

---

## Zusammenfassung

Erste VBA-Schritte und die entsprechenden Kenntnisse werden anhand von einigen Beispielen, wie *Pascal'sches* Dreieck und Kreisberechnung, erklärt. Praktische Hinweise zum Erstellen von Makros und dem Debuggen von Codes werden gegeben.

---

## 2.1 Kopieren (relativ und absolut)

Wir zeigen zuerst erneut, wie wichtig es ist, in Excel Zellen richtig zu **kopieren**. Wir erinnern uns: Wenn man Zellen kopiert, hängt das Ergebnis sehr davon ab, ob wir Texte bzw. Zahlen kopieren oder ob es sich um Formeln bzw. Funktionen handelt. Im ersten Fall reproduzieren wir genau die Zellinhalte. Im zweiten Fall müssen wir vorsichtig sein, denn Excel verändert die Adressen der Zellen, in denen Formeln benutzt werden, d. h. die Zellbezüge werden beim Kopieren verändert, wenn man sie nicht schützt.

### Beispiel

Wir nehmen an, in der Zelle A3 steht die Formel  $=A1+A2$  und wir kopieren sie nach B3. Wir werden feststellen, dass in B3 eine Formel mit veränderten Bezügen stehen wird, nämlich  $=B1+B2$ . Das liegt daran, dass die Ausgangsformel nur **relative** Bezüge enthält, d. h. solche, die kein  $\$$ -Zeichen enthalten und sich beim Kopieren dem Ort der neuen Zelle anpassen.

Das Dollarzeichen unterscheidet einen relativen Bezug von einem absoluten. Damit sich der Bezug beim Kopieren nicht ändert, müssen wir  $\$$  vor die Zellbezüge setzen. D. h. die Formel in A3 muss lauten:  $=\$A1+\$A2$ . Wenn wir das nach A4 kopieren, ergibt sich dort  $=\$A2+\$A3$ , also ebenfalls eine Änderung der Bezüge! Grund: Der Schutz mit nur einem  $\$$ -Zeichen vor den Buchstaben ist nur beim Horizontalkopieren wirksam. Um die Formel wirklich zu schützen, ist es notwendig, jeden Buchstaben zwischen zwei  $\$$ -Zeichen

	A	B	C	D	E	F	G
1	1						
2	1	1					
3	1	2	1				
4	1	3	3	1			
5	1	4	6	4	1		
6	1	5	10	10	5	1	
7	1	6	15	20	15	6	1
8							
9							
10							
11							
12							

**Abb. 2.1** *Pascal'sches Dreieck* – Form 1 [Arbeitsmappe Pascal.xlsm; Blatt: Pascal 1]

zu stellen! Also muss in A3 stehen  $=\$A\$1+\$A\$2$ . Jetzt haben wir **absolute** Bezüge, und jedwedes Kopieren lässt die Bezüge unverändert.

Das *Pascal'sche* Dreieck gibt uns eine gute Gelegenheit, das Kopieren von Formeln zu üben. Die Absicht ist, dieses berühmte Dreieck von ganzen Zahlen später fast automatisch in einem Excel-Blatt aufzubauen. Wir betrachten zwei Dreiecksformen.

Für die erste Dreiecksform (vgl. Abb. 2.1) belegen wir die Zellen A1:A7 und B2 mit einer 1. In B3 schreiben wir die Formel  $=A2+B2$ , die wir bis B7 kopieren. Danach werden die Zellen B2:B7 nach rechts bis G2:G7 kopiert. Die entstehenden Nullen löschen wir anschließend "von Hand" mit *Entf*. Später werden wir sehen, wie man die Nullen mit einem einfachen Befehl verbirgt.

- Um eine Formel von Hand zu kopieren, zeigen wir mit dem Cursor auf das Ausfüllkästchen in der rechten unteren Ecke der Zelle, in der sich die Formel befindet. Der Cursor verwandelt sich in ein  $+$  – Zeichen. Mit gedrückter linker Maustaste zieht man das Kästchen bis zur Zielzelle.

Das *Pascal'sche* Dreieck wird von den binomischen Zahlen gebildet:

$$\begin{aligned}
 \binom{0}{0} &= 1 \\
 \binom{1}{0} &= 1; \quad \binom{1}{1} = 1 \\
 \binom{2}{0} &= 1; \quad \binom{2}{1} = 2; \quad \binom{2}{2} = 1 \\
 \binom{3}{0} &= 1; \quad \binom{3}{1} = 3; \quad \binom{3}{2} = 3; \quad \binom{3}{3} = 1
 \end{aligned}$$

.....

