



Inhaltsverzeichnis

Inhaltsverzeichnis	2
1 Makros aufzeichnen	4
1.1 Einführung	4
1.2 Entwicklertools	5
1.3 Makrosicherheit	6
1.4 Makroaufzeichnung starten	7
1.5 Makro aufrufen	9
1.6 Symbolschaltfläche	12
1.7 Menüleiste	14
2 Visual Basic für Anwendungen	16
2.1 VBA Editor	16
2.2 Programmcode	18
2.3 Sub Prozeduren	19
2.4 Hilfen bei der Programmierung	20
3 VBA Allgemein	22
3.1 With Anweisung	22
3.2 InputBox	22
3.3 MsgBox	23
3.4 Abfragen	25
3.5 Schleifen	27
3.6 Datums und Zeitfunktionen	29
3.7 Textfunktionen	30
4 VBA-Excel	32
4.1 Zellen und Bereiche	32
4.2 Spalten und Zeilen	33
4.3 Werte eintragen	37
4.4 Column und Row	39
4.5 Tabellen	40
4.6 Formeln	42
4.7 Tabellenfunktionen aus Excel	43
4.8 Ereignisprozeduren	44
5 Daten	46
5.1 Variablen	46
5.2 Datentypen	47
5.3 Datenfelder	48



5.4	Konstanten	49
5.5	Variablen Gültigkeit	50
6	Formulare.....	52
6.1	Benutzerdefinierte Formulare.....	52
7	Benutzerdefinierte Funktionen.....	54
7.1	Funktion erstellen.....	54
7.2	Funktionen verwenden	55
7.3	Optionale Argumente	55
7.4	Beschreibung für Funktionen	56
7.5	Kategorie zuordnen	57
7.6	Add-In verwenden.....	58
8	Wichtiges	60
8.1	Programmcode drucken.....	60
8.2	Hilfe für VBA	60
8.3	Programmtest und Fehlersuche	61
8.4	Fehlerbehandlung.....	63
9	Beispiele	64
	Makro Zellschutz	64
	Mehrere Tabellen schützen	66
	Makros für Zahlenformat	67
	Liter in Kilogramm	68
	Zeile einfügen	69
	Rechnungsnummer erhöhen.....	70
	Umfrageauswertung	71
	Projektzeiten	72
	Eingabemaske für Projektzeiten.....	74
	Spezialfilter mit Makro	77
	Diagramm anpassen	78
	Tabellenblätter sortieren.....	80
	Liste der Tabellenblätter erstellen	81
	Dateiliste erstellen.....	82
	Mit Schaltfläche starten.....	83
	Rechnungen in Liste übertragen.....	84
	Adresse in Rechnung eintragen.....	85
	Rechnungsnummer aus Liste übernehmen	86
	Artikel in Rechnung eintragen	88
	Funktionen	90



7 BENUTZERDEFINIERTER FUNKTIONEN

Sie können mit VBA eigene Funktionen erstellen. Diese stehen Ihnen dann genau so zur Verfügung wie Funktionen die in Excel bereits enthalten sind.

Diese selbst erstellten Funktionen stehen dann im Funktionsassistenten in der Kategorie **Benutzerdefiniert** zur Verfügung.

7.1 Funktion erstellen

Rufen Sie in Excel den VB-Editor mit **[Alt]+[F11]** auf.

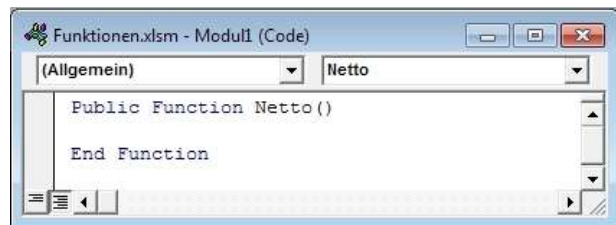
Im Editor wählen Sie den Menübefehl **Einfügen / Prozedur**. Falls der Befehl noch nicht zur Verfügung steht, wählen Sie vorher im gleichen Menü den Befehl **Modul**, dann können Sie anschließend die Prozedur erzeugen.



Übung:
Funktionen..... 90



Im Fenster **Prozedur hinzufügen** geben Sie den Namen der gewünschten Funktion ein, im Beispiel ist das *Netto*, und wählen als Typ *Function*. Im Codefenster erscheint nun der Prozedurrumpf. In diesem müssen Sie Ihren Programmcode noch ergänzen.



