Fach: Anwendungsentwicklung/Programmierung	FIS	print(), println(), printf()

1. Schreiben Sie ein Java-Programm „AnwendenVonPrint“, das den folgenden Text im Format wie dargestellt ausgibt:

```
Hallo Klasse,

Das ist mein erstes eigenstaendig erstelltes Programm.

Es sollen noch viele weitere Programme folgen.

Was kommt wohl als Naechstes?
```

Verwenden Sie alle drei Methoden: System.out.print()/println()/printf().

Weisen Sie den Start und das Ende des Code-Blocks mit jeweils einem Kommentar aus.

```
public class
{
  public static void main (String[] args)
  {









  }
}
```

Fach: Anwendungsentwicklung/Programmierung	FIS	Datentypen, Literale, Variablen, Zuweisungen, Arithmetische Op.

2. Geben Sie das Programm ein, compilieren Sie es und führen es aus.
   Variieren Sie die Variableninhalte z.B. so wie vorgeschlagen.
   Was ist das Ergebnis?
   Notieren Sie sich alles, was Ihnen auffällt.

```
public class Zuweisungen
{
  public static void main (String[] args)
  {
     // Deklaration von Variablen
     byte varByte;
     short varShort;
     int varInt;
     long varLong;
     float varFloat;
     double varDouble;
     char varChar;
     boolean varBoolean;

    /* verschiedene
       Wertzuweisungen */

    varByte = 100;                                  // testen Sie mal 256
    System.out.println("Wert in varByte: " + varByte);
    varShort = -25000;                               // testen Sie mal mit 34000
    System.out.println("Wert in varShort: " + varShort);
    varInt = 100000;                                // testen Sie mal mit 3000000000
    System.out.println("Wert in varInt: " + varInt);
    varLong = 300000000;                            // testen Sie mal mit 10000000000000000000
    System.out.println("Wert in varLong: " + varLong);
    varFloat = 12.34F;                              // testen Sie mal mit 12345.6789F
    System.out.println("Wert in varFloat: " + varFloat);
    varDouble = 12345.6789;                         // testen Sie mal mit 123456789.123456789
    System.out.println("Wert in varDouble: " + varDouble);
    varChar = 'A';                                  // testen Sie mal mit '/n'
    System.out.println("Wert in varChar: " + varChar);
    varBoolean = false;                             // testen Sie mal mit 1
    System.out.println("Wert in varBoolean: " + varBoolean);
  }
}
```

Fach: Anwendungsentwicklung/Programmierung	FIS	Datentypen, Literale, Variablen, Zuweisungen, Arithmetische Op.

3. Erstellen Sie ein Programm **Lohn**, das errechnet, wie hoch der Arbeitslohn eines Arbeiters ist, der 40 Arbeitsstunden bei einem Stundenlohn von 15.0 leistet und Steuern mit einem Steuersatz von 12% zahlen muß.

   Verwenden Sie zur Berechnung Variablen des benötigten Typs, geben Sie sowohl die Arbeitsstunden als auch den Bruttoarbeitslohn und den Steuerbetrag aus.

   Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

4. Schreiben Sie ein Programm **Niederschlag**, das die durchschnittliche Niederschlagsmenge für die drei Monate April (12), Mai (14) und Juni (8) berechnet.

   Deklarieren und initialisieren Sie eine Sie eine int-Variable für jeden Monat. Berechnen Sie den Durchschnitt und geben Sie sowohl die Monatsniederschläge als auch den Durchschnittswert aus.

   Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

Fach: Anwendungsentwicklung/Programmierung	FIS	Objektdaten

5. Erstellen Sie heute ein Programm **EntferneZeichen**, das aus dem String-Objekt ’’ Abensberg ist eine schöne Stadt, die Vielerlei zu bieten hat. “ durch Anwendung der String-Methode trim() ein neues String-Objekt erzeugt.

   Geben Sie den ersten String, wie angegeben, d.h. mit Leerzeichen an Anfang und Ende, ein. Geben Sie beide Strings und ihre Länge aus.

   Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

6. Erstellen Sie ein Programm **AendereZeichen**, das aus dem String-Objekt “Abensberg ist eine schöne Stadt, die Vielerlei zu bieten hat.“ durch Anwendung der String-Methoden toUpperCase() und toLowerCase() zwei neue String-Objekte erzeugt.

   Geben Sie alle Strings aus.

   Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

Fach: Anwendungsentwicklung/Programmierung	FIS	Input/Output

7. Erstellen Sie ein Programm **KreisFlaeche**, das den Radius des Kreises als Input anfordert, die Kreisfläche berechnet und ausgibt.

   Verwenden Sie die *nextInt()*-Methode, um den Radius als Ganzzahl einzulesen.

   Verwenden Sie die Konstante PI der Klasse *Math*, um die Fläche mit der Kreisflächenformel F= r*r*PI auszurechnen.

   Geben Sie das Ergebnis mit *println()* aus.

   Der Dialog sollte so aussehen:
   Geben Sie den Radius ein:
   3
   Der Radius ist: 3. Die Fläche beträgt: 28.274333882308138.

   Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

8. Erstellen Sie ein Programm **CentToEuro**, das eine Zahl in Cent einliest und in Euro und Cent ausgibt.

   Der Dialog sollte so aussehen:
   Geben Sie die Cent ein:
   378
   378 Cent ergeben 3 Euro und 78 Cent.

   Verwenden Sie Ganzzahl-Arithmetik und vermeiden Sie Gleitpunkt-Arithmetik.

   Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

9. Erstellen Sie ein Programm **GasRechnung**, das den Preis pro Kubikmeter Gas und die verbrauchten Kubikmeter einliest und den sich ergebenden Betrag ausgibt.

   Der Dialog sollte so aussehen:
   Geben Sie den Preis pro Kubikmeter ein:
   14,56
   Geben Sie die verbrauchten Kubikmeter ein:
   1268
   Die Jahreskosten sind: 18462,08

   Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

Fach: Anwendungsentwicklung/Programmierung	FIS	Input/Output

10. Erstellen Sie ein Programm **OhmschesGesetz**, das den Widerstand eines Stromkreises nach der Formel R = U / I errechnet.

    Lesen Sie die Integer-Werte U für die Spannung und I für die Stromstärke ein und berechnen Sie R als double (Ausgabe mit 3 Dezimalstellen).

    Geben Sie das Ergebnis mit *printf()* aus.

    Der Dialog sollte so aussehen:
    Geben Sie die Spannung ein:
    230
    Geben Sie die Stromstaerke ein:
    15
    Der Widerstand betraegt: 15,333

    Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

11. Erstellen Sie ein Programm **ParWid**, das den Gesamtwiderstand zweier parallel geschalteter Widerstände nach der Formel $R_g = (R_1 * R_2) / (R_1 + R_2)$ errechnet.

    Lesen Sie die Double-Werte für die Widerstände R_1 und R_2 ein berechnen Sie R_g als double (Ausgabe mit 3 Dezimalstellen).

    Der Dialog sollte so aussehen:
    Geben Sie den Widerstand 1 ein:
    12,5
    Geben Sie den Widerstand 2 ein:
    18,5
    Der Gesamt-Widerstand betraegt: 7,460

    Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

12. Erstellen Sie ein Programm **BenzinVerbrauch**, das nach Eingabe der gefahrenen Kilometer und der Menge des verbrauchten Kraftstoffs den Durchschnitts-verbrauch auf 100 KM berechnet und ausgibt..

    Der Dialog sollte so aussehen:
    Verbrauchter Kraftstoff: 67,5
    gefahrene Strecke in KM: 600:
    Durchschnittsverbrauch: 11,25 l/100km

    Drucken Sie Ihr Programm aus, kopieren Sie es auf einen Stick und bringen Sie es zur nächsten Stunde mit.

Fach: Anwendungsentwicklung/Programmierung	FIS	Input/Output

13. Erstellen Sie ein Programm **BerechneMittel**, das neben dem arithmetischen Mittel auch das harmonische Mittel zweier ganzer Zahlen berechnet.

    Formeln:
    Arithmetisches Mittel: (x+y)/2
    Harmonisches Mittel: 2/(1/x+1/y)

    Lesen Sie die Integer-Werte x und y für zwei Zahlen ein.

    Berechnen Sie das arithmetisches Mittel als Zahl vom Datentyp float.

    Berechnen Sie das harmonische Mittel als Zahl vom Datentyp double.

    Wenden Sie für das arithmetische Mittel die Datenkonvertierung vom Typ *casting* und für das harmonische Mittel die Datenkonvertierung vom Typ *numeric promotion* an.

    Geben Sie die Ergebnis mit *printf()* aus.

    Der Dialog sollte so aussehen:
    Geben Sie die erste Zahl ein: 12
    Geben Sie die zweite Zahl ein: 16
    Arithmetisches Mittel: 14,000
    Harmonisches Mittel: 13,714

    Drucken Sie Ihr Programm aus und kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	If-Anweisung

14. Erstellen Sie ein Programm **SonderPreis**, das für alle Einkäufe über 100€ einen Rabatt von 20% gewährt.

    Verwenden Sie Ganzzahl-Arithmetik.

    Der Dialog sollte so aussehen:
    Geben Sie den Gesamtpreis ein: 1200
    Der rabattierte Preis ist: 960

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

15. Erstellen Sie ein Programm **DeinAlter**, das das Geburtsjahr und das aktuelle Jahr kodiert in zwei Ziffern (also z.B. 84 für 1984, 13 für 2013) abfragt und das Alter des Anwenders ausgibt.

    Verwenden Sie Ganzzahl-Arithmetik.

    Der Dialog sollte so aussehen:
    Geburtsjahr: 62
    Aktuelles Jahr: 99
    Alter: 37

    Oder
    Geburtsjahr: 62
    Aktuelles Jahr: 04
    Alter: 42

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Boolsche Ausdrücke

Bewerten Sie den Ausdruck	true	false
1 + 2 == 3		
25 < 5 * 5		
8 + 1 >= 3 * 3		
5 > 2 || 12 <= 7		
5 > 2 && 12 <= 7		
3 == 8 && 6 != 6		
3 == 8 || 6 != 6		
1 + 2 > 4 – 2 && 12 < 23		
1 + 2 > 4 – 2 || 12 < 23		
1 + 2 > 4 – 2 && 12 > 23		
1 + 2 > 4 – 2 || 12 > 23		
!((23 + 17) == 40)		
((23.0 + 17) != 40.0) && true		
true || (2 > 3)		
(!('a' == 'b')) || (!(!(2 == 2)))		
('a' == 'a') && (2 < 3)		
(true && ('x' == 'x')) || false		
(''ab'' + ''cd'') == ''ab cd''		
((6.6 / 3.3)) == 2		
(10 / 4 == 2) || ('a' != 'b')		
((10 / 3) – 3) * 1234567891234 > 0		
'Q' == 'q'		
!(''Hallo'' != ''Hallo'')		
(!('A' == 'a')) == true		
!((23 + 17) == 40)		

F**ach**: Anwendungsentwicklung/Programmierung	FIS	Verschachtelte If-Anweisung

16. Erstellen Sie ein Programm **GenauesAlter**, das das aktuelle Datum und das Geburtsdatum einliest und das aktuelle Alter ausgibt.

    Verwenden Sie Ganzzahl-Arithmetik.

    Der Dialog sollte so aussehen:
    Welches Datum haben wir heute?
    Tag: 14
    Monat: 10
    Jahr: 2014

    Wann sind Sie geboren?
    Tag: 20
    Monat: 03
    Jahr: 1993

    Sind sind also xx Jahre alt.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	switch, while, for-Anweisung

17. Erstellen Sie ein Programm **TaschenRechner**, der die Operatoren +, -, *, / beherrscht und den Operator auf zwei Zahlen anwendet.

    Lesen Sie Zahl1, Operator und Zahl2 ein. Verwenden Sie die switch-Anweisung zur Auswahl des Code-Zweiges.

    Der Dialog sollte so aussehen:
    Operator: +
    Zahl1: 13,14
    Zahl2: 14,15
    13.14 * 14.15 = 27.29

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

18. Erstellen Sie ein Programm **Teiler1**, das die ganzen Zahlen von 1 bis 50 untersucht, ob sie erstens durch 3 teilbar sind und zweitens nicht durch 5 aber durch 4 teilbar sind.