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1											1								
2										1		1							
3					0	0	0	0	1	0	2	0	1	0	0	0	0		
4					0	0	0	1	0	3	0	3	0	1	0	0	0		
5					0	0	1	0	4	0	6	0	4	0	1	0	0		
6					0	1	0	5	0	10	0	10	0	5	0	1	0		
7					1	0	6	0	15	0	20	0	15	0	6	0	1		
8																			

**Abb. 2.2** *Pascal'sches Dreieck – Form 2* [Arbeitsmappe Pascal.xlsm; Blatt: Pascal 2]

Die Summen der Zahlen in diagonal liegenden Zellen, z. B. der hellgrau-bzw. dunkelgrau gefärbten, sind die *Fibonacci*-Zahlen:  $1 + 4 + 3 = 8$ ;  $1 + 5 + 6 + 1 = 13$ . ... (Wir werden dieses Thema im Abschn. 3.2.2 ausführlich behandeln).

Die Summe der binomischen Zahlen in einer Zeile ist eine Potenz von 2:

$$2^0, 2^1, 2^2, 2^3 \dots$$

In der Abb. 2.2 sehen wir eine andere Form des Dreiecks, die in China schon 1300 v. Chr. auftauchte. Hier wird eine 1 in K1, J2 und L2 eingetragen. In K3 steht die Summe  $=J2+L2$ . Kopiere diesen Ausdruck mit dem Ausfüllkästchen nach links (bis E3) und nach rechts (bis Q3). Anschließend kopieren wir den Inhalt von E3:Q3 mit dem Ausfüllkästchen in alle Zellen bis E7:Q7.

Schließlich können wir alle Nullen verbergen. Wähle dazu alle Zellen von E1 bis Q7 (bei gedrückter *Shift*-Taste und Linksklick in Q7) und suche in *START* das Menü *Zellen*. Hier *Format > Zellen formatieren > Zahlen > Benutzerdefiniert*. Unter Typ tragen wir ein: **0;;; –** alle Nullen verschwinden, wenn Sie *OK* drücken (Siehe Abb. 2.3).

Das Ergebnis nach Drücken auf *OK* sehen Sie in Abb. 2.4. Die benutzerdefinierten Zellenformate in Excel gehorchen folgender Syntax:

<Format für positive Zahlen>; <Format für negative Zahlen>; <Format für Null>; <Format für Text>

### Beispiele

- 0;;; →** zeige nur eine Ziffer für positive Zahlen, verberge negative Zahlen, Nullen und Texte.
- ;-0;; →** zeige nur eine Ziffer für negative Zahlen, verberge positive Zahlen, Nullen und Texte.
- 0;-0;; →** zeige nur eine Ziffer für positive und negative Zahlen, verberge Nullen und Texte.
- ;;; →** verberge alle Daten.

War doch alles einsichtig, nicht wahr?

