

Sequenzen und Verzweigungen

Scanner, if-else und switch case



Stand: **06.11.2019**

Autor: **M. Völkl**

1. Eingaben über die Tastatur

Zur Eingabe von Daten über die Tastatur steht die **Klasse Scanner** zur Verfügung. Dafür muss das **Klassenpaket java.util.Scanner**, in dem sie hinterlegt ist, eingebunden werden.

Die verschiedenen Datentypen müssen nun mit dem entsprechenden Codefragment eingelesen werden:

Datentyp	Codefragment (Methode)	Beschreibung
String	next()	Einlesen eines String-Wertes (bis zum ersten Leerzeichen).
String	nextLine()	Einlesen eines String-Wertes (bis zum Zeilenende).
byte	nextByte()	Einlesen eines byte-Wertes
short	nextShort()	Einlesen eines short-Wertes
int	nextInt()	Einlesen eines int-Wertes
long	nextLong()	Einlesen eines long-Wertes
float	nextFloat()	Einlesen eines float-Wertes. Der Eingabewert muss durch ein Dezimal komma (z. B. 0,3) getrennt sein, der Rückgabewert enthält einen Dezimal punkt (z. B. 0.3). Ein f für float muss nicht angegeben werden.
double	nextDouble()	Einlesen eines double-Wertes. Der Eingabewert muss durch ein Dezimal komma getrennt sein, der Rückgabewert enthält einen Dezimal punkt .
boolean	nextBoolean()	Einlesen eines boolean-Wertes (true/false).

fsi_sequenzen_Vk_20191106.docx

Beispielcode

```
import java.util.Scanner;
class Addition {
    public static void main (String[] args) {
        Scanner berta = new Scanner(System.in); // wird einmal pro Klasse benötigt
        System.out.print("Geben Sie eine ganze Zahl ein: ");
        int zahl1 = berta.nextInt();
        System.out.print("Geben Sie eine zweite ganze Zahl ein: ");
        int zahl2 = berta.nextInt();

        System.out.println("Die erste eingegebene Zahl war: "+ zahl1);
        System.out.println("Die zweite eingegebene Zahl war: "+ zahl2);
    }
}
```



Erstellen Sie ein Programm, das nach Eingabe von zwei Zahlen sämtliche Grundrechenarten durchführt.



Programmierauftrag 1: Temperatur-Umrechner

Plant man eine Reise nach Amerika, können die Temperaturangaben sehr trügerisch sein, 60° muss nicht so heiß sein, wie es sich auf den ersten Blick liest.

Entwerfen Sie ein Programm, das für den Anwender eine beliebige Temperaturangabe von Grad Celsius nach Grad Fahrenheit umrechnet!

$$^{\circ}\text{F} = ^{\circ}\text{C} * 1,8 + 32$$

Eine weitere Temperaturskala wurde von William Thomson, dem späteren Lord Kelvin, geprägt. Die Umrechnung von Grad Celsius nach Kelvin berechnet sich wie folgt:

$$\text{K} = ^{\circ}\text{C} + 273,15$$

Ergänzen Sie Ihr Programm so, dass es die eingegebene Temperatur zusätzlich in Kelvin umrechnet!



Daniel Gabriel Fahrenheit
<https://www.dkfindout.com/us/science/fam>

Weiterführende Information

Daniel Gabriel Fahrenheit verwendete als Nullpunkt seiner Skala die tiefste Temperatur, die er mit einer Mischung aus Eis, Wasser und Salmiak (= Ammoniumchlorid) oder Seesalz (Kältemischung) erzeugen konnte: -17,8 °C. Als zweiten und dritten Fixpunkt legte Fahrenheit 1714 den Gefrierpunkt des reinen Wassers bei 32 °F und die Körpertemperatur eines gesunden Menschen bei 96 °F. Allerdings entsprechen 96 °F rund 35,6 °C; dieser Wert liegt, verglichen mit heute üblichen Messmethoden, deutlich unterhalb des menschlichen Normaltemperaturbereichs.