    Verwenden Sie for-Schleifen als Kontrollstruktur.

    Die Ausgabe sollte so aussehen:
    Alle Zahlen zwischen 1 und 50, die durch 3 teilbar sind:
    3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48

    Alle Zahlen zwischen 1 und 50, die nicht durch 5 aber durch 4 teilbar sind:
    4 8 12 16 24 28 32 36 44 48

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

19. Erstellen Sie ein Programm **FakultaetBis5**, das für alle ganzen Zahlen n von 1 bis 5 die Fakultät n! errechnet und ausgibt (n! = 1*2*3*…*n).

    Verwenden Sie eine while-Schleifen als Kontrollstruktur.

    Die Ausgabe sollte so aussehen:
    Die Fakultaeten sind:
    1! = 1
    2! = 2
    3! = 6
    4! = 24
    5! = 120

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	switch, while, for-Anweisung

20. Erstellen Sie ein Programm **Teiler2**, das die ganzen Zahlen von 1 bis 50 untersucht, ob sie erstens durch 3 teilbar sind und zweitens nicht durch 5 aber durch 4 teilbar sind.

    Verwenden Sie while-Schleifen und die continue-Anweisung als Kontrollstruktur.

    Die Ausgabe sollte so aussehen:
    Alle Zahlen zwischen 1 und 50, die durch 3 teilbar sind:
    3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48

    Alle Zahlen zwischen 1 und 50, die nicht durch 5 aber durch 4 teilbar sind:
    4 8 12 16 24 28 32 36 44 48

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

21. Erstellen Sie ein Programm **PruefeName**, das zur Überwachung von Personennamen eingesetzt wird. Es werden Namen eingelesen und wieder ausgegeben. Falls der Name Saenger auftaucht, soll das Programm eine Warnmeldung ausgeben und abbrechen.

    Verwenden Sie eine do-Schleife und die break-Anweisung als Kontrollstruktur.

    Die Ausgabe sollte so aussehen:
    Butte geben Sie einen Namen ein:
    Maier – OK
    Butte geben Sie einen Namen ein:
    Mueller – OK
    Butte geben Sie einen Namen ein:
    Saenger – Achtung, Sicherungsmassnahmen ergreifen!

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Sichtbarkeit und Lebensdauer

22. Betrachten Sie das folgende Programm.

Geben Sie an, welche Variablen an den mit /* i */ (i = 1, 2, 3) gekennzeichneten Stellen leben und welche davon sichtbar sind.

```
public class Aufgabe1
{
        static int meineZahl = 10;

        static int maxab(int a, int b)
        { /* 3 */
                if (a > b)
                {
                        return a;
                }
                else
                {
                        return b;
                }
        }

        static int berechneZahl(int x)
        { /* 2 */
                int y;

                y = meineZahl * maxab(x,0);
                return y;
        }

        public static void main (String[] args)
        { /* 1 */
                int n;

                n = 10;
                System.out.println(berechneZahl(n));
    }
}
```

23. Erstellen Sie ein Programm **FlaecheDreieck**, das zu drei gegebenen Seitenlängen die Fläche eines Dreiecks errechnet.

    Der Vorgang soll wiederholt werden, bis für a eine 0 eingegeben wird.

    Lesen Sie die Seitenlängen als Integer ein, geben Sie die Fläche als Double mit zwei Dezimalstellen aus.

    Berechen Sie die Fläche in einer Methode *area(a, b, c)* nach der Formal von Heron s = (a+b+c)/2, F= Wurzel[s(s-a)(s-b)(s-c)]. Die Wurzel errechnet man mittels Math.sqrt(x).

    Der Dialog sollte so aussehen:

    Programmbeginn!

    Geben Sie die Laenge von a ein (Ende = 0): 10
    Geben Sie die Laenge von b ein: 11
    Geben Sie die Laenge von c ein: 12
    Die Flaeche ist: 51,52

    …

    Programmende!

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

24. Erstellen Sie ein Programm **QuersummeMal2**, das die finale Quersumme einer eingelesenen Zahl iterativ und rekursiv berechnet.

    Unter der finale Quersumme versteht man die einstellige Quersumme einer Zahl (Beispiele: 6 -> 6, 24 -> 2+4=6, 789 -> 7+8+9=24 -> 2+4=6).

    Der Dialog sollte so aussehen:

    Geben Sie eine Zahl ein: 789
    Finale Quersumme (iterativ): 24
    Finale Quersumme (rekursiv): 24

Fach: Anwendungsentwicklung/Programmierung	FIS	Sichtbarkeit und Lebensdauer

22. Betrachten Sie das folgende Programm.
Geben Sie an, welche Variablen an den mit /* i */ (i = 1, 2, 3) gekennzeichneten Stellen leben und welche davon sichtbar sind.

```
public class Aufgabe1
{
        static int meineZahl = 10;

        static int maxab(int a, int b)
        { /* 3 – es leben: meine Zahl, args, n, x, y, a, b; sichtbar sind; meineZahl, a, b*/
            if (a > b)
            {
                return a;
            }
            else
            {
                return b;
            }
        }

        static int berechneZahl(int x)
        { /* 2 – es leben: meineZahl, args, n, x, y; sichtbar sind: meine Zahl, x, y*/
            int y;

            y = meineZahl * maxab(x,0);
            return y;
        }

        public static void main (String[] args)
        { /* 1 - es leben: meine Zahl, args, n; sichtbar sind: meineZahl, args, n */
            int n = 10;

            System.out.println(berechneZahl(n));
    }
}
```

Fach: Anwendungsentwicklung/Programmierung	FIS	Kontrollstrukturen,Struktogramme

25. Es soll ein Programm **GesWid** erstellt werden, das wahlweise den Gesamtwiderstand R_g zweier parallel ($R_g = (R_1 * R_2)/(R_1 + R_2)$) oder in Reihe ($R_g = R_1 + R_2$) geschalteter Widerstände R_1 und R_2 berechnet.

    Lesen Sie den String für Parallel oder Reihen-Schaltung, die double-Werte für die Widerstände R_1 und R_2 ein und berechnen Sie R_g als double (Ausgabe mit 3 Dezimalstellen). Verwenden Sie die if-Anweisung zur Strukturierung.

    Der Dialog sollte so aussehen:
    Widerstandsberechnung
    (P)arallel- oder (R)eihenschaltung? P
    R1 in Ohm: 12000
    R2 in Ohm: 10000
    Der Gesamtwert der Parallelschaltung beträgt: 5454,545 Ohm.
    Ende Widerstandsberechnung

    Erstellen Sie zunächst ein Struktogramm und dann das Programm anhand des Struktogramms.

26. Es soll ein Programm **SummeKehrwert** erstellt werden, das die Summe von 1/1 + 1/2 + 1/3 + …. + 1/n errechnet. n wird vom Anwender eingegeben.

    Lesen Sie n als int ein. Berechnen Sie die Summe als double. Verwenden Sie die for-Anweisung zur Strukturierung.

    Der Dialog sollte so aussehen:
    Geben Sie n ein: 9
    Die Summe der Kehrwerte ist: 2,83

    Erstellen Sie zunächst ein Struktogramm und dann das Programm anhand des Struktogramms.

27. Es soll ein Programm **ZahlWort** erstellt werden, das nach Eingabe der Zahl 1 (eins, one, uno), 2 (zwei, two, due), 3 (drei, three, tre), 4 (vier, four, quattro) oder 5 (fuenf, five, cinque) das deutsche, englische und italienische Wort ausgibt.

    Lesen Sie die Zahl als int ein. Verwenden Sie die switch-Anweisung zur Strukturierung.

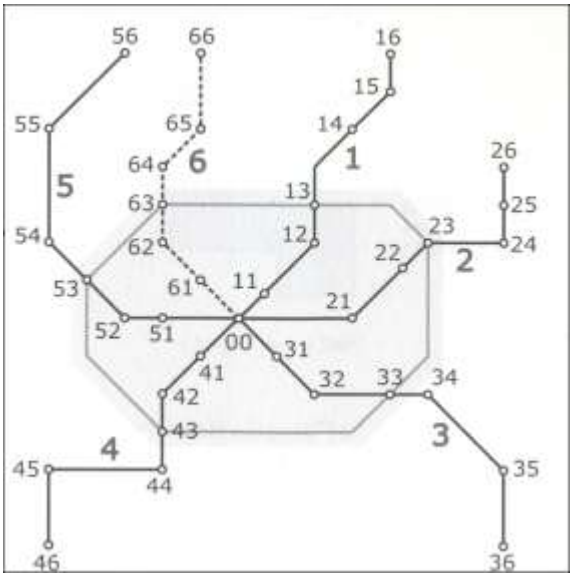
    Der Dialog sollte so aussehen:
    Geben Sie eine Zahl ein: 3
    3: drei, three, tre

    Erstellen Sie zunächst ein Struktogramm und dann das Programm anhand des Struktogramms.

Fach: Anwendungsentwicklung/Programmierung	FIS	S-Bahn von Byteburg

28. Erstellen Sie ein Programm **BahnTarif**, das zwei Haltepunkte im S-Bahnnetz von Byteburg als Parameter übergeben bekommt und den entsprechenden Fahrpreis berechnet.

    Das S-Bahnnetz von Byteburg ist regelmäßig aufgebaut.
    - es gibt derzeit 5 Hauptlinien und eine Ringlinie,
    - die nächste Linie 6 ist bereits im Bau, wird aber noch nicht benutzt,
    - die Stationen der Hauptlinien sind von der Mitte zum Außenbereich von 1 bis 6 durchnummeriert,
    - die Station in der Mitte, das Zentrum, hat die Bezeichnung 00,
    - die Ringlinie verbindet die Stationen 3 der Hauptlinien,
    - die Stationen sind mit einem zweistelligen Code gekennzeichnet: die Zehnerstelle nennt die Hauptlinie und die Einerstelle nennt die Station,
    - im Innenraum des S-Bahnnetzes liegen das Zentrum und alle Stationen inklusive der Ringlinie,
    - alle anderen Stationen liegen im Außenraum.



Folgende Regeln gelten für die Fahrpreise:

- eine Fahrt kostet 2BT = Byte-Taler (z.B. für 11 nach 13),
- jede überquerte Zonengrenze kostet 1BT zusätzlich (z.B. für 11 nach 14 = 3BT)
- benutzte Endstation kostet 1BT zusätzlich (z.B. für 14 nach 16 = 3BT)
- Fahrt zwischen 2 benachbarten Stationen kostet immer nur 1BT, auch wenn eine Zonegrenze dazwischen liegt oder eine Endstation benutzt wird (z.B. für 13 nach 14 = 1BT, für 15 nach 16 = 1BT, für 13 nach 53 = 1BT)

Berücksichtigen Sie, dass nach Linie 6 noch weitere Hauptlinien gebaut werden können, die die Ringlinie an Station 3 schneiden.
Aber auch dann bleiben die Hauptlinien im Uhrzeigersinn fortlaufend durchnummeriert.

Das Programm soll in der Zukunft mit möglichst wenig Aufwand an weitere Linien anpassbar sein.

Sie sollten zunächst generelle Überlegungen anstellen, die zu verwenden Variablen festlegen und dann ein Struktogramm erstellen.

Drucken Sie Ihr Struktogramm und ihr Programm aus oder kopieren Sie beides auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Übungen zu GdP

29. Erstellen Sie ein Programm **Maexchen**, das ein einfaches Würfelspiel simuliert.

    Es wird mit zwei Würfeln gespielt, die Punkte der Würfel werden als Kommandozeilen-Parameter übergeben.

    Der Wert des Wurfs ergibt sich wie folgt:

    a) der Wurf 1, 2 heißt Mäxchen und ist 1000 Punkte wert
    b) ein Wurf mit zwei gleichen Augenzahlen ist ein Pasch und 100 Punkte wert
    c) sonst ist der Wert: 10*(höhere Zahl)+(niedrige Zahl), z.B. 3, 5 für 53 Punkte

    Das Programm soll den Wert des Wurfes berechnen und ausgeben.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

30. Erstellen Sie ein Programm **QuWurzel**, das eine Zahl n > 0 einliest und die Quadratwurzel x der Zahl n anhand des Newton'sche Approximationsverfahren ermittelt.

