

Und-Verknüpfung – (Konjunktion)

Beispiel:

A: Das Leuchtmittel ist ok.

B: Der Lichtschalter ist eingeschaltet.

Q: Die Lampe leuchtet.

Hinweis:

in der positiven Logik bedeutet: 0 -> AUS, Low, falsch, false,

in der positiven Logik bedeutet: 1 -> AN, High, wahr, true (Bei der negativen Logik verhält sich das genau andersherum.)

Wahrheitstabelle für die Konjunktion:

A	B	Q	Ergebnis
0	0	0	FALSCH
0	1	0	FALSCH
1	0	0	FALSCH
1	1	1	WAHR

Funktionsgleichung:

$$Q = A \wedge B$$

Oder-Verknüpfung (Disjunktion)

Beispiel:

A: 130 km/h in einer Baustelle auf der A42

B: Bei rot über die Ampel

Q: Der Führerschein ist weg.

Wahrheitstabelle:

A	B	Q	Ergebnis
0	0	0	FALSCH
0	1	1	WAHR
1	0	1	WAHR
1	1	1	WAHR

Funktionsgleichung

$$Q = A \vee B$$

Antivalenz – exklusives Oder (-A bedeutet nicht A!!!)XOR

Die Aussage ist genau dann wahr, wenn genau EINE Teilaussage wahr ist.

Wertetabelle hierzu:

A	B	Q	Ergebnis
0	0	0	FALSCH
0	1	1	WAHR
1	0	1	WAHR
1	1	1	FALSCH

Funktionsgleichung:

$$Q = (-A \wedge B) \vee (A \wedge -B) \quad \text{- bedeutet: Q ist WAHR, wenn B nicht A ist und A nicht B}$$

Paritybyte (berechnen)

Eine zweimalige Anwendung der XOR - Funktion hebt sich selbst auf.

$$A \text{ XOR } B \text{ XOR } B = A$$

Beispiel:

4 Datenbytes

: Daraus wird das ParityByte berechnet:

$$((D1 \text{ XOR } D2) \text{ XOR } D3) \text{ XOR } D4$$

D1	1	0	1	0	0	1	0	1
D2	1	1	1	1	0	0	0	0
D3	0	0	1	1	1	1	0	0
D4	1	0	1	1	1	0	0	1
ParityByte	1	1	0	1	0	0	0	0

Diese Werte werden auf 5 HDDs gespeichert. Die HDD Nummer 3 stirbt.

Mit dem Parity Byte kann man nun den Wert D3 rekonstruieren:

$$((D1 \text{ XOR } D2) \text{ XOR } D3) \text{ XOR } P$$

D1	1	0	1	0	0	1	0	1
D2	1	1	1	1	0	0	0	0
ParityByte	1	1	0	1	0	0	0	0
D4	1	0	1	1	1	0	0	1
D3	0	0	1	1	1	1	0	0

Vergleichsoperationen (Wichtig!)

Schreibweisen logischer Operationen sind von der verwendeten Programmiersprache abhängig.

Beispiel:

$$A = B \text{ (Vergleich)} \quad \text{-->} A == B$$

$$A = B \text{ (Wertzuweisung)} \quad \text{-->} A = B$$

Unterschied Interpreter / Compiler (Was ist ein Compiler !!!!!)

Ein Compiler ist ein Programm, das ein fertiges Maschinen-Programm erzeugt, das dann komplett ausgeführt wird. Ein Interpreter ist ein Programm, das ein Hochsprachen-Programm während der Laufzeit analysiert (Parse-Baum aufbauen) und gleichzeitig ausführt.

Warum sind Kommentare wichtig?

- sind ohne Einfluss auf die Programme
- aber dienen der Lesbarkeit
- auch komplexer Quellcode bleibt nachvollziehbar

Zugriffsattribut Public / Private

- In einer Klasse können Daten, Abläufe und Anweisungen hinterlegt werden.
- eine mit dem Zugriffsattribut public versehene Klasse kann von allen anderen Klassen eines Programms benutzt werden
- in einer Datei können auch mehrere Klassen definiert werden
- von diesen darf nur eine das Zugriffsattribut public haben
- die Datei muss so heißen, wie der Name der public-Klasse es vorgibt

Main-Methode = Hauptmethode

- Klassen bestehen unter anderem aus Methoden
- in einer Methode sind in der Regel mehrere Anweisungen zusammengefasst,

- so dass eine Methode eine bestimmte Teilfunktionalität realisiert – im
- Beispiel gibt die main-Methode lediglich etwas aus.
- Die main-Methode ist Hauptmethode eines jeden Java-Programms, d.h.jedes
- Java-Programm enthält genau eine Methode mit Namen main.
- Die Ausführung eines Java-Programms beginnt mit der Ausführung der main-
- Methode, d.h. der Maschinencode zur main-Methode ist der Code, der bei
- Programmstart ausgeführt bzw. aufgerufen wird
- Die in einer Methode hinterlegten Anweisungen werden im Methodenrumpf
- zwischen { und } geklammert.

