

---

# PROGRAMMIERUNG – EINFÜHRUNG

## Inhaltsverzeichnis

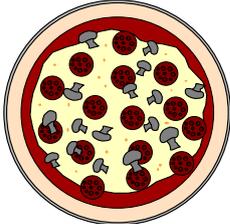
---

<b>1. Vom Problem zum Programm</b> .....	<b>1-1</b>
1.1 Wie löst man ein Problem?	
1.2 Einfache Designtechnik	
<b>2. Sequenzen</b> .....	<b>2-1</b>
<b>3. Verzweigungen</b> .....	<b>3-1</b>
<b>4. Schleifen</b> .....	<b>4-1</b>
4.1 kopfgesteuerte Schleifen	
4.2 fussgesteuerte Schleifen	
<b>5. Einführung in die Programmierung</b> .....	<b>5-1</b>
5.1 Mein erstes Programm	
5.2 Variablen	
<b>6. Grundlagen von Visual BASIC</b> .....	<b>6-1</b>
6.1 Komponenten eines Visual BASIC Programms	
6.2 Objekt- und ereignisorientierte Programmierung	
6.3 Form und Toolbox	
6.4 Eigenschaftenfenster und Farbpalette	
6.5 Das Codefenster	
<b>7. Verzweigungen mit Visual BASIC</b> .....	<b>7-1</b>
<b>8. Schleifen mit Visual BASIC</b> .....	<b>8-1</b>
8.1 DO ... LOOP - Schleifen	
8.2 FOR-TO-NEXT-Schleifen	
8.3 Weitere Beispiele	

# 1. Vom Problem zum Programm

# 1

## 1.1 Wie löst man ein Problem?

<p><b>Problem:</b> Pizza backen</p>	<p><b>Lösung:</b> Rezept</p>
	<ul style="list-style-type: none"> <li>• Teig herstellen</li> <li>• kreisrund ausbreiten</li> <li>• Mozzarella, Tomaten und weitere Zutaten auflegen</li> <li>• im Holzofen backen</li> </ul>

### Das Problem

Im täglichen Leben treffen Sie viele Probleme an, bei deren Bewältigung Ihnen eine Anleitung hilft, z.B. Pizza backen mit Rezept, eine Stereoanlage aufbauen nach Bedienungsanleitung, ein Billett an der Tramstation nach Anweisung lösen, Regeln für Notfälle (Feuer, Erste Hilfe) befolgen.

### Die Anleitung - der Algorithmus

Anleitungen, die in der richtigen Reihenfolge ausgeführt ein Problem lösen, nennt man Algorithmen. Um das Problem zu lösen, müssen Sie nur die Anleitung verstehen und einen Schritt nach dem anderen durchführen. Um eine Anleitung für ein Problem zu schreiben, müssen Sie nicht nur wissen, wie das Problem zu lösen ist - also den richtigen Algorithmus herausfinden - , sondern auch wie die Anleitung zu schreiben ist.

### Das Programm

Den Algorithmus in einer für einen Computer verständlichen Sprache nennt man Programm.

Beispiele für Probleme die ein Computer lösen kann:

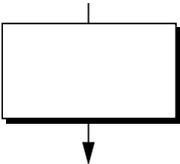
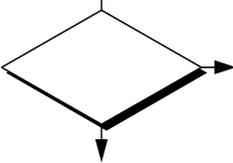
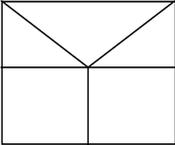
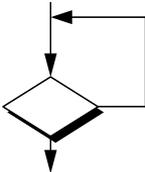
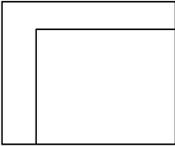
- Er soll Ihre Adressen nach Postleitzahlen sortieren.
- Er soll einen Text, den Sie geschrieben haben, speichern und ausdrucken.
- Er soll die Zeit auf dem Bildschirm anzeigen.

### Also ...

<i>Problem</i>	Hier ist die Ausgangslage, dort ist das Ziel
<i>Algorithmus</i>	logische Folge von Schritten zum Ziel
<i>Programm</i>	entsprechende Befehle oder Taten

## 1.2 Einfache Designtechnik

Um Algorithmen grafisch darzustellen, verwendet man in der Informatik vorwiegend Flussdiagramme oder Struktogramme. Dabei beschränkt man sich im Wesentlichen auf drei Strukturen:

Vorgang	Flussdiagramm	Struktogramm
Sequenz		
Verzweigung		
Schleife		

*Die Sequenz*

ein Schritt folgt nach dem anderen.

*Die Verzweigung*

abhängig von einer Bedingung wird der eine oder andere Weg besritten

*Die Schleife*

ein Programmteil oder eine Anzahl Schritte wird mehrmals wiederholt.

Mit diesen drei Grundstrukturen lassen sich alle Algorithmen darstellen.

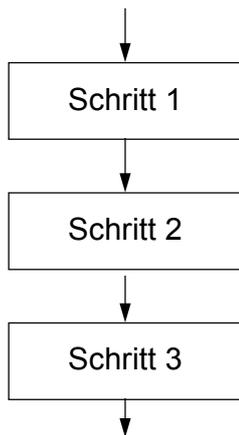
## 2. Sequenzen

# 2

### Darstellung von Sequenzen

Einzelne Schritte, die hintereinander folgen, nennt man Sequenzen.