    Man beginnt das Verfahren mit einem Schätzwert, z.B. q=1.
    Daraus ergeben sich folgende Möglichkeiten:
    a) q=x: q*q = n, d.h. die Wurzel ist gefunden
    b) q < x: q*x < x^2, d.h. q*x < n, also q < x < n/q
    c) q > x: entsprechend n/q < x < q

    Als neuer Schätzwert wird das arithmetische Mittel von q und n/q verwendet.

    Das Verfahren wird wiederholt, bis q^2 nahe an n liegt, z.B. $|n - q^2| < n*10^{-8}$
    Den Betrag kann mit der Klassen-Methode Math.abs(n- q*q) berechnen.

    Lesen Sie in einer Schleife Werte für n ein und geben Sie den Approximations-wert und die Anzahl der Appriximationsschritte aus.

    Das Programm wird beendet, wenn 0 eingegeben wird.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Übungen zu GdP

31. Erstellen Sie ein Programm **SekZuZeit**, das in einer Schleife Sekunden als Integer einliest und als Stunden:Minuten:Sekunden ausgibt.

    Das Einlesen soll in einer Methode erfolgen.
    Das Umrechnen und Ausgeben soll in einer Methode erfolgen.

    Der Dialog sollte so aussehen:
    Programmbeginn.
    Geben Sie die Sekunden ein (Ende=0): 1234
    1234: 0:20:34
    ….
    Programmende.

    Eingaben kleiner 0 sollen als Fehler abgewiesen werden.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

32. Erstellen Sie ein Programm **APotenzB**, das zwei Zahlen ganze Zahlen a und b (jeweils größer als 0) einliest und die Potenz a**b in zwei Methoden mit unterschiedlichen Schleifen (for und while) errechnet und ausgibt.

    Der Dialog sollte so aussehen:
    Geben Sie einen Wert fuer a ein: 3
    Geben Sie einen Wert fuer b ein: 4
    Methode mit for-Schleife ergibt: 81
    Methode mit while-Schleife ergibt: 81

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eindimensionale Arrays

33. Erstellen Sie ein Programm **QuadratArray**, das die Quadratzahlen der Zahlen von 1 bis zu einer ganzzahligen Obergrenze berechnet, speichert und ausgibt.

    Die ganzzahlige Obergrenze soll vom Anwender angefragt werden.

    Die Berechnung der Quadratzahlen soll in einer Methode **berechneQuadrate** (mit der ganzzahligen Obergrenze als Übergabe-Parameter) erfolgen, die die Ergebnisse in einem Array speichert und dieses als Ergebnis zurückgibt.

    Mit einer weiteren Methode **printQuadrate** (mit dem Ergebnis-Array als Übergabe-Parameter) sollen die Quadrate auf die Konsole ausgegeben werden.

    Gestalten Sie die Ein- und Ausgabe Anwender-freundlich, verwenden Sie geeignete und sprechende Namen, sparen Sie nicht mit Kommentaren.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

34. Erstellen Sie ein Struktogramm, auf dessen Basis die Methode **EANPruef** erzeugt werden kann, das die Prüfziffer zu einer zwölfstelligen EAN (Europäische Artikel Nummer) erstellt.

    Die 12-stellige EAN wird als byte-Array übergeben. Die ermittelte Prüfziffer wird als byte-Wert zurückgegeben.

    Die Prüfziffer wird wie folgt berechnet:
    - die 12 Ziffern der EAN werden von links nach rechts addiert
    - vor der Addition werden die Ziffern der gerade Stellen mit 3 multipliziert
    - die Summe wird durch 10 dividiert
    - der Rest wird als ganze Zahl von 10 subtrahiert
    - die Einerstelle der Differenz ist die Prüfziffer

    Drucken Sie Ihr Struktogramm aus oder kopieren Sie es auf einen Stick.

35. Erstellen Sie ein Programm **NotenFIS11AEP**, das die Noten einer Schulaufgabe verwaltet.

    Das Programm hat vier Funktionen:
    a. Noten der Schulaufgabe einlesen, wobei die Anzahl der Schüler abgefragt wird
    b. Notenliste der Schulaufgabe ausgeben
    c. Durchschnittsnote der Klasse ausgeben
    d. Programm beenden

    Die Funktionen werden über ein wiederkehrendes Menü angesteuert.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Mehrdimensionale Arrays

36. Erstellen Sie ein Programm **Stundenplan**, das den Inhalt eines Stundenplans einliest und ausgibt.

    Das Programm hat drei Funktionen:
    a. Stundenplan ausgeben
    b. Stunde eingeben
    c. Programm beenden

    Die Funktionen werden über ein wiederkehrendes Menü angesteuert.

    Der Menue-Dialog könnte so aussehen:

```
***** Stundenplanmanager *****
(1) Stundenplan ausgeben
(2) Stunde eingeben
(3) Programm beenden
Bitte waehlen Sie einen Menuepunkt: 2
```

    Die Ausgabe des Stundenplans könnte so aussehen:

```
Mo      Di      Mi      Do      Fr
--      --      --      --      --
M       P       D       B       E
M       P       D       B       E
E       G       L       D       F
S       G       L       D       F
S               K
                K
```

    Der Dialog zur Eingabe der Stunde könnte so aussehen:

```
** Stunden eintragen **
Waehlen Sie ein Fach (D, E, M, L, F, S, K, G, B, P):  D
Waehlen Sie den Tag(Mo=1, Di=2, Mi=3, Do=4, Fr=5): 1
Waehlen Sie eine Stunde von 1 bis 6: 2
```

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

37. Erstellen Sie ein Programm **NotenSchnitt**, das die Durchschnittsnote einer beliebigen Anzahl von Schülern einer Klasse berechnet.

    Beachten Sie, dass jeder Schüler eine andere Anzahl an Noten gesammelt haben kann.

    Eine Gewichtung der Einzelnoten wird nicht vorgenommen.

    Erstellen Sie zunächst ein Struktogramm und dann ein ablauffähiges Programm, dass folgenden Dialog erzeugt:

Berechnung der Durchschnittsnoten

Anzahl der Schueler: 5

Anzahl der Noten von Schueler 1: 3
Schueler 1, Note 1: 2
Schueler 1, Note 2: 2
Schueler 1, Note 3: 1

Anzahl der Noten von Schueler 2: 5
Schueler 2, Note 1: 1
Schueler 2, Note 2: 2
Schueler 2, Note 3: 1
Schueler 2, Note 4: 3
Schueler 2, Note 5: 2

.
.
.

Durchschnittsnote von Schueler 1: 1,67
Durchschnittsnote von Schueler 2: 1,80
.
.
.

Ende der Berechnung

Drucken Sie Ihr Struktogramm und Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	ArrayList, Stringbuilder

38. Erstellen Sie ein Programm **FussballMannschaft**, das die Mannschaften einer Fußballliga in einer ArrayList speichert, die Liste zum Termin Saisonbeginn ausgibt, die Liste zum Saisonende überarbeitet und dann wieder ausgibt.

    Zum Saisonbeginn gehören der Liga 6 Mannschaften an, zum Saisonende steigen 2 Mannschaften ab und zwei Mannschaften aus der nächst tieferen Liga steigen auf.

    Der Dialog sollte so aussehen:

    ***** Liste der Mannschaften zum Saisonbeginn *****
    1. FC Wadenbrecher
    SC Schienbeintreter
    VfL TrittInsKnie
    Fortuna PfuetzenSchluck
    FC Knickebein 93
    VfB Eisenfuss

    ***** Liste der Mannschaften zum Saisonende *****
    1. FC Wadenbrecher
    VfL TrittInsKnie
    FC Knickebein 93
    VfB Eisenfuss
    Eintracht Haxenbruch
    Borussia ToteErde

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

39. Erstellen Sie ein Programm **Anagramm**, das zwei Worte einliest und feststellt, ob diese Anagramme sind, d.h. aus den gleichen Buchstaben bestehen.

    Groß- und Klein-Schreibung soll dabei ignoriert werden.

    Verwenden Sie die String-Methode
    char[ ] a = s.toCharArray .

    Tip: Sortieren könnte hilfreich sein!

    Der Dialog sollte so aussehen:

    Geben Sie das erste Wort ein: Maus
    Geben Sie das zweite Wort ein: Saum
    Maus und Saum sind ein Anagramm.

    Geben Sie da erste Wort ein: Maus
    Geben Sie das zweite Wort ein: Haus
    Maus und Haus sind kein Anagramm.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

40. Erstellen Sie ein Programm **FloatToString**, das das Divisions-Ergebnis von zwei einzulesenden float-Zahlen als float berechnet und in einen String umwandelt und ausgibt. Der String soll 2 Nachkomma-Stellen berücksichtigen.

    Der Dialog sollte so aussehen:
    Geben Sie die erste Zahl ein: 3300,25
    Geben Sie die zweite Zahl ein: 22,5
    Das Ergebnis ist: 146,68

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

41. Erstellen Sie ein Programm **ArrayToString**, das zehn Buchstaben einliest.

    Das daraus entstandene Wort soll in eingegebener und umgekehrter Reihenfolge ausgegeben werden.
    Dann sollen die mittleren 4 Buchstaben rauslöscht und das verbliebene Wort nochmals vorwärts und rückwärts ausgegeben werden.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Zufallszahlen 1

42. Erstellen Sie ein Programm **RateZahl**, das ein Ratespiel simuliert, in dem der Computer sich eine Zahl zwischen 1 und 10 ausdenkt und der Anwender drei Versuche zum Erraten der Zahl hat.

    Der Dialog sollte so aussehen:

    Ich denke an eine Zahl von 1 bis 10.
    Sie haben 3 Versuche diese Zahl zu erraten.
    Raten Sie!
    Versuch 1: 2
    Falsch
    Versuch 2: 4
    Falsch
    Versuch 3: 7
    Falsch
    Die richtige Zahl war 9.
    Sie haben das Spiel verloren.

    Ich denke an eine Zahl von 1 bis 10.
    Sie haben 3 Versuche diese Zahl zu erraten.
    Raten Sie!
    Versuch 1: 3
    Falsch
    Versuch 2: 5
    RICHTIG!
    Sie haben das Spiel gewonnen.

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

43. Ändern Sie das Programm **Maexchen** zu **MaexchenZufall** so, das die Würfelwerte nicht mehr beim Programmaufruf übergeben werden, sondern das Programm solange den Wurf mit zwei Würfeln simuliert, bis ein Mäxchen geworfen wird.

    Der Dialog sollte so aussehen:

    Programmbeginn!
    1. Wurf: 5 und 3 – 53 Punkte
    2. Wurf: 4 und 4 – 100 Punkte
    …
    n. Wurf: 1 und 2 – 1000 Punkte
    Programmende!

    Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Zufallszahlen 2

44. Erstellen Sie ein Programm **WuerfelStatistik**, das 100 Würfe mit einem Würfel simuliert und die erreichten Häufigkeiten ausgibt

Die Ausgabe sollte so aussehen:

```
<terminated> WuerfelStatistik [Java Application] C:\Program Files\Java\jre8\bin\javaw.exe (15.12.2014 16:55:59)
Bei 100 Würfen mit einem Würfel traten folgende Häufigkeiten auf:
1 - 17
2 - 14
3 - 20
4 - 18
5 - 15
6 - 16
Versuchen Sie es gleich nochmal!
```

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

45. Erstellen Sie ein Programm **RouletteStatistik**, das 1000 Roulette-Versuche simuliert und folgende Daten errechnet:
    i. die am häufigsten geworfene Zahl
    ii. die Verteilung 0, gerade, ungerade

Die Ausgabe sollte so aussehen:

```
<terminated> RouletteStatistik [Java Application] C:\Program Files\Java\jre8\bin\javaw.exe (15.12.2014 16:59:03)
Bei 1000 Versuchen wurde die Zahl 12 mit der Anzahl 36 am haeufigsten erreicht

Statistik für Gerade-Ungerade
0         - 20
gerade    - 477
ungerade  - 503
```

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Zeit

46. Erstellen Sie eine Klasse Zeit.

Gehen Sie vor, wie bei der Erstellung der Klasse **Datum**.

Die Klasse besteht aus der Stunde (primitiver Datentyp *int*, von 0 bis 24), der Minute (primitiver Datentyp *int*, von 0 bis 59) und der Sekunde (primitiver Datentyp *int*, von 0 bis 59).

