Schleifen verstehen

Schleifen zu verstehen ist für einen Programmierneuling eine echte Herausforderung.
Deshalb sollte dieses Dokument dafür dienen in möglichst kleinen Schritten die verschieden Möglichkeiten der Schleifen-Programmierung zu erläutern.

1 Die FOR-NEXT-Schleife

Sehen wir uns folgendes Programm an, welches die Aufgabe hat in den Zeilen 1 bis 10 der Spalte A das Quadrat der jeweiligen Zeile zu schreiben. Die Prozedur ohne Inhalt sieht so aus

```
SUB SchleifenBeispiel 01

End Sub
```

Um in eine Zelle eine Zahl zu schreiben, benutzen wir den Cells-Befehl.
Die grundsätzliche Schreibweise des Cells- Befehl lautet:

Cells(<Zeilen-Nr.>, <Spalten-Bezeichnung als Nummer oder Buchstabe>) = < Wert>

Beispiel:

```
Cells(3, 1) = 12          In die Zelle A3 wird die Zahl 12 geschrieben
Cells(3, "C") = 23        In die Zelle C3 wird die Zahl 23 geschrieben
```

Ohne Schleifenkennnisse würde dann unser Programm wie folgt aussehen:

```
SUB SchleifenBeispiel 01
    Cells(1, 1 ) = 1 * 1
    Cells(2, 1 ) = 2 * 2
    Cells(3, 1 ) = 3 * 3
    Cells(4, 1 ) = 4 * 4
    Cells(5, 1 ) = 5 * 5
    Cells(6, 1 ) = 6 * 6
    Cells(7, 1 ) = 7 * 7
    Cells(8, 1 ) = 8 * 8
    Cells(9, 1 ) = 9 * 9
    Cells(10, 1 ) = 10 * 10
End Sub
```

Nun kommt eine weitere Programmier-Option zum Zuge, die Variable.

Wie der Name schon sagt, ist eine Variable ein Programmierelement in Form einer Folge von Buchstaben, Zahlen und Sonderzeichen, dem ein Wert in Form einer Zahl, eines Text, eines Wahrheitswertes und vieles mehr zuordnen kann.

Es müssen die Regeln für einen Variablennamen beachtet werden, die da heißen:

- Das **erste Zeichen** muss **immer ein Buchstabe** sein (Klein- oder Groß-Buchstabe)
- Alles weiteren Zeichen können Buchstaben, Zahlen oder der Unterstrich sein.
- **Sonderzeichen** dürfen **nicht** verwendet werden, mit Ausnahme des Unterstrichs
- Ebenso ist das **Leerzeichen** in einem Variablennamen absolut tabu
- Deutsche Umlaute sind zwar möglich, können aber Probleme bei unterschiedlichen Systeme (Windows / Apple-Betriebs-System) hervorrufen
- Für einen Variablennamen können maximal **255 Zeichen** verwendet werden
- Der Name eines Variable sollte möglichst sinnvoll und sprechend sein
  Wenn zum Bespiel eine Excel-Zeile über eine Variable im Cells-Befehl angesprochen werden soll, verwenden wir als Variablenname z.B. Zeile:
  Cells(Zeile,1) = 12
  je nach Variablenwert wird eine 12 in die entsprechende Excel-Zelle geschrieben
  Es können natürlich auch Variablennamen wie: A, B, I, Erwin oder Paula benutzt werden.
  Diese sind aber nicht unbedingt sprechend.

Für unser kleines Programm benötigen wir die Variable Zeile und wollen unser Beispiel erst einmal wie folgt verschlimmbessern:

```
SUB SchleifenBeispiel_01
    Zeile = 1
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 2
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 3
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 4
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 5
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 6
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 7
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 8
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 9
    Cells(Zeile, 1 ) = Zeile * Zeile
    Zeile = 10
    Cells(Zeile, 1 ) = Zeile * Zeile
End Sub
```

Auffällig ist nun, dass jeder Cells-Befehl gleich aussieht:

```
Cells(Zeile, 1 ) = Zeile * Zeile
```

Die Idee ist nun den Inhalt der Variable Zeile mittels eines Programmier-Elementes zu steuern:

Diese geschieht z.B. mir der FOR-NEXT-Anweisung, welches die Mutter aller Programmier-Schleifen ist.

Schauen wir uns die Schreibweise dieses Schleifentyps einmal an

Schreibweise einfache Form

`For <Schleifenvariable> = <Anfangszahl> To <Endzahl>`
`    <Weitere Programmier-Befehle>`
`Next <Schleifenvariable>`

Unser Programm würde nun wie folgt aussehen:

```
   SUB SchleifenBeispiel 01
1      For Zeile = 1 To 10
2          Cells(Zeile, 1 ) = Zeile * Zeile
3      Next Zeile
   End Sub
```

Pos	Beschreibung
1	Es wird festgelegt mit welchen Zahlen die Variable Zeile belegt werden soll. In diesem Fall also von 1 bis 10
2	Ausgabe der Zellinformation
3	In dieser Zeile wird die Variable Zeile um 1 erhöht und anschließend überprüft ob die Variable den Wert 10 überschritten hat. Ist dies der Fall wird die Schleife verlassen

Die For-Next-Schleife kann auch komplexer angewendet werden

Schreibweise komplexe Form

`For <Schleifenvariable> = <Anfangszahl> To <Endzahl> Step <Schrittweite>`
`  <Weitere Programmier-Befehle>`
`  If <Bedingung> Then Exit For`
`Next <Schleifenvariable>`

2 Die DO-LOOP-Schleife

Eine weitere Möglichkeit unter VBA eine Schleife zu programmieren ist die DO-LOOP-Schleife, welche in verschiedenen Ausführungen zur Verfügung steht.

Im Allgemeinen besteht diese Schleife aus 3 Bestandteilen:

- Den Schleifenkörper
- Eine Bedingung, ob die Schleife aufrechterhalten oder verlassen werden soll.
- Etwas, was zur Änderung der Bedingung führt, z.B. Änderung einer Variable durch Hochzählung.

Schreibweisen der DO-LOOP-Schleife:

Variante	Schreibweise	Beispiel

1	`<Variable> = 0`
	`Do`
	`  <Weiterer Programm-Code>`
	`Loop Until <Bedingung>`
		`Zeile = 0`
		`DO`
		`  Zeile = Zeile + 1`
		`  Cells(Zeile, 1) = Zeile * Zeile`
		`Loop Until` `Zeile = 10`

2	`<Variable> = 0`
	`Do`
	`  <Weiterer Programm-Code>`
	`Loop While <Bedingung>`
		`Zeile = 0`
		`DO`
		`  Zeile = Zeile + 1`
		`  Cells(Zeile, 1) = Zeile * Zeile`
		`Loop While` `Zeile < 10`

3	`<Variable> = 0`
	`Do Until <Bedingung>`
	`  <Weiterer Programm-Code>`
	`Loop`
		`Zeile = 0`
		`DO Until` `Zeile = 10`
		`  Zeile = Zeile + 1`
		`  Cells(Zeile, 1) = Zeile * Zeile`
		`Loop`

4
HAW-Variante
	`<Variable> = 0`
	`Do While <Bedingung>`
	`  <Weiterer Programm-Code>`
	`Loop`
		`Zeile = 0`
		`DO While` `Zeile < 10`
		`  Zeile = Zeile + 1`
		`  Cells(Zeile, 1) = Zeile * Zeile`
		`Loop`

5	`<Variable> = 0`
	`Do`
	`  <Weiterer Programm-Code>`
	`  If <Bedingung> Then Exit DO`
	`Loop`
		`Zeile = 0`
		`DO`
		`  Zeile = Zeile + 1`
		`  Cells(Zeile, 1) = Zeile * Zeile`
		`If` `Zeile = 10` **`Then Exit Do`**
		`Loop`

3 Die WHILE-WEND-Schleife

Ein weiterer Schleifen-Typ ist die WHILE-WEND-Schleife

Schreibweise:

Schreibweise	Beispiel

`<Variable> = 0`
`While <Bedingung>`
`  <Weiterer Programm-Code>`
`Wend`
	`Zeile = 0`
	`While Zeile < 10`
	`  Zeile = Zeile + 1`
	`  Cells(Zeile, 1) = Zeile * Zeile`
	`Wend`

Es ist sicher unschwer zu erkennen, dass die Arbeitsweise der WHILE-WEND-Schleife der DO-LOOP-Schleife sehr ähnlich ist.