Flussdiagramm

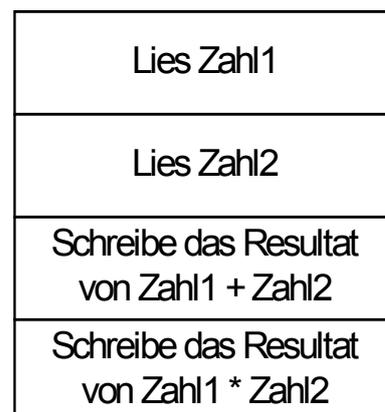
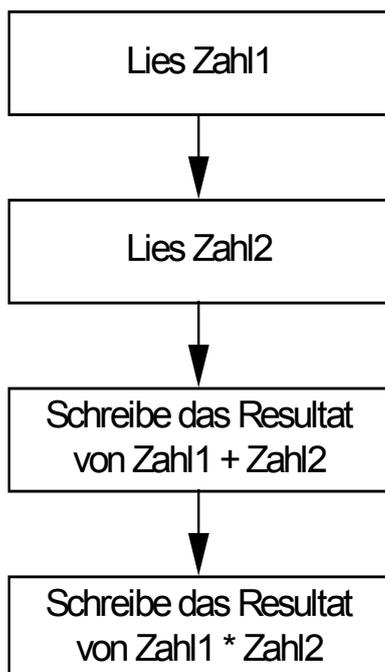


Struktogramm



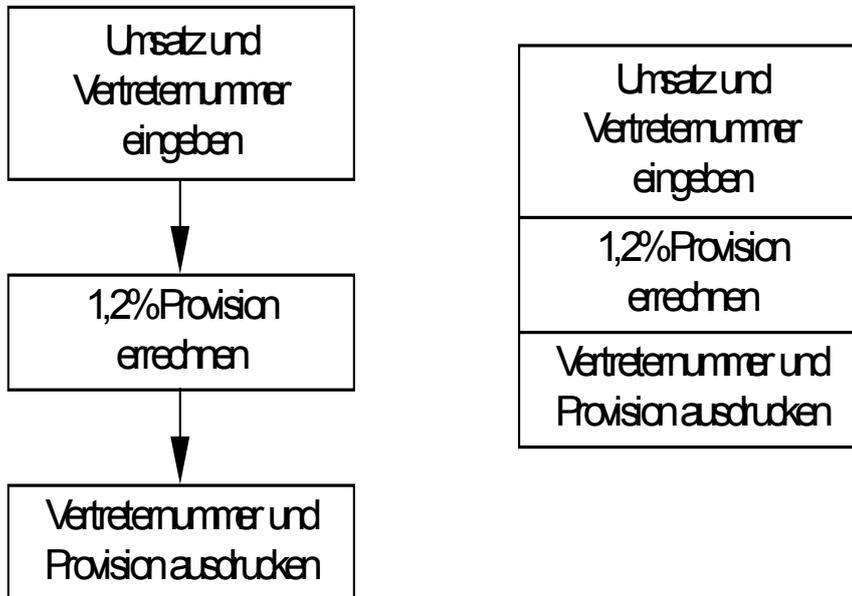
### Beispiele «Zwei Zahlen»

Der Computer soll zwei Zahlen einlesen, sie addieren bzw. multiplizieren und die jeweiligen Ergebnisse ausgeben.



**Beispiel «Vertreter»**

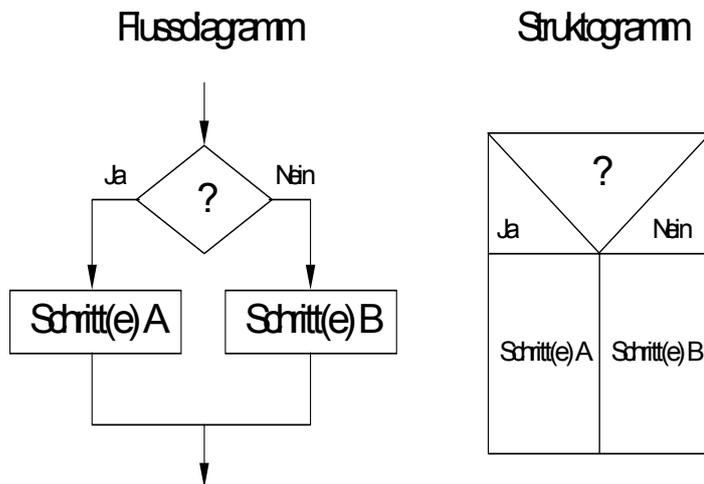
Der Computer soll aufgrund der Vertreternummer und des Umsatzes die Provision von 1.2% berechnen und ausgeben.



## 3. Verzweigungen

# 3

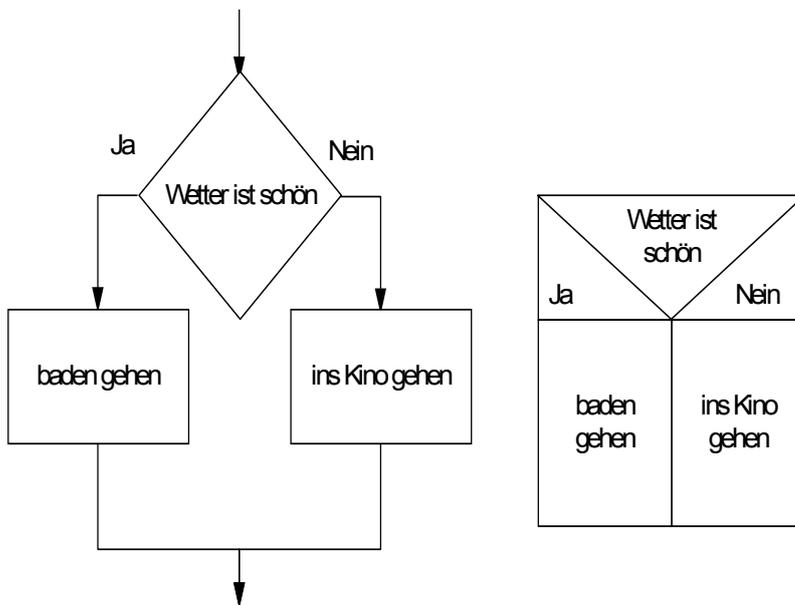
Darstellung von Verzweigungen



### Beispiel «Wetter»

Ein Teenager sagt zu seiner Mutter:

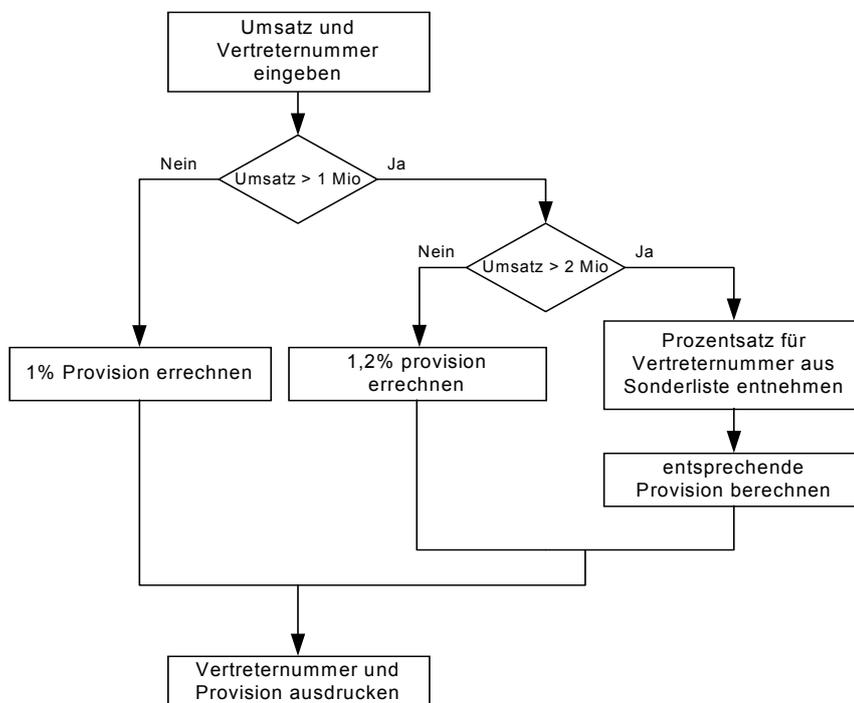
*Wenn es heute nachmittag schönes Wetter ist, gehe ich baden, sonst gehe ich ins Kino.*



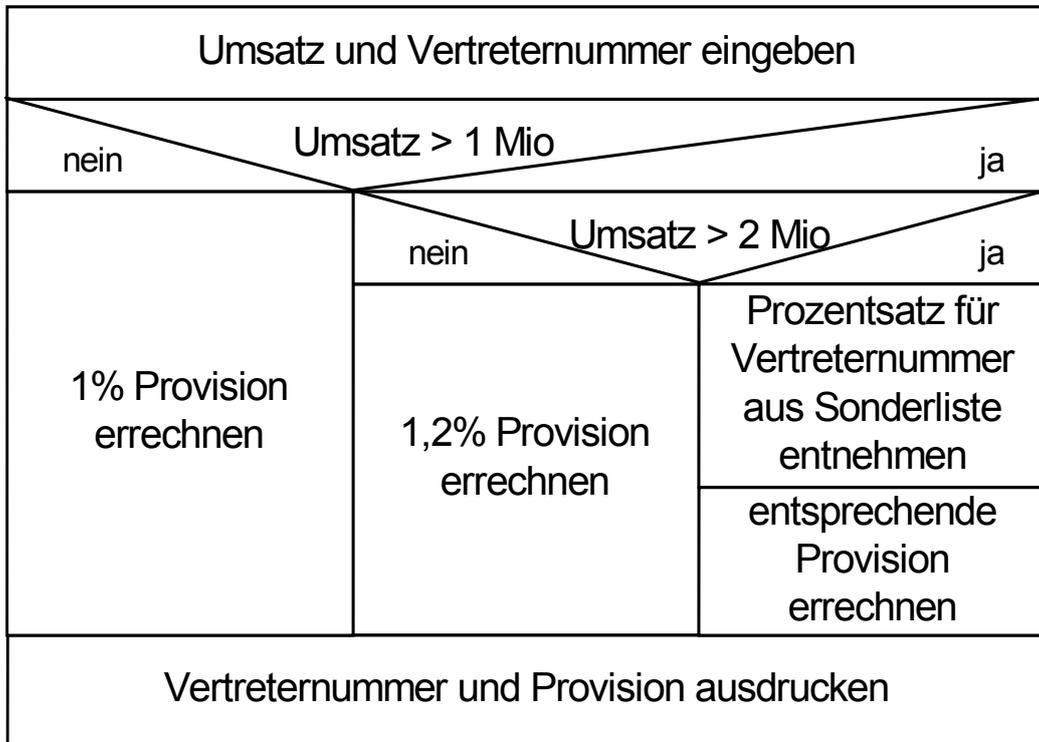
### Beispiel «Vertreter»

Für die Vertreter eines Unternehmens, die mit ihrer Vertretersnummer erfasst sind, soll eine Provision errechnet und einzeln ausgedruckt werden. Vertreter mit einem Umsatz bis zu einer Million bekommen 1% Provision, Vertreter mit 1 bis 2 Millionen Umsatz bekommen 1.2% Provision. Mit Vertretern, die mehr als 2 Millionen Umsatz gemacht haben, wurden Einzelverabredungen getroffen, die in einer Sonderliste ausgewiesen sind.

Flussdiagramm



Struktogramm



## 4. Schleifen

# 4

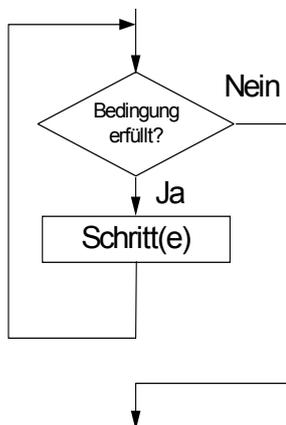
### 4.1 kopfgesteuerte Schleifen

#### Definition

- Bedingung am Anfang
- Anweisungen werden nie ausgeführt, falls Bedingung schon am Anfang falsch ist.

