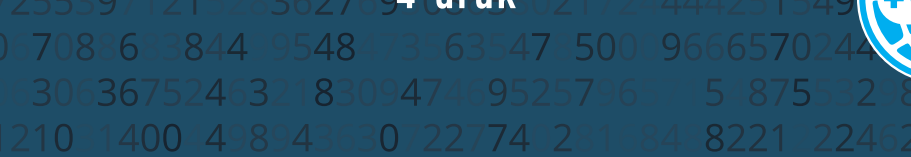
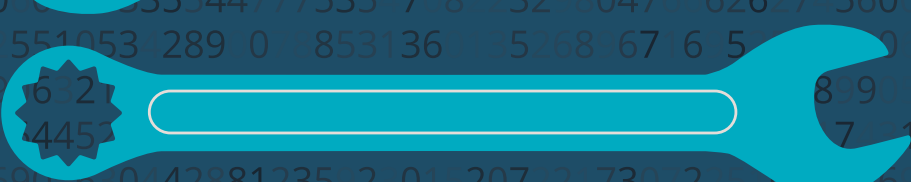
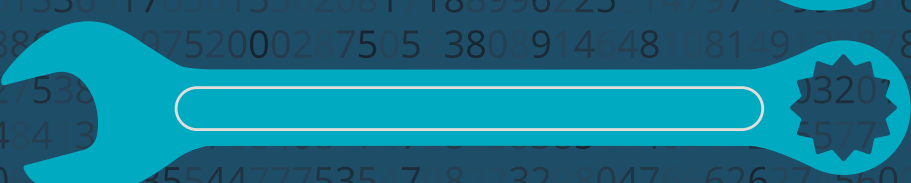
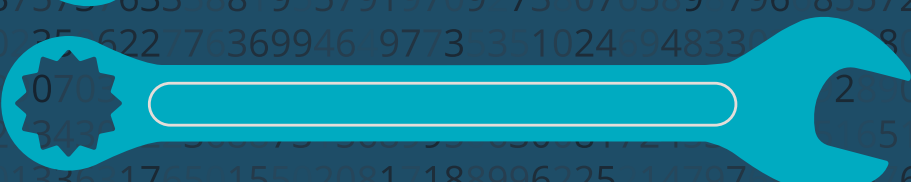
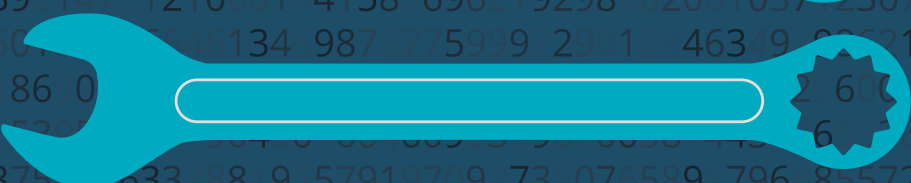
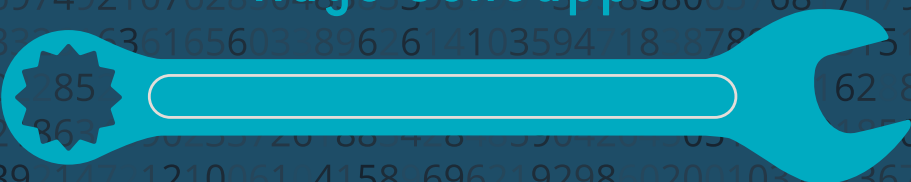


Hugo Schouppe



AAN DE SLAG MET EXCEL VBA

GESCHIKT VOOR EXCEL 2010-2019

4^e druk



Boom

Aan de slag met Excel VBA

Voor Nadine

Aan de slag met Excel VBA

Geschikt voor Excel 2010-2019

Hugo Schouppe

Boom

**inclusief
website!**



Met behulp van onderstaande unieke activeringscode kunt u toegang krijgen tot www.aandeslagmetexcelvba.nl voor extra materiaal. Deze code is persoonsgebonden en gekoppeld aan de 4e druk. Na activering van de code is de website 3 jaar toegankelijk. De code kan tot zes maanden na het verschijnen van een volgende druk geactiveerd worden.

