PR1 – Formular für Lesenotizen
WS2020/21

Nachname	Vorname	Matrikelnummer	Abgabedatum:
Abdel Kader	Schehat	1630110	12.11.2020

Bedingte Ausführung (L4.4 – 4.6) & Programmlogik und indefinite Schleifen (L.5.1 – 5-3)

Lernzielfragen:

a) Wandeln Sie den folgenden Programmtext in eine einzige Zuweisung der Form phase = ... ; um:

```
if (alter < 1) {
  phase= "Baby";
} else if (alter < 3) {
  phase= "Kleinkind";
} else if (alter < 6) {
  phase= "Vorschulkind";
} else if (alter < 13) {
  phase= "Schulkind";
} else if (alter < 20) {
  phase= "Teenager";
} else {
  phase= "Erwachsen";
}
```

```
public class Aua {
    public static void main(String[] args) {
        int alter = 20;
        String phase;

        phase = alter < 1? "Baby" : alter < 3? "Kleinkind" : alter < 6?
        "Vorschulkind" : alter < 13? "Schulkind" : alter < 20? "Teenager" : "Erwachsen";

        System.out.println(phase);
    }
}
```

b) Formulieren Sie eine sinnvolle Vorbedingung für die folgende Methode. Ergänzen Sie in der Methode eine Überprüfung der Vorbedingung, welche zu einer geeigneten Exception führt, falls die Vorbedingung nicht erfüllt ist.

```
public static double log(double arg, double base) {
    return Math.log(arg) / Math.log(base);
```

```
public class Aub {
    public static void main(String[] args) {
        double x = log(2.71, 2.71);
        System.out.println(x);
    }
    /** Vorbe.: arg u. base > 0 u. != 1
        Nachbe.: liefer ln(arg) den Exponeten heraus
    */
    public static double log(double arg, double base) {
        if (arg <= 0 || base <= 0) {
            throw new IllegalArgumentException("Werte > 0 angeben!");
        }
        if (arg == 1 || base == 1) {
            throw new IllegalArgumentException("Werte != 1 angeben!");
        }
        return Math.log(arg) / Math.log(base);
    }
}
```

c) Schreiben Sie ein Programm, das eine sich zufällig entwickelnden Zahlenfolge simuliert:
Startwert: 1 mit Wahrscheinlichkeit von 60% wird 1 addiert, mit einer Wahrscheinlichkeit von 30% wird 1 subtrahiert, mit einer Wahrscheinlichkeit von 10% bleibt der Wert konstant. 100 Durchläufe

```
public static void main(String[] args) {
    Random rand = new Random();
    int n = 1;
    for (int i = 0; i < 100; i++) {
        int x = rand.nextInt(10);
        if (x <= 5) {
            n++;
        }
        else if (x >= 6 && x < 9) {
            n--;
        }
    }
    System.out.println(n);
}
}
```

d) Schreiben Sie ein Programm, das den Benutzer nach einem neuen Passwort fragt, und zwar so lange, bis die Eingabe mindestens 8 Zeichen lang ist. Am Ende soll die Anzahl der Versuche ausgegeben werden.
   - unnötig

e) Was stimmt mit den folgenden Ausdrücken nicht?
   - a != 5 || 6 => 6 ist kein boolescher Ausdruck
   - 5 <= i <= 8 => Zahlenintervalle werden nicht so in Java angegeben

f) Welchen Wert haben die folgenden Ausdrücke (nehmen Sie an, dass die Variable lottogewinn vom Typ boolean existiert)?
   - 17 < 15 && lottogewinn = > false
   - 17 > 15 || lottogewinn = > true
   - 13 != 12 ^ 12 != 11 = > false
   - !(Math.PI < 4) = > false

Notizen:

- Präzedenzen

Operator	Rang	Typ	Beschreibung
++, --	1	Arithmetisch	Inkrement / Dekrement
+, -	1	Arithmetisch	Unäres Plus und Minus
!	1	boolean	Negation
(Typ)	1	Jeder	Typumwandlung
*, /, %	2	Arithmetisch	Multiplikative Op.
+, -	3	Arithmetisch	Additive Op.
+	3	String	String-Konkatenation
<, <=, >, >=	5	Arithmetisch	Numerische Vergleiche
==, !=	6	Primitiv	Gleich-/Ungleichheit von Werten
==, !=	6	Objekt	Gleich-/Ungleichheit von Referenzen
^	8	boolean	Logisches exkl. Oder
&&	10	boolean	Logisches Und
||	11	boolean	Logisches Oder
=	13	Jeder	Zuweisung
*=, /=, %=, +=, -=	14	Jeder	Zuweisung mit Operation

Return

- Alle if/else Pfade müssen einen return Befehl enthalten
- Compiler kann nicht Bedingungen vergleichen z.B.: if (a <= b) u. else if (a > b) return Statements besitzen kommt es trotzdem zu einer Fehlermeldung. Immer ende Methode return
- For-schleife mit if/else muss auch return berücksichtigt werden wenn Schleife/if kein Mal läuft

Random

- Import java.util.Scanner
- Methoden: nextInt(<max. Zahl>), nextDouble(<max Zahl>)
- Erzeugung Intervall min. – max.: nextInt(max – min + 1) + min

Ternäre Operator

- Bedingte Auswertung – Syntax: <type ><varname> = <boolean expr>? <expr1> : <expr2>;
  ⇨ Wenn <boolean expr> wahr, dann <expr1> zugewiesen sonst <expr2>

Indefinite Schleife - while-Schleife

- Syntax: while (<condition>) {
  <statement>;
  }

Sentinel-Schleife: Schleife, die bis zu Sentinel-Wert läuft. Nutzt häufig das Zaunpfahlstil

Sentinel-Werte

- Ein spezieller (Eingabe-)Wert, der das Ende einer Folge von Daten(-eingaben) signalisiert