#### Darstellung

Flussdiagramm



Struktogramm

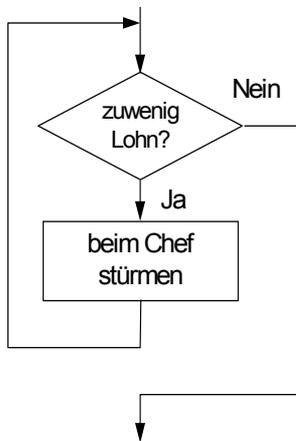


#### Beispiel

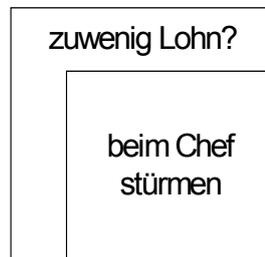
##### *Arbeitswelt*

Solange Du zuwenig Lohn hast, «stürme» beim Chef um mehr Lohn.

Flussdiagramm



Struktogramm

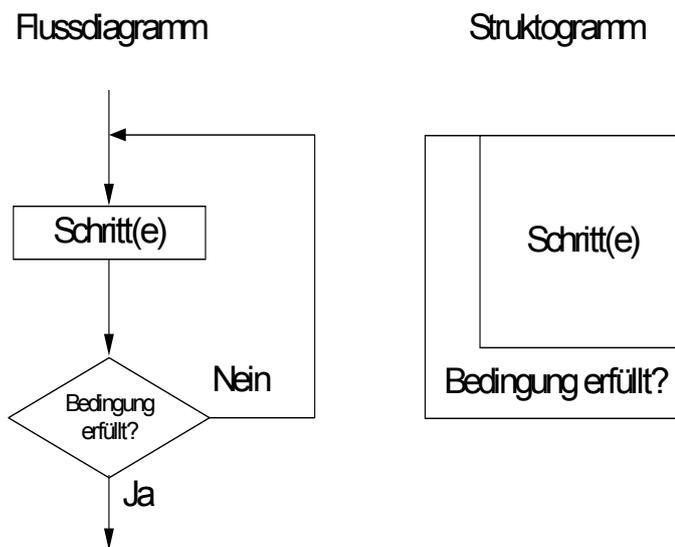


## 4.2 fussgesteuerte Schleifen

### Definition

- Bedingung am Schluss
- Anweisungen werden *mindestens einmal* ausgeführt.

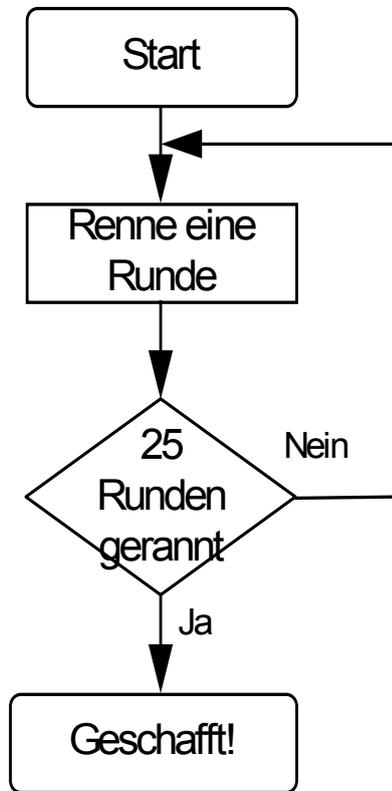
### Darstellung



### Beispiele

*10'000-Meter-Lauf*

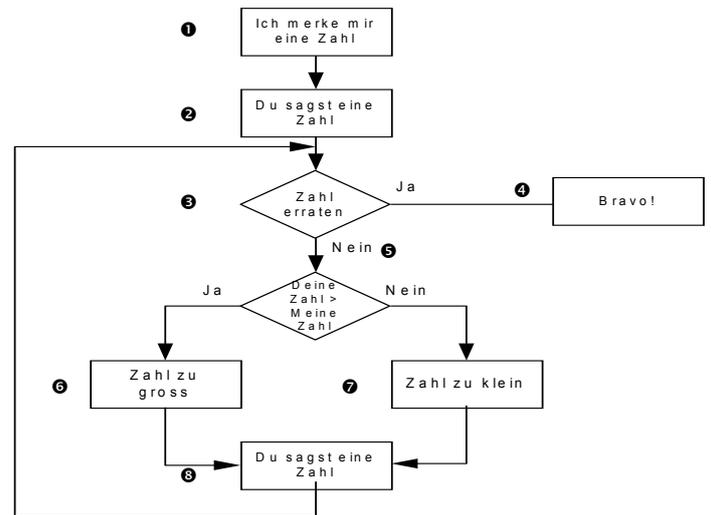
Renne, bis Du 25 Runden gedreht hast.



### Spiel «Zahlen erraten»

Ich denke mir eine Zahl und Sie müssen sie erraten. Ich sage «Zu klein!» oder «Zu gross!» bis Sie sie erraten haben.

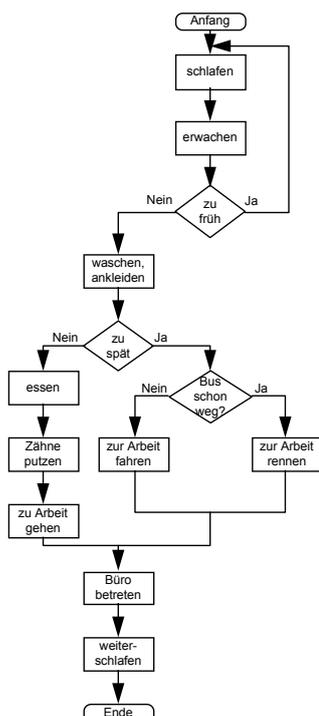
1. Zu Beginn merke ich mir eine Zahl.
2. Im nächsten Schritt versuchen Sie diese Zahl zu erraten, indem Sie eine Zahl sagen.
3. Jetzt kommt die Abfrage der Bedingung: «Haben Sie die Zahl erraten?».
4. Falls ja, ist das Spiel fertig.
5. Falls nein, sage ich, je nach dem ob die von Ihnen genannte Zahl zu gross oder zu klein ist :
6. «Zahl zu gross!» oder
7. «Zahl zu klein»
8. Mit dieser zusätzlichen Information, können Sie den nächsten Rateversuch unternehmen.