Programmierauftrag 2: Brutto-Netto Rechner

Erstellen Sie ein Programm, das nach Eingabe eines Bruttolohns den Nettolohn ausgibt! Rechnen Sie mit den folgenden Abzügen:



Lohnsteuer	14 %
Pflegeversicherung (vom Gehalt)	1,275 %
Solidaritätszuschlag (von der Lohnsteuer)	5,5 %
Kirchensteuer (von der Lohnsteuer)	9 %
Rentenversicherung (vom Gehalt)	9,35 %
Arbeitslosenversicherung (vom Gehalt)	3,0 %
Krankenversicherung (vom Gehalt)	7,3 %



Programmierauftrag 3: Bildgrößen Rechner

Ein zum Schuljahresanfang abgegebenes Passfoto wird als bmp-Datei eingescannt. Für dieses Bild soll die Dateigröße nach folgender Formel ermittelt werden:

$$\text{Dateigröße} = \text{Anzahl der Bildpunkte} * \text{Farbtiefe} + \text{Dateiheader}$$

Anzahl der Bildpunkte: 1024 x 768

Farbtiefe: 24 Bit (Truecolor)

Dateiheader (bei Truecolor): 432 Bit

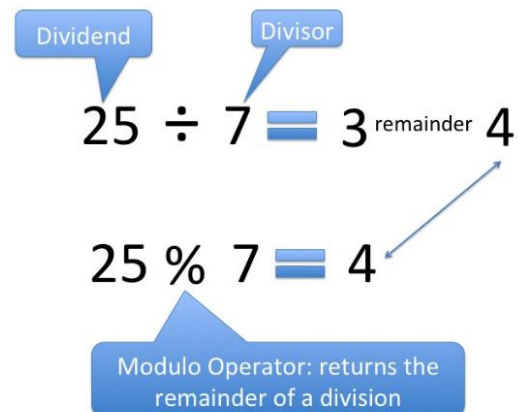
Erstellen Sie ein Programm, das nach Eingabe der Länge und Breite (Auflösung) die Dateigröße für dieses Bild berechnet und als Bit-, Byte- sowie kiB-Wert ausgibt!

Umrechnungen	
1 Byte	8 Bit
1 kiB (Kibibyte)	1024 Bytes
1 kB (Kilobyte)	1000 Bytes
Die umgangssprachlichen Bezeichnungen und Umrechnungen sind jedoch nach wie vor verbreitet.	

2. Modulo Operator

Definition

Modulo berechnet den Rest (*Englisch: remainder*) einer Division. Man kann eine Funktion definieren, welche jedem Zahlenpaar den Teilerrest zuordnet. Diese nennt man „Modulo“. In vielen Programmiersprachen – wie auch in JAVA – wird sie durch % dargestellt und als Operator behandelt



Beispiele

Zu welchen Ausgaben führen die folgenden Programmzeilen?

```
System.out.println( 22 % 7 );
System.out.println( 8 % 4 );
System.out.println( 11 % 4 );
System.out.println( 13 % 3 );
System.out.println( 14 % 3 );
```



Programmauftrag 4: Flaschen-Berechnung

1435 Flaschen Bier sollen in Kästen eingefüllt werden. In einen Kasten passen jeweils 12 Flaschen. Ein Kasten muss immer vollständig mit 12 Flaschen gefüllt werden. Wie viele Flaschen bleiben übrig?

Erweitern Sie das Programm so, dass der Dividend per Tastatur eingegeben wird.

Anschließend soll auch der Divisor über die Tastatur eingegeben werden.



Programmauftrag 5: Prüfziffer

Bei Kreditkarten wird die Prüfziffer aus Teilen der Kreditkartennummer berechnet!

Entwickeln Sie ein Programm, das eine verkürzte und vereinfachte Version davon darstellt!

Entwickeln Sie ein Programm mithilfe des folgenden Algorithmus:

int ziffer1 bis ziffer4
ziffer1 bis ziffer4 ist Eingabe Benutzer
summe := ziffer1 bis ziffer4
rest := summe%3
Ausgabe rest

Nur für Profis: Informieren Sie sich selbstständig über die tatsächliche Prüfzifferberechnung einer von Ihnen ausgewählten Kreditkarte (Visa, Mastercard, etc.) und erstellen Sie ein entsprechendes Programm!