Programmcode erstellen

Als erfahrener Anwender können Sie diesen Prozedurrumpf auch manuell eintippen.

Die Funktion soll aus einem Bruttobetrag den Nettobetrag errechnen unter Verwendung des angegebenen Prozentsatzes.

Ergänzen Sie die Prozedur, indem Sie in die Klammern zunächst die Argumente eintragen, die übergeben werden müssen. Hier sind es *Brutto* und *Prozent*. Danach erstellen Sie die Formel für die Berechnung.

```
Public Function Netto(Brutto, Prozent)
    Netto = Brutto / (1 + Prozent)
End Function
```

In der Formel müssen Sie vor dem = Zeichen den Namen der Funktion angeben, in diesem Fall **Netto** so dass das Ergebnis an die Funktion zurückgegeben wird.

Speicherort

Eine Funktionsprozedur wird in der Mappe gespeichert, in der sie erstellt wurde. Damit steht diese nur dann zur Verfügung, wenn die Mappe geöffnet ist. Das kann beabsichtigt sein.

Um benutzerdefinierte Funktionen immer zur Verfügung zu haben, können erfahrene Anwender diese in der persönlichen Arbeitsmappe speichern. Eine Beschreibung dazu finden Sie unter Makro speichern in. Siehe Seite 8



7.2 Funktionen verwenden

Um die benutzerdefinierte Funktion zu verwenden, rufen Sie im Funktionsassistenten die Kategorie **Benutzerdefiniert** auf, dann werden Ihnen in der rechten Spalte alle Funktionen angezeigt, die Sie selbst erstellt haben.



7.3 Optionale Argumente

Sie können Argumente als optional festlegen, diese müssen dann nicht angegeben werden, aber Sie können angegeben werden. Um ein Argument in einer Funktionsprozedur als **optional** zu deklarieren fügen Sie das Wort *Optional* vor das entsprechende Argument. Optionale Argumente müssen am Ende der Aufstellung stehen.

Funktion: Netto nach Abzug von Rabatt errechnen, Prozentwert optional
 Diese Funktion errechnet einen Nettobetrag, bei dem vom Bruttobetrag ein Rabatt subtrahiert wird, jedoch ist der Prozentwert optional, das bedeutet, er muss nicht angegeben werden. Es reicht, wenn nur der Bruttobetrag als Argument angegeben wird. Wird kein Prozentwert angegeben, wird mit 10% gerechnet.

```
Public Function Rabatt(Brutto, Optional Prozent As Single = 0.1)
    Rabatt = Brutto * (1 - Prozent)
    Rabatt = Excel.Application.Round(Rabatt, 2)
End Function
```

Übung:
 Zahl in
 Worten91

Zum Runden auf zwei Dezimalstellen wird hier die Funktion **Runden** aus Excel verwendet, die hier in Englisch geschrieben werden muss. (*Round*)

In den nachfolgenden Abbildungen sehen Sie, dass die Funktion sowohl mit einem Argument als auch mit zwei rechnet.

Nur ein Argument: Zellbezug

<i>f_x</i>	=Rabatt(B1)	
B	C	
1.000,00 €	900,00 €	

Zwei Argumente: Zellbezug und Prozentsatz

<i>f_x</i>	=Rabatt(B1;B2)	
B	C	
1.000,00 €		
25%	750,00 €	



7.4 Beschreibung für Funktionen



Zu benutzerdefinierten Funktionen können Sie einen beschreibenden Text erzeugen der im Funktionsfenster dem Anwender zusätzliche Informationen liefert. Diese Beschreibung können Sie im Fenster Makrooptionen sehr einfach eingeben.

Öffnen Sie im Register **Entwicklertools**, Gruppe **Code** mit der Schaltfläche **Makros** das Fenster **Makro**. In diesem Fenster wird die benutzerdefinierte Funktion noch nicht angezeigt. Sie müssen im Feld **Makroname:** den Namen ihrer benutzerdefinierte Funktion eintippen. Sobald der Name vollständig ist, ist die Schaltfläche **Optionen...** aktiv.