☞ Das Spiel dauert so lange, bis Sie die Zahl richtig erraten haben.

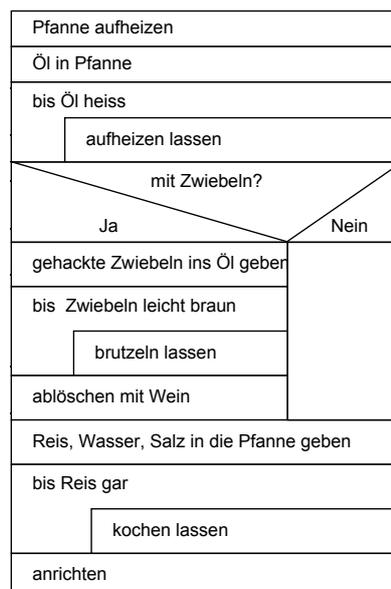
### Montagsmorgen-Drama

Flussdiagramm



### Reis kochen

Struktogramm



## 5. Einführung in die Programmierung

# 5

### 5.1 Mein erstes Programm

Schauen wir uns Schritt für Schritt die Entstehung eines kleinen Programmes in Visual BASIC an.

1. Starten Sie Visual BASIC.
2. Öffnen Sie über das Menü [Datei], <Projekt öffnen> das Projekt FIRSTPRG.MAK.
3. Wählen Sie die Schaltfläche <Code anzeigen>.
4. Im Fenster FIRSTPRG.BAS tippen Sie das folgende Programm:

```
Sub Main ()
Dim A, B as string
A = InputBox(«Geben Sie Ihren Vornamen ein:»)
B = InputBox(«Geben Sie Ihren Nachnamen ein:»)
MsgBox («Ihr vollständiger Name lautet «& A & ««& B)
End Sub
```

5. Jetzt starten Sie das Programm mit der Taste **[F5]** oder mit einem Klick auf das entsprechende Symbol.
6. Der erste Befehl des Programmes verlangt Ihren Vornamen und speichert ihn unter dem Namen A ab.
7. Der zweite Befehl verlangt den Nachnamen und speichert ihn unter B ab.
8. Im letzten Schritt wird ihr vollständiger Name in einer sogenannten MessageBox ausgegeben.

#### Sequenz



#### Handhabung des Programmes

Visual BASIC-Programme können Sie über das Menü, mit Tastaturbefehlen oder mit Symbolen steuern

##### Starten



[Ausführen] <Starten>



##### Beenden



[Ausführen] <Beenden>



##### Unterbrechen



[Ausführen] <Unterbrechen>



##### Schrittweise ausführen



[Testen] <Einzelschritt>



##### Haltepunkte ein- und ausschalten



[Testen], <Haltepunkt ein-/ausschalten>



## 5.2 Variablen

Variablen wie A und B im obigen Beispiel sind Platzhalter für Zahlen und Textzeichen und werden dazu verwendet, Zahlenwerte oder Text während des Programmablaufs zu speichern.

Variablennamen dürfen maximal 40 Zahlen oder Buchstaben lang sein und müssen mit einem Buchstaben beginnen. Sie dürfen keine Sonderzeichen ausser - oder \_ enthalten. Es wird nicht zwischen Gross- und Kleinschreibung unterschieden.

### Beispiele für Variablen

<i>Zulässige Namen</i>	<i>unzulässige Namen</i>
summe	1Produkt
breite	x+y
Quartals_Budget	such.resultat

### Zuweisung

Die Zuweisung von Werten an Variablen erfolgt durch das Gleichheitszeichen '='. Bei einer Zuweisung wird in der links stehenden Variablen der Wert des rechts stehendes Ausdrucks gespeichert. Der Ausdruck rechts kann auch eine Berechnung darstellen.

#### Beispiele

```
breite = 10
länge = 20
fläche = breite * länge
ergebnis = ergebnis + 100
Meldung = «Geben Sie nur ganze Zahlen ein.»
```

### Variablen-Deklaration

Die Namen und Typen der Variablen, die in einem Programm verwendet werden, müssen am Anfang des Programm deklariert werden. Dadurch wird der benötigte Speicherplatz reserviert und alle Variablen im Programm auf die korrekte Schreibweise überprüft.

#### Syntax

```
Dim variable as datentyp
```

#### Anmerkung

<i>variable</i>	bezeichnet den Namen der Variablen.
<i>datentyp</i>	deklariert den Typ der Variablen. Dieser kann <i>integer</i> für ganze Zahlen oder <i>string</i> für einen Text sein.

#### Beispiele

```
Dim breite, länge as integer
Dim Meldung as string
```

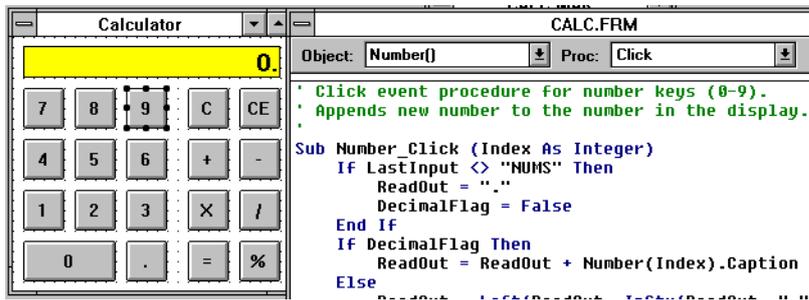
## 6. Grundlagen von Visual BASIC

# 6

### 6.1 Komponenten eines Visual BASIC Programms

Ein Visual BASIC Programm besteht aus 2 Komponenten:

- 1 oder mehrere Forms
- Programmcode.