1e druk 2002

2e druk 2011

3e druk 2016

4e druk 2019

Basisontwerp binnenwerk: Studio Bassa, Culemborg

Opmaak: Nu-nique grafische vormgeving, Goor

Basisontwerp omslag: Dog e Pony, Amsterdam

Omslagontwerp: DPS, Amsterdam

© 2019 H. Schouppe e Boom uitgevers Amsterdam

ISBN: 9789024402052

ISBN: 9789024402069 (e-book)

NUR: 123/989

www.boomhogeronderwijs.nl

www.aandeslagmetexcelvba.nl

Alle rechten voorbehouden. Alle intellectuele eigendomsrechten, zoals auteurs- en databankrechten, ten aanzien van deze uitgave worden uitdrukkelijk voorbehouden. Deze rechten berusten bij Boom uitgevers Amsterdam en de auteur.

Behoudens de in of krachtens de Auteurswet gestelde uitzonderingen, mag niets uit deze uitgave worden verveelvoudigd, opgeslagen in een geautomatiseerd gegevensbestand of openbaar gemaakt in enige vorm of op enige wijze, hetzij elektronisch, mechanisch, door fotokopieën, opnamen of enige andere manier, zonder voorafgaande schriftelijke toestemming van de uitgever.

Voor zover het maken van reprografische verveelvoudigingen uit deze uitgave is toegestaan op grond van artikel 16 h Auteurswet, dient men de daarvoor wettelijk verschuldigde vergoedingen te voldoen aan de Stichting Reprorecht (Postbus 3051, 2130 KB Hoofddorp, www.reprorecht.nl). Voor het overnemen van gedeelte(n) uit deze uitgave in bloemlezingen, readers en andere compilatiewerken (artikel 16 Auteurswet) dient men zich te wenden tot de Stichting PRO (Stichting Publicatie- en Reproductierechten Organisatie, Postbus 3060, 2130 KB Hoofddorp, www.cedar.nl). Voor het overnemen van een gedeelte van deze uitgave ten behoeve van commerciële doeleinden dient men zich te wenden tot de uitgever.

Hoewel aan de totstandkoming van deze uitgave de uiterste zorg is besteed, kan voor de afwezigheid van eventuele (druk)fouten en onvolledigheden niet worden ingestaan en aanvaarden de auteur(s), redacteur(en) en uitgever deswege geen aansprakelijkheid voor de gevolgen van eventueel voorkomende fouten en onvolledigheden.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the publisher's prior consent.

While every effort has been made to ensure the reliability of the information presented in this publication, Boom uitgevers Amsterdam neither guarantees the accuracy of the data contained herein nor accepts responsibility for errors or omissions or their consequences.

Voorwoord

De meeste leerboeken over Excel VBA gaan ervan uit dat je reeds kan programmeren in een andere programmeertaal. Je leert er de syntaxis van een nieuwe taal maar de basisconcepten van programmeren worden snel afgehandeld. In het beste geval kan je uit de voorbeelden afleiden wat een goede programmeerstijl zou zijn.

Dit boek gaat ervan uit dat jij nog geen ervaring hebt met programmeren. Het combineert het beste van twee werelden. Enerzijds is er gekozen voor een omgeving die je vandaag wellicht al gebruikt: Microsoft Excel. Anderzijds beperkt het boek zich niet louter tot een bespreking van de kleinste details van de taal. Klassieke programmeerconcepten worden uitgebreid besproken en voorzien van voorbeelden. Er wordt gewerkt aan de hand van een groot aantal oefeningen die stap voor stap worden opgebouwd en die resulteren in een uitgewerkte code. Verder kan je dit boek gebruiken in combinatie met de meest gangbare versies van Microsoft Excel (Excel 2010, 2013, 2016 en 2019; en Office 365).