Legen Sie drei Konstruktoren an:
- Zeit(int int1, int int2, int int3): int1 = Stunde, int2 = Minute, int3 = Sekunde
- Zeit(int int1): int1 = Sekunden (zu zerlegen in Stunde/Minute/Sekunde)
- Zeit(): setzt Stunde, Minute und Sekunde auf 0

Verifizieren Sie die Parameter auf Gültigkeit (0<Stunde<=24, 0<=Minute<=59, 0<=Sekunde<=59).
Bei Ungültigkeit setzen Sie Stunde, Minute und Sekunde auf 0.

Implementieren Sie folgende Methoden:
- String alsString(): gibt ''hh:mm:dd“ zurück
- int als Integer(); gibt die Zeit als Integer h*3600+m*60+s zurück
- boolean vglZeit(Zeit z): true bei Gleichheit, sonst false

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

47. Testen Sie die Klasse Zeit mit einem Programm **ZeitTester,** in dem Sie drei Zeit-Objekte anlegen (eine mit jedem Konstruktor) und diese dann wie folgt ausgeben:

12:12:12 entspricht 43932 Sekunden
19:21:23 entspricht 69683 Sekunden
00:00:00 enstpricht 0 Sekunden

12:12:12 != 19:21:23

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

48. Erstellen Sie ein Programm **DatumZeitTester**, das die Eingabe der aktuellen Zeit und des aktuellen Datums anfordert und diese dann folgend ausgibt:

Geben Sie das aktuelle Datum ein (yyyymmdd): 20131216
Geben Sie die aktuelle Zeit ein (hhmmss): 141022
Wir haben heute den 16. Dezember 2013 und es ist augenblicklich 14:10:22.

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Datum und Zeit

49. Erweitern Sie das Programm **DatumZeitTester**, das die Eingabe der aktuellen Zeit und des aktuellen Datums anfordert und diese dann folgend ausgibt:

```
Geben Sie das aktuelle Datum ein (yyyymmdd): 20131217
Geben Sie die aktuelle Zeit ein (hhmmss): 141022
Ihrer Meinung nach haben wir heute den 17. Dezember 2013 und es ist augenblicklich 14:10:22.
Laut System haben wir heute den 17. Dezember 2013 und es ist augenblicklich 14:10:32.
```

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

50. Erstellen Sie ein Programm **AktuellesAlter**, das das Geburtsdatum abfragt und das Alter des Anwenders ausgibt.

Verwenden Sie Ihr Programm GenauesAlter als Basis.

Der Dialog sollte so aussehen:

```
Geburtsdatum (yyyymmdd): 19620521
Sind sind  51 Jahre alt.
```

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Punkt

51. Erstellen Sie das Programm **QuadratFlaeche**, das die Fläche des von den Punkten 1.0/1.0 und 5.0/5.0 aufgespannten Quadrats errechnet.

Der Dialog sollte so aussehen:
1. Punkt – x-Koordinate: 1.0, y-Koordinate: 1.0
2. Punkt – x-Koordinate: 5.0, y-Koordinate: 5.0
Die Flaeche des aufgespannten Quadrats betraegt: 16,00

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

52. Erstellen Sie das Programm **RechteckFlaeche**, das die Koordinaten zweier Punkte einliest und die Fläche des von den Punkten aufgespannten Rechtecks errechnet.

Der Dialog sollte so aussehen:
x-Koordinate 1. Punkt: 1.0
y-Koordinate 1. Punkt: 2.0
x-Koordinate 2. Punkt: 6.0
y-Koordinate 2. Punkt: 8.0
Die Flaeche des aufgespannten Rechtecks betraegt: 30,00

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Linie

53. Erstellen Sie das Programm **RechteckFlaecheL**, das die Koordinaten zweier Linien einliest und die Fläche des von den Linien aufgespannten Rechtecks errechnet (der erste Punkt beider Linien muss identisch sein, die x-Koordinaten der ersten Linie müssen identisch sein, die y-Koordinaten der zweiten Linie müssen identisch sein).

Der Dialog sollte so aussehen:

Linie1:
x-Koordinate 1. Punkt: 1.0
y-Koordinate 1. Punkt: 2.0
x-Koordinate 2. Punkt: 1.0
y-Koordinate 2. Punkt: 6.0

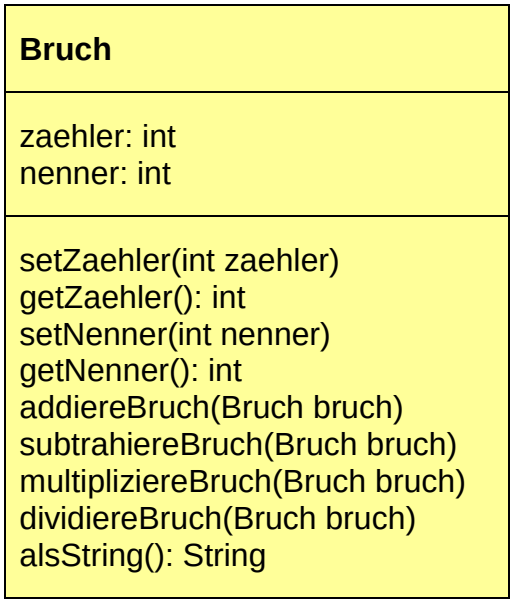
Linie 2:
x-Koordinate 1. Punkt: 1.0
y-Koordinate 1. Punkt: 2.0
x-Koordinate 1. Punkt: 6.0
y-Koordinate 1. Punkt: 2.0

Die Flaeche des aufgespannten Rechtecks betraegt: 25,00

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Bruch 1

54. Erstellen Sie unsere fünfte eigene Klasse **Bruch**, die aus
   i. den int-Variablen *zaehler* und *nenner* besteht,
   ii. die Konstruktoren *Bruch(int x, int y), Bruch()* beinhaltet
   iii. *Getter*- und *Setter*-Objekte für die Variablen beinhaltet.
   iv. die Operations-Methoden *addiereBruch, subtrahiereBruch, multipliziereBruch und dividiereBruch* definiert
   v. eine Methode *alsString* zur Ausgabe des Bruches beinhaltet



55. Erstellen Sie ein Programm **BruchTester**, das für zwei Brüche Zähler und Nenner einliest, den Anfangsstand ausgibt, die Getter- und Setter-Methoden sowie die Operationsmethoden ausführt und die Ergebnisse ausgibt.

Der Dialog sollte so aussehen:

Geben Sie den Zaehler von Bruch 1 ein: 5
Geben Sie den Nenner von Bruch1 ein: 8
Geben Sie den Zaehler von Bruch 2 ein: 7
Geben Sie den Nenner von Bruch 2 ein: 12

Bruch 1: 5/8
Bruch 2: 7/12

Bruch 1: 7/8	(nach getZaehler() + 2)
Bruch 2: 7/13	(nach getNenner() +1)
Bruch 1: 147/104	(nach Addition 1 mit 2)
Bruch 1: 1183/1352	(nach Subtraktion 2 von 1)
Bruch 2: 8281/17576	(nach Multiplikation 2 mit 1)
Bruch 2: 11195912/20792408	(nach Division 2 durch 1)

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Bruch 2

56. Ergänzen Sie unsere fünfte eigene Klasse **Bruch** mit der statischen Variablen *bruchZaehler*, die mittels Klassenkonstruktor mit 0 initialisiert wird und bei jeder Objekt-Erzeugung um 1 hochgezählt wird sowie einer statischen Methoden *setzeBruchZaehlerZurueck*, die die Variable *bruchZaehler* auf 0 setzt.

Bruch
bruchZaehler: int zaehler: int nenner: int
setzeBruchZaehlerZurueck() setZaehler(int zaehler) getZaehler(): int setNenner(int nenner) getNenner(): int addiereBruch(Bruch bruch) subtrahiereBruch(Bruch bruch) multipliziereBruch(Bruch bruch) dividiereBruch(Bruch bruch) alsString(): String

57. Erweitern Sie das Programm **BruchTester** um dreimalige Ausgaben der statischen Variablen *bruchZaehler* zu Anfang des Programms, nachdem die Brüche erzeugt sind und zu Ende des Programms, nachdem die Methode *setzeBruchZaehlerZurueck* aufgerufen wurde.

Der Dialog sollte so aussehen:

```
Bruchzaehler: 0

Geben Sie den Zaehler von Bruch 1 ein: 5
Geben Sie den Nenner von Bruch1 ein: 8
Geben Sie den Zaehler von Bruch 2 ein: 7
Geben Sie den Nenner von Bruch 2 ein: 12
Bruch 1: 5/8
Bruch 2: 7/12
Bruchzaehler: 2

…

Bruch 2: 8281/17576                    (nach Multiplikation 2 mit 1)
Bruch 2: 11195912/20792408             (nach Division 2 durch 1)

Bruchzaehler: 0
```

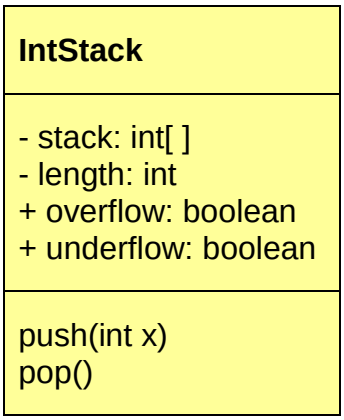
Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Stack

58. Erstellen Sie eine Klasse **IntStack**, die einen Stack nach dem LIFO-Prinzip bereitstellt. Die Klasse soll, wie im UML-Diagramm spezifiziert, implementiert werden.

Der Konstruktor *IntStack* konstruiert *stack* als array in der als int-Parameter übergebenen Grösse, setzt *length* auf 0 sowie *overflow* und *underflow* auf false.

Die Methode *push(int x)* legt einen *int*-Wert auf den *stack* und erhoeht den Zähler *length* (Vorsicht: *overflow –> true*, falls vorgesehener Platz überschritten wird). Die Methode *int pop()* gibt den letzten Eintrag zurück und reduziert den Zähler *length* (Vorsicht: *underflow -> true* bei leerem Stack).



Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

59. Erstellen Sie ein Programm **IntStackTester**, das die Klasse **IntStack** testet z.B. mit folgenden Aktionen:

i. Anlegen eines Stacks der Groesse 10
ii. Nehmen eines Elements vom leeren Stack (mit entsprechender Meldung)
iii. Ablegen von 3, 5, 7 auf dem Stack
iv. Multiplizieren der drei Stack-Elemente (mit Ergebnis-Ausgabe)
v. Nehmen eines Elements vom aktuellen Stack (mit entsprechender Meldung)
vi. Ablegen von 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 auf dem Stack mit jeweiliger Abfrage des Stack-Überlaufs (mit entsprechender Ausgabe)

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Wörterbuch

60. Erstellen Sie eine Klasse **WBGesamt**, die die Funktionalität eines einfachen Wörterbuchs (Begriff und Übersetzung) implementiert. Die Klasse soll, wie im UML-Diagramm spezifiziert, implementiert werden und die einzelnen Einträge in Objekten der Klasse **WBEintrag** ablegen.
Das erste Wort, der Begriff, soll als Schlüssel dienen, das zweite Wort, die Übersetzung, als Inhalt.
Dem Konstruktor *WBGesamt* wird die maximale Anzahl der Einträge übergeben, damit ein Array der benötigten Größe angelegt werden kann.
Dem Konstruktor WBEintrag werden ein Schlüssel und ein dazugehöriger Inhalt übergeben, damit ein Wörterbuch-Element angelegt werden kann.

WBGesamt
- eintrag: WBEintrag[ ] - naechsterPlatz: int - aktuelleGroesse: int
einfuegenEintrag(String schluessel, String inhalt) suchenEintrag(String schlussel): String

WBEintrag
- wSchlussel: String - wInhalt: String
setWSchluessel(String s) getWSchluessel(): String setWInhalt(String s) getWSchluessel(): String

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

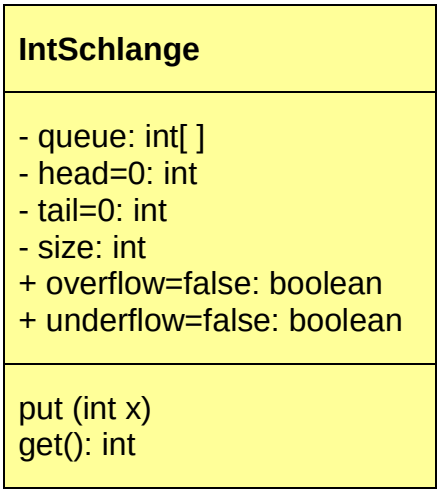
61. Erstellen Sie ein Programm **WoerterBuch**, das ein Wöterbuch mit maximal 10 Einträgen anlegt, mit den Paaren „Haus, house“, „Buch, book“, „Baum, tree“, „Sonne, sun“, „Mond, moon“, „Hund, dog“ füllt und die Ergebnisse von drei Eintrags-Suchen für Buch, Vogel, Mond ausgibt.