4 Die FOR-EACH-Schleife

Und zu guter Letzt die FOR-EACH-Schleife

Schreibweise:

Schreibweise	Beispiel

`Dim <Unterobject> As Object`
 	
`For Each <Unterobject> In <Object>`
`  <Weiterer Programm-Code>`
`Wend`
	`Dim Zelle As Object`
	
	`For Each Zelle In Range("A1:A10")`
	`  Zelle = Zelle.Row ^ 2`
	`Next Zelle`

Bei disem Schleifentyp wird hauptsächlich mit Objekten gearbeitet, und zwar in der Form, dass alle Unterobjekte eines Objektes durchlaufen werden. Zum Beispiel besteht ein Zellbereich (das Objekt) aus einzelenen Zellen (Unterobjekte). Im obren Beispiel besteht der Bereich Range(“A1:A10“) aus den einzelene Zellen A1, A2, … bis A10. Diese Zellen werden in der FOR-EACH-Schleife einzelnd angesprochen. In den nachfolgendesen Beispielen werden wir noch mal auf diesen Schleifentyp zurückkommen und auch den Vorteil ggenüber den anderen Schleifentypen bei speziellen Problemen erkennen.

5 Zusammenfassung aller Schleifentypen

Schleife	Schreibweise	Beispiel
FOR NEXT Einfachste Form	<Var> = 0 For <Var> = <Anfg> To <Ende> <Weiterer Programm-Code> Next <Var>	**For** Zeile = 1 **To** 10 Cells(Zeile, 1) = Zeile * Zeile **Next** Zeile

For Next
 	
Komplexe Form
	<Var> = 0
	For <Var> = <Anfg> To <Ende> Step <Schritt>
	  <Weiterer Programm-Code>
	  If <Bedingung> Exit For
	Next <Var>
		For Zeile = 1 **To** 100 **Step** 2
		  Cells(Zeile, 1) = Zeile * Zeile
		  If Zeile * Zeile > 1000 **Then**
		  Exit For
		  End If
		Next Zeile

DO-LOOP Variante 1	<Variable> = 0
	Do
	  <Weiterer Programm-Code>
	Loop Until <Bedingung>
		Zeile = 0
		DO
		  Zeile = Zeile + 1
		  Cells(Zeile, 1) = Zeile * Zeile
		Loop Until Zeile = 10

DO-LOOP Variante 2	<Variable> = 0
	Do
	  <Weiterer Programm-Code>
	Loop While <Bedingung>
		Zeile = 0
		DO
		  Zeile = Zeile + 1
		  Cells(Zeile, 1) = Zeile * Zeile
		Loop While Zeile < 10

DO-LOOP Variante 3	<Variable> = 0
	Do Until <Bedingung>
	  <Weiterer Programm-Code>
	Loop
		Zeile = 0
		DO Until Zeile = 10
		  Zeile = Zeile + 1
		  Cells(Zeile, 1) = Zeile * Zeile
		Loop

DO-LOOP Variante 4 **HAW-Variante**	<Variable> = 0
	Do While <Bedingung>
	  <Weiterer Programm-Code>
	Loop
		Zeile = 0
		DO While Zeile < 10
		  Zeile = Zeile + 1
		  Cells(Zeile, 1) = Zeile * Zeile
		Loop

DO-LOOP Variante 5	<Variable> = 0
	Do
	  <Weiterer Programm-Code>
	  If <Bedingung> Then Exit DO
	Loop
		Zeile = 0
		DO
		  Zeile = Zeile + 1
		  Cells(Zeile, 1) = Zeile * Zeile
		  If Zeile = 10 **Then Exit Do**
		Loop

While-Wend	<Variable> = 0
	While <Bedingung>
	  <Weiterer Programm-Code>
	Wend
		Zeile = 0
		While Zeile < 10
		  Zeile = Zeile + 1
		  Cells(Zeile, 1) = Zeile * Zeile
		Wend

EACH-FOR	Dim <Unterobject> As Object For Each <Unterobject> In <Object> <Weiterer Programm-Code> Wend	Dim Zelle As Object **For Each** Zelle **In** Range("A1:A10") Zelle = Zelle.Row ^ 2 **Next Zelle**

6 Beispiele.

Nachfolgende sollen die verschiedenen Schleifentypen anhand von diversen Problemstellungen erläutert werden.

6.1 Aufgabe 1: Quadratische Reihe

Es sollen die Werte der Zellen A6 bis A20 mit dem Faktorwert in der Zellen B3 multipliziert werden:

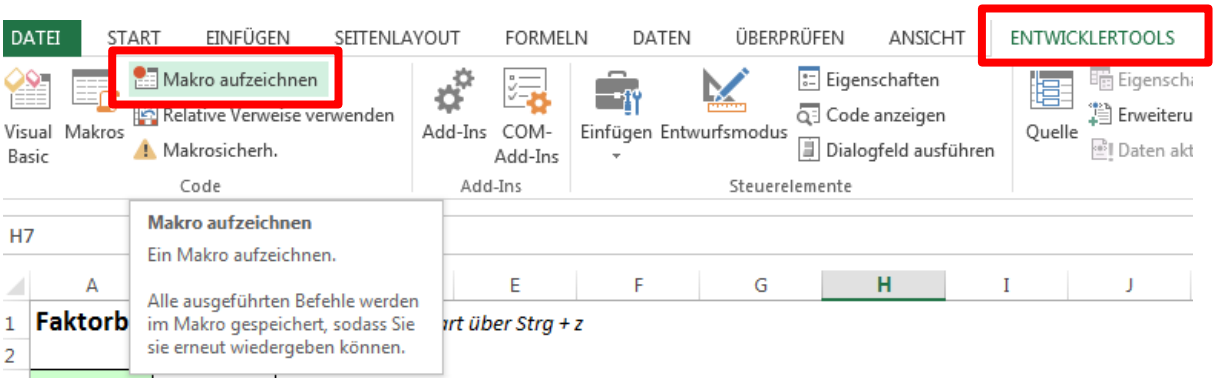
	A	B	C	D	E
1	**Faktorberechnung**			*Prozedurstart über Strg + z*	
2					
3	Faktor:	23			
4					
5	Zahl	Eregbnis			
6	161				
7	188				
8	177				
9	159				
10	158				
11	198				
12	181				
13	122				
14	135				
15	155				
16	124				
17	152				
18	112				
19	138				
20	199				

Prozedurstart: Über die Tastenkombination STRG + z
Prozedurname: FaktorMultiplikation

Ersteellen der Prozedur.

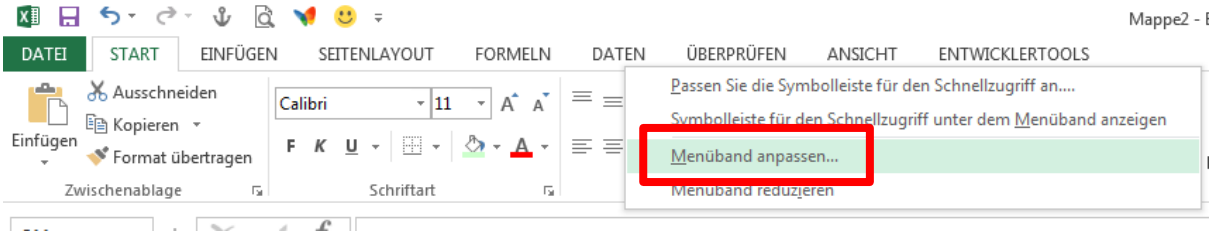
Für diese Aufgabenstellung werden wir eine leere Prozedor-Hülle mit Hilfe des Makrorekorder erstellen. Der Makrorekorder kann entweder über die Schaltfläche unten links der Excel-Oberfläche gestartet werden:

oder im Menüband “ENTWICKLERTOOLS“ über die Schaltfläche “Makro aufzeichnen“



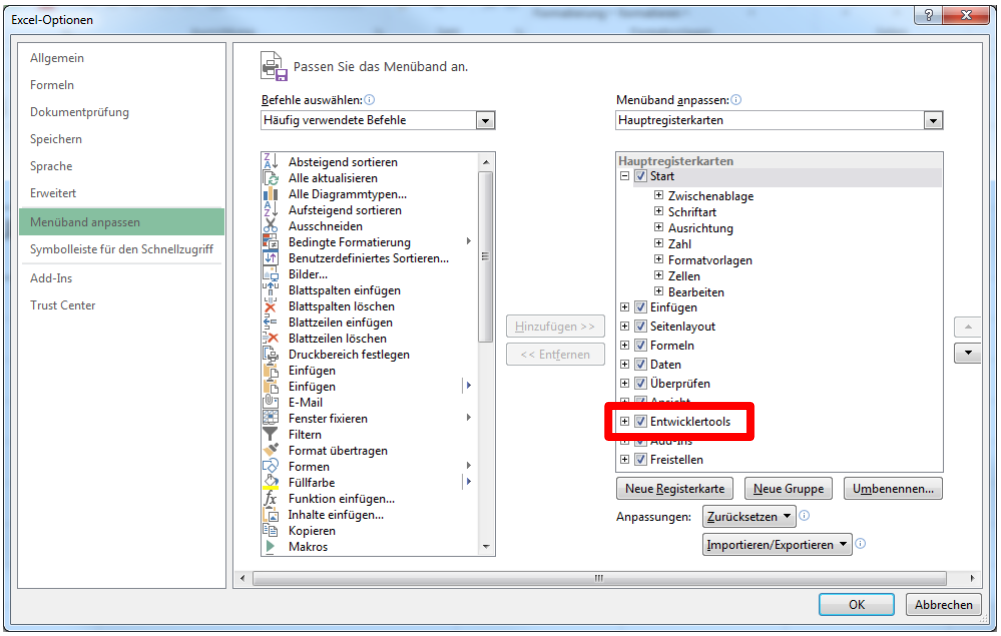
Sollte das Menüband “ENTWICKLERTOOLS“ nicht sichtbar sein, kann es mit folgenden Schritten sichtbar gemacht werden:

Klicken Sie mit der rechten Maustaste auf einen der Menüleisten-Titel z.B. “Start“und wählen Sie aus dem sichtbaren Kontextmenü die Option „“Menüband anpassen“

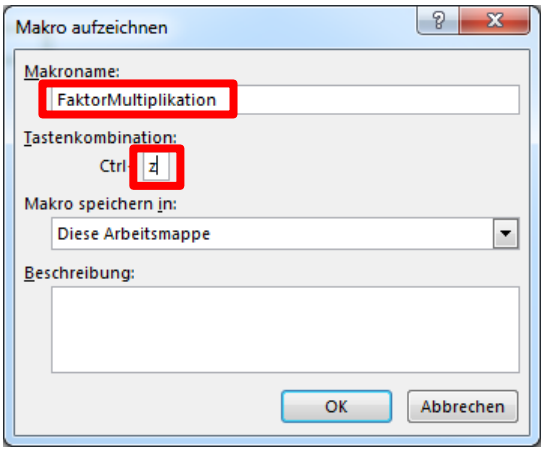


Diese Vorgehenweise gilt nur für Excel 2010, 2013 und 2016 !

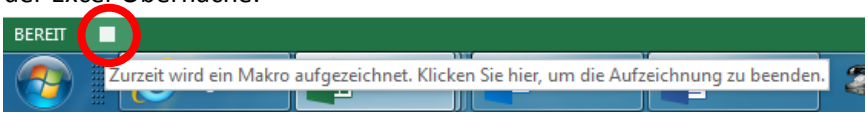
Dort aktivieren Sie das Häkchen bei “Entwicklertools“



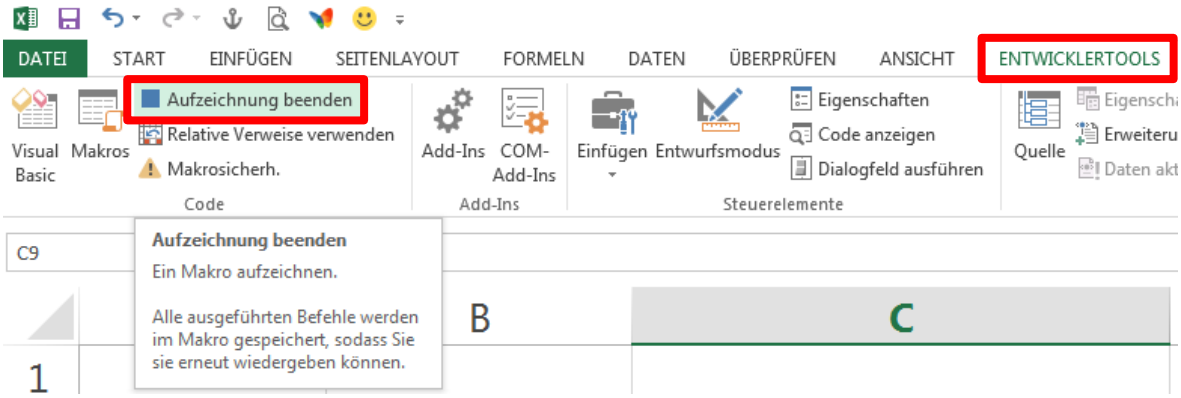
Nachdem wir den Makrorekorder erfolgreich gestartet haben, geben wir im vorhandenen Fenster folgenden Werte ein und schließen das Fenster mit “OK“.



Es ist nun eine leere Prozedur erstellt worden. Bevor wir uns diese anschauen ist es wichtig den Makrorekorder wieder abzuschalten. Dieses geschieht entweder über die Schaltfläche unten links in der Excel-Oberfläche:

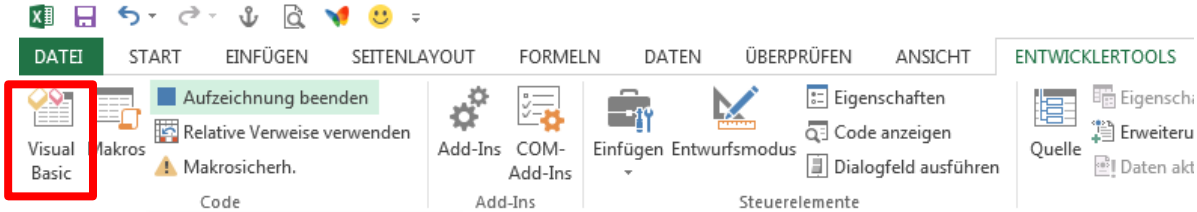


Oder im Menüband „“ENTWICKLERTOOLS“ über die Schaltfläche “Aufzeichnung beenden“



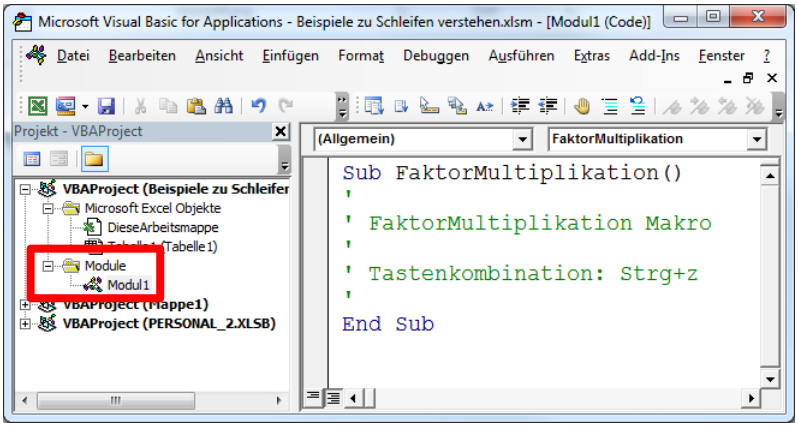
Die durch den Makrorecorder aufgezeichnete Prozedur kann nun wie folgt angeschaut werden:

Aktivieren des VBA-Editors im Menüband „“ENTWICKLERTOOLS“ über die Schaltfläche “Visual Basic“:



Oder über die Tastenkombination Alt + F11.

Im VBA-Editor klicken wir auf das Pluszeichen vor „Module“ (falls dieses Element noch nicht aufgeklappt ist) und aktivieren per Doppelklick auf “Modul1“ die dort befindliche Prozedur:

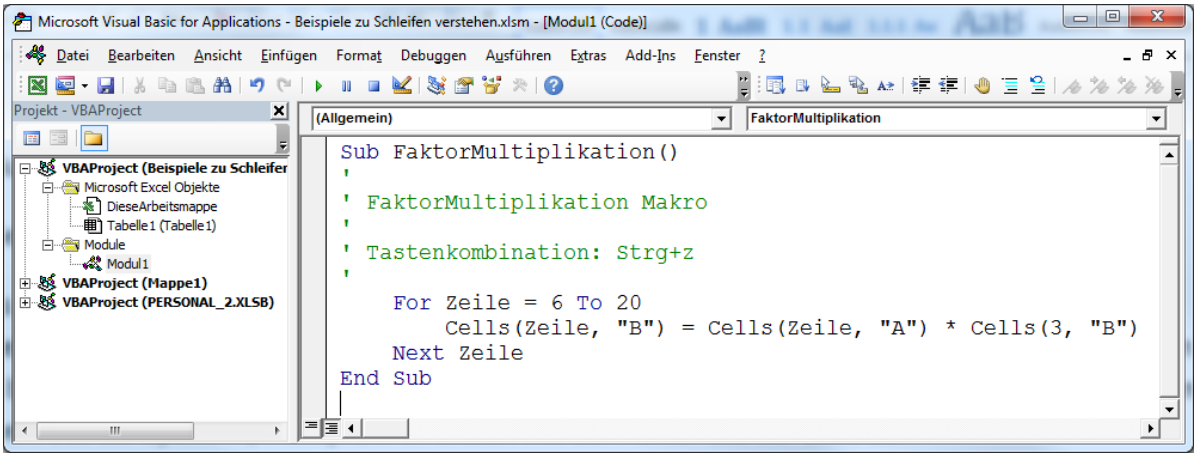


Nun gilt es, die bestehende Prozedur mit dem noch fehlenden Programmcode zu ergänzen.