Form mit Codefenster

Die Form stellt die Benutzeroberfläche des Anwenders dar, im Code werden die benötigten Funktionen erstellt. Da Visual BASIC auf **Ereignisse** reagiert, erstellt man zuerst die Formen und legt anschliessend fest, wie auf welche Ereignisse reagiert werden soll.

#### Das Projekt

Alle Teile eines Visual BASIC Programm werden in einer \*.mak-Datei, dem Projekt, zusammengefasst.



Das Projektfenster

Im Projektfenster sieht man alle im Projekt vorkommenden Formen (\*.frm), BASIC-Module (\*.bas) und Zusatzelemente (\*.vbx).

Das Projektfenster wird beim Erstellen eines Projekts automatisch geladen.

## 6.2 Objekt- und ereignisorientierte Programmierung

### Objekte im Alltag

Ein Objekt (z.B. ein Telefon) kann auf verschiedene Arten definiert werden:

#### *Eigenschaften*

Das Objekt Telefon besitzt Eigenschaften, z.B. ist es rot (Eigenschaft Farbe = rot), es ist alt (Eigenschaft Zustand = alt) und es besitzt Tasten (Eigenschaft Tasten = ja).

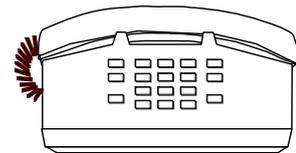
#### *Methoden*

Mit dem Objekt Telefon kann man verschiedenes machen, man kann z.B. den Hörer abnehmen, auflegen, eine Nummer wählen etc. Man spricht in diesem Zusammenhang von «Methoden».

#### *Ereignis*

Ein wichtiges Ereignis, das bei einem Telefon eintreten kann, ist das Klingeln.

### Telefon



#### **Eigenschaften**

*Telefon.Farbe = weiss*

#### **Methode**

*Telefon\_abnehmen*

#### **Ereignis**

*Telefon\_klingeln*

### Objekte unter Windows

Alles, was wir unter Windows antreffen, wird als Objekt aufgefasst, z.B. Fenster, Schaltflächen, Menüs oder die Zwischenablage. Jedes Objekt hat:

#### *Eigenschaften*

diese bestimmen das Aussehen und Verhalten eines Objektes, z.B. Grösse und Farbe

#### *Methoden*

Funktionen und Prozeduren, die beim Objekt selber gespeichert sind. z.B.: Form.hide blendet eine Form aus.

#### *Ereignisse*

Jedes Objekt registriert verschiedenste Ereignisse, z.B. Klicken mit der Maus, Änderung eines Textfeldes etc. Zu jedem Ereignis lässt sich eine Prozedur schreiben, die dann ausgeführt wird, wenn das Ereignis eintritt.

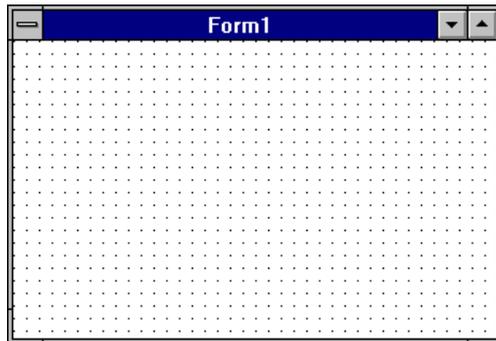
Ein Visual BASIC Programm wird nicht linear von oben nach unten abgearbeitet, sondern besteht aus einer Vielzahl von Prozeduren, die zu einem durch den Benutzer bestimmten Zeitpunkt ausgeführt werden.

Ereignisprozeduren sind dabei beim jeweiligen Objekt hinterlegt. Daneben kann man auch globale Funktionen und Prozeduren schreiben, die alle Objekte aufrufen können.

## 6.3 Form und Toolbox

### Die Form

Beim Erstellen eines neuen Projekts wird automatisch eine leere Form (oder Formularfenster) erzeugt.



Dieses Formularfenster verfügt bereits über alle Elemente, die ein Windows-Fenster ausmachen: Titelleiste, Systemmenü, Symbol- und Vollbildschaltflächen.

Der gepunktete Hintergrund ist ein *Raster*, an welchem neue Elemente ausgerichtet werden.

### Die Toolbox

Um eine Form zu erweitern, dient die Toolbox.

Hier finden Sie alle Elemente, die Sie einer Form hinzufügen können. Dabei sind in der Professional-Version von Visual BASIC mehr Elemente verfügbar als in der Standardversion.

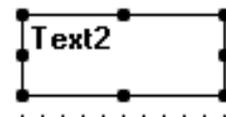
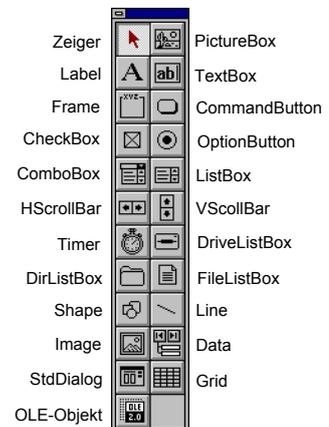
#### *Ein Element zu einer Form hinzufügen*

- Ziehen Sie das Element aus der Toolbox an die gewünschte Stelle
- oder doppelklicken Sie auf das Element fügt dieses in der Mitte des Formulars hinzu.

#### *Ein Element verschieben/Grösse ändern*

Wenn Sie ein Element anklicken, erhält es wie abgebildet 8 Rechtecke entlang des Randes. Ziehen Sie mit der Maus an diesen, um die Grösse zu verändern.