Je begint eenvoudig: met de macrorecorder (hoofdstuk 2). Daarna leer je stelselmatig hoe je de code die de macrorecorder oplevert, kan aanpassen in de VBA editor (hoofdstuk 3). Gestructureerd programmeren komt aan bod in hoofdstuk 4. De programmastructuur wordt telkens geïllustreerd met behulp van ProgrammaStructuurDiagrammen. PSD's verplichten je om gestructureerd te werken. Vele programma's worden, naarmate ze uitbreiden, steeds onoverzichtelijker (en ongestructureerder). Opsplitsen in modules of modulair programmeren (hoofdstuk 5) kan helpen om het overzicht te bewaren. Software moet niet alleen gestructureerd geschreven zijn, maar ook herbruikbaar zijn. Daarvoor kan je programma onderdelen en gegevens verpakken in objecten. Excel VBA is een objectgeoriënteerde taal. Dit wil zeggen dat je bestaande Excel-objecten zoals draaitabellen en grafieken kan gebruiken (hoofdstuk 6) maar ook dat je eigen formulieren (hoofdstuk 8) en objecten (hoofdstuk 9) kan creëren. Een zeer belangrijk Excel-object wordt uiteraard gevormd door tabellen (hoofdstuk 7). Het boek wordt afgesloten met een korte exploratie van recursieve algoritmen (hoofdstuk 10).

Het meeste rendement haal je uit dit boek door de voorbeelden en oefeningen na te maken op de computer. Neem het boek bij de hand, zet de pc aan en begin! Je kan alle codefragmenten uit dit boek downloaden via de website van het boek (www.aandeslagmetexcelvba.nl). Onder de rubriek 'Studenten' is er per hoofdstuk een Excel-werkmap opgenomen met daarin alle codevoorbeelden uit het boek en de oplossingen van oefeningen. Daarnaast vind je er ook screencasts met voorbeelden.

Hugo Schouppe
januari 2019

Inhoud

Voorwoord	v
1 De Excel-werkomgeving	1
1.1 Wat is Excel?	1
1.1.1 Ontstaansgeschiedenis	1
1.1.2 Kenmerken	2
1.2 Wat is Excel VBA?	3
1.3 Waarom Excel VBA?	7
1.3.1 Elegante oplossing met een Excel-macro	7
1.3.2 Niet zo elegante oplossing met een Excel-macro	9
1.4 Instellen van de Excel VBA werkomgeving	10
1.4.1 Keuze van de pakkettaal	11
1.4.2 Zichtbaar maken van het menu Developer (Ontwikkelaars)	12
1.4.3 Aanpassen Macro Security (Macrobeveiliging)	12
1.4.4 Aanpassen van enkele werkmapopties	14
1.4.5 Instellen van de VBA-editor	14
2 De Excel-macrorecorder	15
2.1 Voorbeeld: gebruik van de macrorecorder	15
2.2 Starten en stoppen van de macrorecorder	17
2.2.1 Starten van de recorder	17
2.2.2 Stoppen van de recorder	19
2.3 Opnemen met de macrorecorder	20
2.3.1 De macro bekijken in de Visual Basic editor	22
2.3.2 Relatief of absoluut opnemen	24
2.4 Uitvoeren van een macro	29
2.4.1 Macro uitvoeren met het menu	29
2.4.2 Macro uitvoeren met een sneltoets	30
2.4.3 Macro uitvoeren met een knop op het werkblad	30
2.5 Oefeningen	31
3 De Visual Basic-editor	33
3.1 Voorbeeld: Hello world!	33
3.2 Gebruik van de Visual Basic-editor	34
3.2.1 Starten van de Visual Basic-editor	34
3.2.2 Sluiten van de Visual Basic-editor	34
3.2.3 Schakelen tussen de editor en de werkmap	35
3.3 Onderdelen van de editor	36
3.3.1 De Standard toolbar (werkbalk Standaard)	38
3.3.2 De Project Explorer (Projectverkenner)	39
3.3.3 Properties Window (venster Eigenschappen)	40
3.3.4 Code Window (venster Programmacode)	41
3.4 Schrijven van macro's in de editor	43