3. Programmablaufplan und Struktogramm

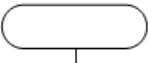
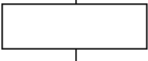
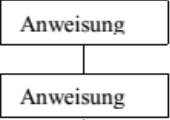
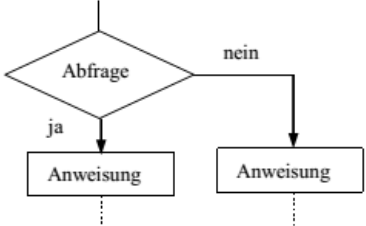
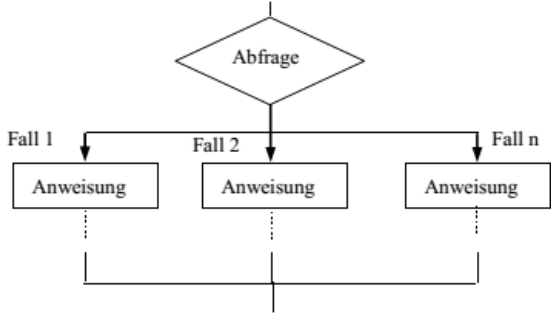
Programmablaufplan (PAP)

Ein Programmablaufplan ist ein Ablaufdiagramm für ein Programm und dient zur graphischen Darstellung zur Umsetzung eines Algorithmus in einem Programm. Es beschreibt die Folge von Operationen zur Lösung einer Aufgabe. Die Symbole für Programmablaufpläne sind in der **DIN 66001** genormt. Programmablaufpläne werden auch zur Darstellung von Prozessen und Tätigkeiten eingesetzt (z. B. als Beschreibung des Arbeitsablaufs bei der Angebotserstellung in einem Handelsunternehmen).

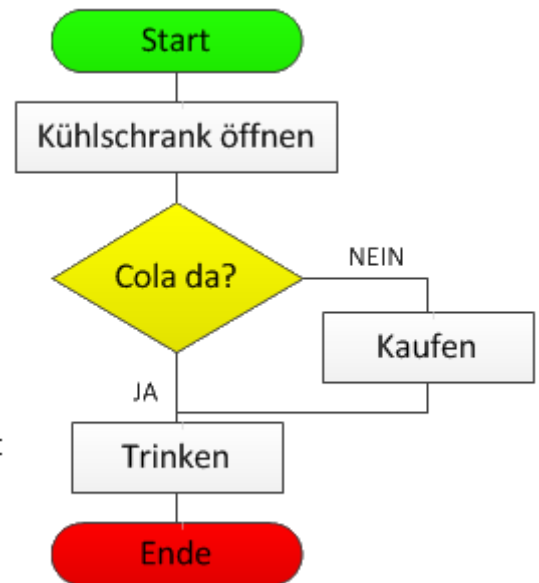
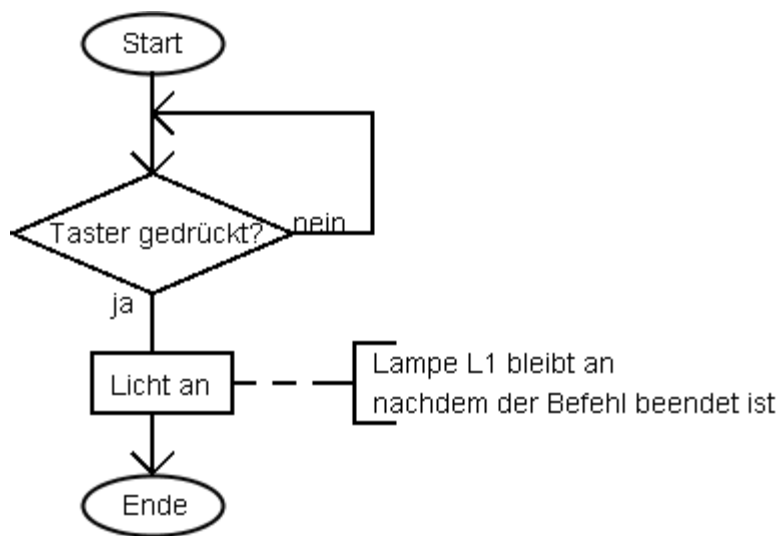
Die Beschriftung in den Symbolen muss allgemeingültig und programmiersprachenunabhängig sein!