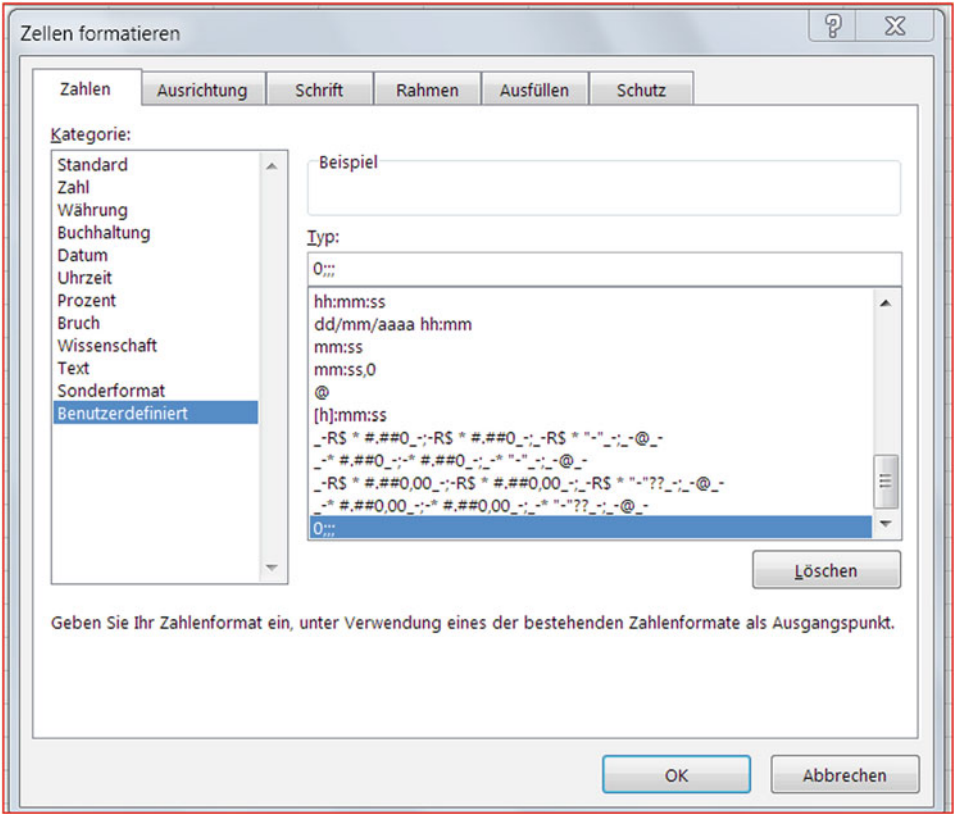
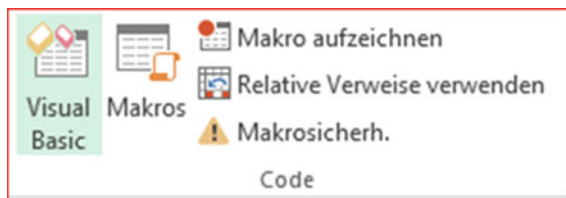


Abb. 2.3 Benutzerdefinierte Zahlenformate

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1											1								
2										1		1							
3									1		2		1						
4								1		3		3		1					
5							1		4		6		4		1				
6						1		5		10		10		5		1			
7					1		6		15		20		15		6		1		
8																			
9																			

Abb. 2.4 *Pascal'sches* Dreieck "Nullen bereinigt" [Arbeitsmappe Pascal.xlsm; Blatt: Pascal 2]

**Abb. 2.5** Code-Gruppe

## 2.2 Makrorekorder

Jetzt werden wir den Kopiervorgang für die Formel  $=J2+L2$  mithilfe eines **Makros** automatisieren.

► Ein **Makro** ist ein automatisierter Prozess, der das wiederholte manuelle Ausführen bestimmter Kommandos unnötig macht.

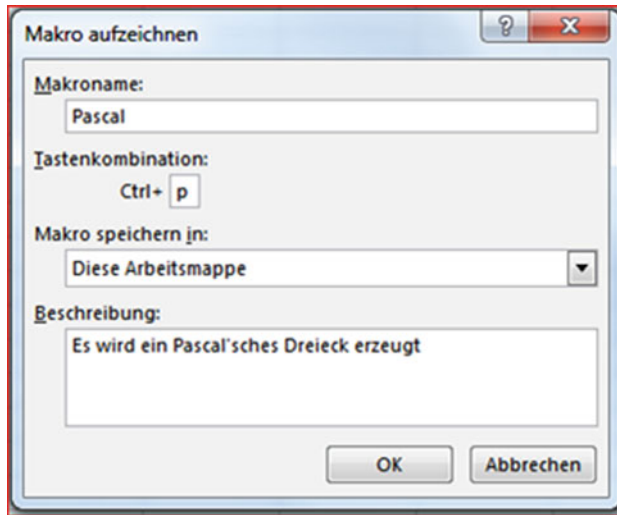
Für die Aufzeichnung eines Makros brauchen wir die Registerkarte *ENTWICKLER-TOOLS*.

► Wenn die Registerkarte *ENTWICKLER-TOOLS* nicht angezeigt wird, gehen Sie folgendermaßen vor:

- Klicken Sie auf die Registerkarte *DATEI*.
- Klicken Sie auf *Optionen* und dann auf *Menüband anpassen*.
- Aktivieren Sie in der Kategorie *Menüband anpassen* in der Liste *Hauptregisterkarten* das Kontrollkästchen *Entwicklertools*, und klicken Sie dann auf *OK*.