3.5	Uitvoeren van een macro in de editor	45
3.6	Testen van een macro	46
3.7	Onderbreken van een macro	48
3.8	Oefeningen	49
4	Gestructureerd programmeren	51
4.1	Variabelen en constanten	51
4.1.1	Gegevenstype van variabelen en constanten	52
4.1.2	Operaties op variabelen en constanten	58
4.1.3	Input- en outputoperaties	60
4.2	Basisprogrammastructuren	66
4.2.1	De sequentie	66
4.2.2	De selectie	69
4.2.3	De iteratie	74
4.2.4	Combinatie van de basisstructuren	80
4.3	Oefeningen	82
4.3.1	Sequentie	83
4.3.2	Selectie	83
4.3.3	Iteratie	83
4.3.4	Combinatie	84
5	Modulair programmeren	85
5.1	Functies en subroutines	85
5.1.1	Functies	86
5.1.2	Subroutines	92
5.1.3	Doorgeven van parameters	94
5.1.4	Bereik van een variabele	96
5.2	Top-downontwerp	97
5.3	Bottom-upontwerp	101
5.4	Ingebouwde Excel VBA-functies	106
5.4.1	Wiskundige en statistische functies	106
5.4.2	Conversiefuncties	106
5.4.3	Tekstfuncties	106
5.4.4	Datum- en tijdfuncties	107
5.4.5	Informatiefuncties	107
5.5	Oefeningen	108
6	Objectgeoriënteerd programmeren	111
6.1	Excel-objecten	113
6.1.1	Het Excel-objectmodel	114
6.1.2	De Object Browser (venster Objectenoverzicht)	116
6.2	Het Application-object	120
6.3	De objecten Workbook en Worksheet	123
6.4	Het object Range	128
6.4.1	Verschillende manieren om een Range-object te verkrijgen	133
6.5	Het object PivotTable (draaitabel)	139

6.6	Het object Chart	144
6.7	Oefeningen	146
7	Tabellen en bestanden	147
7.1	Tabellen	147
7.1.1	Zoekalgoritmen	156
7.1.2	Sorteer algoritmen	161
7.2	Bestanden	166
7.2.1	Aanmaken van een bestand	167
7.2.2	Een lijst maken van gegevens uit één bestand	170
7.2.3	Groepsonderbreking	171
7.2.4	Samenvoegen van bestanden	172
7.3	Oefeningen	174
8	Formulieren en gebeurtenissen	177
8.1	Het venster UserForm en de werkset Controls	182
8.1.1	Het venster UserForm	182
8.1.2	De werkset Controls	183
8.1.3	Code schrijven voor de UserForm en besturingselementen	185
8.2	Integratie van formulieren met een Excel-toepassing	187
8.3	Gebeurtenissen in het werkblad	191
8.3.1	Gebeurtenissen van het werkblad	191
8.3.2	Gebeurtenissen van de werkmap	193
8.4	Oefeningen	195
9	Eigen objecten en gekoppelde lijsten	197
9.1	Door de gebruiker gedefinieerde datatypen	197
9.2	Door de gebruiker gedefinieerde objecten	200
9.2.1	Samenvatting: aanmaken van eigen objecten	207
9.3	Gekoppelde lijsten	208
9.3.1	Voorstelling en notatie	208
9.3.2	Tabelimplementatie	211
9.3.3	Wijzerimplementatie	212
9.4	Oefeningen	221
10	Recurisie	223
10.1	Voorbeeld: fractalen	223
10.2	Iteratie versus recursie	224
10.2.1	Berekenen van de faculteit van een getal	226
10.2.2	Berekenen van de macht van een getal	229
10.2.3	De reeks van Fibonacci	230
10.2.4	De torens van Hanoi	232
10.3	Recurisie en wijzers	235
10.4	Recurisie en tabellen	238
10.5	Oefeningen	241
	Index	243

Hoofdstuk 1

De Excel-werkomgeving

Excel is een rekenbladprogramma dat door Microsoft in 1985 werd uitgebracht voor de Apple Macintosh. In enkele jaren tijd werd het marktleider. Het succes van Excel heeft vele redenen, maar een ervan is ongetwijfeld de programmeertaal: Visual Basic for Applications (VBA).

