welches Problem kann mit diesem Verfahren auftreten und warum?

`'e'`, 15%:
`Z %5Z*z#5z)%+z+5%z,%5Z)ywzz)z%`

`'e'`, 10%:
`Ol +S"sy+q}t !+vpty+R}l`