Folgen Sie jetzt den folgenden Anweisungen, um das Makro zu erzeugen:

1. Schreibe 1 in K1, J2 und L2, in K3  $=J2+L2$ .
2. Klicke in *ENTWICKLER-TOOLS* > *Code* > *Makro aufzeichnen* an (vgl. Abb. 2.5). In dem Dialogfenster *Makro aufzeichnen* (siehe Abb. 2.6), geben Sie dem Makro einen Namen, z. B. "Pascal". Für die Tastenkombination können Sie z. B. "p" wählen. Diese Tastenkombination setzt eine gleichnamige in Excel standardmäßig vorhandene Tastenkombination außer Kraft, solange das Makro geöffnet ist.
3. Nachdem *OK* angeklickt wurde, beginnt die Aufzeichnung des Makros (die Schaltfläche *Makro aufzeichnen* verwandelt sich in ein blaues Quadrat). Alle Schritte, die wir jetzt tun, werden registriert: wir kopieren K3 mit dem Ausfüllkästchen nach links (bis E3) und nach rechts (bis Q3). Anschließend kopieren wir den Inhalt von E3:Q3 mit dem Ausfüllkästchen in alle Zellen bis E7:Q7.
4. Klicke in die blaue Schaltfläche, um die Aufzeichnung des Makros zu beenden.



**Abb. 2.6** Dialogfenster *Makro aufzeichnen*

Um das Makro auszuprobieren, löschen wir alle Zellen und starten das Makro mit *Strg-p*. Das *Pascal'sche* Dreieck von Abb. 2.4 wird schlagartig ausgefüllt.

Makros müssen in **Excel 2013** im Dateiformat *Excel-Arbeitsmappe mit Makros (\*.xlsm)* gespeichert werden. Deswegen speichern wir die Arbeitsmappe als "Pascal.xlsm".

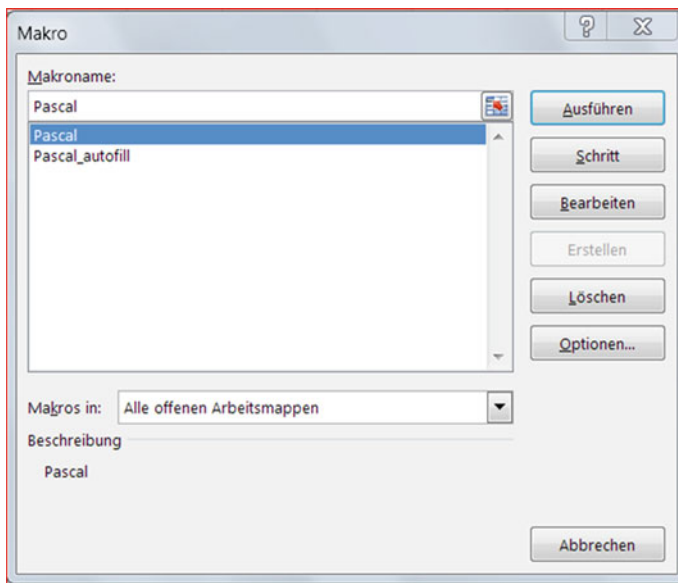
Um Fehler zu vermeiden, muss man genau wissen, welche Schritte man während der Aufzeichnung ausführen will. Bei einem Fehler ist es besser, von vorne zu beginnen, als zu versuchen, den Fehler zu korrigieren.

Wenn Sie den erzeugten Code sehen wollen, gehen Sie in den *ENTWICKLERTOOLS* zum *Code-Menü* und wählen *Makros anzeigen*. Es erscheint das Dialogfenster *Makro* aus Abb. 2.7.