In dit hoofdstuk krijg je een korte beschrijving van de ontstaansgeschiedenis en een bespreking van zowel het rekenbladgedeelte als de programmeertaal van Excel. Je zal ook te weten komen waarom en wanneer je Excel VBA nodig hebt en hoe je de Excel-werkomgeving het best instelt om handig te werken in VBA.

1.1 Wat is Excel?

Excel is een rekenbladprogramma of een spreadsheet. De naam 'spreadsheet' komt uit de boekhouderswereld. In bedrijven hield men vroeger de kostenstaat handmatig bij op een groot kostenblad van soms wel bijna 1 m². Deze grote bladen, die men op het bureaublad moest uitspreiden, noemde men spreadsheets.

1.1.1 Ontstaansgeschiedenis

Spreadsheets liggen aan de basis van het succes van pc's. Het eerste elektronische rekenblad, VisiCalc, werd in 1978 ontwikkeld voor de Apple II-computer, een machine met 64 kB intern geheugen. VisiCalc was een rekenblad met 254 rijen en 63 kolommen. Een kolom was standaard 9 tekens breed en kon verbreed worden tot maximaal 37 tekens. Het programma kende al de meeste basisfuncties zoals SUM, AVERAGE, COUNT, ABS, INT, en was slechts 27.520 bytes groot.

Tip

Je kan VisiCalc nog steeds uitproberen op je pc, indien je dit zou willen. Open de website van één van de makers (<http://bricklin.com/visicalc.htm>) en download een nog steeds werkende versie ervan.

In januari 1983 kwam de firma Lotus Development Corporation op de markt met Lotus 1-2-3. Dit pakket combineerde een krachtig rekenprogramma met elementaire grafieken en een kleine maar handige database.

Microsoft had even daarvoor in 1982 zijn eerste spreadsheet, Multiplan, gelanceerd. Dit programma was allesbehalve succesvol en in 1985 kwam Excel voor de Apple Macintosh op de markt. Pas in november 1987 werd Excel voor Windows voorgesteld aan de pers. Men begon maar meteen met versie 2.0.

Van toen af ging het snel: versie 3 kwam in 1990 uit, in 1992 was het de beurt aan versie 4 en in 1993 aan versie 5. In 1995 is er de versie voor Windows 95 (officieel versie 7 genoemd; dus ook hier slaat men een nummer over). Versie 8.0 komt uit als onderdeel van Office 97. In 1999 is het de beurt aan versie 9.0 of de zogenaamde Excel 2000 uitvoering. De XP-versie of Excel 2002 werd gelanceerd in mei 2001. Vanaf versie 2003 werd gesproken van 'Microsoft Office Excel' om duidelijk te maken dat Excel een onderdeel was van de Microsoft Office Suite. Vervolgens kwamen Excel 2007, 2010, 2013, 2016 en de huidige versie Excel 2019.

Momenteel kan je Excel in drie varianten aanschaffen: de stand alone versie Excel 2019, waar je dus eenmalig voor betaalt maar die je dus opnieuw zal moeten kopen bij de volgende upgrade. Je kan je echter ook abonneren op de Office 365-service. Tegen een maandelijks of jaarlijkse prijs beschik je dan steeds over de nieuwste Office-versie, waar Excel, naast Word, PowerPoint, Access, Outlook en Publisher, een onderdeel van is. Tenslotte kan je ook de gratis Office online apps gebruiken in je webbrowser of smartphone. Deze hebben niet alle functionaliteiten van de stand alone- of Office 365-versies (bijv. 'Protect Workbook'), maar komen toch zeer dicht in de buurt. Belangrijk is echter dat je er geen Excel-macro's kunt mee aanmaken. Rekenbladen die macro's bevatten kunnen wel worden geopend in de online versie, maar niet uitgevoerd.