Syntaxfehler

werden vom Compiler angezeigt. Er stoppt. Alle Syntaxfehler müssen beseitigt werden

Variablen

aus Programmiersicht: benannter Hauptspeicherbereich, in den man einen Wert ablegen, später wieder abrufen und manipulieren kann

Datentypen

bestimmt, was in einer Variablen gespeichert werden kann und hierzu die Größe des zur Aufnahme eines Variablenwertes nötigen Speicherbereiches bestimmt zugleich, was mit der Variablen gemacht werden kann

Wozu benötigt man unterschiedliche Datentypen?

Weil der programmtechnisch abzubildende Problembereich aus Objekten und Objekteigenschaften unterschiedlichen Typs besteht (Name vs. Hubraum vs. Durchschnittsverbrauch eines Kfz.) Weil diese problemnah und speicherschonend abgebildet werden sollen

Man sollte einige nennen können

- Ganzzahldatentypen (int)
- Gleitkommatdatentypen (float)
- Zeichendatentypen (char)
- Logikdatentypen (boolean)
- Zeichenketten (String)

Welcher Datentyp welche Größe /Wertebereich bietet (z.B Pi mit int möglich?) Nein , besser float

Konstante

– wie Variablen nur mit „final“ davor – z.B. final float pi

Reservierte Wörter in Java

Schlüsselwörter - sind Wörter mit in der Programmiersprache festgelegter Bedeutung , diese dürfen als Bezeichner nicht verwendet werden .

Unterschied Operand / Operatoren

Operanden: x, y, t, c, gruss, vorname, "Hallo", "Knuth"

Operatoren: =, +, /, *, () und Math.sqrt()

Operatoren verknüpfen Operanden

Beispiel: gruss = "Hallo" + vorname;

Modulo

Diese Operatoren sind auf Ganzzahlen und Gleitkommazahlen anwendbar.

%-Operator (Modulo-Operator) liefert den Rest einer Ganzzahldivision

3 % 4 liefert 3

4 % 2 liefert 0

5 % 3 liefert 2

Zuweisungsoperator

– Werte tauschen durch Hilfsvariable

Vergleichsoperatoren – (Wichtig)

== gleich

!= ungleich

> größer als

< kleiner als

>= größer gleich

<= kleiner gleich

Liefert bei entsprechender Wahrheit den Wert „true“ oder „false“

Logische Operatoren

! - nicht - (!true-->false, !false-->true)

&& und (true && true --> true, sonst: false)

|| oder (false || false --> false, sonst: true)

& und (true und true --> true, sonst: false) es werden

BEIDE operatoren ausgewertet im Gegensatz zu &&

^ xor (Siehe Wertetabelle aus Unterlage: „001“)

Inkrementoperatoren - ++ und –

Präinkrement: ++x

x erst um 1 erhöhen, dann den aktualisierten x-Wert im Ausdruck verwenden

Postinkrement: x++

x-Wert erst im Ausdruck verwenden und dann erst um 1 erhöhen

Kurzformen

Die Zuweisung x = x + 5 lässt sich kompakter schreiben als x += 5;

Aufgabe 1 im Skript 002.pdf

ist wichtig. Programm erklären können (Ausgaben)

1) Welche Ausgabe liefert das Programm

„Arithmetiker.java“? Compilieren Sie es zur Kontrolle Ihrer Ausarbeitung.

```
public class Arithmetiker {
    public static void main(String[] args) {
        int y, x = 10;
        y = x + 1;
        System.out.println("1. y=" + y); // 1. y= 11
        y--;
        System.out.println("2. y=" + y); // 2.y =10
        y = ++x + 1;
        System.out.println("3. y=" + y + " x=" + x); // 3.y= 12 x= 11
        y = x++ + 1;
        System.out.println("4. y=" + y + " x=" + x); // 4. y=12 x=12
        y += -y - 2 * 3;
        System.out.println("5. y=" + y); // 5. y= -6
        x = 10;
        y = --x/5;
        System.out.println("6. y=" + y + " x=" + x); // 6. y=1 x=9
    }
}
```