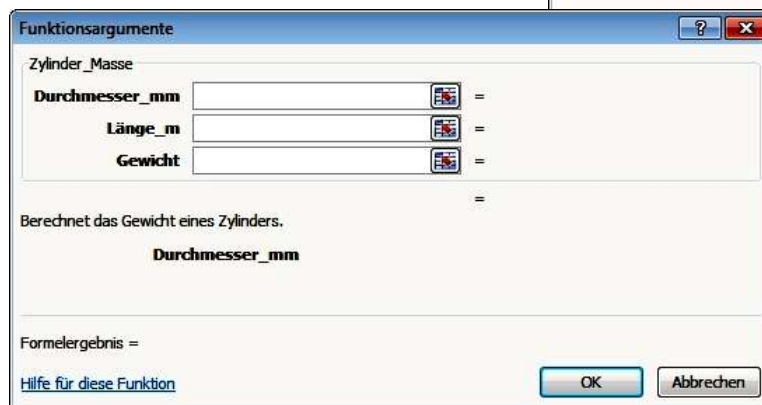
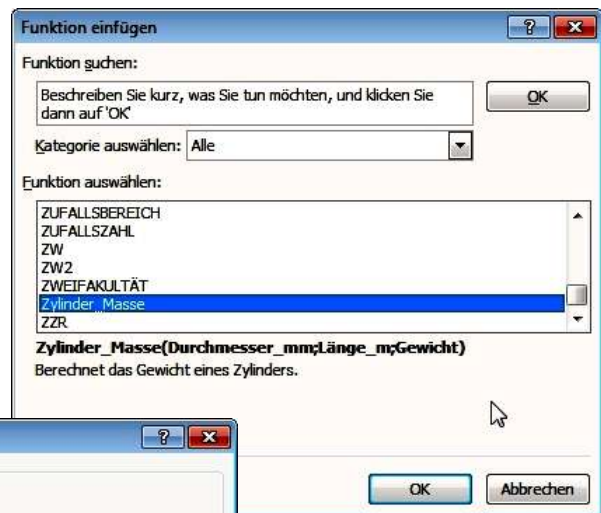
Wenn Sie auf diese Schaltfläche klicken wird das Fenster **Makrooptionen** angezeigt.



In diesem Fenster können Sie im Feld **Beschreibung:** den erklärenden Text erfassen. Dieser Text ist für die Anwender sichtbar.

Der Beschreibungstext wird nun in Fenster **Funktion einfügen**, bei der Auswahl der Funktion am unteren Rand des Fensters angezeigt.

Und auch im Fenster **Funktionsargumente** unterhalb der Auflistung der Argumente.





7.5 Kategorie zuordnen

Benutzerdefinierte Funktionen werden in der Kategorie **Benutzerdefiniert** abgelegt. Sie können aber bestimmen in welcher Kategorie Ihre benutzerdefinierte Funktion gefunden werden kann. Dafür verwenden Sie den VBA Befehl `MacroOptions`. In dem Sie beim Argument `Category` eine Nummer der vorhandenen Kategorien angeben.



Kategorie	Bezeichnung
1	Finanzmathematisch
2	Datum & Zeit
3	Mathematik. & Trigonometrie
4	Statistik
5	Matrix
6	Datenbank
7	Text
8	Logik
9	Information
...	

Diese Befehlszeile fügt die Funktion `Zylinder_Masse` in die Kategorie **Mathematik. & Trigonometrie** ein.

```
Application.MacroOptions Macro:="Zylinder_Masse", Category:=3
```

Eigene Kategorie erzeugen

Sie können jedoch auch eigene Kategorien festlegen indem Sie bei `Category` anstelle einer Zahl den Namen Ihrer Kategorie angeben. Diesen müssen Sie in Anführungszeichen schreiben.

Mit dem Befehl `MacroOptions` können Sie zusätzlich mit dem Argument `Description` einen Hilfetext angeben, wenn Sie diesen nicht im Fenster **Optionen** erzeugen wollen.

Der nachfolgende Code erzeugt einen Beschreibungstext und legt die Funktion in die neue Kategorie **Kreis**.

```
Public Sub FunktionsOptionen_einfügen()
    Text = "Berechnet die die Masse eines zylindrischen Körpers in kg. "
    Text = Text & "Wenn der Durchmesser in 'mm' die Länge in 'm' "
    Text = Text & "und das Gewicht in 'kg' angegeben ist."

    Application.MacroOptions Macro:="Zylinder_Masse", _
        Description:="Text", Category:="Kreis"
End Sub
```

Zum Aktivieren der Funktionsoptionen müssen Sie bei jedem Start der Datei mit den Funktionen diese Routine ausführen. Am besten ist es, wenn Sie beim Öffnen der Arbeitsmappe die nachfolgend gezeigte Ereignisprozedur `Workbook_Open` ausführen lassen. Siehe Seite 45.

```
Private Sub Workbook_Open()
    Call FunktionsOptionen_einfügen
End Sub
```



7.6 Add-In verwenden

Ein Add-In ist eine Excel-Datei in einem besonderen Format, in der Programme, Formulare und/oder Funktionen enthalten sind, aber keine Tabellen. Sie sind geeignet um Programme an andere PCs weiter zu geben, diese Add-Ins können in Excel deaktiviert werden, so dass Ihre Funktionalität ausgeschaltet ist. Sie können aber jederzeit wieder aktiviert werden.

