Java Zusammenfassung

Aufbau eines Java-Programms

public class Klassenname {       
 public static void main(String[] args) {            
 // Programmcode

}

}

Variabel Deklaration

Static Datentyp VariabelName;

z.B. static int Zahl

Definieren der Variabel

Datentyp Variabel\_Name = Wert;

z.B. int Zahl = 5

Kommentare

// dann geht der Kommentar bis zum Ende der Zeile.

/\* bis \*/ dann kann der Kommentar über mehrere Zeilen gehen.

Ausgabe

System.out.println; gibt eine Ausgabe mit einem Zeilenumbruch aus.

System.out.print; gibt eine Ausgabe ohne Zeilenumbruch aus.

Mit Variabel

Z.B. Ausgabe: "Text1" + Variable + "Text2"

Daten einlesen

Zu oberst im Skript muss man import java.util.Scanner; schreiben.

Mit Scanner scanner\_Name=new Scanner(System.in); erstellt man einen neuen Scanner.

Aufbau

import java.util.Scanner;

public class Klassenname {

public static void main(String[] args) {

Scanner scanner\_Name = new Scanner(System.in);

ganze Zahlen einlessen (Rückgabetyp int)  
z.B. izahl = scanner\_Name.nextInt();

gebrochene Zahlen einlessen (Rückgabetyp double)   
z.B.  dzahl = scanner\_Name.nextDouble();

Zeichenketten einlessen (Rückgabetyp String)  
 z.B. wort = scanner\_Name.next();

Zeile einlessen; inklusive Leerzeichen (Rückgabetyp String)   
z.B. zeile = scanner\_Name.nextLine();

Kontrollstrukturen

If/else

if (Bedingung) {

// Anweisungen werden ausgeführt,  
// falls Bedingung wahr ist

}

else {

// Anweisungen werden ausgeführt,  
// falls Bedingung nicht wahr ist

}

Fallunterscheidung

switch (Variable) {

case Wert1:  // Einstiegspunkt für den Fall

// Variable == Wert1                   
break;

 case Wert2:  // Einstiegspunkt für den Fall

// Variable == Wert2                 
break;

…

default:   // Anweisungen, falls die

// Variable keinen der zur

// Auswahl stehenden Werte

// enthält

}

Der Datentyp von Variable darf nur ein primitiver Datentyp, String oder eine Wrapperklasse sein. Wert1, Wert2, … sind Konstanten!

Wiederholung

while – Schleife (solange wie die Bedingung erfüllt ist)

while (Bedingung) {

// Anweisungen, welche ausgeführt werden,       
// solange die Bedingung erfüllt ist }

do while – Schleife (mindestens einmal aber solange wie die Bedingung erfüllt ist)

do {

// Anweisungen, werden mindestens einmal      
// ausgeführt, und dann so oft wiederholt,      
// solange die Bedingung erfüllt ist

}   
while (Bedingung);

for – Schleife (Spezialfall: Zählschleife)

for (Initialisierung;Bedingung;Schrittweite){

// Anweisungen

}

Beispiel: Schleife wird 10 mal durchlaufen, bei i++ wird die Variabel i eines Gösser pro durchlauf. for (int i = 0; i  < 10; i++) {

//Anweisung

}

Vorzeitiges Beenden einer Scheleife

continue    
- springt zum Ende des Schleifenrumpfes

break (Abbruch)   
- Schleife / switch wird an dieser Stelle verlassen und nächste Anweisung hinter der Schleife / dem switch wird ausgeführt