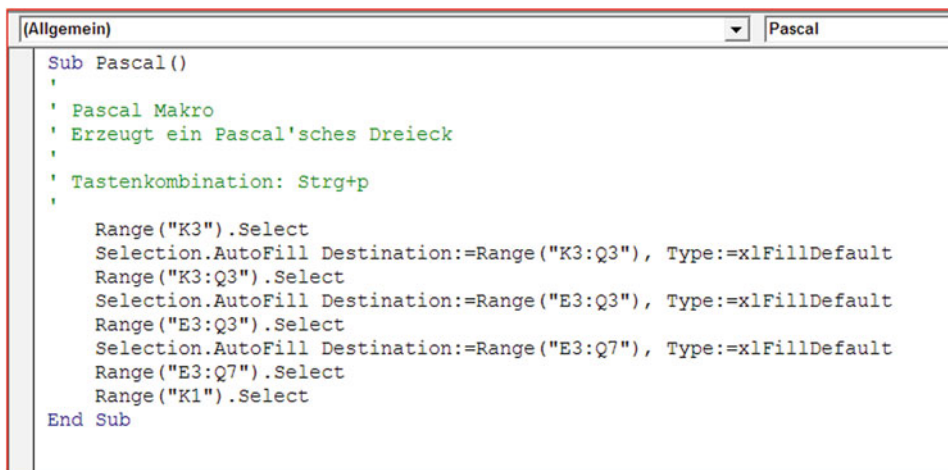
Hier kann man ein Makro *löschen* oder auch *bearbeiten*. *Bearbeiten* liefert den erzeugten Code (siehe Abb. 2.8).

Der Makrorekorder erzeugt eine **Subroutine**, daher sehen wir am Anfang die Bezeichnung *Sub*. Jede Subroutine muss mit *End Sub* abgeschlossen werden. Was wir sehen, ist ein Programm mit dem Namen "Pascal" in der Sprache **VBA** (Visual Basic for Applications). VBA ist eine Programmiersprache, die in dem Office-Paket enthalten ist – und die in Zukunft für uns von großer Bedeutung sein wird.

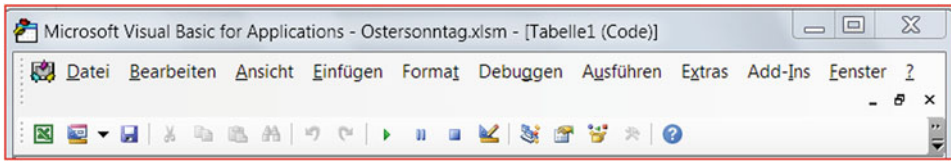
- Der **VBA-Editor** von **Excel 2013** ist praktisch identisch mit dem der früheren Versionen. Er kann durch *ENTWICKLERTOOLS > Visual Basic* oder mit *ALT-F11* geöffnet werden. Makros können auch in der Registerkarte *ANSICHT* aufgezeichnet und angezeigt werden.



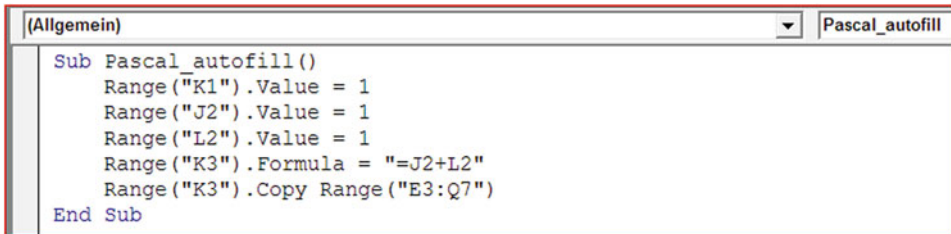
**Abb. 2.7** Dialogfenster *Makro*



**Abb. 2.8** Code für *Pascal'sches Dreieck* [Arbeitsmappe: Pascal.xlsm; Makro: Pascal]



**Abb. 2.9** Das Menüband des VBA-Editors



**Abb. 2.10** Code für Pascal\_autofill [Arbeitsmappe: Pascal.xlsm; Makro: Pascal\_autofill]

## 2.3 Weitere Beispiele für VBA

Wir wollen uns noch einige VBA-Beispiele anschauen.

### Beispiel 1

Im ersten Beispiel erzeugen wir wieder ein *Pascal'sches*-Dreieck, aber dieses Mal ohne zuerst die Startwerte in die Tabelle zu schreiben. Um den Code zu schreiben, benutzen wir den VBA-Editor (*ALT-F11*), dessen Menüband in Abb. 2.9 zu sehen ist.

Nachdem der VBA-Editor aufgerufen wurde (*ALT-F11*), wird *Einfügen > Modul* gewählt. Schreibe den Code in das Codefenster (vgl. Abb. 2.10). Auch mit *ALT-F11* kommt man wieder in die Excel-Tabelle.

Nun löschen wir alle Zellen des Arbeitsblatts "Pascal 2" und führen den Code aus: *ALT-F8 > pascal\_autofill > Ausführen*. Das Programm erzeugt nun das ganze Dreieck.