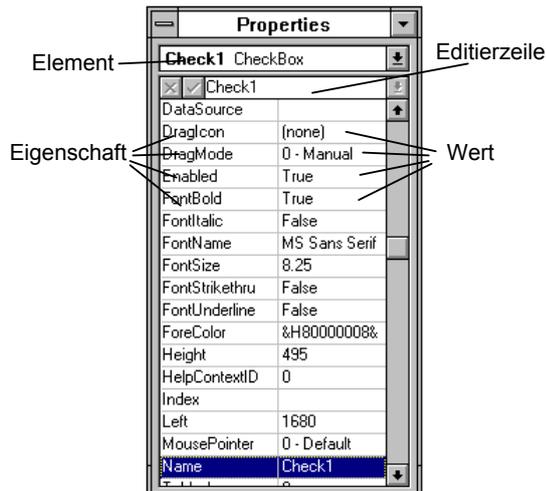
Um es zu verschieben, Klicken Sie in das Element und ziehen es an seine neue Position.



## 6.4 Eigenschaftenfenster und Farbpalette

### Das Eigenschaftenfenster

Zu jedem Element lassen sich verschiedene *Eigenschaften einstellen*, die das Aussehen und Verhalten bestimmen. Dies erfolgt im Eigenschaftenfenster.



### Ändern einer Eigenschaft

Zum Ändern einer Eigenschaft klicken Sie in die entsprechende Zeile und stellen den gewünschten Wert ein. Sie befinden sich automatisch in der Editierzeile, sobald Sie eine Taste drücken.

☞ Als Alternative gelangen Sie mit **[F2]** in die Editierzeile.

In der Editierzeile können folgende Symbole vorhanden sein:



öffnet einen Dialog, z.B. zur Auswahl eines Bildes

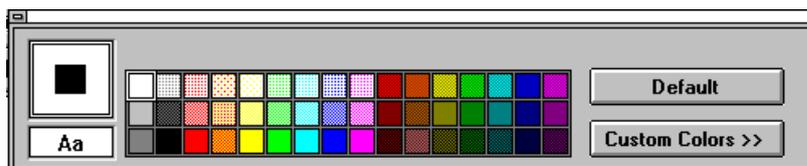


öffnet eine Liste mit möglichen Werten, z.B. True/False

☞ Beide Symbole lassen sich mit **[ALT]+[↓]** öffnen.

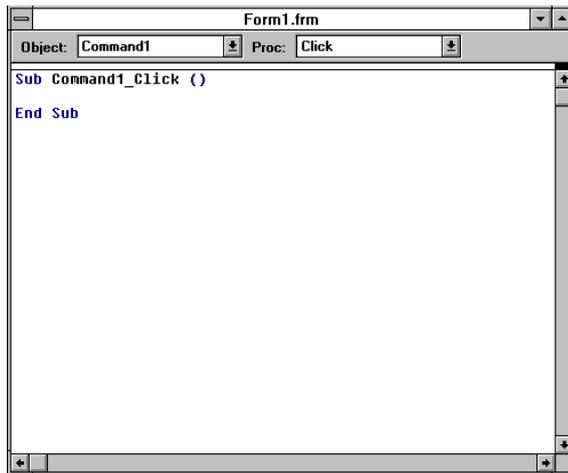
### Farbpalette

Zum Ändern der Farben dient die **Farbpalette**.



## 6.5 Das Codefenster

Wenn Sie auf ein Element im Form **doppelklicken**, öffnet sich das Codefenster zu dem Ereignis, das am häufigsten für dieses Element verwendet wird:



Das Codefenster

Hier können Sie Ihre Prozedur schreiben.

## 7. Verzweigungen mit Visual BASIC

# 7

Die einfache Verzweigung wird mit dem IF-Befehl programmiert.

### Der IF-Befehl

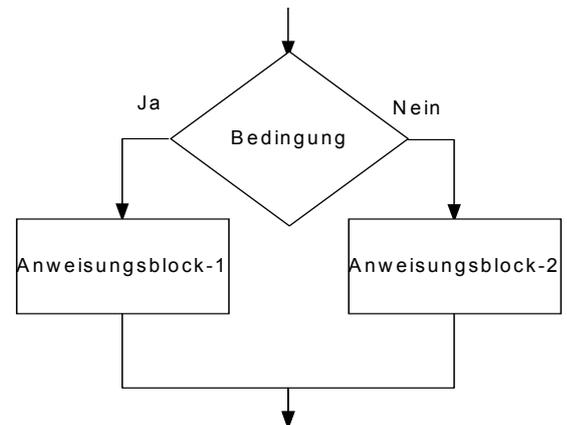
Erlaubt eine bedingte Ausführung von Anweisungen aufgrund der Auswertung eines Ausdrucks.

#### Syntax

```
If Bedingung Then
    Anweisungsblock -1
Else
    Anweisungsblock -2
End If
```

#### Anmerkung

If	Reserviertes Wort, das den IF-Befehl einleitet.
Bedingung	Entspricht Bedingung
Then	Reserviertes Wort für auszuführende Aktionen, falls eine Bedingung erfüllt worden ist.
Anweisungsblock-1	Eine oder mehrere Visual BASIC-Anweisungen in einer oder mehreren Zeilen.
Else	Reserviertes Wort für auszuführende Aktionen, falls die Bedingungen nicht erfüllt ist.
Anweisungsblock-2	Ein oder mehrere ausgeführte Visual BASIC-Anweisungen, falls Bedingung falsch ist.
End If	Reserviertes Wort, das den IF-Befehl beendet.



#### Beispiel

Vergleichen Sie das Beispiel im Kapitel «3. Verzweigungen»

```
Sub Main ()
Dim wetter As String
wetter = InputBox(«Wie ist das Wetter? \
    (schön/regnerisch)», «Wetter?», «schön»)
If wetter = «schön» Then
    MsgBox «Gehe baden», , «Antwort»
Else
    MsgBox «Gehe ins Kino», , «Antwort»
End If
End Sub
```

#### Lesbarkeit

Beachten Sie, dass für eine bessere Lesbarkeit des Programmes innerhalb einer Schleife die Zeilen eingerückt sind.