Der Dialog sollte so aussehen:
Buch: book
Vogel: null
Mond: moon

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Schlange

62. Erstellen Sie eine Klasse **IntSchlange**, die einen Schlange nach dem FIFO-Prinzip bereitstellt. Die Klasse soll, wie im UML-Diagramm spezifiziert, implementiert werden.
Der Konstruktor *IntSchlange(int size)* konstruiert *queue* als array in der als int-Parameter übergebenen Grösse.
Der Konstruktor *IntSchlange()* konstruiert *queue* als array der Grösse 128.
Die Methode *put()* fügt einen *int*-Wert an die S*chlange* an und erhoeht den Ende-Zähler *tail* (Vorsicht: *overflow –> true*, falls vorgesehener Platz überschritten wird). Die Methode *int get()* gibt den ersten Eintrag zurück und erhöht den Beginn-Zähler *head* (Vorsicht: *underflow -> true* bei leerer Schlange).



Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

63. Erstellen Sie ein Programm **IntSchlangenTester**, das die Klasse **IntSchlange** testet z.B. mit folgenden Aktionen:

i. Anlegen einer Schlange der Groesse 10
ii. Nehmen eines Elements von leerer Schlange (mit entsprechender Meldung)
iii. Erweitern der Schlange um 3, 5, 7
iv. Nehmen von drei Elementen von der Schlange (mit Ergebnis-Ausgabe)
v. Nehmen eines Elements von der Schlange (mit entsprechender Meldung)
vi. Erweitern der Schlange mit 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 mit jeweiliger Abfrage des Schlangen-Überlaufs (mit entsprechender Ausgabe)
vii. Nehmen von drei Elementen von der Schlange (mit Ergebnis-Ausgabe)

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Dynamische Datenstrukturen 1

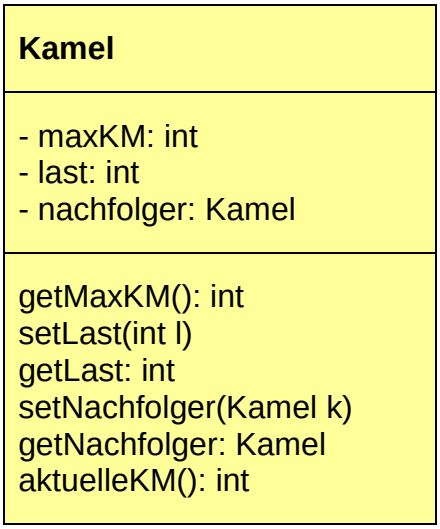
64. Erstellen Sie ein Programm **Josephus**, das errechnet, welcher Gefangene eines Josephus-Ringes am Ende übrig bleibt.

Die Sage sagt:
Vor langer Zeit, als die Sitten noch rau waren, wurden Josephus und eine Anzahl weiterer Gefangener zu einer schlimmen Strafe verurteilt. Nachdem aber der König an diesem Tag Geburtstag hatte, sollte einer der Verurteilten zum Ruhme des Königs freigesprochen werden.
Um eine zufällige Auswahl zu treffen, mussten sich die Gefangenen im Kreis aufstellen. Nun wurde, beginnend mit dem ersten Gefangenen, reihum von 1 bis zu einer vorher festgelegten „fatalen“ Zahl abgezählt.
Wen die Zahl traf, an dem wurde das Urteil sofort vollstreckt und das Abzählen mit dem nächsten Verurteilten neu begonnen.
Der Letzte, der übrig blieb, wurde begnadigt.

Dem Programm soll die Anzahl der Gefangenen und die „fatale“ Zahl über die Kommandozeile übergeben werden.
Das Programm spielt das Abzählen nach, weist aus, welche Gefangenen ausscheiden und welcher Glückliche übrig bleibt.

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

65. Erstellen Sie eine Klasse **Kamel**, eine Klasse **Karawane** und ein Programm **Karawanserei**, das den Einsatz der Klasse vorstellt.



Kamel hat die Instanzvariablen **maxKM** (maximale Anzahl KMH, die das Kamel unbeladen laufen kann; wird beim Erzeugen festgelegt), **last** (Anzahl an Ballen Ladung; jeder Ballen senkt die aktuellen KMH, die das Kamel laufen kann, um 1; falls **last** >= **maxKM** ist, legt sich das Kamel nieder) und **nachfolger** (Adresse des nächsten Kamels; **null**, wenn es sich um das letzte Kamel handelt).

Kamel hat **Getter** für alle drei Variablen und **Setter** für **last** und **nachfolger** (**maxKM** wird beim Erzeugen eingetragen und nicht mehr geändert).

Karawane
- erstesKamel=null: Kamel
aktuelleKM(): int hinzufuegenKamel(Kamel k) entfernenKamel(Kamel k) entladenAlle() erhoeheLast(int l)

Karawane hat die Instanzvariable erstes Kamel, die beim Erzeugen auf null gesetzt wird.
aktuelleKM: gibt die aktuellen KMH, die die Karawane laufen kann (entspricht den KMH des langsamsten Kamel)
hinzufuegenKamel: fügt ein Kamel der Karawane hinzu
entfernenKamel: nimmt ein Kamel aus der Karawane heraus
entladenAlle: entfernt die Last von alle Kamelen
erhoeheLast: verteilt zusätzliche Ballen auf die Kamele der Karawane (dabei sollen die KMH der Karawane möglichst hoch bleiben)

Das Programm Karawanserei hat folgenden Ablauf:

i. Erzeugen der Kamele preiseAllah mit der 8 KMH und wuestenWind mit 7 KMH
ii. Erzeugen der Karawane saharaExpress
iii. Erweitern des saharaExpress mit preiseAllah und wuestenWind
iv. Ausgeben der Reisegeschwindigkeit von saharaExpress
v. Beladen der Karawane mit 5 Ballen Ladung
vi. Ausgeben der Reisegeschwindigkeit von saharaExpress
vii. Entladen der Karawane
viii. Ausgeben der Reisegeschwindigkeit von saharaExpress

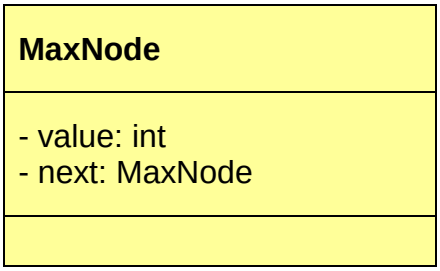
Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Dynamische Datenstrukturen 2

66. Erstellen Sie ein Programm **BestimmeMaximum**, das in einer dynamisch verketteten Liste aus Knoten, die Zahlen enthalten, die größte Zahl sucht und ausgibt.

Vorgehensweise:

i. Erstellen Sie zunächst eine Klasse MaxNode mit den Instanzvariablen value und next und einem Konstruktor, der Werte für beide Variablen übernimmt.



ii. Lesen Sie in einer statischen Methode die zu betrachtenden Zahlen von der Konsole ein (Endekriterium 0) und erzeugen Sie damit eine Liste von Knoten.

iii. Bestimmen Sie in einer weiteren statischen Methode das Maximum der Zahlen.

iv. Geben Sie das Maximum aus.

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Filehandling 1

67. Erstellen Sie ein Programm **SchuelerFIS11**, das die Namen der Schüler unserer Klasse von der Konsole einliest und in eine Datei SchuelerFIS11.txt einträgt.
Falls die Datei noch nicht existiert, soll sie angelegt werden.

Der Dialog könnte so aussehen:
Erfassung der Schueler der Klasse FIS11!
Geben Sie die Namen der Schueler ein (Ende zum Beenden der Erfassung):

Coscin, Dirie
…
Zötl, Christoph
Ende

Vielen Dank fuer Ihr Muehen!

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

68. Erstellen Sie ein Programm **KopiereDatei**, das den Namen einer Input-Datei und den Namen einer Output-Datei von der Konsole einliest, die Existenz der Input-Datei prüft (und ggf. mit einer Fehlermeldung eine neue Eingabe anfordert), die Output-Datei bei Bedarf anlegt und den Inhalt der Input-Datei in die Output-Datei kopiert.

Der Dialog könnte so aussehen:
KopiereDatei gestartet!

Input-Datei: xxx.txt
Output-Datei: yyy.txt

KopiereDatei beendet.

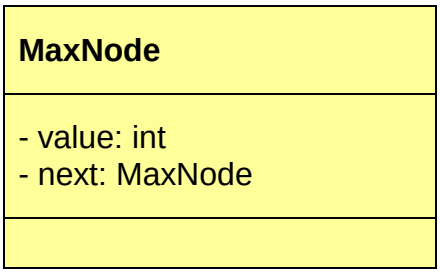
Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Filehandling 1

69. Erstellen Sie ein Programm **BestimmeMaximumF**, das in einer dynamisch verketteten Liste aus Knoten, die Zahlen enthalten, die größte Zahl sucht und ausgibt.

Vorgehensweise:

i. Erstellen Sie zunächst eine Klasse MaxNode mit den Instanzvariablen value und next und einem Konstruktor, der Werte für beide Variablen übernimmt.



ii. Lesen Sie in einer statischen Methode die zu betrachtenden Zahlen aus einer Datei ein und erzeugen Sie damit eine Liste von Knoten.

iii. Bestimmen Sie in einer weiteren statischen Methode das Maximum der Zahlen.

iv. Geben Sie das Maximum aus.

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Filehandling 2

70. Erstellen Sie ein Programm **ZugangsKontrolle**, das die eine achstellige UserId und ein sechsstelliges Password von der Konsole einliest und auf Gültigkeit prüft.

Die Eingabe der UserId/Password-Kombination soll maximal dreimal möglich sein.

Zur Verifikation werden die UserId und das Passwort mit den Einträgen in der Datei UIDPWD.txt verglichen, die pro verfügbarem Eintrag eine Zeile mit UserId (Spalte 1 bis 8) und Password (Spalte 10 bis 15) enthält.

UIPDPWD.txt

```
BraHmann EDuaD
LICklede RkUrt
WasSERMa nnUDO
```

Der Dialog könnte so aussehen:

```
Zugangskontrolle!

Geben Sie bitte ihre achstellige UserId ein:
Geben Sie bitte ihr sechstelliges Password ein:
UserId oder Password nicht gueltig!

Geben Sie bitte ihre achstellige UserId ein:
Geben Sie bitte ihr sechstelliges Password ein:
UserId oder Password nicht gueltig!

Geben Sie bitte ihre achstellige UserId ein:
Geben Sie bitte ihr sechstelliges Password ein:
OK, Sie erhalten Zugang zum System.

oder

Sorry, die Anzahl der gueltigen Versuche wurde ueberschritten,
wenden Sie sich bitte an Ihren System-Administrator.
```

Machen Sie erst Grundsatzüberlegungen zu Struktur, Funktionen und Variablen, erstellen Sie dann ein Struktogramm, bevor Sie das Programm implementieren.

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

71. Erstellen Sie ein Programm **LogCheck**, das die tägliche LOG-Datei zur Plattenbelegung eines Server-Verbundes auswertet und Meldungen für alle Platten ausgibt, deren Auslastung über 80% liegt.

