Kontrollstrukturen

Verzweigungen in JAVA

Scanner, if-else und switch-case

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 eins String-Wertes (bis zum ersten Leerzeichen).
String	nextLine()	Einlesen eins 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).

Beispielcode

```
import java.util.Scanner;
class Addition {
	public static void main (String[] args) {
		Scanner berta = new Scanner(System.in);
		System.out.print("Geben Sie eine ganze Zahl ein: ");
		int zahl = berta.nextInt();
		System.out.print("Die Quadratwurzel der Zahl ist: "+
	Math.sqrt(zahl));

	}
}
```

Erstellen Sie ein Programm, dass die Engabe von „Hallo Klasse IT11“ einliest! **Testen** Sie das Ganze mit der Methode **next()** und **nextLine()**!

Beschreiben Sie den Unterschied von **next()** und **nextLine()**!

Zeichenketten (Strings)

Besonderheiten beim Vergleich

Der Vergleich von zwei Zeichenketten ist mit dem **Gleichheitsoperator (==)** im Allgemeinen nicht möglich. Dieser vergleicht lediglich den Speicherort (Referenz) der String-Objekte und nicht deren Inhalte. Um den Inhalt der Zeichenketten auf Gleichheit zu überprüfen, verwendet man für Strings die Methode **equals( )**. Diese achtet aber beim Vergleich ganz genau auf die Groß- und Kleinschreibung. Mit der Methode **equalsIgnoreCase( )** können Werte unabhängig von Groß – und Kleinschreibung miteinander verglichen werden.

Beispielcode

```
Scanner sc = new Scanner(System.in);
String str1 = sc.nextLine();

if (str1.equals("Celcius") ) { //Nur Celcius wird als wahr ausgewertet
            System.out.println("Temperatur in: " +str1);
      } else {
            System.out.println("Der Name ist falsch!");
}


if (str1.equalsIgnoreCase("Celcius") ) { //Celcius und celcius sind wahr
            System.out.println("Temperatur in: " +str1);
      } else {
            System.out.println("Der Name ist falsch!");
}
```

Weitere Methoden der Klasse String

length() : Liefert die aktuelle Länge des String-Objekts.

subString(int startIndex, int endIndex) : Liefert den Teilstring, der an Position startIndex beginnt und an Position endIndex endet. Wobei die Zeichenkette bei endIndex nicht mehr ausgegeben wird.

toLowerCase() : Alle Buchstaben des Strings werden durch Kleinbuchstaben ersetzt.

toUpperCase() : Alle Buchstaben des Strings werden durch Großbuchstaben ersetzt.

charAt(int index) : Diese Methode liefert das entsprechende Zeichen an einer Stelle, die im Index genannt wird.

Verzweigungen

Definition

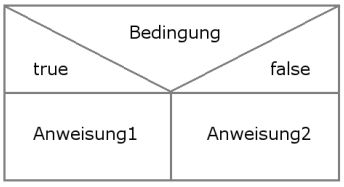
Häufig ist ein linearer Ablauf des Programms nicht möglich. So kann die Auswahl von Kriterien für den weiteren Programmablauf bestimmend sein, um ein richtiges Ergebnis zu liefern.

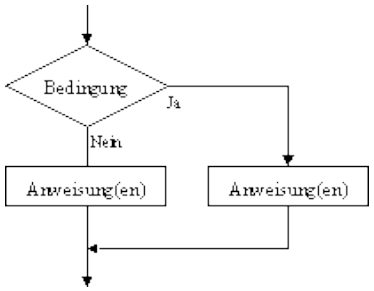
Eine **Verzweigung** legt fest, welche Programmabschnitte, abhängig von einer **Bedingung**, ausgeführt werden.

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

IF-ELSE-Anweisung

Ein Teil es Programmcodes wird nur dann bearbeitet, wenn eine bestimmte Bedingung erfüllt ist. In dieser **Bedingung** werden Vergleichsoperatoren (z.B. **==** , **>** , **<** , **<=** , **>=**, **!=**) eingesetzt. Gibt es keine besondere Alternative, kann der else-Block entfallen.





Beispielcode

```
boolean volljaehrig = true;

if (volljaehrig == true) { //Bedingung muss wahr sein
      System.out.println("Ich bin mindestens 18 Jahre alt."); //Ausgabe
} else {
      System.out.println("Ich bin unter 18 Jahre alt.");
```

Mehrere Verknüpfungen

Es kann notwendig sein mehrere Bedingungen zu verknüpfen.

```
boolean sonne = true;
int temperatur = 23;

if (sonne == true && temperatur > 20) { //logische UND-Verknüpfung
      System.out.println("Ich fahre mit dem Fahrrad."); //Ausgabe
} else {
      System.out.println("Ich fahre mit dem Auto.");
}


if (sonne == true || temperatur > 20) { //logische ODER-Verknüpfung
      System.out.println("Ich fahre mit dem Fahrrad."); //Ausgabe
} else {
      System.out.println("Ich fahre mit dem Auto.");
}
```

IF-ELSE-Anweisung für Mehrfachverzweigungen

Für Mehrfachverzweigungen kann die **ELSE-IF-Anweisung** eingesetzt werden. Abhängig vom Wert einer Variablen lassen sich verschiedene Anweisungen ausführen. Dabei ist zu beachten, dass nur die Anweisung ausgeführt wird, in der die erste Bedingung wahr wird. Alle anderen Zweige werden danach vernachlässigt.

Beispielcode

```
if (alter < 14) {
      System.out.println("Ich bin ein Kind im Sinne des Gesetzes.");
} else if (alter >= 14 && alter < 18) {
      System.out.println("Ich bin Jugendlicher im Sinne des Gesetzes.");
} else {
       System.out.println(„Ich bin Erwachsener im Sinne des Gesetzes!");
}
```

Kontrollfragen

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

größer als		größer gleich als	
kleiner als		kleiner gleich als	
gleich		ungleich	

b) Mit welchen **Methoden** werden Zeichenketten verglichen?