## 8. Schleifen mit Visual BASIC



### 8.1 DO ... LOOP - Schleifen

Wiederholt einen Block von Anweisungen, solange eine Bedingung wahr ist oder bis eine Bedingung wahr wird.

#### Syntax

##### *kopfgesteuerte Schleife:*

```
Do [While/Until Bedingung]
    [Anweisungsblock]
Loop
```

##### *fussgesteuerte Schleife*

```
Do
    [Anweisungsblock]
Loop [While/Until Bedingung]
```

Do	Muss die erste Anweisung in einer Do...Loop-Schleife sein.
While	Zeigt an, dass die Schleife ausgeführt wird, solange Bedingung wahr ist.
Until	Zeigt an, dass die Schleife ausgeführt wird, bis Bedingung wahr ist.
Bedingung	Numerischer oder Zeichenfolgenausdruck, der wahr oder falsch ist.
Anweisungsblock	Programmzeilen zwischen den Do- und Loop-Anweisungen, die wiederholt werden, solange oder bis Bedingung wahr ist.

#### Beispiel

##### *kopfgesteuerte Schleife:*

```
Sub Form_Paint ()
    Dim x
    x = 3
    Do While x < 6
        Print x
        x = x + 0.5
    Loop
End Sub
```

Das Programm gibt die Werte 3, 3.5, 4, 4.5, 5 und 5.5 aus. Bei der nächsten Erhöhung um 0.5 erhält x den Wert 6 und ist somit nicht mehr kleiner als 6. Die *while*-Bedingung ist nicht mehr erfüllt und die Schleife stoppt.

##### *fussgesteuerte Schleife*

```
Sub Form_Paint ()
    Dim x
    x = 10
    Do
        Print x
        x = x - 1
    Loop Until x = 0
End Sub
```

Das Programm gibt die Werte 10, 9, 8 usw bis 1 aus. Ist x = 0, stoppt die Schleife .

## 8.2 FOR-TO-NEXT-Schleifen

Wiederholt mehrmals eine Folge von Befehlen.

### Syntax

```
For Zähler = Startwert To Endwert [Step Schrittgröße]
  [Anweisungsblock]
Next [Zähler]
```

 In einer absoluten Schleife ist genau festgelegt, wie oft die Anweisungen ausgeführt werden. Die Variable Zähler startet bei Startwert und wird um den Wert Schrittgröße erhöht bis sie den Wert der Variablen Endwert erreicht. Bei jeder Erhöhung werden die Befehle des Anweisungsblocks ausgeführt.

For	Beginnt eine For...Next-Struktur. Muss vor allen anderen Teilen der Struktur erscheinen.
Zähler	Numerische Variable, die als Schleifenzähler verwendet wird.
Startwert	Anfangswert des Zählers.
To	Trennt Startwert und Endwert voneinander.
Endwert	Endwert des Zählers.
Step	Gibt an, daß Schrittgröße explizit angegeben ist.
Schrittgröße	Betrag, um den der Zähler bei jedem Schleifendurchlauf geändert wird. Wenn Sie keinen Wert für Step angeben, erhält Schrittgröße den Standardwert 1.
Anweisungsblock	Programmzeilen zwischen For und Next, die so oft wie angegeben ausgeführt werden.
Next	Beendet eine For...Next-Schleife. Veranlasst, dass Schrittgröße zu Zähler hinzuaddiert wird.

### Beispiel

```
For i=1 to 10
  Print i
Next i
```

### Beispiel-Programm

Dieses kleine Programm liefert den Ausdruck der Zahlen 1 bis 10. Folgende Schritte werden der Reihe nach ausgeführt.

1.	i=1	Der Variablen i wird der Wert 1 zugewiesen.
2.	to 10	Der Wert von i wird mit 10 verglichen. Ist er grösser, stoppt die Schleife, sonst wird der nächste Befehl ausgeführt.
3.	Print i	Dieser Befehl druckt den Wert von i aus, in diesem Moment also 1.
4.	Next i	Der Wert der Variablen i wird um 1 erhöht, ist also jetzt 2.
5.	to 10	i wird wieder mit 10 verglichen. i ist noch nicht grösser als 10, also wird die Schleife weiter ausgeführt.
6.	Print i	2 wird ausgedruckt
...		usw bis 10 ausgedruckt wird.
31.	Next i	i wird jetzt 11
32.	to 10	i ist jetzt grösser als 10. Die Schleife stoppt

i	Print i
1	1
2	2
3	

## 8.3 Weitere Beispiele

Was machen die folgenden Beispiele?

### Beispiel «Verzinsung»

```
For i=1 to Laufzeit
  Zinsen = Kapital * Zinssatz / 100
  Kapital = Kapital + Zinsen
Next i
```

### Beispiel «Farbige Rechtecke»

```
For rot = 0 To 255 Step 25
  fillcolor = RGB(rot, 0, 0)
  x0 = rot * 20
  y0 = 0
  x1 = x0 + 500
  y1 = y0 + 500
  fillstyle = 0
  Line (x0, y0)-(x1, y1), , B
Next rot
```

### Verschachtelte Schleifen

For-To-Next -Schleifen können auch verschachtelt werden, das heisst innerhalb einer Schleife kann eine zweite Schleife ausgeführt werden.

### Beispiel «Farbiges Parkett»

```
For rot = 0 To 255 Step 25
  For grün = 0 To 255 Step 25
    fillcolor = RGB(rot, grün, 0)
    x0 = grün * 20
    y0 = rot * 20
    x1 = x0 + 500
    y1 = y0 + 500
    fillstyle = 0
    Line (x0, y0)-(x1, y1), , B
  Next grün
Next rot
```

Zuerst wird rot auf den Wert 0 gesetzt. Jetzt wird die ganze Schleife For grün = 0 To 255 Step 25 abgearbeitet. Danach wird rot um 25 erhöht und wieder wird die ganze Schleife für grün ausgeführt. Rot wird wiederum erhöht und so weiter bis rot den Wert 255 überschritten hat.