Die Struktur der Datei:

Spalte 01-10: Name des Servers
Spalte 11-20: Name des Platte
Spalte 21-30: Größe der Platte in GB
Spalte 31-40: Freier Platz auf der Platte in GB

LogVol140126.txt

```
SERV111111HD0000000100001000000000500000
SERV010101HD0000011100002000000000028000
SERV010101HD0000111100003000000000600000
SERV111111NAS0000011000010050000000700000
SERV111112NAS0001111000020050000000800000
SERV022033NAS0002220002000000000001800000
SERV022033NAS0000222003000000000000234567
SERV022033NAS0002223045000000000005678901
SERV023034NAS0053333050000000000044444444
SERV024035HD0000022200030000000000500000
SERV056780HD0001222200020000000000400000
SERV454545HD5000222200010000000000300000
```

Der Dialog könnte so aussehen:

```
LogCheck gestartet!

Achtung: Platte HD00000111 - Server SERV010101 - Auslastung 86%
Achtung: Platte NAS0000222 - Server SERV022033 - Auslastung 91%

LogCheck beendet.
```

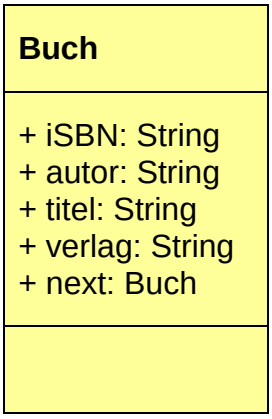
Machen Sie erst Grundsatzüberlegungen zu Struktur, Funktionen und Variablen, erstellen Sie dann ein Struktogramm, bevor Sie das Programm implementieren.

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

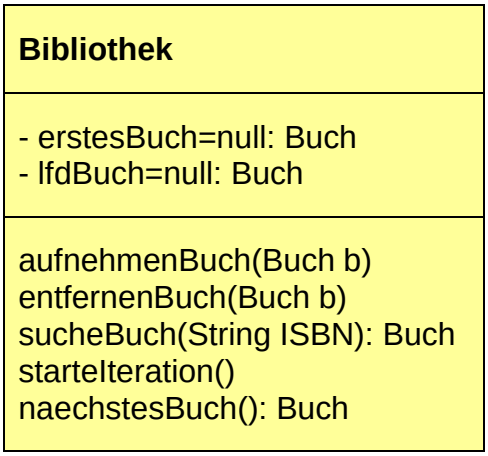
Fach: Anwendungsentwicklung/Programmierung	FIS	Dyn. Datenstr. + File Handling

72. Erstellen Sie eine Klasse **Buch**, eine Klasse **Bibliothek** (dynamisch verkettete Liste von Büchern) und ein Programm **Verwaltung**, das den Einsatz der Klasse vorstellt.

Buch ist eine einfache Klasse, die ein Buch (mit Zeiger auf ein weiteres Buch) repräsentiert.
Bibliothek repräsentiert eine Menge an Büchern (sortiert nach der ISBN). Sie bietet die Funktionen Aufnehmen eines Buches, Suchen eines Buches, Entfernen eines Buches und Iterieren über alle Bücher, d.h. Ausgaben aller Bücher Buch für Buch.
Verwaltung liest zur Initialisierung den Gesamtbestand aus einer Datei **Buecher.txt** ein, füllt damit eine Bibliothek und testet die Funktionalität durch Suchen, Löschen und Hinzufügen von Büchern.



Buch hat die Instanzvariablen **ISBN** (eindeutig, daher der Schlüssel), **autor, titel, verlag** und **next** (zeigt auf das nächste Buch der Bibliothek).
Die Variablen vom Typ String werden mit ’’ ’’ initialisiert, die Variable next mit Null, d.h. es ist weder ein Konstruktor noch andere Methoden notwendig.



Bibliothek hat die Instanzvariable **erstesBuch**, die auf das erste Buch zeigt und **lfdBuch**, die auf das aktuelle Buch einer Iteration zeigt..
aufnehmenBuch: fügt ein Buch hinzu
entfernenBuch: löscht ein Buch
sucheBuch: prüft die Existenz des Buches
starteIteration: setzt **lfdBuch** auf den Anfang der Bibliothek
naechstesBuch: liefert in der Iteration den nächsten Eintrag und setzt **lfdBuch**

Bibliothek.txt liefert den Startbestand. Ein Satz der Datei hat die Form:
01-10: ISBN, z.B. 2-33-123-4
11-20: Autor, z.B. Simpel
21-50: Titel, z.B. Programmierung leicht gemacht
51-70: Verlag, z.B. Schulbuch-Verlag

Das Programm **Verwaltung** hat folgenden Ablauf:

- Einlesen der Datei
- Füllen einer Bibliothek
- Ausgeben des Buchbestands
- Suchen eines Buches
- Löschen eines Buches
- Aufnehmen eines neuen Buches
- Ausgeben des Buchbestands

Drucken Sie Ihr Programm aus oder kopieren Sie es auf einen Stick.

Fach: Anwendungsentwicklung/Programmierung	FIS	Eigene Klassen: Bruch

73. Die Klasse **Bruch** aus W17 hatte das Problem, das die Brüche mit zunehmenden Aktionen immer größer wurden.

Abhilfe schafft das Teilen durch den ggt von Zähler und Nenner nach jeder Aktion.

Ein einfacher Algorithmus zum Bestimmen des ggt zweier Zahlen x und y lautet:

```
int ggt(int x, int y)
{
    int r = x % y;

    while (r != 0)
    {
            x = y;
            y = r;
            r = x % y;
    }
    return y;
}
```

Erstellen Sie eine Klasse **BruchKurz** als Kindklasse von Bruch, in der zum Abschluß der mathematischen Methoden der Bruch um den ggt gekürzt wird.

74. Erweitern Sie das Programm **BruchTester** zu **BruchTesterKurz**, indem anstatt der Klasse Bruch die Kindklasse BruchKurz verwendet wird.

Fach: Anwendungsentwicklung/Programmierung	FIS	Vererbung 1

75. Erstellen Sie eine abstrakte Klasse **Haustier**. Die Klasse besteht aus:

- den Instanzvariablen
  - String name;
  - double futtervorrat;
  - int anzahlTage; // gibt die Anzahl Tage aus, die der Futtervorrat reicht

- einem Konstruktor, der die beiden Instanzvariablen *name* und *futtervorrat* initialisiert (dieser Kontruktor kann zwar nicht direkt genutzt werden, da die abstrakte Klasse Haustiere nicht instanziiert werden kann, aber er kann von den Subklassen verwendet werden)

- den Methoden
  - public void fuettern()
    {
    // Aufruf der Methode sprich()
    // solange der Futtervorat reicht, die Methode friss() aufrufen
    // Anzahl der Tage hochzählen, die der Vorrat reicht
    }
  - public void anzeigen()
    {
    // Ausgabe: Namen des Haustiers, wie viele Tage der Futtervorrat reicht
    }
  - public abstract void friss();
  - public abstract void sprich();

Leiten Sie von der abstrakten Klasse **Haustier** die Klassen **Hund** und **Katze** ab.

Schreiben Sie für beide Kindklassen einen Konstruktor, der den Konstruktor der Elternklasse aufruft.

Implementieren Sie in den Kindklassen die abstrakten Methoden
- sprich(): gibt "Wuff!" bzw "Miau!" aus
- friss(): verringert den Wert der Instanzvariablen futtervorrat um 1.0 (Hund) bzw. um 0.5 (Katze) und gibt den Namen des Haustiers mit dem jeweiligen Stand des Futtervorrats aus

Testen Sie das Programm in einer Klasse **HaustiereFuettern**. Die Ausgabe sieht dann ungefähr wie folgt aus:

Miau!
Minka: 5.0
Minka: 4.5
…
Minka: 1.0
Minka: 0.5
Der Vorrat fuer Minka reicht 10 Tage.

Wuff!
Strolchie: 5.0
…
Strolchie: 1.0
Der Vorrat fuer Strolchie reicht 5 Tage.

76. Die Kindklassen der Programmieraufgabe **Haustier** sollen erweitert werden.

Die Klasse Hund bekommt zusätzlich eine Instanzvariable *kategorie*(Typ String). Es soll 3 Kategorien geben:

- 1 - Kleinhunde
- 2 - Mittelgrosse Hunde
- 3 - Große Hunde

Die Klasse Katze bekommt eine Instanzvariable *haltung* (Typ String). Es wird zwei Typen geben:

- 1 - Wohnung
- 2 - Artgerecht

Erweitern Sie die Konstruktoren der Kindklassen um eine int-Variable, über die die Instanzvariablen *kategorie* bzw. *haltung* entsprechend initialisiert werden.

Überschreiben Sie in den Kindklassen die Methode *anzeigen()* der Elternklasse. Sie soll zusätzlich die Kategorie bzw. den Typ der Haltung ausgeben.

Testen Sie das Programm in einer Klasse **HaustiereFuettern.** Die Ausgabe sieht dann ungefähr wie diese aus:

Miau!
Minka: 3.0
Minka: 2.5
Minka: 2.0
Minka: 1.5
Minka: 1.0
Minka: 0.5
(Haltung: Wohnung) Der Vorrat fuer Minka reicht 6 Tage.

Wuff!
Strolchie: 5.0
Strolchie: 4.0
Strolchie: 3.0
Strolchie: 2.0
Strolchie: 1.0
(Kategorie: Kleinhunde) Der Vorrat fuer Strolchie reicht 5 Tage.

Fach: Anwendungsentwicklung/Programmierung	FIS	Enumerationen 1

77. Erstellen Sie zunächst einen Enumerationstypen **Wochentag** mit den sieben Wochentagen als Inhalt.

    Erstellen Sie dann eine Klasse **Statistik** mit einer Variablen **daten** vom Typ int-Array[7] und den Methoden **addiereEreignis(Wochentag t),** die für den Wochentag das entsprechende Arrayfeld inkrementiert und **gebeEreignis(Wochentag t),** die den für den Tag gespeicherten Arrayinhalt zurückgibt.

    Testen Sie den Enumerationstyp und die Klasse mit dem Programm **StatistikTest,** das in einer Schleife die Ereignisse pro Wochentag einliest und nach Eingabe eines Ende-Kriteriums ausgibt..

    Der Dialog könnte so aussehen:

```
Wochentag (MO, DI, MI, DO, FR, SA, SO), Ende = E): MO
Wochentag (MO, DI, MI, DO, FR, SA, SO), Ende = E): MI
Wochentag (MO, DI, MI, DO, FR, SA, SO), Ende = E): MI
Wochentag (MO, DI, MI, DO, FR, SA, SO), Ende = E): DO
Wochentag (MO, DI, MI, DO, FR, SA, SO), Ende = E): SA
Wochentag (MO, DI, MI, DO, FR, SA, SO), Ende = E): E

MO: 1
DI: 0
MI: 2
DO: 1
FR: 0
SA: 1
SO: 0
```

Fach: Anwendungsentwicklung/Programmierung	FIS	Vertiefung: Arrays/ArrayList 1

78. Erstellen Sie ein Programm **ErrateDieHauptstadt**, das aus der Datei *Land+Hauptstadt.txt* (1-15: Land, 16-30: Hauptstadt) die dort gegebenen 25 Paare „Land“ + „Hauptstadt“ einliest und diese in einem zweidimensionalen Array lUh abspeichert..

    Mittels Zufallsgenerator wird eine Land+Haupstadt-Kombination ausgewählt. Der Anwender wird mit der Ausgabe des Landes aufgefordert, die Hauptstadt einzugeben (maximal drei Antworten möglich).

    Bewerten Sie die Antwort auf Richtigkeit und geben Sie entsprechende Meldungen aus.

    Der Dialog könnte so aussehen:

    Was ist die Haupstadt von Deutschland?
    Bonn
    Falsch!
    München
    Falsch!
    Berlin
    Richtig, der Kandidat hat 100 Punkte.

Fach: Anwendungsentwicklung/Programmierung	FIS	Vertiefung: Array/ArrayList 2

79. Erstellen Sie ein Programm **ErrateDieHauptstadtInEuropa,** das aus der Datei *Land-Hauptstadt.txt* (1-25: Land, 26-30: Hauptstadt) die dort gegebenen Paare „Land“ + „Hauptstadt“ beliebiger Anzahl einliest und diese in generic-Array-Listen laender und haupstaedte abspeichert.

    Wählen Sie in einer Schleife von 10 Durchläufen mittels Zufallsgenerator eine Land+Haupstadt-Kombination aus. Fordern Sie den Anwender mit der Ausgabe des Landes auf, die Hauptstadt einzugeben (maximal drei Antworten möglich). Bewerten Sie die Antwort auf Richtigkeit und geben Sie entsprechende Meldungen aus.