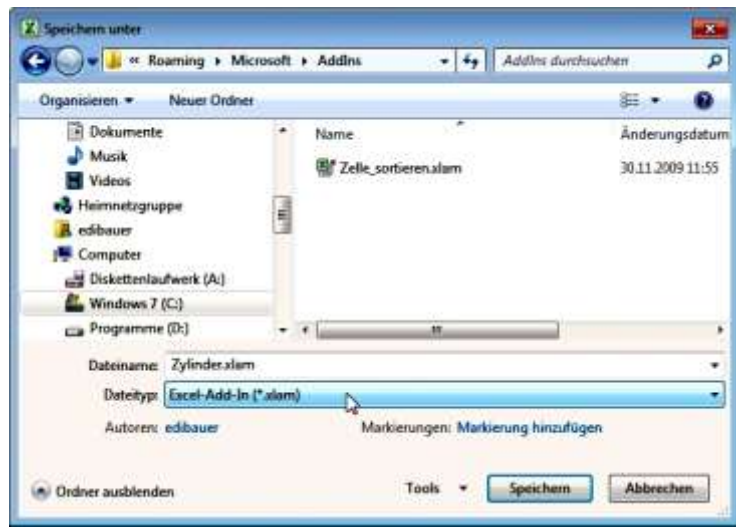


Add-In erstellen

Aus einer fertigen Excel Datei mit VBA Code können Sie sehr einfach ein Add-In erstellen. Öffnen Sie die Datei aus der Sie das Add-In erstellen wollen.

Nicht zwingend notwendig, aber dennoch sehr nützlich: Ändern Sie in **Datei / Informationen** bei Eigenschaften den **Titel** z.B. in *Funktionen zur Zylinderberechnung*.

Wählen Sie das Menü **Datei / Speichern unter**. Im gleichnamigen Fenster wählen Sie den Dateityp *Excel-Add-In (*.xlam)*. Durch die Auswahl wird als Speicherort das AddIns-Dateiverzeichnis automatisch vorgeschlagen.

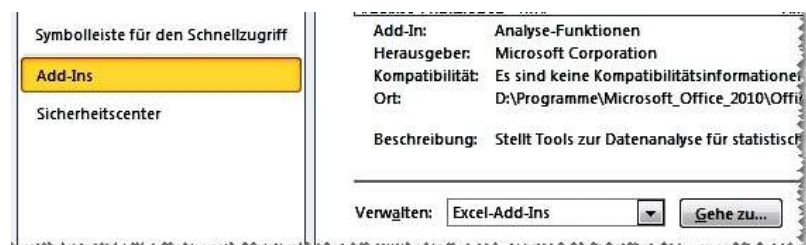


Original aufbewahren

Bewahren Sie die Original -xlsm-Datei gut auf, denn eine xlam-Datei können Sie nicht mehr öffnen. Falls Sie in Ihrem Add-In eine Änderung vornehmen wollen benötigen Sie die Originaldatei, aus der Sie eine neue Add-In-Datei erzeugen können.

Weitergeben

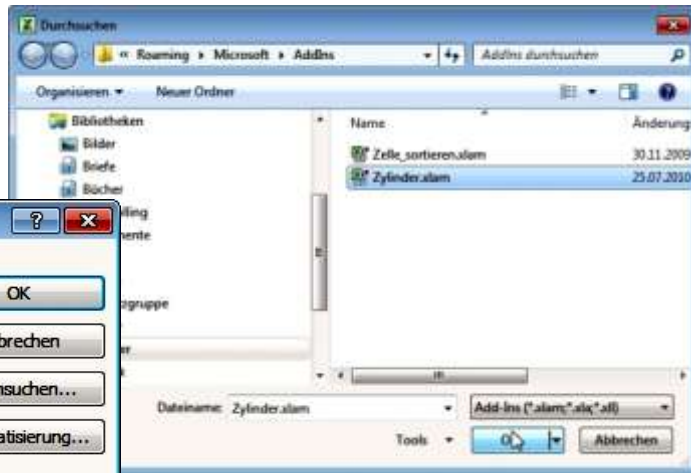
Auf dem Ziel-PC öffnen Sie das Fenster **Optionen** und aktivieren die Kategorie **Add-Ins**. Rechts unten neben **Verwalten: Excel-Add-Ins** klicken Sie auf **Gehe zu...**



Dadurch öffnet sich das **Fenster Add-Ins**. In diesem Fenster klicken Sie auf die Schaltfläche **Durchsuchen...** Darauf öffnet sich das Fenster **Durchsuchen**. Suchen Sie hier die xlam-Datei die Sie verwenden wollen und klicken rechts auf die Schaltfläche **OK**.



