

Weitere einfache Programme in C#

Bevor es an die nächsten Aufgaben geht, noch einige wichtige Punkte:

Ganzzahlige Division

Zur Verdeutlichung schauen wir uns den folgenden Quelltext an:

```
// Die Variablen x und y werden deklariert:
int x,y;
// Den Variablen werden Werte zugewiesen:
x=3;
y=4;
// x wird durch y dividiert und in einer neuen Variablen
// quotient abgespeichert:
int quotient; // (1)
quotient=x/y; // (2)
MessageBox.Show(quotient.ToString());
```

Die Zeilen mit // sind Kommentarzeilen. Alles hinter // wird vom Computer ignoriert und hilft nur den Menschen, die mit dem Quelltext arbeiten.

Wenn bei (1) die Variable `quotient` vom Typ `int` (also als ganze Zahl) definiert ist, dann ist die Ausgabe im Meldungsfenster 0.

Wenn bei (1) die Variable `quotient` vom Typ `double` (also als Dezimalzahl, sprich: Kommazahl) definiert ist, bleibt die Ausgabe weiterhin 0, da `x` und `y` ganze Zahlen sind. Die Division wird weiterhin ganzzahlig ausgeführt.

Erst wenn man angibt, dass man eine Division mit `double`-Zahlen durchführen will, hat man Erfolg. Dies geschieht, indem man (`double`) in Klammern davor schreibt:

```
double quotient; // (1)
quotient=(double)x/y; // (2)
```

Umwandlung eines Textes in eine Zahl

Wenn man eine Textbox verwendet, um eine Zahl einzulesen, muss man erst dafür sorgen, dass man eine „richtige Zahl“ erhält. Mit Hilfe der Eigenschaft `Text` der Textbox kann man auslesen, was in einer Textbox steht. Aber der Wert dieser Eigenschaft ist als Datentyp `String` (Zeichenkette) gespeichert. Auch wenn in ihr eine Zahl gespeichert ist, ist diese Zahl für den Computer etwas anderes als eine normale Zahl, mit der er rechnen kann. Deshalb muss man die Zahl aus ihrer Textdarstellung in etwas für den Computer Sinnvolles umwandeln.

Dies zeigt das folgende Beispiel:

```
// für ganze Zahlen gilt:
int Ganze_Zahl; // Variable deklarieren
// aus textBox1.Text einlesen und mit Hilfe von int.Parse umwandeln
Ganze_Zahl=int.Parse(textBox1.Text);
// für Kommazahlen gilt:
double Kommazahl;
// aus textBox1 einlesen und umwandeln mit Hilfe von double.Parse
Kommazahl=double.Parse(textBox1.Text);
```

Aufgaben:

1. Schreibe ein Programm, mit dem man zwei Dezimalzahlen addieren, subtrahieren, multiplizieren und dividieren kann. Die beiden Zahlen sollen in Textboxes eingegeben werden können.
2. Schreibe ein Programm, mit dem man den Body Mass Index (BMI) berechnen kann. Suche dazu im Internet nach der Formel für die Berechnung. Notiere die verwendete Quelle.

3. Schreibe ein Programm, mit dem man nach der Eingabe von Länge und Breite die Fläche und den Umfang eines Rechtecks berechnen kann. Ergänze es später durch Berechnungsmöglichkeiten für weitere geometrische Formen (z. B. mittels eines `TabControl`s).

Komponenten InfoKurs-Tools

In diesem Kurs verwenden wir zusätzliche Komponenten, die man von der Homepage des Kurses herunterladen kann. Wenn man das Installationsprogramm ausführt, werden die Komponenten installiert und Sharp Develop so eingerichtet, dass man sie ohne Probleme verwenden kann.

Wenn man nach der Installation eine neue Projektmappe anlegt, steht ein zusätzlicher Projekttyp zur Verfügung, genannt „InfoKurs-Anwendung“, den man ab jetzt verwenden sollte. Im Moment sind im Paket drei Komponenten enthalten:

InputBox:	Die Klasse <code>InputBox</code> stellt ein Eingabedialogfeld zur Verfügung, mit dem man ganze Zahlen (<code>int</code>), Gleitkommazahlen (<code>double</code>) und Zeichenketten (<code>String</code>) eingeben kann.
RandomNumbers:	Die Komponente <code>RandomNumbers</code> stellt Zufallszahlen mit Hilfe einer visuellen Komponente zur Verfügung.
Turtle:	Die Komponente <code>Turtle</code> stellt einen Bereich zur Verfügung, in dem auf einfache Weise gezeichnet werden kann.

Die Komponenten werden in Sharp Develop in der Toolbox unter „InfoKursTools“ installiert.

Beispiel für die Turtle:

```
void Button1Click(object sender, System.EventArgs e)
{
    turtle1.GoToMiddle(); // Bewegt Turtle in die Mitte d. Zeichenfläche
    turtle1.Angle=-45; // dreht die Turtle auf 45 Grad
    turtle1.MoveTurtle(50); // bewegt die Turtle um 30 Einheiten
    turtle1.PenIsDrawing=true; // ab jetzt zeichnen
    turtle1.Angle=180;
    turtle1.GoForward(60);
    turtle1.TurnLeft(90);
    turtle1.GoForward(60);
    turtle1.TurnLeft(90);
    turtle1.GoForward(60);
    turtle1.TurnLeft(90);
    turtle1.GoForward(60);
}
```

Aufgaben:

4. Was wird im obigen Beispiel ausgegeben?
5. Schreibe ein Programm, das eine Turtle benutzt, um ein Haus zu zeichnen.
6. Schreibe ein Programm mit der `RandomNumbers` und der `Turtle`-Komponente, in dem die Turtle bei jedem Buttondruck eine zufällige Bewegung macht.
7. Zeichne mit der `Turtle` weitere geometrische Figuren.