1.1.2 Kenmerken

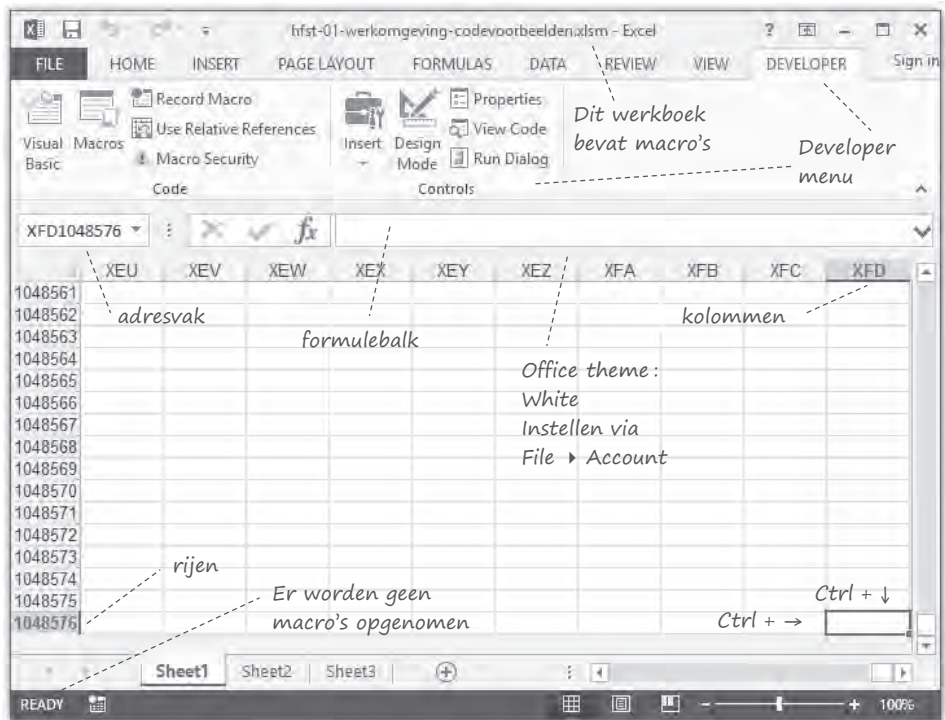
Een spreadsheet wordt wel eens vergeleken met een krachtige rekenmachine. Maar zoals een tekstverwerker veel meer is dan een typemachine mag je ook een spreadsheet niet als een geavanceerde rekenmachine beschouwen. Een rekenpakket verschilt op de volgende essentiële punten.

- De spreadsheet bestaat uit rijen en kolommen (maximaal 1 048 576 rijen x 16 384 kolommen in Excel 2019). Iedere kruising van een rij en kolom vormt een cel waarin je gegevens of formules kan plaatsen (zie figuur 1.1).

Tip

Met het menu File ► Options ► Formulas ► R1C1 reference style (Bestand ► Opties ► Formules ► Verwijzingstype R1K1) kan je omschakelen naar de oude Microsoft Multiplan-stijl, waarin zowel de rijen als kolommen numeriek worden benoemd. Met de toetsencombinatie Ctrl+ → en Ctrl+ ↓ kan je in een leeg werkblad dan controleren of het aantal rijen en kolommen overeenkomt met voorgaande aantallen.

- Deze ingevoerde gegevens en formules worden onthouden en kunnen eventueel bewaard worden op een extern opslagmedium.
- De gegevens en de formules worden duidelijk gescheiden. Indien je de berekeningen opnieuw wil uitvoeren met andere gegevens, hoeft je alleen de gegevens aan te passen.
- Bij wijziging van een gegeven wordt de spreadsheet automatisch opnieuw doorge-rekend. Hierdoor is het zeer eenvoudig nieuwe scenario's uit te rekenen; je hoeft alleen de gegevens te veranderen. Ook kan je (complexe) rekenmodellen construeren waarin later de gegevens worden gegoten.