Elemente im PAP

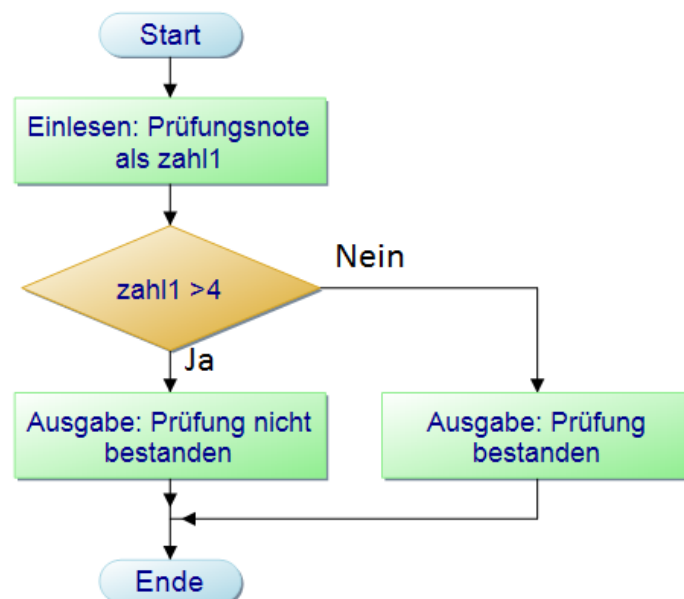
Folgende Tabelle zeigt einen Ausschnitt der nach DIN66001 genormten Symbole.

Symbole	
Grenzstelle (z.B. Anfang, Ende)	
Anweisung (allg.), Verarbeitung	
Sequenz (Folgestruktur)	
Bedingte Verzweigung (einseitige / zweiseitige Auswahlstruktur)	
Fallauswahl (Mehrfachverzweigung)	

Beispiele



Prüfung bestanden



Struktogramme (DIN 66261)

Ein Nassi-Shneiderman-Diagramm (Struktogramm) ist ein Diagrammtyp zur Darstellung von Entwürfen im Rahmen der Methode der strukturierten Programmierung. Er wurde 1972/73 entwickelt und ist in der DIN 66261 genormt.

Da Nassi-Shneiderman-Diagramme Programmstrukturen darstellen, werden sie auch als Struktogramme bezeichnet.

Ein Struktogramm ist die grafische Darstellung eines Programmablaufs in Form eines geschlossenen Blocks. Die Grundform ist ein Rechteck ohne „Dellen“ und „Auswucherungen“. Die verschiedenen Symbole werden ohne Zwischenräume aneinandergereiht bzw. verschachtelt.

Elemente im Struktogramm

Linearer Ablauf (Sequenz)



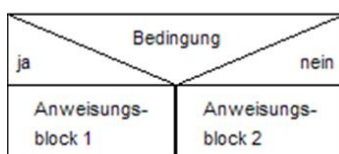
Jede Anweisung wird in einen rechteckigen Strukturblock geschrieben. Die Strukturblocke werden nacheinander von oben nach unten durchlaufen. Leere Strukturblocke sind nur in Verzweigungen zulässig.

Einfache Auswahl (Einfache Verzweigung)



Nur wenn die Bedingung zutreffend (wahr) ist, wird der Anweisungsblock 1 durchlaufen. Ein Anweisungsblock kann aus einer oder mehreren Anweisungen bestehen.

Zweifache Auswahl



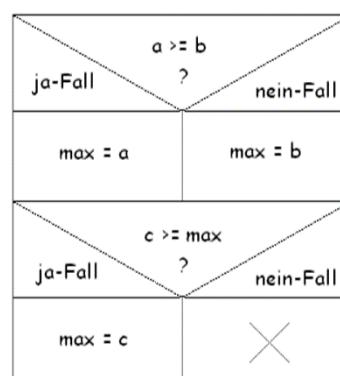
Wenn die Bedingung zutreffend (ja) ist, wird der Anweisungsblock 1 durchlaufen. Trifft die Bedingung nicht zu (nein), wird der Anweisungsblock 2 durchlaufen. Ein Anweisungsblock kann aus einer oder mehreren Anweisungen bestehen.