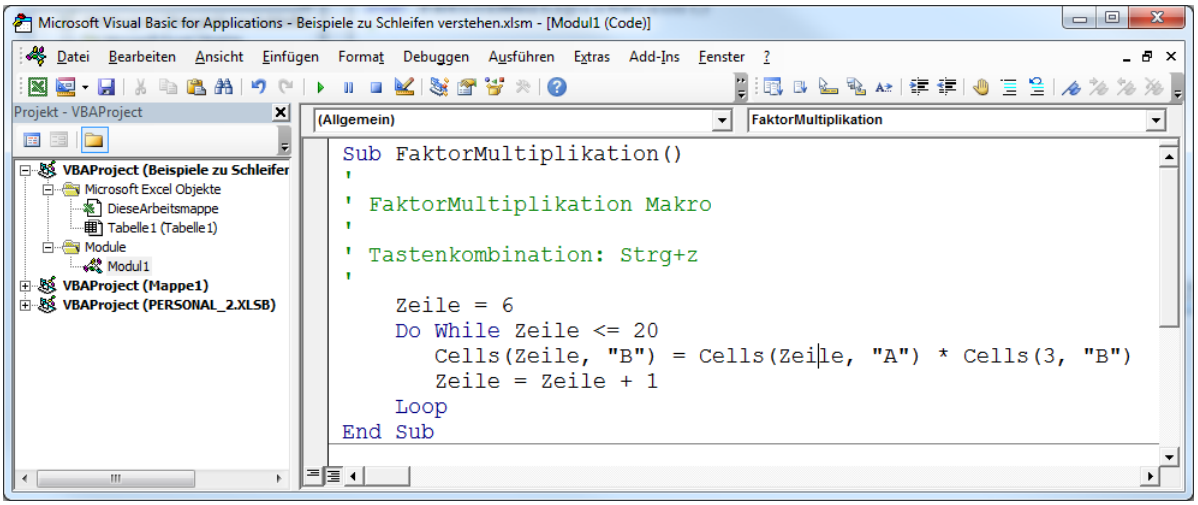
Gehen wir erste einmal in uns und überlegen die Schritte, welche wir programmieren müssen.

Es geht also darum eine Tabelle mit einer gewissen Berechnung zu vervollständigen. Um diese Berechnung durchführen zu können, benötigen wir die Werte der Tabelle in Spalte A und den Zeilen 6 bis 20. Wir werden also mittels Schleife die Zeilen 6 bis 20 durchlaufen, holen uns dabei den Zeilenwert aus der Spalte A, multiplizieren diesen mit den Faktor aus der Zelle B3:

Das Ergebnis mit einer FOR-NEXT-Schleife würde wie folgt aussehen.



Für unsere Freunde der HAW sieht das Ergebnis per DO-LOOP-Schleife so aus



6.2 Aufgabe 2: Spaltenreihe

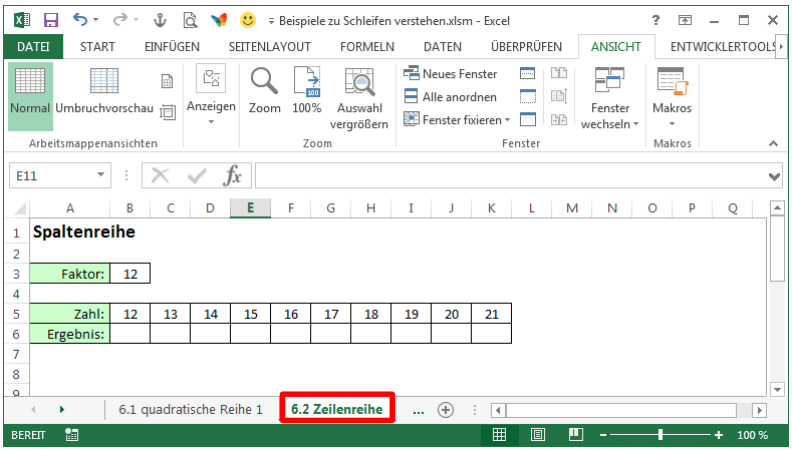
Was zeilenmäßig programiert werden kann, muß natürlich auch spaltenmäßig programmiert werden können.

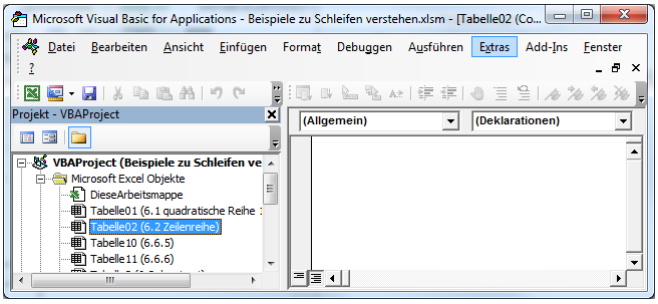
Erstellen Sie eine Zeilenreihe mit folgendem Aussehen:

Die dazugehörige VBA-Prozedur werden wir diesmal allerdings nicht mit einer Tasten-Kombination starten, sonder über das Ereignis CHANGE (Wenn eine Zelle mit einem neuen Wert belegt wird.)

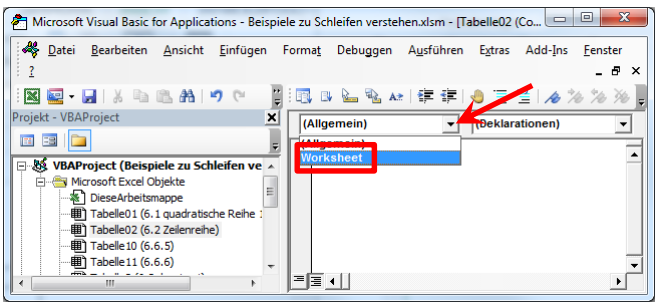
Vorgehensweise:

- Wechseln Sie in den VBA-Editor (z.B. über die Tastenkombination Alt + F11)
- Doppelklicken Sie auf das Excel-Objekt welches den gleichen Namen trägt wie das Betreffende Tabellenblatt, in welches die Ausgabe erfolgen soll:

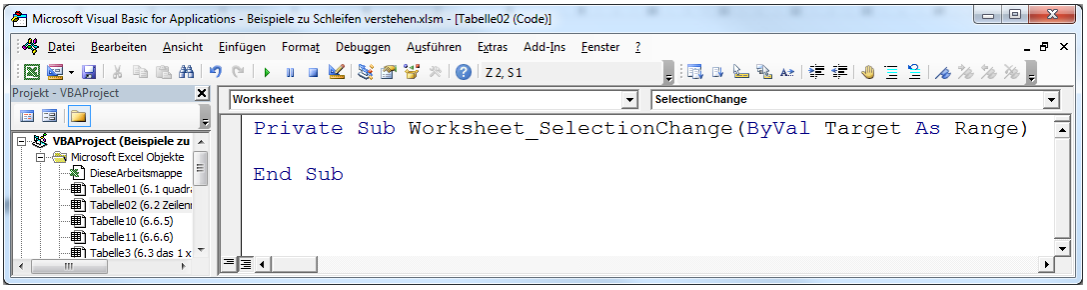




Dort angekommen, wählen wir im 1. Auswahlfeld des VBA-Editors folgende Option:

