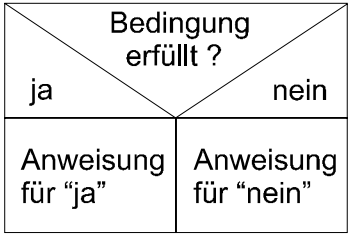
Auswahl (Verzweigungen) in C#

Alle bisherigen Programme in C# waren Sequenzen, d.h. eine Anweisung folgte auf die nächste. Es wurden alle Anweisungen in der gleichen Reihenfolge und vollständig abgearbeitet.

Die ein- und zweiseitige Auswahl:

Im Theorie-Teil haben wir aber schon über die Auswahl (Selektion, Verzweigung) gesprochen.

Struktogramm:



Das links abgebildete Struktogramm bedeutet in die Umgangssprache übersetzt:

```
WENN Bedingung DANN Anweisung für "Ja"
               SONST Anweisung für "Nein"
```

Hier haben wir eine **zweiseitige Auswahl**: Wenn die Bedingung WAHR (`true`) ist, dann wird die Anweisung für "Ja" ausgeführt sonst - d.h. wenn die Bedingung FALSCH (`false`) ist - wird die Anweisung für "Nein" ausgeführt.

Formulierung in C#:

```
if(Bedingung) Anweisung für "Ja";
else Anweisung für "Nein";
```

Besteht die Anweisung aus einer Folge von mehreren Anweisungen, so muss diese Folge durch geschweifte Klammern { } zusammengefasst werden.

Beispiel:

```
double a=4;
if(a<0) a=a*(-1);
else a=2*a;

if(a==0)
{
  MessageBox.Show("Fehler");
}
else
{
  a=5/a;
  MessageBox.Show(a.ToString());
}
```

Die Bedingung `a==0` ist interessant. Es handelt sich hierbei um keine Zuweisung, sondern es wird geprüft, ob die linke Seite gleich der rechten ist. Das Ergebnis dieser Prüfung ist entweder wahr oder falsch.
Man kann die folgenden Operatoren (genannt relationale Operatoren) bei der Formulierung von Bedingungen verwenden:

==	ist gleich mit	!=	Ist ungleich mit
<	ist kleiner als	>	Ist größer als
<=	Ist kleiner oder gleich mit	>=	Ist größer oder gleich mit

Der else-Teil kann auch entfallen. In diesem Fall spricht man von **einseitiger Auswahl**.

Beispiel:

```
if(a<0) MessageBox.Show("Fehler");
```


Mehrere Verzweigungen können auch ineinander verschachtelt werden.

Beispiel:

```
if(a==0){          // Um Platz zu sparen, steht { jetzt oben
  MessageBox.Show("Fehler");
}
else{
  if(a>0) a=5/a;
  MessageBox.Show(a.ToString());
}
```

In solchen Fällen ist eine übersichtliche Struktur des Quelltextes sehr wichtig!

Aufgaben:

1. Erweitere das Programm BMI vom Übungsblatt „Weitere einfache Programme.“, so dass je nach berechnetem Wert ausgegeben wird, ob die Person Übergewicht, Normalgewicht oder Untergewicht hat.
   Hinweis: Betrachte dabei zunächst nur Männer oder Frauen. Ergänze das andere Geschlecht erst, wenn man das Programm läuft.
2. Gegeben ist eine quadratische Gleichung $ax^2 + b\,x + c = 0$ mit beliebigen a, b, c. Nach Eingabe der Zahlenwerte für a, b, c soll der Computer die Gleichung auf Lösbarkeit prüfen und die Lösungen berechnen und ausgeben oder auf die Unlösbarkeit der Gleichung hinweisen.
   Nutze das schon im Unterricht (siehe Thema: Algorithmen) erstellte Struktogramm. Achte auf eine ordentliche Strukturierung des Quelltextes!
3. Es soll das Maximum von zwei eingegeben Zahlen bestimmt werden.
   a) Fertige ein Struktogramm an.
   b) Erstelle das dazu passende C#-Programm.
4. Schreibe ein Programm unter Verwendung der if-Anweisung, das nach Eingabe einer Note von 1 bis 6 diese Note mit Worten (sehr gut, gut, befriedigend, ...) ausgibt und eine unzulässige Eingabe wie 7 oder 3,5 mit einer entsprechenden Mitteilung quittiert.
5. Schreibe ein C#-Programm, mit dem man feststellen kann, ob ein eingegebenes Jahr ein Schaltjahr ist. Verwende das schon erstellte Struktogramm!
   Zur Erinnerung: Alle durch 4 teilbaren Jahre sind Schaltjahre. Ausnahme: Alle durch 100 teilbaren Jahre sind keine Schaltjahre mit Ausnahme der durch 400 teilbaren Jahren. Die sind trotzdem Schaltjahre.
6. Eine Versicherung erstattet ihren Kunden einen Teil des Jahresbeitrages zurück und zwar bei mehr als 5-jähriger Mitgliedschaft 9 %, anderenfalls 4 %. Die Rückerstattungsbeträge sollen berechnet und vom Jahresbeitrag für das folgende Jahr abgezogen werden.
   a) Was sind die Eingabedaten? Was sind die Ausgabedaten?
   b) Formuliere umgangssprachlich einen Lösungsalgorithmus.
   c) Fertige ein Struktogramm an.
   d) Erstelle jetzt erst das C#-Programm.
7. Erstelle eine InfoKurs-Anwendung mit der Turtle. Lass die Turtle in der Mitte ihres Bereichs starten. Füge ein Button hinzu, mit dem die Turtle immer um 20 Schritte nach vorne geht. Wenn sie an die Ränder ihres Bereichs stößt, soll sie ihre Richtung umdrehen und zurücklaufen. Nutze dazu u. a. die Eigenschaften Height und Width der Turtle.