Beispiele

Durchschnittsverbrauch



Maximum bestimmen



Arbeitsaufträge



Bart packt aus?

Erstellen Sie zu dem folgenden Ablauf einen **Programmablaufplan** nach DIN 66001!

Bart Simpson überlegt sich, ob er fleißig in der Schule bleiben soll oder ob er lieber „blau machen“ soll. Bleibt er in der Schule, hat er einen langen, langweiligen Tag vor sich. Bleibt Bart nicht in der Schule, bekommt er auf jeden Fall später Ärger mit seinen Eltern, weil sie es rausfinden werden.



Entscheidet er sich, die Schule zu schwänzen, wird er während seiner neu gewonnenen „Freizeit“ Zeuge eines Verbrechens. Dieses Verbrechen kommt dann später vor Gericht und Bart muss sich entscheiden, ob er vor Gericht als Entlastungs-Zeuge auftreten soll, um den unschuldigen Angeklagten zu entlasten, der ohne Barts Aussage verurteilt wird.

Falls er aussagt, beweist er damit zugleich, dass er die Schule geschwänzt hat und muss somit mit einer Strafe vom Schulrektor rechnen. In diesem Fall muss Rektor Skinner entscheiden, ob er honoriert, dass Bart trotz drohender Strafe vor Gericht aussagt und ihn laufen lässt oder ob er ihn nachsitzen lässt.



LAN Party

Erstellen Sie zu dem folgenden Ablauf ein **Nassi-Shneiderman-Diagramm**.

Sie möchten mit Ihren Freunden am Wochenende eine kleine LAN-Party bei sich zu Hause veranstalten. In Ihrem Zimmer haben insgesamt vier Leute Platz. Im Wohnzimmer der Eltern hingegen könnten insgesamt sechs Leute einen „Spieleabend“ veranstalten. Sie versuchen also Ihre Eltern zu überreden ein „Wellness-Wochenende“ in einem Hotel zu verbringen.



Sollten Ihre Eltern sich entscheiden das Wochenende nicht zu Hause zu verbringen, müssen Sie einen größeren Switch organisieren, Ihren Rechner im Wohnzimmer aufbauen und Verpflegung organisieren. In diesem Fall laden Sie Ihren Freund Sheldon zu der LAN-Party ein, falls er kommt wollen Sie Halo spielen. Ansonsten spielen Sie Call of Duty Ghosts.

Falls Sheldon nicht kommt, laden Sie ihren guten Freund Leonard und seine Freundin Penny ein.

Falls Sheldon kommt, müssen Sie noch thailändisches Essen organisieren, da er sonst nichts essen mag.

Falls Ihre Eltern zuhause bleiben, spielen Sie nur StarCraft 2 mit ihren Freunden. Falls Sie Ihren Eltern Bescheid geben, kochen diese für Sie und es muss nichts weiter organisiert werden. Sonst müssen Sie noch Cola und Chips einkaufen.

Egal was an diesem Abend passiert, Sie müssen am nächsten Tag das ganze Haus aufräumen, da es immer etwas unordentlich wird.



Für Experten

Wandeln Sie das PAP von **Bart packt aus?** in ein Struktogramm um und wandeln Sie das Struktogramm von der **LAN Party** in ein PAP um!

4. Verzweigungen

Definition

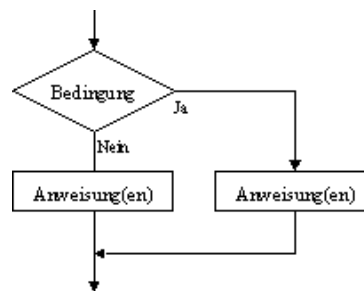
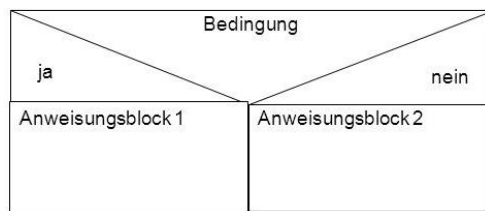
Häufig ist ein linearer Ablauf des Programms nicht möglich. So kann die Auswahl von Kriterien für den weiteren Programmablauf bestimmend sein, um ein richtiges Ergebnis zu liefern.