    Der Dialog könnte so aussehen:

    Was ist die Haupstadt von Deutschland?
    Bonn
    Falsch!
    München
    Falsch!
    Berlin
    Richtig, der Kandidat hat 1 Punkt.
    Was ist die Haupstadt von Belgien?
    ….
    Der Kandidat hat 8 von 10 Punkten.

*

Fach: Anwendungsentwicklung/Programmierung	FIS	Threads

80. Erstellen Sie ein Programm **ThreadEinAus**, dass zwei Threads startet.

    Der erste Thread **Ausgeber** soll in einer Endlosschleife ein Zeichen ausgeben, das in der statischen Variablen *Ausgeber.ch* gespeichert ist (beginnen Sie mit '*').

    Der zweite Tread **Einleser** soll in einer Endlosschleife auf die Eingabe eines Zeichen warten und dieses in *Ausgeber.ch* speichern.

    Wird ein '.' eingegeben, soll sich das Programm beenden.

Fach: Anwendungsentwicklung/Programmierung	FIS	Enumerationen 2

81. Erstellen Sie ein Programm **SiebzehnUndVier**, dass eine einfache Variante des Kartenspiels „Siebzehn und Vier“ für einen Spieler implementiert. Den Test der Implementierung können zwei nebeneinander sitzende Schüler erspielen.

    Dazu benötigt man eine Klasse **Kartenstapel**, die ein Array deklariert in dem jeweils vier Karten der Werte As, König, Dame, Bube, Zehn, Neun, Acht, Sieben, Sechs, Fünf, Vier, Drei und Zwei liegen.

    Die Karten sollen als Enumerationstyp **Spielkarte** implementiert werden, wobei ein As den Wert 11, König, Dame und Bube den Wert 10 und die Zahlenkarten den Wert ihres Namens haben.

    Ein Spieler darf beliebig viele Karten aus dem restlichen Stapel ziehen (die gezogene Karte ermittelt ein Zufallsgenerator), die Punkte der gezogenen Karten werden addiert.

    Gewonnen hat der Spieler, dessen Punkte am nähesten an 21 liegen, den Wert 21 aber nicht überschreiten. Ist der Wert der gezogenen Karten über 21, hat der Spieler auf jeden Fall verloren.

    Der Kartenstapel wird mit jeder gezogenen Karte kleiner.

    Der Dialog könnte so aussehen:

```
Ein neues Spiel (j|n)? j

Neue Karte (j|n)? j
SIEBEN: 7, Summe: 7
Neue Karte (j|n)? j
KOENIG: 10, Summe: 17
Neue Karte (j|n)? j
BUBE: 10, Summe: 27
Verloren, Du bist über 21!

Ein neues Spiel (j|n)? j

Neue Karte (j|n)? j
ZWEI: 2, Summe: 2
Neue Karte (j|n)? j
NEUN: 9, Summe: 11
Neue Karte (j|n)? j
DAME: 10, Summe: 21
Neue Karte (j|n)? n
Du hast 21 Punkte!

Ein neues Spiel (j|n)? n
Servus bis zum naechsten Mal!
```

Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: Buttons 1

82. Erstellen Sie ein Programm **FIS11Button**, dass einen Master-Frame (der das Programm beenden kann) und zwei Slave-Frames (die nur sich selbst beenden können) erzeugt, und die Schüler der ehemaligen Klasse als Buttons auf die Slave-Frames verteilt.

    Der Ergebnis könnte so aussehen:

Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: Buttons 2

83. Erstellen Sie ein Programm **GrossKleinButton**, dass zwei Buttons in einen Frame stellt.

    Ein Button trägt die Aufschrift „Vergrößern“ und soll den Frame um 10% vergrößern, ein Button trägt die Aufschrift „Verkleinern“ und soll den Frame um 10% verkleinern.

    Der Ergebnis könnte so aussehen:

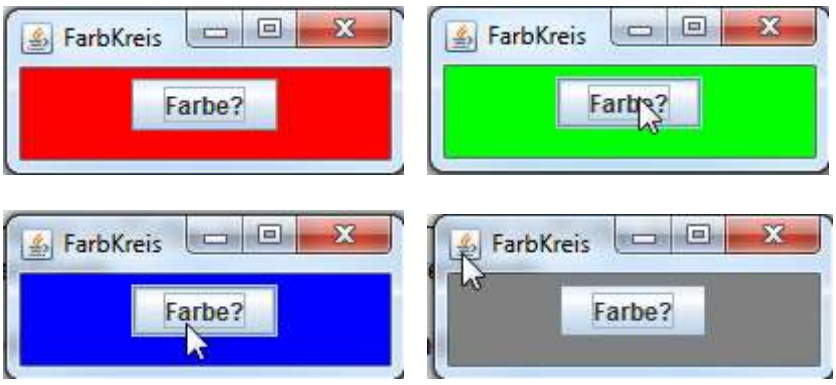
84. Erstellen Sie ein Programm **FarbKreis**, dass einen Button mit der Aufschrift „Farbe“ in einen Frame stellt.

    Das Anklicken des Buttons soll zum Wechsel der Hintergrundfarbe führen. Die Farben sollen in der Reihenfolge rot, grün, blau, grau wechseln.

    Zum Abfragen der aktuellen Hintergrundfarbe verwenden Sie die Klasse *Color* und Methode *getBackgroundColor().*

    Der Aufruf *Color farbe = getContentPane().getBackground();* versorgt das Objekt *farbe* mit der aktuellen Hintergrundfarbe. Diese kann mit *farbe.equals(Color.red)* abgefragt werden.

    Der Ergebnis könnte so aussehen:



und wieder von vorn

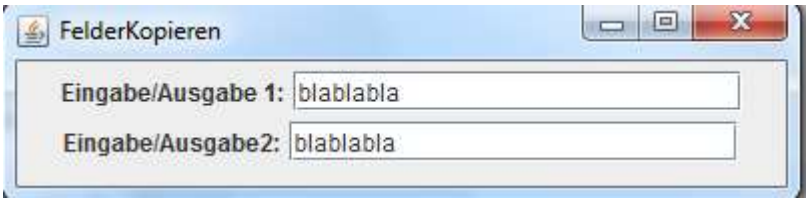
Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: Textfelder 1

85. Erstellen Sie eine Klasse **FelderKopieren**, die zwei GUI-Elemente der Klasse *JTextField* in einen Frame stellt und nach Betätigen der *Return*-Taste den Inhalt des einen Feldes in das andere kopiert.

    Nutzen Sie aus, das bei Instanzen der Klasse *JTextField* das *ActionCommand* den Wert des Textfeldes beinhaltet.

    Die Klasse *FelderKopieren* soll gleich die *main*-Methode beinhalten, die ein Objekt der Klasse konstruiert.

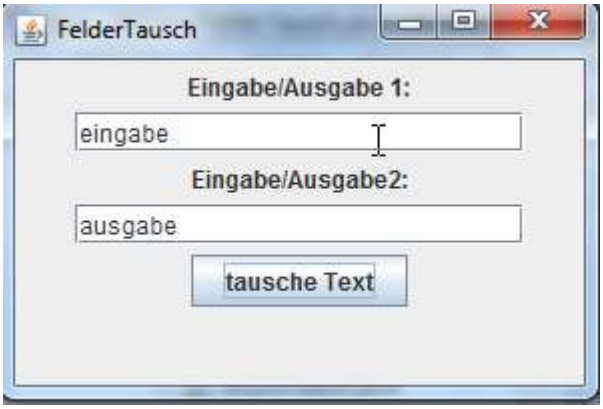
    Der Ergebnis könnte so aussehen:



86. Erstellen Sie eine Klasse **FelderTausch**, die neben den zwei GUI-Elemente der Klasse *JTextField* ein GUI-Element der Klasse JButton in einen Frame stellt und nach Betätigen des Buttons den Inhalt der Felder austauscht.

    Die Klasse *FelderTausch* soll gleich die *main*-Methode beinhalten, die ein Objekt der Klasse konstruiert.

    Der Ergebnis könnte so aussehen:

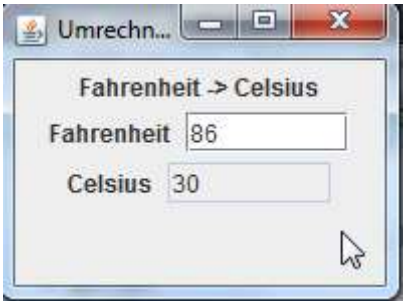


Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: Textfelder 2

87. Erstellen Sie eine Klasse **FtoC**, die eine Temperaturangabe in Fahrenheit einliest, mit der Formel C=(F-32)*5/9 nach Celsius umrechnet und den Celsius-Wert ausgibt.

    Die Klasse *FtoC* soll gleich die *main*-Methode beinhalten, die ein Objekt der Klasse konstruiert.

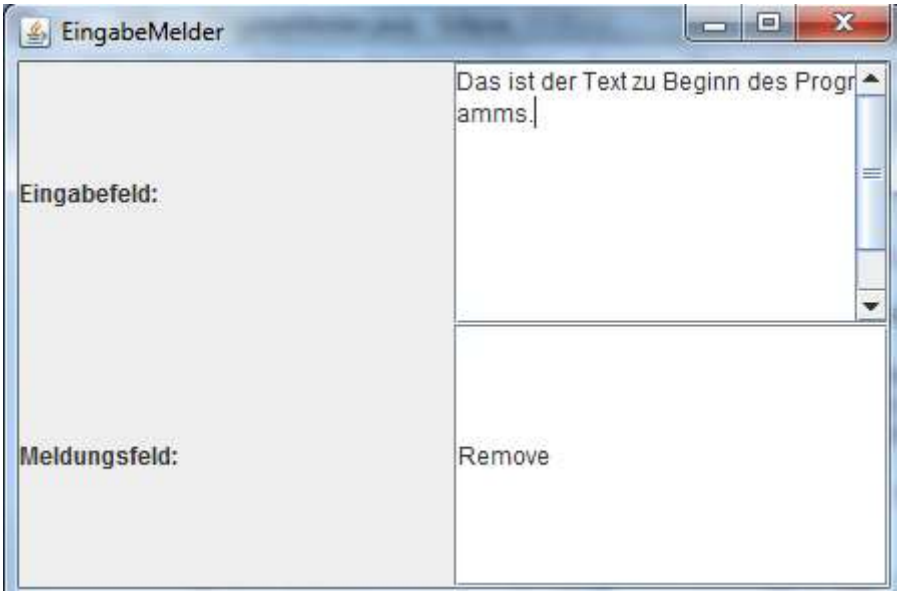
    Der Ergebnis könnte so aussehen:



88. Erstellen Sie eine Klasse **EingabeMelder**, die ein GUI-Element der Klasse *JTextArea* und ein GUI-Element der Klasse *JTextField* in einen Frame stellt und die Art der Veränderung im *JTextArea*-Objekt im *JTextField-Objekt* anzeigt.

    Die Klasse *EingabeMelder* soll gleich die *main*-Methode beinhalten, die ein Objekt der Klasse konstruiert.

    Der Ergebnis könnte so aussehen:



Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: Checkboxes

89. Erstellen Sie eine Klasse **Wochenerfassung**, die Eingabewerte (2 Zeichen) für die Tage der Woche erfasst.

    Die Werte sollen Button-gesteuert entweder für den Meßpunkt am Morgen, am Mittag oder am Abend erfasst werden.

    Die erfassten Werte sollen auf die Konsole ausgeben werden.

    Der GUI-Bildschirm soll so aussehen:

    Die Größe des Bildschirms soll nicht veränderbar sein.