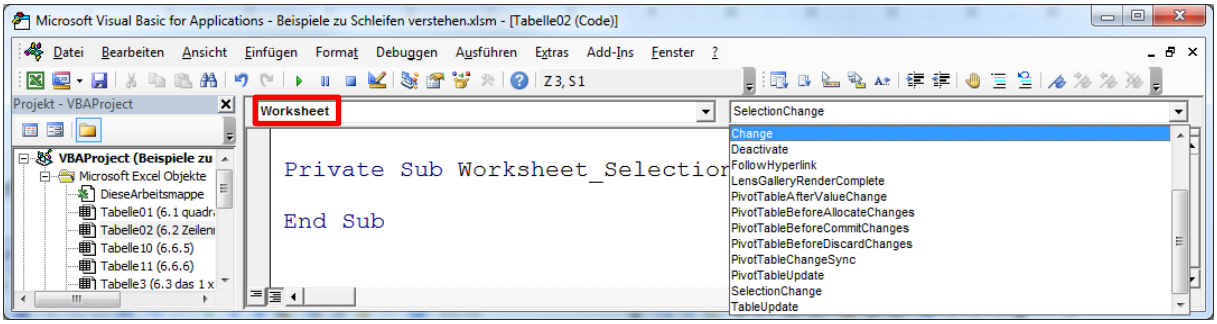


Und erhalten folgende leere VBA-Prozedur:

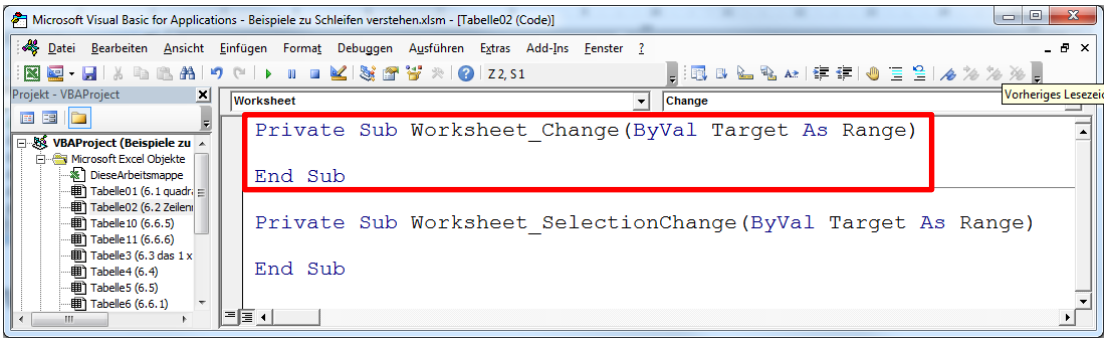


Dieses ist eine Ereignis-Prozedur (Event) welche abläuft, sobald eine ander Zelle im betreffenden Tabellenblatt gewählt wird.

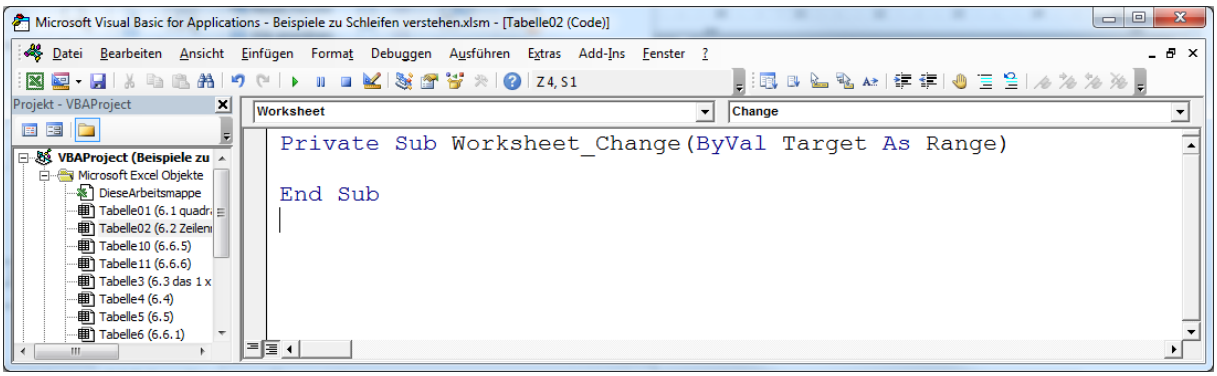
Da wir aber eine andere Eignisprozedur haben möchten, die ablaufen soll wenn eine Zelleingabe erfolgt. Werden wir diese im 2. Auswahl aufrufen:



Das Ergebnis ist eine 2. VBA-Prozedur:



Das uns die Selection_Change-Prozedur nicht interessiert, wird diese gelöscht:

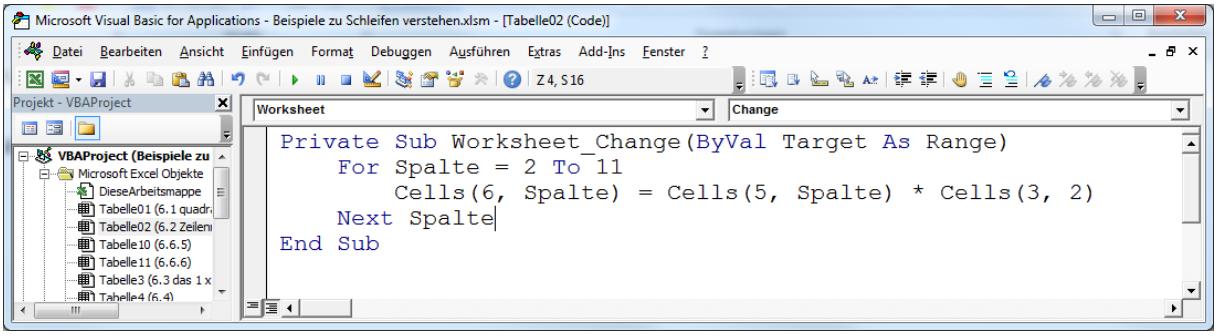


Und nun zur eigentlichen Aufgabe, welche wir in die leere Ereignis-Prozedur schreiben werden.

In Worten heißt dieses:

Berechne für die Felder der 6. Zeile, von Spalte 2 bis Spalte 11, das Produkt aus der jeweilig darüberliegenden Zelle (Zeile 5) und der entsprechenden Spalte mit der Zelle B3 (Zeile 3, Spalte 2).

Das Ergenis mittels FOR-Next Schleife würde wie folgt aussehen:



Diese Prozedur hat allerdings einen kleinen Haken. Jedes mal wenn ein Wert in eine Zelle eingetragen wird, egal ob es händisch im Tabellnbaltt oder über eine VBA-Prozedur passiert, wird diese Prozedur aufgerufen.

Wir müssen also dafür Sorgen tragen, dass unsere Prozedur nur dann abläuft, wenn ein Wert in Zelle B3 eingetragen wird.

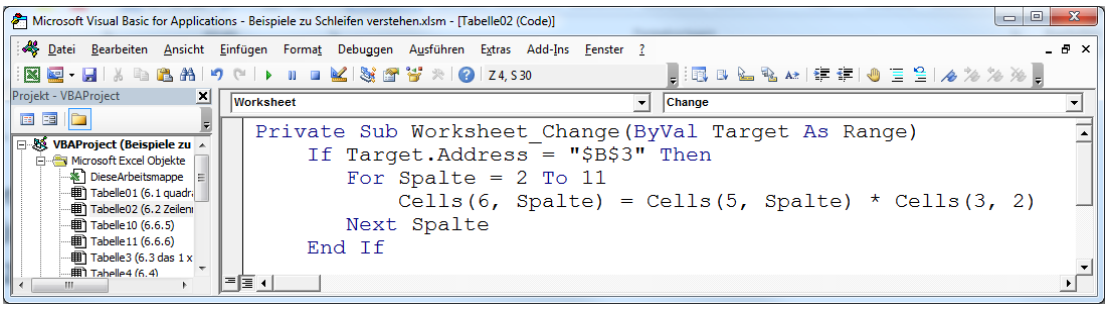
Das Argument Target in der Prozedur ist ein Objekt und steht stellvertretent für die Zelle, in der eine Eingabe erfolgt ist. Über eine Abfrage können wir die Adresse der betreffenden Zelle ermitteln und nur eine bestimmte Zelle zulassen.

Um den Zellbezug der aktuelle Zelle zu ermitteln, verwenden wird den Befehl **Target.Address**.