Aufgabe 3 die do -while-Schleife

```
do{
    r = a % b;
    a = b;
    b = r;
}while (r != 0);
```

Anweisungen und Kontrollstrukturen (Autobeispiel)

```
if((hp > 175 && !benzin && preis < 25000 && alter >= 1
&& alter <= 3) && kilometerstandTacho < 30000)
```

liefert TRUE, wenn Hp größer 175 ist UND es kein Benziner ist UND der Preis unter 25000 ist UND das alter größer 1 UND kleiner oder gleich 3 ist

Unterschied kopf- und fussgesteuerte Schleife

(Frage in der Klausur!!!!)

Bei **kopfgesteuerten** Schleifen wird zuerst die Bedingung geprüft und dann der Code ausgeführt, bei einer **fußgesteuerten** Schleife wird zuerst der Code ausgeführt und dann die Bedingung geprüft. Eine fußgesteuerte Schleife wird also immer mindestens einmal ausgeführt.

Seite 4 – Beispiel Handelsvertreter (Lösung)

Wie sorgt man z.B. im Programm dafür, dass ein Kunde abhängig vom Bestellwert:
keinen Rabatt bekommt, wenn gilt: Bestellwert < 1000 €
5% Rabatt bekommt, wenn gilt: Bestellwert >= 1000 €
und zugleich Bestellwert < 10000 €
10% Rabatt bekommt in allen anderen Fällen (entspricht Bestellwert >= 10000 €)

Hier kommen logische Operatoren zum Einsatz.

Lösung:

```
if(bestellWert < 1000)
    preisNachlass = 0.0
else
    if(bestellWert >= 1000 && bestellWert < 10000)
        preisNachlass = 0.05*bestellWert;
    else
        preisNachlass = 0.1*bestellWert;
```

Arrays, Schleifen

Seite 7 – Arrays – wie setzt man Array zusammen und läßt sie in einer Schleife durchlaufen (Indexierung i) siehe Beispielpogramm Zahlenmerker.java
Code steht in Klausur – man sollte dann erkennen können was das für ein Array ist und was das macht

```
import java.util.Scanner;
public class Zahlenmerker {
    public static void main(String[] args) {
        int i;
        double[] reihe; // reihe ist ein double-Feld
        reihe = new double[5]; // Speicherbereitstellung für
        // 5 Feldelemente vom Typ double
        Scanner in = new Scanner(System.in);
        for(i = 0; i < 5; i++){
            System.out.println("Zahl " + (i+1) + ": ");
            reihe[i] = in.nextDouble();
        }
        System.out.println("Du hast folgende Zahlen eingegeben: ");
        for(i = 0; i < 5; i++)
            System.out.println("Zahl " + (i+1) + ": " + reihe[i]);
        System.out.println("Das war's...");
    }
}
```

Syntax:

Schritt eins: Feldvariable definieren:

```
--> Datentyp [] Feldname;
```

oder

```
--> Datentyp --> Feldname [];
```

Schritt zwei: Speicher für Feldelemente bereitstellen:

```
Feldname = new Datentyp [Anzahl];
```

(siehe Listing oben)

Kombiniert gehts auch:

```
int[] a = new int[5];
```

```
double[][] noten = new int[100][10]; //Noten von 100
Schülern in 10 Fächern
```

Zugriffe auf Feldinhalte - über den Feldindex:

```
nt[] a = new int[5];
```

- Definiert ein Feld, das mit „a“ bezeichnet ist
- besteht aus 5 gleichartigen Elementen (Typ: int)
- wird im Speicher in aufeinanderfolgende Speicherzellen abgelegt
- die Elemente sind von 0...4 durchnummeriert
- über a[0], a[1], a[2]... greift man auf die Elemente zu
- der Index eines n-elementigen Feldes reicht immer von 0... n-1
- mit feldname.length enthält man immer der Anzahl der Elemente eines Feldes (hier: 5)

Prüfung von Feldgrenzen:

Felder lassen sich auch ohne schleifen Initialisieren. Hier ist die Angabe „new“ und die Größenangabe nicht notwendig.
double[] jemandesNoten = {1.0, 1.3, 1.7, 1.3, 2.0};
// Werte der Feldelemente werden durch Komma getrennt
// innerhalb geschweifter Klammern angegeben;
// Compiler bestimmt Größe des Feldes anhand der Anzahl // der angegebenen Werte.

```
System.out.println(„Größe des arrays:“ +
jemandesNoten.length);
```

...gibt die Größe des Arrays aus.