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

Programmierauftrag 1: Tresor

Erstellen Sie ein Programm für einen Tresor, das abhängig von der PIN-Eingabe des Benutzers eine Meldung ausgibt.

Falls die PIN falsch eingegeben wird, erscheint eine Fehlermeldung. **Ansonsten** wird die Meldung angezeigt, dass der Tresor geöffnet werden kann.

Die richtige PIN hat den Wert **615243.**

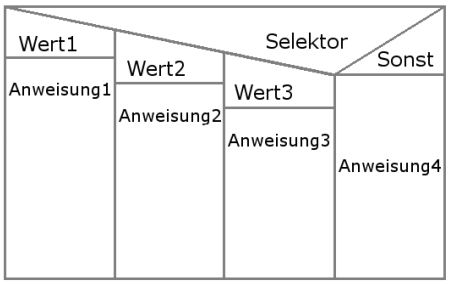
Programmierauftrag 2: Jahreszeiten

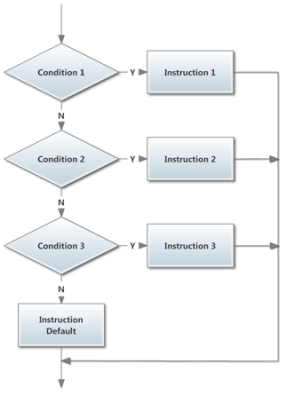
Erstellen Sie ein Programm, das den Benutzer auffordert, einen Kalendermonat einzugeben. Das Programm bestimmt anhand dieser Eingabe, welche Jahreszeit vorliegt. Anschließend wird diese Jahreszeit ausgegeben. Bei der Eingabe einer ungültigen Zeichenkette wird eine Fehlermeldung angezeigt. Folgende Bedingungen gelten:

- Winter (Dezember, Januar, Februar)
- Frühjahr (März, April, Mai)
- Sommer (Juni, Juli, August)
- Herbst (September, Oktober, November)

Switch - Case

In vielen Programmiersprachen gibt es **mehrfache Verzweigungen**, auch Fallunterscheidungen genannt, die abhängig von Bedingungen sind. Ein Teil es Programmcodes wird nur dann bearbeitet, wenn eine bestimmte Bedingung erfüllt ist. Gibt es keine besondere Alternative, kann der Sonst-Block (***default***) entfallen.
Nach dem Programmcode einer jeden Anweisung (bis auf der Sonst-Anweisung) erfolgt ein ***break*** , das den Abschluss einer Verzweigung kennzeichnet.





Beispielcode

```
switch (str) {
	case "Sonne":
		System.out.println("Fahrrad!");
		break;
	case "Regen":
		System.out.println("Auto!");
		break;
	case "Schnee":
		System.out.println("Bus!");
		break;
	default:
		System.out.println("unsicher!?");

}
```

In einem Switch-Case Statement sind nur bestimmte Datentypen sowie Klassen zulässig. Erlaubt sind u.a. **String, char, int, short** und **byte**. Die Datentypen **boolean, long, float** und **double** sind unzulässig.

Das Schlüsselwort **break** am Ende eines jeden Falls (**case**) ist grundsätzlich optional, jedoch meistens notwendig, um die gewünschte Funktionalität zu erreichen. Mit der **break**-Anweisung wird die **Switch-Case-Struktur** sofort **beendet** und nicht mehr weiter ausgeführt. Fehlt die **break**-Anweisung, wird nicht nur der Fall (**case**) mit dem Treffer ausgeführt, sondern auch alle nachfolgenden Alternativen.

Beispielcode für Switch ohne break:

```
switch (zahl) {
	case 1:
		System.out.println("Eins");
	case 2:
		System.out.println("Zwei");
	case 3:
		System.out.println("Drei");
}
```

Hinweis:

- ⇨ Ist die Variable **zahl == 1**, dann wird **Eins**, **Zwei** und **Drei** ausgegeben
- ⇨ Ist die Variable **zahl == 2**, dann wird **Zwei** und **Drei** ausgegeben

Programmierauftrag 3: Schulnoten

Erstellen Sie ein Programm, das Ihnen bei der Eingabe einer Zahl (1-6) die Schulnoten (sehr gut, gut, befriedigend, ausreichend, mangelhaft und ungenügend) ausgibt. Bei einer ungültigen Noten-Eingabe soll eine Fehlermeldung erscheinen.

Programmierauftrag 4: Einfacher Taschenrechner

Erstellen Sie ein Programm, das die vier Grundrechenarten (+, -, /, *) beherrscht. Der Benutzer gibt über die Konsole zwei Zahlen und die gewünschte Rechenoperation ein. Anschließend wird das Ergebnis berechnet und ausgegeben. Bei der Eingabe eines ungültigen Rechenoperators wird eine Fehlermeldung angezeigt.

Erweitern Sie den Taschenrechner mit den Funktonalitäten der Prozentrechnung sowie der Modulorechnung!

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 ohne Programmierung zu lösen. Geben Sie die Werte an, die ausgegeben werden, wenn beim Programmstart

a) a eingegeben wird _____

b) b eingegeben wird _____

c) c eingegeben wird _____

Weitere Programmieraufgaben für Experten

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 zuerst 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 50,00 €.
- 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 CD-Rohlingen:

> 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!

Erstellen Sie ein Abrechnungsprogramm, wobei die Bestellmenge abgefragt wird und anschließend der Rechnungsbetrag (Stückpreis eines Rohlings: 0,50 €) ausgewiesen wird!