Nach dem Klick auf die Schaltfläche **OK** sehen Sie wieder das Add-Ins Fenster mit dem neuen Eintrag.



Obwohl im Beispiel die Datei *Zylinder.xlam* ausgewählt wurde, wird im Add-Ins Fenster der Eintrag *Funktoinen zur Zylinderberechnung* angezeigt. Das ist der Titel, der im Menü **Datei** in der Kategorie **Informationen** angelegt wurde. Wenn Sie keinen Titel für die Datei anlegen, wird im Add-In-Fenster der Dateiname angezeigt.

Geheim

Der Vorteil der Add-Ins ist dass Sie die Datei sehr einfach auf anderen PCs installieren können. Da die xlam-Datei nicht geöffnet werden kann, ist es den Anwendern unmöglich herauszubekommen wie Ihre Berechnungen bzw. Programme funktionieren.

Add-In deaktivieren und löschen

Brauchen Sie ein Add-In vorübergehend nicht mehr, dann können Sie dieses im Add-In-Fenster durch entfernen des Häkchens deaktivieren. Die Datei bleibt auf dem PC aber die Funktionalität ist nicht mehr nutzbar.

Um ein Add-In endgültig aus der Liste zu entfernen, gehen Sie folgendermaßen vor. Sie öffnen im Windows Explorer das Verzeichnis mit den Add-Ins. Um die Datei nur aus der Liste zu entfernen benennen Sie diese einfach um, um Sie endgültig zu entfernen löschen Sie die betroffene xlam-Datei.

In der Add-In-Liste ist der Eintrag aber immer noch vorhanden. Erst wenn Sie versuchen das Add-In in der Liste zu aktivieren, verschwindet es.

Speicherort

Die Add-Ins werden je nach Betriebssystem und Installation an unterschiedlichen Orten gespeichert. Den Ort finden Sie heraus, wenn Sie das Optionfenster öffnen und in der Gruppe **Add-Ins** in der Liste das von Ihnen hinzugefügte Add-In suchen.

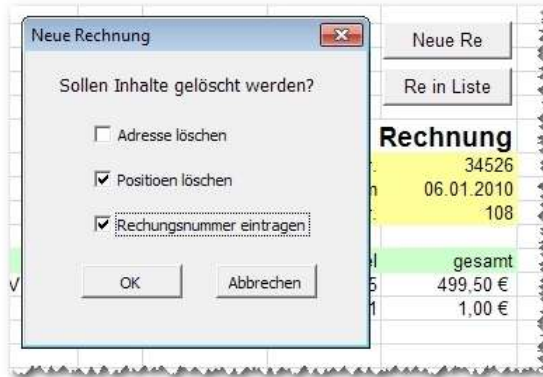


Themen:
Formular
If...Then
Range
Rows
Sheets

Rechnungsnummer aus Liste übernehmen

Sie haben eine Liste mit den bisher ausgestellten Rechnungen, aufsteigend nach Rechnungsnummern sortiert. Beim Schreiben einer neuen Rechnung ist es sinnvoll, die nächste Rechnungsnummer aus dieser Liste zu übernehmen.

In diesem kleinen Programm können Sie in einer Maske auch noch bestimmen, ob die *Adresse* und die *Rechnungspositionen* der vorherigen Rechnung verwendet oder gelöscht werden sollen.



Formular erstellen

- ① Erstellen Sie im VisualBasic-Editor das oben abgebildete Formular und stellen Sie diese Eigenschaften ein.