Beim Compilieren findet keine Bereichsüberprüfung statt.

Man kann über Feldgrenzen hinaus lesen und schreiben. --

> Es kommt zu einem Fehler während der Laufzeit.

Mehrdimensionale Arrays

(Bubblesort wie funktioniert das Programm – erklären können !!!)

```
package bubblesort;
```

```
public class Main {
```

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

```
        int [] a = new int []
```

```
{23,56,45,67,12,34,67,86,43,13,22,24,78,33,123,678};
```

```
        int c;
```

```
        int i;
```

```
        int j;
```

```
        for (i=a.length; i>0; i--){
```

```
            for (j=0; j<i-1; j++){
```

```
                System.out.println(i + " " + j);
```

```
                if (a[j] > a[j+1]){
```

```
                    c= a[j]; //C= HilfsVar
```

```
a[j]=a[j+1];
a[j+1]=c;}}
```

```
for (i=0;i<a.length;i++)
    System.out.print(a[i]+"t");
}}
```

Seite 10 – Array in einer Schleife durchlaufen

wenn man mit einer Schleife ein Array ausliest, muss man darauf achten, dass das Array bei dem Index 0 beginnt.

Daher ist die Schleifen Bedingung auch immer: `for (int i = 0; i < array.length; i++)`

–Wenn Du sagst, dass er bis `“i <= array.length“` laufen soll, zählt der bei einem Array aus zehn Werten (also 0 bis 9) von 0 bis 10, da ja 10 Elemente im Array liegen.

Was heißt Konstruktor übersetzt?

Ein Konstruktor ist eine spezielle öffentliche (public)Methode einer Klasse, die automatisch als Initialisierungsroutine beim Erzeugen eines Objektes ausgeführt wird den Namen der Klasse trägt; dabei wird kein Rückgabetyt angegeben beliebig überladen werden kann und damit in unterschiedlichen Formen zur Verfügung stehen kann (s.u.)

Besitzt eine Klasse keinen Konstruktor, so stellt der Compiler immer einen parameterlosen Standard-Konstruktor bereit. Beispiel:

```
public Rectangle();
```

Was ist überladen von Methoden? Klausur!!!!!!

In Java ist es erlaubt, Methoden zu *überladen*, d.h. innerhalb einer Klasse zwei unterschiedliche Methoden mit demselben Namen zu definieren. Der Compiler unterscheidet die verschiedenen Varianten anhand der Anzahl und Typisierung der Parameter. Es ist also nicht erlaubt, zwei Methoden mit exakt demselben Namen und identischer Parameterliste zu definieren. Dabei werden auch zwei Methoden, die sich nur durch den Typ ihres Rückgabewertes unterscheiden, als gleich angesehen.

Zugriffsattribute public private protected

Einsatz von Eigenschaften/Methoden mit dem Zugriffsattribut private:

„private“-Elemente sind nur in der Klasse selbst sichtbar und zugreifbar, in der sie definiert werden; so kann z.B. in der main-Methode der Klasse testProg nicht auf die private Eigenschaft geschwindigkeit der Kfz-Klasse zugegriffen werden.

Mit Zugriffsattributen schafft man für Klassen ein „Innen“ und ein „Außen“:

- Innen: Eigenschaften/Instanzvariablen und Methoden, die der Nutzer bzw. andere Klassen nicht sehen
- Außen: Eigenschaften/Instanzvariablen und Methoden, die der Nutzer bzw. andere Klassen sehen und zugreifen kann
- Zur Spezifikation des Innen und Außen gibt es die Zugriffsattribute
- private: Zugriff nur innerhalb der Klasse möglich
- public: Zugriff von überall möglich, die public-Elemente stellen die Schnittstelle einer Klasse nach außen dar
- protected: Zugriff nur innerhalb dieser Klasse und

in abgeleiteten Klassen möglich (später)

Methodendefinitionen:

Rückgabedatentyp: gibt an, von welchem Typ der zurückzugebende Parameter sein muss; soll die Methode keinen Wert zurückgeben, so ist der Rückgabedatentyp void.