Eine **Verzweigung** legt fest, welche Programmabschnitte, abhängig von einer **Bedingung**, ausgeführt werden.

Bedingte Anweisungen und Verzweigungen gehören zu den sogenannten **Kontrollstrukturen**.

if-else Anweisung

Ein Teil des Programmcodes wird nur dann bearbeitet, wenn eine bestimmte Bedingung erfüllt ist. In dieser **Bedingung** werden die Vergleichsoperatoren (z.B. ==, >, <, <=, >=, !=) eingesetzt. Die else Bedingung wird ausgeführt, falls die if Bedingungen nicht erfüllt ist.



Beispielcode

```
if (x>1000) {
    System.out.println("Der x Wert ist größer als 1000");
}
else {
    System.out.println("Der x Wert ist kleiner gleich 1000");
}
```

Mehrere Verknüpfungen

Es kann notwendig sein mehrere Bedingungen zu verknüpfen.

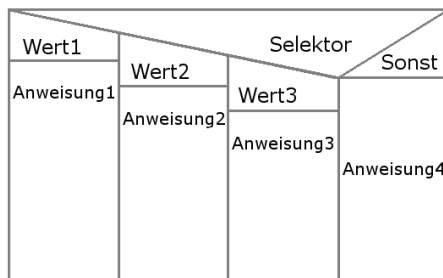
```
if (alter>=13 && alter < 20) { // logische UND-Verknüpfung
    bezeichnung = "Teenager";
}
```

```
if (temperatur > 50 || temperatur < -10){ // logische ODER-Verknüpfung
    alarm = true;
}
```

Switch - Case

In vielen Programmiersprachen gibt es **mehrfache Verzweigungen**, auch Fallunterscheidungen genannt, die abhängig von Bedingungen sind. Ein Teil des Programmcodes wird nur dann bearbeitet, wenn eine bestimmte Bedingung erfüllt ist. Gibt es keine besondere Alternative, kann der Sonst-Block (**default**) entfallen.

Die Anweisungen in einer Verzweigung werden mit einem **break**; abgeschlossen. Beim default Fall ist kein break notwendig.



Beispielcode

```
switch (zahl) {
  case 0:      // Die Zahl hat den Wert 0
    System.out.println("Die Zahl ist 0");
    break;

  case 1:      // Die Zahl hat den Wert 1
    System.out.println("Die Zahl ist 1");
    break;

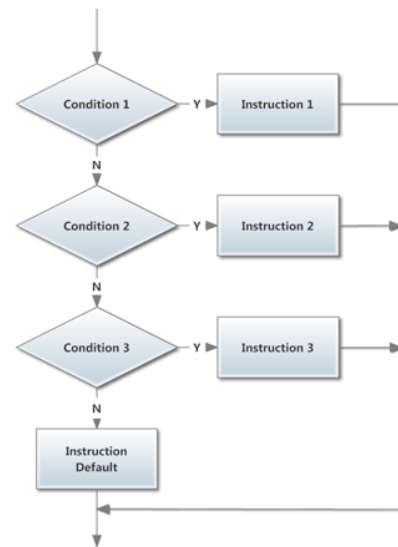
  default:     // Die Zahl hat Werte, die bisher nicht in
               // den case-Bedingungen vorgekommen sind.
    System.out.println("Die Zahl ist irgendwas anderes");
}
```

Sonderfall: Switch ohne break

```
switch (zahl)
{
    case 1: System.out.println("eins");
    case 2: System.out.println("zwei");
    case 3: System.out.println("drei");
}
```

➔ Falls zahl 1 ist, wird **eins**, **zwei** und **drei** ausgegeben.

➔ Falls zahl 2 ist, wird **zwei** und **drei** ausgegeben.



Kontrollfragen

a) Finden Sie die passenden **Vergleichsoperatoren** (>, <, !=, ==, >=, <=).

größer als

größer gleich als

kleiner als

kleiner gleich als

gleich

ungleich

b) Durch welche Zeichen wird die **Bedingungen** im Programmcode eingeschlossen?

c) Welchem Element entspricht die **Verzweigung** im Programmablaufplan nach DIN66001?