Steuerelement	Eigenschaft	Wert
UserForm	(Name)	frmNeueRe
	Caption	Neue Rechnung
Label	(Name)	Label1
	Caption	Sollen die Inhalte gelöscht werden?
checkBox	(Name)	chkAdresse
	Caption	Adresse löschen
checkBox	(Name)	chkPosi
	Caption	Positionen löschen
checkBox	(Name)	chkReNr
	Caption	Rechnungsnummern eintragen
CommandButton	(Name)	cmdOK
	Caption	OK
CommandButton	(Name)	cmdAb
	Caption	Abbrechen



Code im Formular

- ② Öffnen Sie das Codefenster, indem Sie im Formular auf die Schaltfläche **OK** doppelklicken.

```
Private Sub cmdOK_Click()
1  Dim RL As Worksheet
    Set RL = Sheets("Rechnungsliste")
    With Sheets("Rechnung")
4      If chkAdresse = True Then
        .Range("A1:A4").Clear
    End If
7      If chkPosi = True Then
        .Range("A12:E16").Clear
    End If
10     If chkReNr = True Then
        z = RL.Range("A3").CurrentRegion.Rows.Count
        .Range("F7").Value = RL.Cells(z, 1) + 1
13     End If
    End With
    Me.Hide
End Sub
```

Zeile 1-2 Erstellen der Objektvariablen **RL**.

Zeile 4-9 In den zwei Abfragen werden die Bereiche gelöscht, wenn die Check-Boxen aktiviert sind.

Zeile 10- 13 Wenn die **CheckBox** aktiviert wurde wird die letzte Zeile der Rechnungsliste ermittelt, dann die letzte Rechnungsnummer gelesen, diese um den Wert 1 erhöht und in die Rechnung eingetragen.

Zeile 15 Das Formular wird geschlossen.

- ③ Klicken Sie nun im Codefenster auf die Schaltfläche **Abbrechen** um diese Prozedur zu erzeugen. Danach geben Sie den Befehl *Me.Hide* ein.

```
Private Sub cmdAb_Click()
    Me.Hide
End Sub
```

Code im Modul

- ④ Im **Modul1** erstellen Sie manuell diese Routine

```
Sub Neue_Rechnung()
    frmNeueRe.Show
End Sub
```

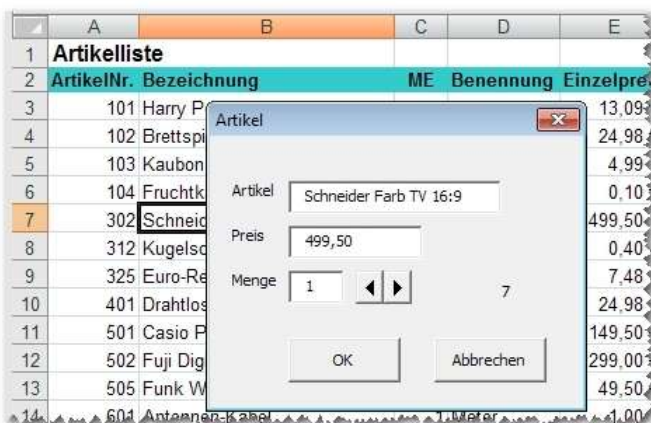
- ⑤ Erstellen Sie in der Tabelle **Rechnung** eine Schaltfläche *Neue Re*. Dieser weisen Sie das zuvor erstellte Makro **Neue_Rechnung** zu.



Artikel in Rechnung eintragen

Sie haben eine Artikelliste, aus dieser wollen Sie die Daten in Ihre Rechnung übernehmen. Nach einem Doppelklick auf den gewünschten Artikel öffnet das Programm ein Formular in dem Artikel und Preis angezeigt werden und die Anwender die Menge einstellen können, indem Sie auf die Schaltfläche mit den zwei Pfeilen klicken.

Themen:
Formular
If...Then
Cells
Sheet



Formular erstellen

① Erzeugen Sie das oben dargestellte Formular und vergeben Sie diese Eigenschaften

Steuerelement	Eigenschaft	Wert
UserForm	(Name)	frmArtikel
	Caption	Artikel
TextBox	(Name)	txtArtikel
TextBox	(Name)	txtPreis
TextBox	(Name)	txtMenge
Label	(Name)	Label1
	Caption	Artikel
Label	(Name)	Label2
	Caption	Artikel
Label	(Name)	Label3
	Caption	Menge
Label	(Name)	lblZeile
	Caption	Zeile
SpinButton	(Name)	spnMenge
	Max	100
CommandButton	(Name)	cmdOK
	Caption	OK
CommandButton	(Name)	cmdAbbreche
	Caption	Abbrechen