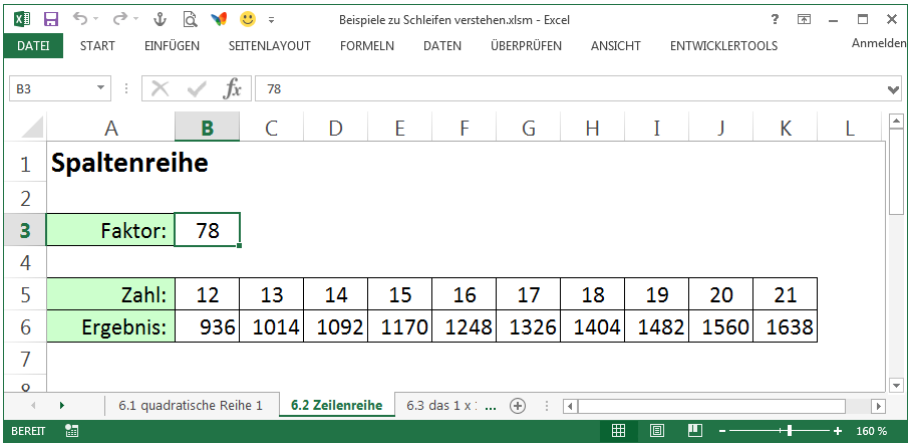
Die Bedingung für die Zelle „B3“ würde wie folgt lauten:

```
If Target.Address = "$B$3" Then
    … Weitere Befehle …
End If
```

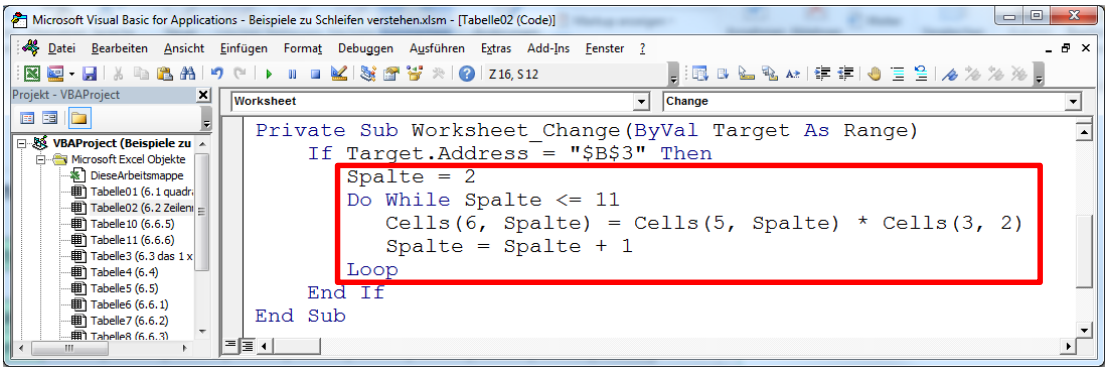
Eingesetzt in unsere Prozedur hätten wir folgendes Ergebnis:



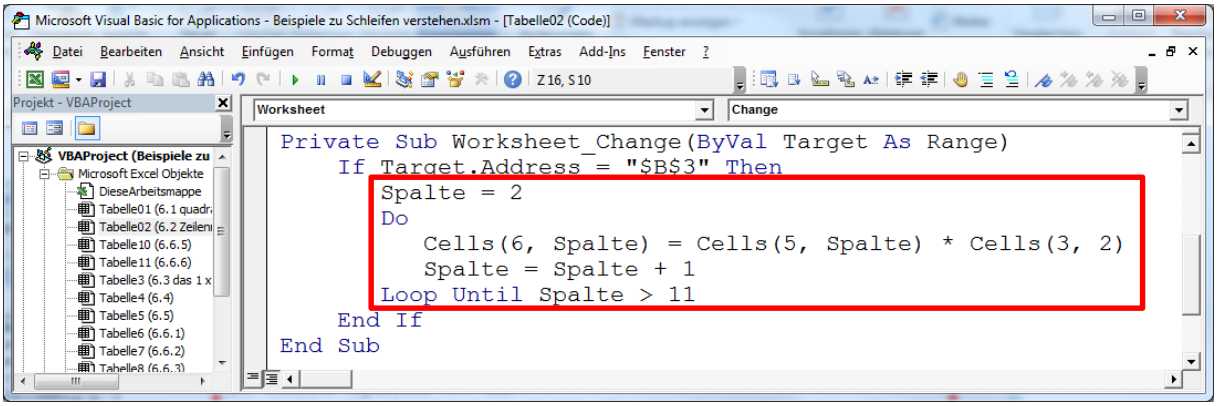
Wenn wir nun in Zelle B3 eine neue Zahl eingen erhalten wir über die von uns geschriebene Ereignis-Prozedur eine neu errechnete Tabelle, wie z.B.:



Um dieses Bespiele mit der DO WHILE-Schleife zu programmieren, würde das Ergebnis wie folgt aussehen:

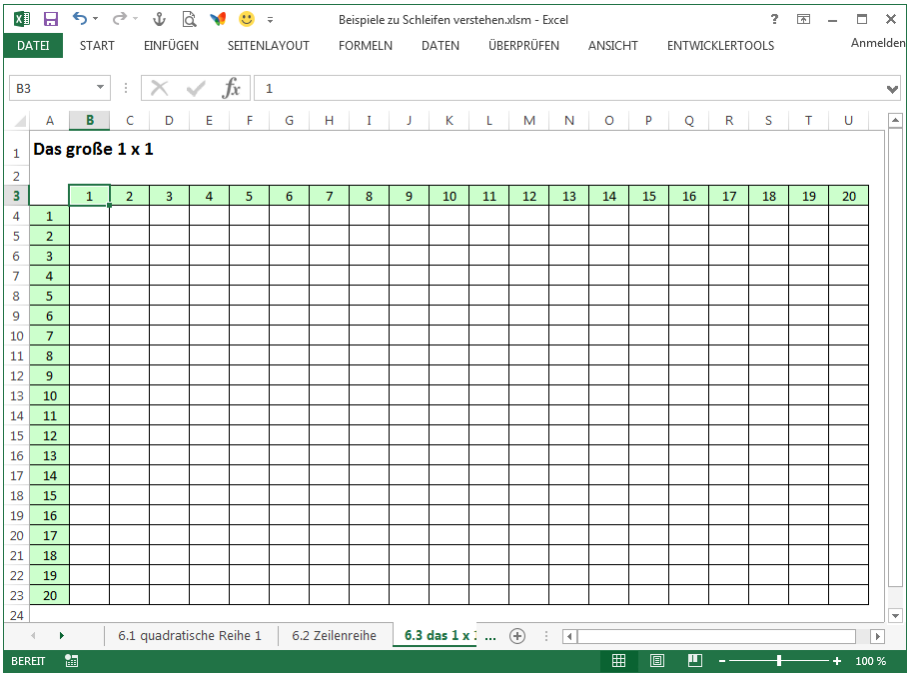


Eine andere Variante (DO-LOOP-UNTIL-Schleife) würde so aussehen:



6.3 Aufgabe 3: die 1 x 1 Tabelle

Bei dieser Aufgabe wollen wir den Inhalt folgender Tabelle füllen

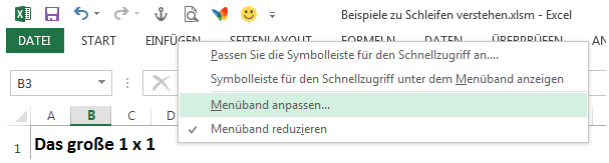


Wir benötigen also eine Prozedur, welche die Zellen im Bereich Zeile 4 bis Zeile 23 und Spalte 2 bis Spalte 21, mit der Multiplikation aus den jeweiligen Spaltenwerten aus der Zeile 3 und den Zeilenwerten aus der Spalte 1 multipliziert.

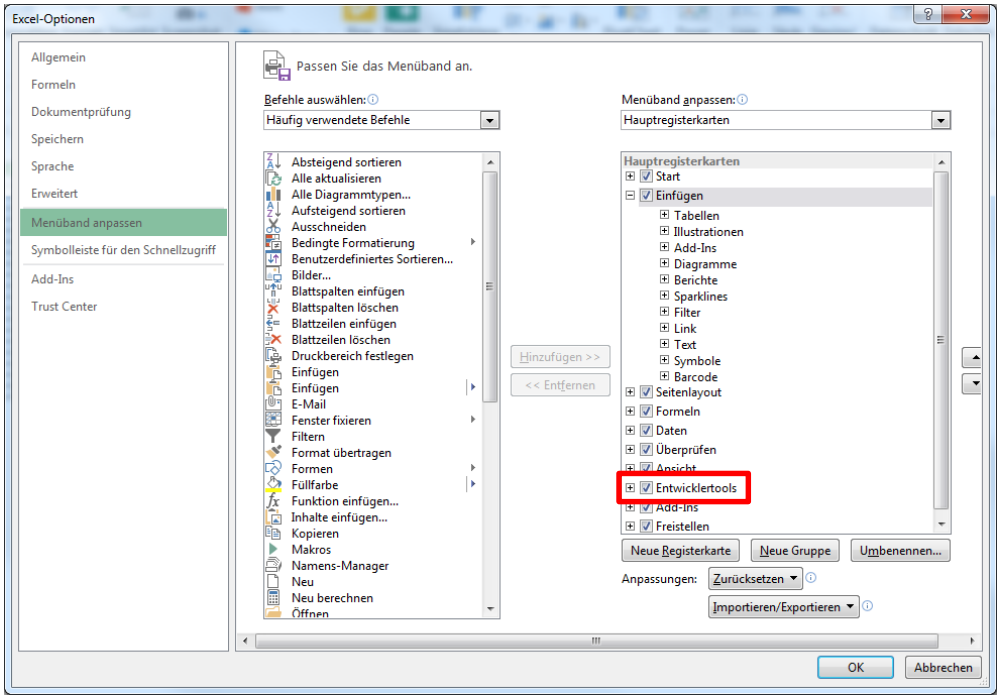
Um die entsprechende Prozedur zu satrten werden wir wieder etwas neues einführen:
Das Starten einer Prozedur über eine ActiveX – Schaltfläche.