Figuur 1.1 Excel-werkboek met het thema Light Gray

- Naast numerieke data kan je ook tekst, datum- en tijdsaanduidingen en logische waarden gebruiken. De beschikbare operaties en functies zijn zeer uitgebreid.
- Een spreadsheet kan als een eenvoudige database fungeren. Ook kunnen de gegevens gemakkelijk in de vorm van grafieken getoond worden.
- Excel is voorzien van een krachtige programmeertaal: Visual Basic for Applications, afgekort als VBA. Hierdoor kan je routinezaken eenvoudig uitvoeren, aangepaste formulieren voor gegevensinvoer ontwerpen en reageren op gebeurtenissen en zodoende de berekeningen van de spreadsheet aanpassen aan de specifieke situatie van het moment.

1.2 Wat is Excel VBA?

Vele handelingen in een rekenbladprogramma hebben een repeterend karakter. Meestal wil je meer dan één cel opmaken of moet je meer dan eens een rij tussenvoegen. Het automatiseren van deze taken stond dan ook op het verlanglijstje van vele gebruikers.

Automatiseren wil zeggen dat een serie handelingen herleid wordt tot het uitvoeren van een automatisch verloopend en meestal kleiner aantal handelingen. Stel dat je een cel een rode achtergrondkleur wil geven. In Excel moet je daarvoor achtereenvolgens: (1) de cel selecteren, (2) klikken op het menu Home (Start), (3) klik-

ken op Font (Lettertype), (4) kiezen voor het tabblad Fill (Opvullen), (5) een opvulkleur kiezen en (6) klikken op de OK-knop. Dit zijn heel wat 'klikken' met de muis. Automatiseren wil zeggen dat je deze zes handelingen kan vervangen door bijvoorbeeld het indrukken van een sneltoets.

In de eerste spreadsheets verliep dit automatiseren hoofdzakelijk via het toetsenbord. Er werd gebruikgemaakt van een 'recorder' die de toetsaanslagen opnam die nodig waren om een bepaalde taak uit te voeren. Het programma dat hieruit resulteerde was bijna een exacte kopie van de serie toetsaanslagen. In Lotus 1-2-3 zag je dan bijvoorbeeld regels code zoals `/fsbudget~r` wat betekende 'bewaar het gewijzigde werkboek onder naam budget.wks'. Om dit handmatig te doen, moest je het menu File (f) activeren (aangegeven door de schuine streep /), dan het submenu Save (s), dan de naam van het werkboek invullen (budget; de extensie wks werd er automatisch bijgeplaatst), op de Enter-toets drukken (weergegeven door -) en ten slotte kiezen voor de replace-optie (r).

Tip

Alhoewel het Lotus-programma niet meer bestaat, vind je toch nog referenties ernaar in Excel 2019; bijvoorbeeld onder File ► Options ► Advanced ► Lotus Compatibility (Bestand ► Opties ► Geavanceerd ► Lotus compatibiliteit). Blader naar de onderkant van de pagina.

Excel was vanaf het begin minder gebonden aan de specifieke toetsaanslagen, maar ook hier zag je initieel (tot versie 5 of 1993) nog deze aanpak. Een nadeel van deze manier van programmeren was dat de betekenis van de toetsaanslagen niet mocht wijzigen tussen de verschillende versies van het programma. Als `/fs` een andere betekenis zou krijgen in een latere versie, zou bovenstaande macro niet meer het beoogde doel bereiken.