**Programmierauftrag 1: if-else**

Erstellen Sie ein Programm, das abhängig von der PUK-Eingabe des Benutzers eine Meldung ausgibt.

Falls der PUK falsch eingegeben wird, erscheint eine Fehlermeldung. **Ansonsten** wird die Meldung "Auf Werkseinstellungen zurück gesetzt" angezeigt.

Der richtige PUK hat den Wert **12345678**.

**Programmierauftrag 2: switch-case**

Erstellen Sie ein Programm, das Ihnen bei Eingabe einer Zahl (1-6) die Schulnoten (Sehr gut, gut, ...) ausgibt. Bei einer ungültigen Note soll eine Fehlermeldung erscheinen.

**Schon fertig?**

Beginnen Sie mit den Programmieraufgaben im Kapitel 6!

Theorieaufgabe switch-case

Von einem Programm liegt folgender Programmausschnitt vor.

```
...
int zahl1 = 2;
int zahl2 = 4;
int erg = 0;
switch (auswahl) // In auswahl wird die Variable gespeichert
{
    case 'a': erg = zahl1 + zahl2 + erg;
    case 'b': erg = zahl1 + zahl2 + erg;      break;
    case 'c': erg = zahl1 + zahl2 + erg;
}
System.out.println(erg);
...
```



Versuchen Sie die Aufgabe zuerst auf dem Blatt Papier zu lösen. Geben Sie die Werte an, die ausgegeben werden, wenn beim Programmstart für die Variabel **auswahl**

- a) a eingegeben wird _____
- b) b eingegeben wird _____
- c) c eingegeben wird _____

**Vergleichen von Strings**

Das Vergleichen von Strings kann nicht mit den Vergleichoperatoren für die elementaren Datentypen durchgeführt werden.

Überprüfen Sie ob in der Variablen **name** der Inhalt **Huber** steht! Geben Sie für denn Fall "Hallo Herr Huber!" aus!

5. Inkrement und Dekrement

Bei der schrittweisen Erhöhung oder Verminderung einer Größe oder Variablen ist das Inkrement (von lateinisch incrementare ‚vergrößern‘) bzw. Dekrement (von lat. decrementare ‚vermindern‘) der festgelegte Betrag der Änderung.

Gegeben ist folgender Programmcode:

```
class Inkrement {  
    public static void main (String[]args) {  
        int var = 0;  
        System.out.println(var++ + " " + ++var + " " + var++ + " " + ++var);  
        var = 10;  
        System.out.println(var-- + " " + --var + " " + var-- + " " + -var);  
    }  
}
```



Wie lautet die Bildschirmausgabe? Notieren Sie die Ausgabe!



Begründen Sie das zustande kommen der Ausgabe, indem Sie den Inkrement- und Dekrement-Operator **beschreiben**!



Bestimmen Sie die Werte, die sich ergeben!

```
int a = 5;  
int x = 10 + a++;
```

x = ____

a = ____

```
int a = 2;  
int x = 8 + a--;
```

x = ____

a = ____

```
int b = 5;  
int y = 10- ++b;
```

y = ____

b = ____

```
int b = 4;  
int y = 9/ --b;
```

y = ____

b = ____

6. Programmieraufgaben

Aufgabe 1 "Volljaehrigkeit" - Voll.java

Der Benutzer gibt auf die Aufforderung hin "Bitte Alter eingeben" sein Alter per Tastatur ein. Daraufhin erscheint am Bildschirm abhängig von der Eingabe ein Hinweis: "Du bist noch nicht volljaehrig!" oder "Sie sind volljaehrig!"

Aufgabe 2 "Fahrenheit - Celsius" - Temp.java

In manchen Ländern wird die Temperatur im Unterschied zu unserer Messung (Celsius) in Fahrenheit angegeben. Erstellen Sie ein Programm, das den Benutzer erstens auffordert anzugeben, ob er Fahrenheit in Celsius oder umgekehrt umgerechnet haben will. Zweitens soll er dann den entsprechenden Wert eingeben. Das Programm rechnet den Wert um und gibt das Ergebnis am Bildschirm aus.



Aufgabe 3 "Prämienvergabe" - Bonus.java

Programmieren Sie eine geeignete Lösung für folgende Situation!