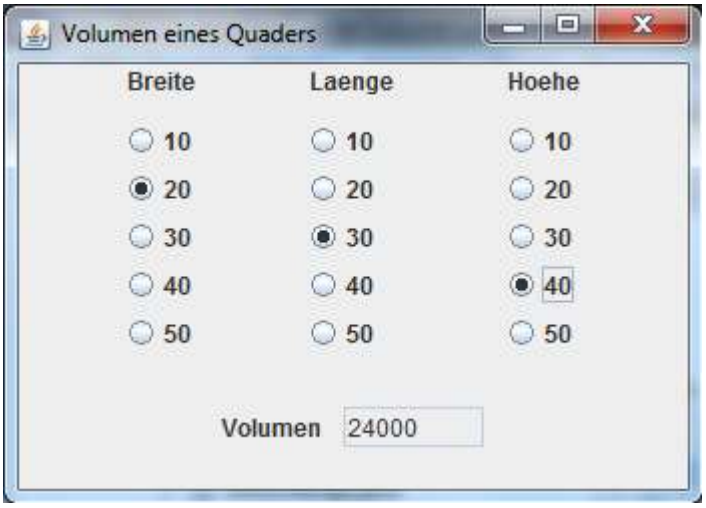
    Die Ausgabe soll so aussehen:

```
Abends - Montag: 1, Dienstag: 2, Mittwoch: 3, Donnerstag: 4, Freitag: 5, Samstag: 6, Sonntag: 7
Mittags - Montag: 2, Dienstag: 3, Mittwoch: 4, Donnerstag: 5, Freitag: 6, Samstag: 7, Sonntag: 8
Morgens - Montag: 3, Dienstag: 4, Mittwoch: 5, Donnerstag: 6, Freitag: 7, Samstag: 8, Sonntag: 9
```

Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: RadioButtons

90. Erstellen Sie eine Klasse **QuaderVolumen**, die die Maße des Quaders via Radio-Buttons erfasst und das daraus errechnete Volumen in einem nicht editierbarem Textfeld ausgibt.

Der GUI-Bildschirm soll so aussehen:



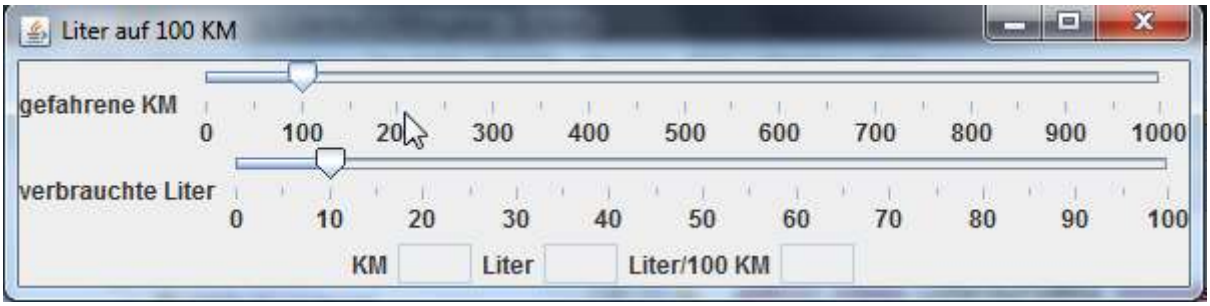
Zwischen dem Titel der Buttonleiste und der Buttonleiste sowie zwischen den Buttonleisten und dem Volumenfeld soll jeweils ein fix definierter Abstand vorhanden sein.

Zwischen dem Rand und den Buttonleisten sowie zwischen den Buttonleisten soll „Kleber“ enthalten sein, der die Abstände beim Vergrößern/Verkleinern sich synchron mit verändern lässt.

Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: Slider

91. Erstellen Sie eine Klasse **LiterAuf100Km,** die den Verbrauch eines Autos auf 100KM Fahrstrecke errechnet.

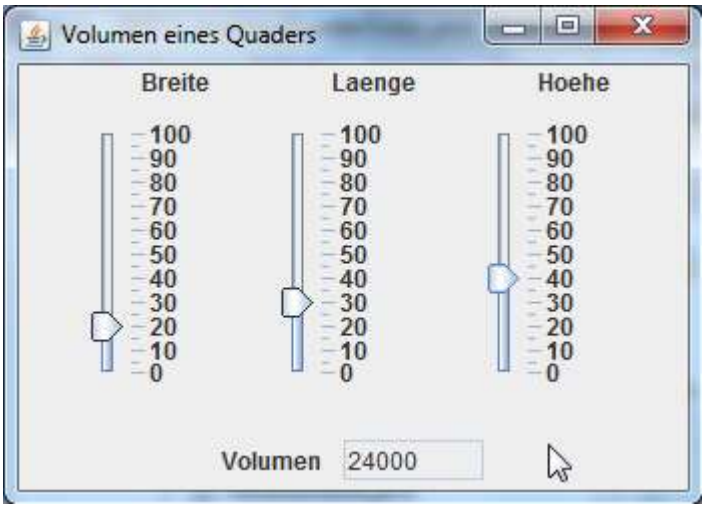
    Der GUI-Bildschirm soll so aussehen:



    Die gefahrenen KM und die verbrauchten Liter werden via Slider erfasst.
    Die erfassten Werte und der resultierende Verbrauch Liter/100 KM werden bei jeder Veränderung ausgegeben.

92. Überarbeiten Sie die Klasse **QuaderVolumen** in eine Klasse **QuaderSlider**, die die Maße des Quaders via Objekten der Klasse *JSlider* erfasst und das daraus errechnete Volumen in einem nicht editierbarem Textfeld ausgibt.

    Der GUI-Bildschirm soll so aussehen:



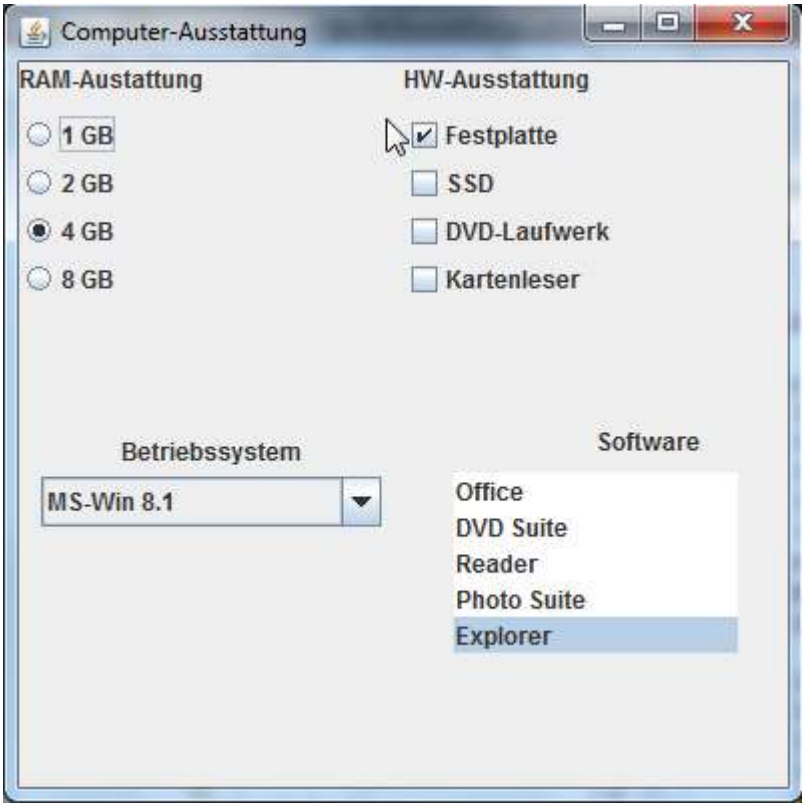
    Zwischen den Titeln der Slider und den Slidern sowie zwischen den Slidern und dem Volumenfeld soll jeweils ein fix definierter Abstand vorhanden sein.

    Zwischen dem Rand und den Slidern sowie zwischen den Slidern soll „Kleber“ enthalten sein, der die Abstände beim Vergrößern/Verkleinern sich synchron mit verändern lässt.

Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: diverse Elemente 1

93. Erstellen Sie die Klasse **ComputerAusstattung1**, die die Ausstattung eines Computers mittels GUI-Elementen abfragt und auf die Konsole ausgibt.

Der GUI-Bildschirm soll zum Start so aussehen:



RAM-Ausstattung: nur ein Wert selektierbar (Radio Button), ein Wert vorbesetzt
HW-Ausstattung: mehrere Werte selektierbar (Checkbox), ein Wert selektiert
Betriebssystem: nur ein Wert selektierbar (Combobox), ein Wert selektiert
Software: alle Werte selektierbar (List), ein Wert selektiert

Die Ausgabe auf die Konsole erfolgt nach jeder Änderung und sollte so aussehen:

```
Computer-Ausstattung
RAM-Speicher: 4 GB,
HW-Ausstattung: Festplatte, DVD-Laufwerk,
Betriebssystem: SUN-Solaris OS5.11,
SW: Office, Reader, Explorer,
```

Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: diverse Elemente 2

94. Erweitern Sie die Klasse der letzten Stunde zur **ComputerAusstattung2**, die die Ausstattung eines Computers mittels GUI-Elementen abfragt und auf die Konsole ausgibt.

    Ergänzen Sie eine Menüleiste mit dem Menü Vorbesetzung und den Menüpunkten Standard-Auswahl (8GB, Festplatte+DVD-Laufwerk, Liniu Ultimate Edition 3.5, Ofiice+Reader+Explorer) und Initialzustand (Zurücksetzen auf die Start-Einstellung).

    Der GUI-Bildschirm soll zum Start so aussehen:

    RAM-Ausstattung: nur ein Wert selektierbar (Radio Button), ein Wert vorbesetzt
    HW-Ausstattung: mehrere Werte selektierbar (Checkbox), ein Wert selektiert
    Betriebssystem: nur ein Wert selektierbar (Combobox), ein Wert selektiert
    Software: alle Werte selektierbar (List), ein Wert selektiert

    Die Ausgabe auf die Konsole ändert sich nicht.

Fach: Anwendungsentwicklung/Programmierung	FIS	GUI: Projektarbeit

95. Zum Beginn eines neuen Schuljahrs sollen alle neuen Schüler erfasst werden.

Erstellen Sie eine Klasse **NeueSchueler**, die die neuen Schüler mit allen zugehörigen Daten in eine Datei NeueSchueler.txt erfasst.

Der Datensatz eines Schülers besteht aus folgenden Informationen:
Name (20 Zeichen)
Vorname (20 Zeichen)
Geschlecht (1 Zeichen): w / m
Geburtsdatum (8 Zeichen): mmddjjjj
Wohnort (20 Zeichen)
PLZ (5 Zeichen, numerisch)
Bundesland (20 Zeichen)
Sprachen (3 Zeichen): efs (für Englisch, Französisch, Spanisch)
Fähigkeiten (5 Zeichen): pmnsc (Sport, Mathematik, Naturw., Sprachen, Computer)
Kommentar (3x50Zeichen)

Das Erfassen der Schüler kann nicht in einem Durchgang erfolgen, d.h. der jeweilige Erfassungsstand muss zum Beginn des Programms aus der Datei geladen und zum Ende des Programms in die Datei geschrieben werden.

Der Ablauf soll menügesteuert erfolgen.

Folgende Funktionen sollen bereitgestellt werden:
Laden der bereits erfassten Schüler (aus einer Datei)
Anzeigen aller erfassten Schüler in einer Selektionstabelle
Anlegen eines neuen Schüler (mit Erfassen der Daten)
Ändern der Daten eines Schülers
Löschen eines bestehenden Schülers
Sichern der bearbeiteten Schüler (in eine Datei)
Beenden der Erfassung

Überlegen Sie, welche Objekte benötigt werden und wie diese in Beziehung zueinander stehen. Was steht im Mittelpunkt aller Objekte?
Versuchen Sie, die Aufgabenstellung systematisch in einzelne Teilaufgaben zu zerlegen.
Überlegen Sie, welche im Schuljahr erlernten Funktionalitäten benötigt werden.

Fach: Anwendungsentwicklung/Programmierung	FIS	Enumerationen/Rekursionen

96. Erstellen Sie ein Programm **Operator**, dass eine static-Methode ***int*** berechne(int x, Operation.?, int y) mit Operation.? als enum-Typ aus ***{PLUS, MINUS, STAR, SLASH}*** bereitstellt, die entsprechend des enum-typs das Ergebnis der arithmetischen Operation auf x und y zurückgibt.

    Testen Sie das Programm in der main-Methode des Programms.

    Der Dialog könnte so aussehen:

```
x: 2+5=7
y: 10-4=6
x*y: 42
x/y: 1
```

97. Erstellen Sie ein Programm **RekursivZahl**, dass eine rekursive Methode print(n) enthält, die alle Ziffern einer positiven Zahl n nacheinander ausgibt.

    Die Methode soll ausser n keine zusätzliche Variable verwenden. print(123) soll nacheinander 1, 2 und 3 ausgeben.

    Der Dialog könnte so aussehen:

```
123:123
4:4
765656:765656
```