Code im Formular

② Öffnen Sie das Codefenster, indem Sie im Formular auf die Schaltfläche **OK** doppelklicken.

```
Private Sub cmdOK_Click()
1 Dim RE As Worksheet
  Set RE = Sheets("Rechnung")
  For z = 12 To 100
```



```

4      If RE.Cells(z, 1) = "" Then Exit For
      Next z
      x = lblZeile.Caption
7      With Sheets("Artikelliste")
          RE.Cells(z, 1).Formula = .Cells(x, 1) 'ArtNr
          RE.Cells(z, 2).Formula = txtArtikel 'Bezeichnung
10     RE.Cells(z, 3).Value = Val(txtMenge) 'Menge
          RE.Cells(z, 4).Formula = .Cells(x, 4) 'Benennung
          RE.Cells(z, 5).Value = .Cells(x, 5) 'Preis
13     End With
      Me.Hide
End Sub

```

Zeile 1-2 *Objektvariable RE erstellen.*

Zeile 3-5 *Suchen der nächsten freien Zeile in der Rechnung.*

Zeile 6 *Die in der Artikelliste gewählt Zeilennummer wird vom Label im Formular gelesen.*

Zeile 8-12 *Artikeldaten lesen und in die Rechnung eintragen.*

Zeile 14 *Das Formular schließen.*

③ Code für die Pfeile im Steuerelement *SpinButton* und für *Abbrechen* erstellen.

```

Private Sub spnMenge_SpinDown()
    txtMenge = Val(txtMenge) - 1
End Sub

Private Sub spnMenge_SpinUp()
    txtMenge = Val(txtMenge) + 1
End Sub

Private Sub cmdAbbrechen_Click()
    Me.Hide
End Sub

```

Code in der Tabelle

④ Erzeugen Sie in der **Tabelle Artikel** eine Routine *BeforeDoubleClick*, siehe Seite 44.

```

Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, _
    Cancel As Boolean)
1   Cancel = True
    z = Target.Row
    With frmArtikel
4       .txtArtikel = Cells(z, 2)
        .txtPreis = Format(Cells(z, 5), "0.00")
        .lblZeile = z
7       .Show
    End With
End Sub

```

Zeile 1 *Bearbeitungsmodus ausschalten*

Zeile 2-6 *Zeile ermitteln und Werte in das Formular eintragen.*

Zeile 7 *Formular Artikel anzeigen*



A		
Absolute Formeln.....	42	
Add-In		
deaktivieren	59	
erstellen	58	
löschen.....	59	
verwenden.....	58	
weitergeben.....	58	
Ändern der Markierung.....	45	
Ansichtauswahl	17	
Anzeigename	13	
Arrays	48	
ASCII-Code	31	
Aufzeichnen	5	
Aufzeichnung		
Absolut	8	
Relativ	8	
Aufzeichnung beenden.....	7, 8	
Auswahlliste bei		
Steuerelementen.....	20	
Auto-Vervollständigen.....	20	
B		
Befehle		
auswählen	12	
Beim Aktivieren.....	44	
Beim Öffnen	45	
Beim Schließen	45	
Beim Verlassen	44	
Beispiele		
Adresse in Rechnung	85	
Artikel in Rechnung.....	88	
Dateiliste.....	82	
Diagramm anpassen.....	78	
Eingabemaske.....	74	
Liste der Tabellenblätter	81	
Liter in Kilogramm	68	
Rechnung in Liste	84	
Rechnungsnummer	70	
Rechnungsnummer aus		
Liste.....	86	
Sortieren	72	
Spezialfilter.....	77	
Tabellen schützen	66	
Tabellenblätter		
sortieren.....	80	
Umfrage.....	71	
Zahlenformat	67	
Zeile einfügen	69	
Zellschutz	64	
Benutzerdefiniert		
Formulare	52	
Benutzerdefiniertes Kürzel .	9	
Bereiche.....	32	
Bereichsnamen.....	36	
Blattschutz	64	
C		
Caption	11	
Cells.....	35	
Choose	25	
Code		
anpassen	73	
Fenster	17	
Column	39	
Columns.....	39	
Const.....	49	
Count	39	
D		
Dateityp	5	
Datenfelder	48	
Daten-Format xlsx	5	
Datentypen.....	47	
Dateserial.....	29	
Datevalue	29	
Day	29	
Deklaration erzwingen.....	46	
Deklariieren	46	
Direktbereich	62	
testen	62	
Direktfenster	17	
Do...Loop.....	28	
Doppelklick	45	
E		
Eigenschaften	11	
Fenster.....	16	
Else-Abfrage.....	25	
Entwicklertools.....	5	
Entwurfsmodus beenden....	11	
Ereignisliste	17	
Ereignisprozeduren	44	
Excel-Optionen	12	
Exit Do	28	
F		
Fehlermeldung	63	
Fehlersuche.....	61	
Feiertage	93	
For...Next.....	27	
Formel anzeigen	91	
Formeln	42	
Formular		
anzeigen	53	
erstellen	52	
schließen	53	
Formulare	18	
Formularfenster	17	
Funktionen		
beschreiben	56	
erstellen	54	
verwenden	55	
G		
Großschreibung	31	
Gruppe		
Code.....	5	
Umbenennen	14	
Gruppen anlegen.....	14	
H		
Haltepunkte	17, 61	
Hilfe bei Programmierung ..	20	
Hilfe für VBA.....	60	
Hour	29	
I		
If...Then-Abfrage.....	25	
IIF-Abfrage	25	
Inhalt aktivieren.....	6	
InputBox.....	22	
K		
Kalenderwoche	92	
Kategorie		
erzeugen	57	
zuordnen.....	57	
Kegelberechnung.....	90	
Kennwort.....	65	
Kleinschreibung	31	
Kommentare	19	
Konstanten.....	49	
L		
LCase	31	
Left	30	
Len	30	
Lesezeichen	17, 61	
Lokalfenster.....	17, 62	
LTrim	30	
M		
Makro		



