Dr. J. Kohlmeyer
S. Wieluch
Servicegruppe Informatik

Präsenzaufgabe 4.1

Drücken Sie die folgenden Anweisungen jeweils durch äquivalente Schleifen der angegebenen Art aus! Die Variable `k` sei jeweils mit einem (hier unbekannten) Wert initialisiert.

a) Mittels einer Do-While-Schleife:

```
int k = ...;
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 4.1 (*5 Punkte*).
Schreiben Sie einen Algorithmus in Java, der zu Beginn einen Satz und einen Buchstaben einliest. Der Algorithmus soll nun jedes Vorkommen des Buchstabens im Satz mit einem `+` markieren. Die Ausgabe (hier im Beispiel für ein gesuchtes 'a') soll wie folgt aussehen:

```
Das ist mein ganz furchtbar toller Satz
 +           +          +          +
```

Hinweis: Verwenden Sie die `charAt()` Funktion der Strings

Aufgabe 4.2 (*5 Punkte*).
Implementieren Sie `10 PRINT` in Java: `10 PRINT` war ein einzeiliges Programm für den C64, das zufällig einen Forward- oder einen Backslash ausgab. Daraus resultierte ein zufällig generiertes spannendes Muster. Mehr über die Geschichte kann man hier nachlesen: https://10print.org/

Ihr Programm soll zwei Zahlen einlesen: die Höhe und die Breite des Musters. Danach soll das oben beschriebene Muster ausgegeben werden. Ihr Programm darf natürlich mehr als eine Zeile Code beinhalten. Für ein besseres Ergebnis können Sie anstelle der Slashes auch die Unicodesymbole U+2571 und U+2572 verwenden. Einen zufälligen Doublewert im Bereich $[0, 1[$ erhalten Sie mit `Math.random()`.

Aufgabe 4.3 (*10 Punkte*).

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 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:
   `|}}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`
   `PVOGrv#Pdqiuhg#xqg#Orhwnroehq#Oxgzlj`

b) Schreiben Sie ein äquivalentes Javaprogramm *Encrypt*, das unter Angabe eines Strings und der Rotation den gegebenen String wie oben genannt verschlüsseln kann.