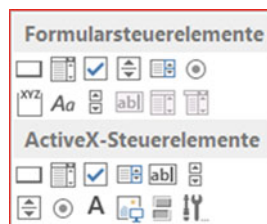
Wenn man ein Makro zum ersten Mal lädt, erscheint aus Sicherheitsgründen der Hinweis, dass die Makros deaktiviert wurden, auch Ihr eigenes! Wenn man kein Risiko sieht, kann man auf die Schaltfläche *Inhalt aktivieren* klicken.

### Beispiel 2

Zu den *ENTWICKLERTOOLS* gehören auch *Steuerelemente* (z. B. Schaltknöpfe), von denen es zwei Sorten gibt: *ActiveX-Steuerelemente*<sup>1</sup> (leistungsfähige VBA-Objekte) und

<sup>1</sup> **Anmerkung:** Am 9.12.2014 hat Microsoft Security-Updates für Office installiert. Diese Updates führten dazu, dass keine *ActiveX-Steuerelemente* mehr funktionierten bzw. keine neue eingefügt werden können. *Formular-Steuerelemente* sind davon nicht betroffen. Inzwischen hat Microsoft den *KB-Beitrag 3025036* veröffentlicht, der durch ein herunterladbares *FixIt*, das Problem löst. Wir haben diese Prozedur am 28.01.2015 mit Erfolg durchgeführt.



**Abb. 2.11** Steuerelemente

*Formularsteuerelemente* (einfache Funktionalitäten, die an Makros geknüpft werden). Man findet diese Elemente in *ENTWICKLERTOOLS > Einfügen > Steuerelemente einfügen*.

Wir wollen zunächst ein *ActiveX-Steuerelement* verwenden, und zwar einen *CommandButton*. Dafür wählen wir in *Steuerelemente > Einfügen* die *Befehlsschaltfläche*, erstes Element in der Liste in Abb. 2.11.

Der Cursor verwandelt sich in ein **+**-Zeichen, mit dem wir den Button auf das Arbeitsblatt zeichnen. Dann klicken wir auf *Code anzeigen*. Der Visual Basic-Editor wird geöffnet (es ist vielleicht einfacher mit der rechten Maustaste auf den gezeichneten Button zu klicken und *Code anzeigen* auswählen).

Wir tragen nur eine Codezeile ein: `Range("K3").Copy Range("E3:Q7")` (vgl. Abb. 2.12).

Anschließend, noch im Visual Basic-Editor, in der Registerkarte *Datei* auf *Schließen und zurück zu Microsoft Excel* klicken. Zurück in Excel, deaktivieren wir durch Anklicken den *Entwurfsmodus*.

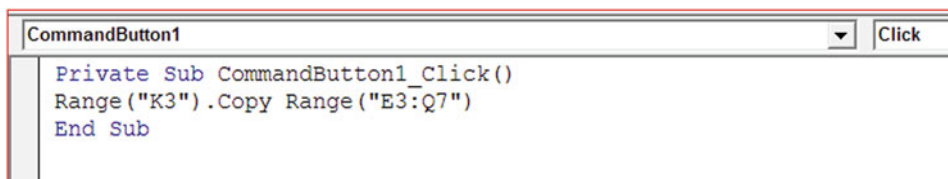
Wieder tragen wir in K1, J2 und L2 eine 1 ein und in K3 die Formel `=J2+L2`.

Den Buttontext "CommandButton1" kann man – wieder im *Entwurfsmodus* – ändern (Rechtsklick auf den Button und *Eigenschaften* aufrufen). Unter *Caption* kann man dann einen anderen Button-Text schreiben (auch der Font kann geändert werden). Hier haben wir "Pascal'sches Dreieck" als *Caption* gewählt.

Nach Anklicken des Buttons "Pascal'sches Dreieck" erhalten wir das schon gut bekannte Dreieck.

### Beispiel 3

Wenn wir alle Zellen  $> 0$  gelb anfärben wollen, so erreichen wir das mit dem VBA-Code aus Abb. 2.13.

**Abb. 2.12** Code für "CommandButton1"



Excel und VBA

Einführung mit praktischen Anwendungen in den  
Naturwissenschaften

Mehr, F.J.; Mehr, M.T.

2015, XIII, 375 S. 329 Abb., Softcover

ISBN: 978-3-658-08885-9