- aufrufen 9
 speichern 8
 speichern in 7
 zuweisen 10
 Makroaufzeichnung 7
 Makroname 7
 Makrorecorder 7
 Makros
 einfügen 15
 umbenennen 15
 Makros In 9
 Makrosicherheit 6
 Markierung verschieben 34
 Menüaufruf 9
 Menüleiste 14
 Mid 30
 Mid-Anweisung 30
 Minute 29
 Modul 18
 Month 29
 MsgBox 23
 mit Rückgabewert 24
 Multiplikation und
 Runden 90

N
 Name 11
 Neue Arbeitsmappe 8

O
 Objekte 18
 Objektkatalog 49
 Objektliste 17
 Offset 34
 Ohne Duplikate 77
 On Error 63
 Optionale Argumente 55
 Ostertermin 93

P
 Parameter 19
 Parameterinfo 21
 Persönliche
 Arbeitsmappe 8, 13
 Private 48, 50
 Programmcode
 drucken 60
 erstellen 54
 neu erstellen 18
 verändern 18
 Programmtest 61
 Projekt Explorer 16
 Public 48, 50

R
 Range Objekt 32
 Register
 Entwicklertools 5
 Register anlegen 14
 Registerposition
 verändern 15
 Relative Formeln 42
 Right 30
 Routine
 aufrufen 19
 automatisch starten 44
 Row 39
 Rows 39
 RTrim 30

S
 Schaltflächen 9, 67
 Ändern 12
 erzeugen 10
 Second 29
 Select...Case 26
 Sicherheitswarnung 6
 Spalten 33
 ausblenden 33
 einblenden 33
 markieren 33
 Spaltenbreite 33
 Speichern unter 5
 Speicherort 54
 Static 48, 50
 Stelle eines Zeichens 31
 Steuerelement
 ActiveX 11
 Eigenschaften 53
 einfügen 52
 formatieren 10
 Sub Prozedur 19
 Symbole erzeugen 83
 Syntaxüberprüfung 21
 Systemdatum 29

T
 Tabelle
 hinzufügen 40
 indizieren 38
 kopieren 41
 löschen 40
 umbenennen 40
 Tabellenfunktionen 43
 Tastenkombination 7, 9
 Text in Zahl 31
 Textfunktionen 30
 Trennzeichen 13
 Trim 30

U
 Überwachungsfenster ... 17, 62
 UCase 31
 Uhrzeit 29

V
 Variablen 42, 46
 Gültigkeit 50
 Lokale 50
 Öffentliche 50
 Private 50
 Static 50
 VBA-Editor 16
 Vertrauensstellungscenter 6

W
 Werkzeugsammlung 17
 Werte
 anzeigen 61
 in andere Mappen 38
 in andere Tabellen 37
 in Zellen 37
 With Anweisung 22

X
 xlam-Datei 58
 xlsm-Datei 58

Y
 Year 29

Z
 Zahl in Text 31
 Zahl in Worten 91
 Zeilen
 ausblenden 33
 einblenden 33
 Zeilenhöhe 33
 Zeilenumbruch 18
 Zellen 32