Methodenname: legt den Namen fest, unter dem die Methode aufgerufen werden kann; der Name folgt der Bezeichnersyntax. Paar aus Parametertyp und Parametername: beschreibt von welchem Typ die angegebene Eingabevariable ist; hier können beliebig viele Parameter samt Typ - durch Kommata getrennt - aufgeführt werden.

Anweisung(en): die zu einem Block geklammerten Anweisungen dienen der Berechnung und ggf. Rückgabe eines Rückgabewertes. Gültigkeit von Variablen: Auf die in der Methode definierten Variablen kann man nur innerhalb der Methode zugreifen.

Returnanweisung (Seite 5) Klausur!!!!

Return-Anweisung wird benutzt zur Rückgabe des in der Methode berechneten Rückgabewertes.

```
public double ermittleUmfang() {
    double umfang;
    umfang = 2*laenge + 2*breite;
    return umfang;
}
```

Wert des Ausdrucks und Kontrollfluss gehen an die Aufrufstelle zurück.

- Fehlt der Ausdruck, geht lediglich der Kontrollfluss an die Aufrufstelle zurück;
- erlaubt nur bei void – Methoden.
- Typ des Ausdrucks muss dem Rückgabetyt der Mode entsprechen bzw. in diesen wandelbar sein.

Korrespondenz zwischen aktuellen und formalen Parametern gemäß Position in den Parameterlisten (n-ter aktueller Parameter gehört zu n-tem formalen Parameter)

Beispiel:

```
s.substring(0,4) (0: 1. Parameter, 4: 2.Parameter)
s: Objekt
Methode
```

Arbeiten mit Dateien

Einlesen ;

```
FileReader fr = new FileReader("Eingabe.txt"); // FileReader Objekt für angegebene Datei erzeugen
```

```
Scanner in = new Scanner(fr); // Scanner soll den FileReaderScannen
String s;
s=in.nextLine(); // die erste Zeile der Datei auslesen
```

```
System.out.println("Erste Zeile der Datei: \n" +s);
```

welche Klassen benötigt,
import java.io.PrintWriter;
import java.io.FileReader;
import java.util.Scanner;

was muss man machen um mehrere Zeilen lesen zu können.

```
FileReader fr = new FileReader(dateiname);
Scanner infile = new Scanner(fr);
String s;
System.out.println("Hier kommen die Dateiinhalte...");
System.out.println("...suche die Datei...: " +dateiname);
while(infile.hasNextLine())
{
    s = infile.nextLine();
    System.out.println(s);
}
System.out.println("Ende der Ausgabe...");
```

Ebenso schreiben von Dateien (Seite 3 Beispiel)

```
PrintWriter pw = new PrintWriter ("Ausgabe.txt"); // Dateiojekt
erzeugen
pw.println("Jees... hier ist etwas");
pw.println("...auch in 2 Zeilen");
pw.close();
```

009.pdf

SchildkroeteNEck.java (eventuell selbst coden können)

es soll folgende Ausgabe : strich, Dreieck, viereck,
fünfeck, sechseck.... zehneck

```
public class ZeichneNEck {

    public static void main(String[] args) {
        SchildkroetenGrafik schildkroetenGrafik = new
SchildkroetenGrafik("Mehrere n-Ecke", 600, 300);
        Schildkroete schildkroete =
schildkroetenGrafik.createSchildkroete();

        for (int n = 1; n < 11; n += 1) {
            schildkroete.positionieren(30 + 55 * n, 250);
            zeichneNEck(n, schildkroete);
        }

        schildkroetenGrafik.warten();
    }

    private static void zeichneNEck(int n, Schildkroete
schildkroete) {
        for (int schritte = 0; schritte < n; schritte++) {
            schildkroete.drehen(360.0 / n);
            schildkroete.laufen(100.0 / n);
        }
    }
}
```

Zusatzinfos:

io. Exceptions (was macht gerne wenn man dividiert –
wegen 0)

Das folgende unten aufgeführte Programm soll mittels
Exception-Handling sicher gemacht werden. Das Problem
im Quellcode ist die Division durch 0.

```
public class Main {
    public static void main(String[] args) {
        int a = 0;
        int b = 20;
        int c = 0;
        try {
            a = b / c;
        }
        catch(ArithmeticException e) {
            System.out.println(
                "Bitte anderen Divisor eingeben!");
            System.out.println("Meldung: "+e.getMessage());
        }
    }
}
```