Fangen wir mit der Erstellung der Schaltfläche an:

- Dazu klicken wir auf das Menüband „Entwickler-Tools“.
  Sollte diese Schaltfläche nicht sichtbar sein, kann Sie wie folgt aktiviert werden:
  - Klicken Sie mit der rechten Maustaste auf ein belibiges Menüband z.B. Start:

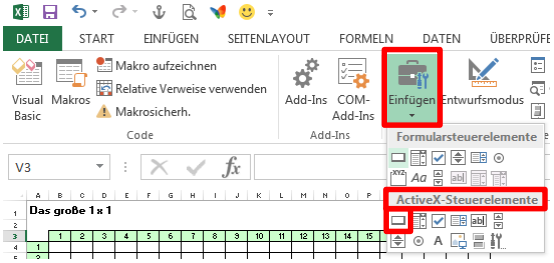


- Wählen Sie die Option „Menüband anpassen“:

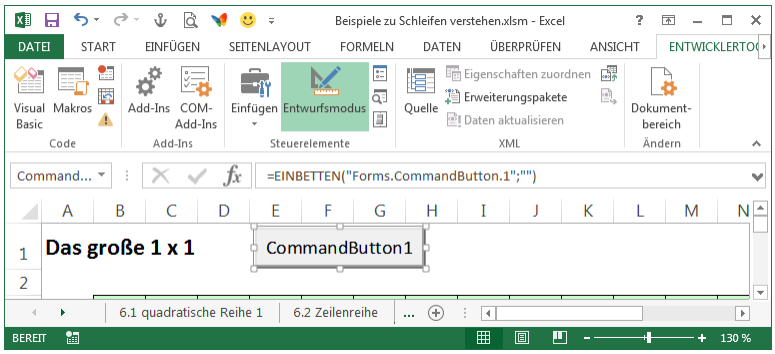


- Aktivieren sie das Kontrollkästchen vor „Entwicklertools“

- Wenn das Menüband „Entwicklertools“ angeklickt worden ist, finden wir in der Gruppe „Steuerelemente“ die Schaltfläche „Einfügen“ und darunter im Bereich Active-X Steuerelemente das Symbolt für eine ActiveX-Schaltfläche

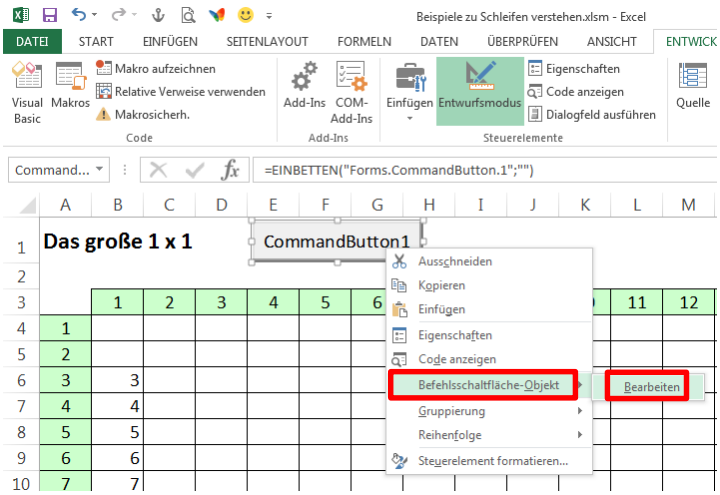


- Nachdem dieses Symbol angeklickt worden ist, kann im Tabellenblatt ein Rechteck gezeichnet werden:
  - angefangen mit der Position der linken oberen Ecke
  - linke Maustaste klicken und geklickt halten und bei geklickter linker Maustaste auf die Position der unteren rechte Ecke des Rechtecks zeiehen.
  - Nach Loslassen der linken Maustaste erhalten wir die Schaltfläche:

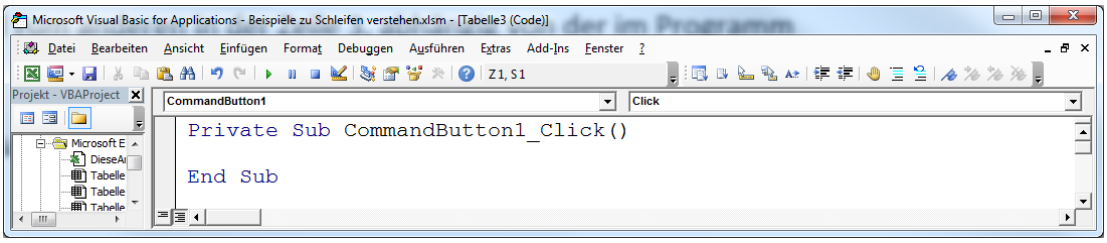


- Um den Beschriftung der Schaltfläche zu Ändern, klicken wir mit der rechten Maus Taste auf die Schaltfläche und wählen aus dem kleinen Kontextmenü die Option „Befehlsschaltfläche-

Objekt“ und anschließend „Bearbeiten“



- Nun kann der Inhalt z.B. auf „Aktualisieren“ geändert werden.
- Durch einen Klick auf eine beliebige Zelle außerhalb der Schaltfläche und einen Doppelklick auf die Schaltfläche, gelangen wir in den VBA-Editor mit einer automatisch erstellen leeren Prozedur:

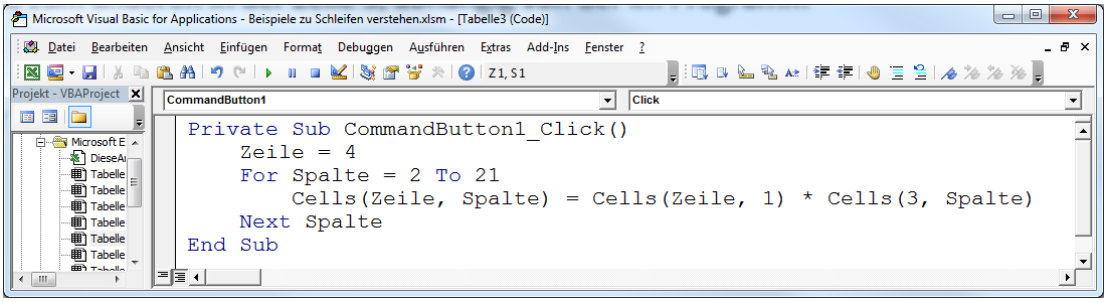


- Diese Prozedur wird dann ausgeführt, wenn auf die soeben erstellte Schaltfläche geklickt wird.

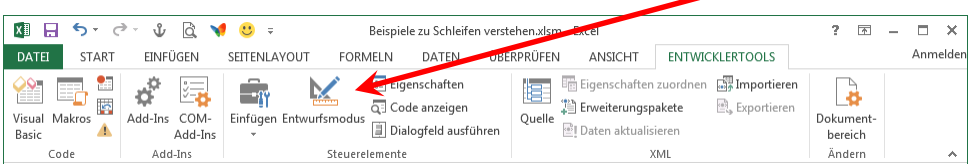
Kommen wir nun zum Erstellen des benötigten VBA-Code um unser Problem zu lösen:

Um nur eine Zeile (z,B, Zeile 4) der Tabelle zu füllen, können wir über eine Schleife von Spalte 2 bis 21 die ensprechenden Zellen füllen. Die beiden Werte zur ausführung der Multiplikation stehen zu einem in der Zelle A4 und zum anderen in der Zeile 3, abhängig von der im Programm angesprochenen Spalte:

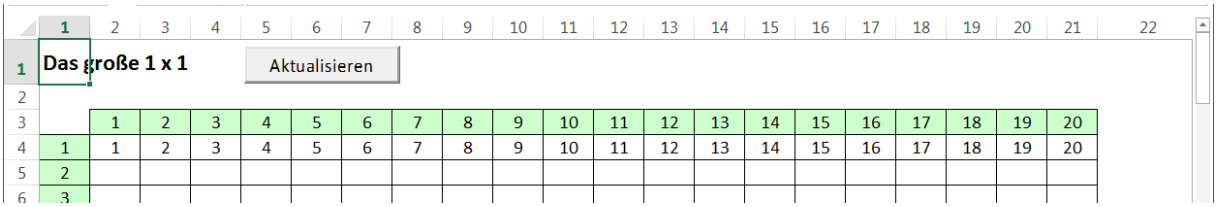
Eine Umsetzung unserer Vorüberlegung mittels FOR-NEXT-Schleife würde wie folgt aussehen:



Zum Testen des Programms wechseln wir auf das Excel-Tabellenbaltt, überprüfen, das im im Menüband „Entwicklertools“ die Schaltfäche „Entwurfsmodus“ **nicht** eingedrückt ist:

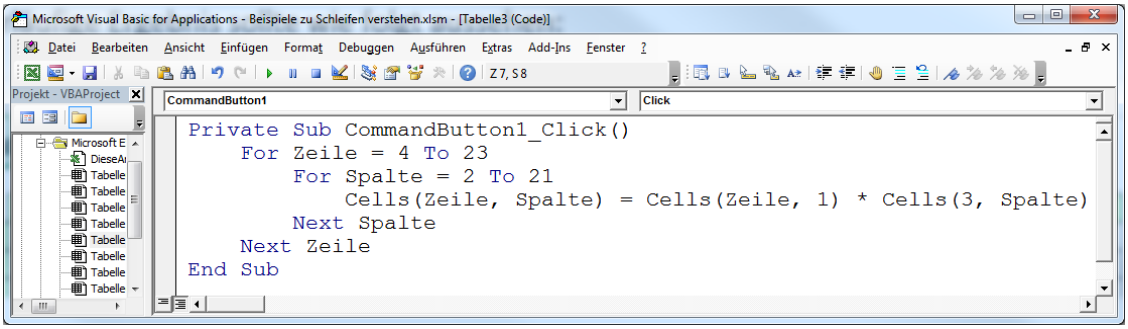


Das vorläufige Ergebnis sollte wie folgt aussehen:



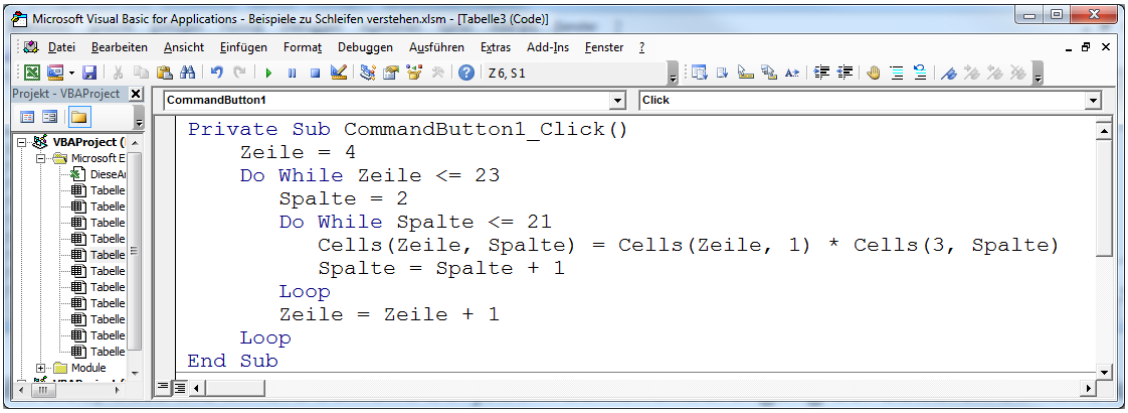
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
2																				
3																				

Um nun auch die weiteren Zeilen von 5 bis 23 auszufüllen ist eine weitere Schleife, welche wir um die bestehende programmieren, notwendig. Das Ergebnis sieht so aus:

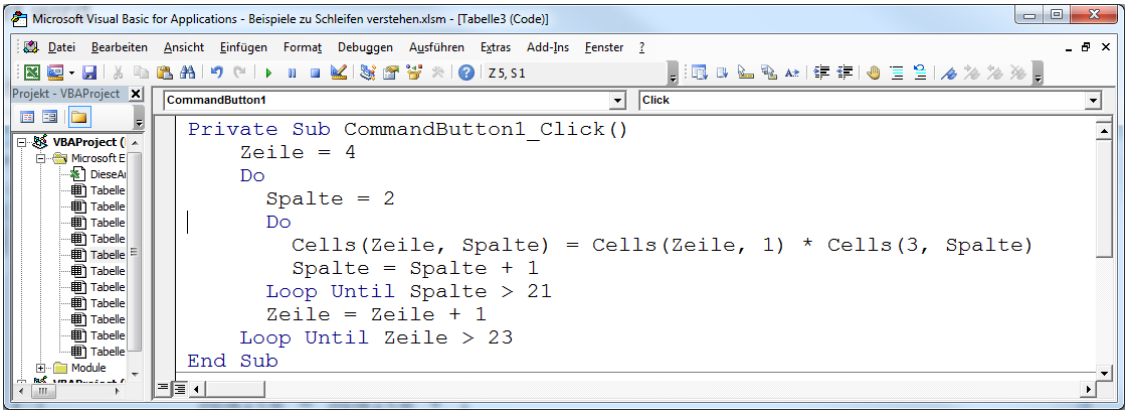


Das heißt also, dass die innere Schleife bei jedem Schleifendurchgang der äußeren Schleife komplett durchgeführt wird.

Das Ergebnis mit einer DO-WHILE-Schleife:



Das Ergebnis mit einer DO-LOOP-Schleife:



6.4 Aufgabe 4: Maximum und Minimum bestimmen

In diesem Übungsbeispiel geht es darum aus einer Menge von Daten das Maximum und das Minimum einer Spalte zu bestimmen.

6.4.1 Aufgabe 4.1

Bestimme aus einer Liste aus 100 Zahlen im Bereich A2:A101 das Minimum und das Maximum und schreibe das Ergebnis in die Zellen D1 und D2.

6.4.2 Aufgabe 4.2

Bestimme aus einer Tabelle aus 1000 Zahlen im Bereich A2:J101 das Minimum und das Maximum und schreibe das Ergenis in die Zelle L1 und L2

6.4.3 Aufgabe 4.3

Bestimme aus einer Spalte in einer bestehenden Tabelle aus 1000 Zahlen im Bereich A2:J101 das Minimum und das Maximum und schreibe das Ergebnis unterhalb der Spalte getrennt von einer Leerzelle

6.4.4 Aufgabe 4.4

Bestimme aus einer Spalte in einer bestehenden Tabelle aus 1000 Zahlen im Bereich A2:J101 das Minimum und das Maximum und schreibe das Ergebnis unterhalb der Spalte getrennt von einer Leerzelle

6.4.5 Aufgabe 4.5

Bestimme aus einer Tabelle aus 1000 Zahlen im Bereich A2:J101 das Minimum und das Maximum und markiere das Minimum in der Tabelle mit einem grünen Hindetrgrund und das Maximum mit einem roten Hintergrund

6.5 Aufgabe 5: Spaltenreihe separieren

6.5.1 Ausgabe 6.5.1

Schreibe alle positiven Zahlen welche sich in der Spalte A befinden in Spalte C

6.5.2

In der Spalte A befinden sich Tiernamen welche zu einer Tiergruppe gehöhren. Diese Information befindet sich in der Spalte B.

Schreiben Sie alle Tiernamen aus der Spalte A und schreiben Sie diese ind die Spalte D (Säugetier) Spalte E (Vogel) und F (Fisch).

6.6 Aufgabe 6: Zellen einfärben

In den nachfolgenden Beispielen sollen über die Schleifenprogrammierung um die akteulle Zelle verschiedene geometrische Figunren durch das einfärben von Zellen erstellet werden.

Die Schreibweise für das Einfärben einer Zelle sieht wie folgt aus:

ActiveCells

6.6.1 Aufgabe 6.1 Das rote Kreuz

6.6.2 Aufgabe 6.2 Das rote x

6.6.3 Aufgabe 6.3 Ein Quadrat aus der Exke

6.6.4 Aufgabe 6,4 Ein Quadrat aus der Mitte

6.6.5 Aufgabe 6.5 Ein Diamant

6.6.6 Aufgabe 6.6 Eine Spirale

6.7 Aufgabe 7: Dateien eines Verzeichnisses bestimmen

6.8 Aufgabe 8: Bild aus Farbiformationen erstellen

Link auf Datei Bild.xls