- Alle Mitarbeiter über 35 Jahre erhalten eine einmalige Zulage in Höhe von 50 €.
- Alle Betriebsangehörige, welche mindestens 25 Jahre in der Firma arbeiten, erhalten einen einmaligen Bonus in Höhe von 500 €.

Der Benutzer gibt zu Beginn des Programms seinen Namen, sein Alter und die Dauer der Betriebszugehörigkeit an!



Aufgabe 4 "Mengenrabatt" - Rabatt.java

In einem Werbeprospekt des Versandhandelsunternehmens LieferQuick findet sich folgendes Angebot an die Kundschaft für die Abnahme von USB-Sticks:



Wir bieten Rabattprozente!

- Seien Sie schlau! Kaufen Sie mindestens 100 Stück und Sie erhalten 8% Rabatt auf den Einkaufspreis!
- Seien Sie schlauer! Kaufen Sie mindestens 200 Stück und Sie erhalten 15% Rabatt auf den Einkaufspreis!
- Seien Sie am schlauesten! Kaufen Sie mindestens 500 Stück und Sie erhalten sogar 30% Rabatt auf den Einkaufspreis!

Codieren Sie ein Abrechnungsprogramm, bei dem die Bestellmenge abgefragt wird und anschließend der Rechnungsbetrag ausgewiesen wird! Stückpreis eines USB-Sticks: 8,50€
Erstellen Sie zunächst ein PAP und ein Struktogramm bevor Sie mit der Programmierung beginnen!

Aufgabe 5 "Bonuskarte gewähren"

Zur Verbesserung der Kundenbindung will die Dress-Trend GmbH Kundenkarten einführen. Die Marketingabteilung gibt Ihnen dazu folgende Informationen:



Kunden, die im laufenden Geschäftsjahr eine bei der Dress-Trend GmbH einkaufen, können eine Kundenkarte erhalten. Welcher Kartentyp den Kunden aus-
gehändigt wird, hängt allein vom dem im laufenden Geschäftsjahr getätigten Umsatz ab:

- | | |
|--|----------------|
| • Kunden mit Umsatz < 500,00 € | Standard-Karte |
| • Kunden mit Umsatz \geq 500,00 € und < 1.000,00 € | Plus-Karte |
| • Kunden mit Umsatz \geq 1.000,00 € | VIP-Karte |

Von dem Typ der Kundenkarte (Standard-Karte, Plus-Karte, VIP-Karte) ist der jeweils zu gewährende Bonus abhängig. Folgende Bonusstufen liegen vor:

- | | |
|--|------------|
| • Kunden mit einer Standard-Karte erhalten | 2 % Bonus |
| • Kunden mit einer Plus-Karte erhalten | 3 % Bonus |
| • Kunden mit einer VIP-Karte erhalten | 5 % Bonus. |

Die Boni werden in Form von Einkaufsgutscheinen gewährt. Erstellen Sie ein Programm, das nach Eingabe des Umsatzes den entsprechenden Kartentyp und den entsprechenden Bonus ausgibt!

Aufgabe 6 "Modulo Lernhilfe" - Modulo.java

Erstellen Sie ein Programm, das den Benutzer auffordert, den Dividenten und den Divisor einzugeben. Anschließend soll das Programm den Benutzer auffordern den Restwert (Modulo) der Division einzugeben. Das Programm überprüft das berechnete Ergebnis des Benutzers. Die Ausgabe soll wie folgt aussehen:

```
Geben Sie den Divident ein: 12
Geben Sie den Divisor ein: 5
Berechnen Sie den Restwert von : 12 % 5 = 2
Korrekte Antwort
```

Aufgabe 7 "Interactive Fiction" - Adventure.java

Als Interactive Fiction bezeichnet man ein Computerspielgenre, in dem der Spieler einen Charakter kontrolliert, dessen Spielumgebung und Interaktionen mit der Umgebung als Text beschrieben werden und der mittels Texteingabe mit seiner Umgebung interagiert.

Erstellen sie ein solches Text-Adventure in JAVA! Seien Sie kreativ!

Inspiration finden sie unter anderem hier: <http://ifwizz.de/if-textadventures-online-spielen.html>