Vanaf versie 5 zien we dat Excel zich ontwikkelt naar een volledige programmeeromgeving. Men spreekt dan ook niet meer over een macrotaal of het schrijven van macro's, maar over het schrijven van programma's in een programmeertaal. Men noemt dit ook wel coderen of het schrijven van 'code'. Het begrip 'macro' is overgenomen uit de wereld van de assembler-programmeur. De assembleertaal is een zeer primitieve taal, die dicht bij de machinetaal staat. Een nadeel van deze assembleercode was dat men zelfs voor de meest simpele dingen (zoals het tonen van een teken op het scherm) een uitgebreid programma met verschillende regels code moest schrijven. Programmeurs begonnen dan ook afkortingen te gebruiken voor bepaalde veelvoorkomende regels code. Op de plaats waar die regels moesten komen in het programma, werd dan alleen de afkorting genoteerd. Naast de tijdsbesparing verhoogde dit dikwijls ook de leesbaarheid van het programma. Op het moment dat het programma klaar was en men er een uitvoerbare versie wilde van maken, werden deze afkortingen dan omgezet naar de volledige sequentie van programmaregels waarvoor ze stonden. Men noemde deze afkortingen 'macro's', mogelijk omdat ze voor iets 'groters' stonden.

In Excel versie 5 werd de ‘oude’ macrotaal dus vervangen door een volwaardige programmeertaal. Beide namen (macro en programmacode) worden echter nog steeds door elkaar gebruikt. De programmeertaal werd Visual Basic for Applications genoemd. Voor een goed begrip van deze naam dien je het verband te kennen met twee andere programmeertalen. BASIC (Beginner’s All-purpose Symbolic Instruction Code) was één van de eerste hogere programmeertalen en werd ontwikkeld in 1964. Terwijl deze eerste hogere programmeertalen zoals BASIC, FORTRAN en COBOL tekstgeoriënteerd waren, is Microsoft in 1991 op de markt gekomen met een versie van BASIC die gebruikmaakte van ‘WIMP’ (Windows, Icons, Menus, Pointing devices) of, anders gezegd, visueel georiënteerd was. Zij noemden deze taal daarom Visual Basic.

VBA is afgeleid van Visual Basic (en dus van BASIC) maar speciaal aangepast aan een applicatie. De bedoeling van Microsoft was tot een gemeenschappelijke programmeeromgeving te komen voor de verschillende applicaties uit de Office-reeks. De eerste twee producten waar VBA werd geïmplementeerd, waren Excel en Project (een projectmanagementsysteem). Naderhand zijn daar Access, Word, PowerPoint en Outlook bijgekomen. We kennen nu bijvoorbeeld Excel VBA, Word VBA en Access VBA. Excel VBA is dus een programmeertaal die lijkt op Visual Basic, maar die speciaal aangepast is aan Excel.

■ Voorbeeld 1.1 Bladzijdenummering plaatsen als rechtervoettekst in Excel

Stel dat je in de voettekst van een document de paginanummering wenst op te nemen. Dit document kan zowel een Excel-spreadsheet als een Word-document zijn. In Excel VBA zal deze macro er zo uitzien.

```
Sub InvoegenBladzijdeNr_Excel()  
    ActiveSheet.PageSetup.RightFooter = "&P"  
End Sub
```

Tip

Je kan alle codefragmenten uit dit boek downloaden via www.aandeslag-metexcelvba.nl. Van ieder hoofdstuk is een Excel-werkmap opgenomen met daarin alle codevoorbeelden en oplossingen van oefeningen.

Deze code is goed leesbaar voor iemand die Excel kan bedienen en verraaft tegelijkertijd nog de band met de vroegere toetsenbordmacro's. Om in Excel met de hand het paginanummer rechtsonder aan de bladzijde te krijgen, moet je (let op de Engelstalige benaming van de menu's):

- het werkblad selecteren waarin je een voettekst wil (ActiveSheet),
- klikken op het menu Page Layout (Pagina-indeling),
- kiezen voor Page Setup (Pagina-instelling),
- het tabblad Header/Footer (Koptekst/voettekst) selecteren,

- klikken op Custom footer ... (Aangepaste voettekst ...),
- de Right Section (Rechts) selecteren,
- en klikken op de knop voor het paginanummer, wat de code `&[Page]` oplevert.

Dit wordt allemaal weergegeven door de Excel VBA-instructie:

```
ActiveSheet.PageSetup.RightFooter = "&P"
```

■ Voorbeeld 1.2 Invoegen van bladzijdenummering als rechtervoettekst in Word

In Word ziet deze macro er als volgt uit:

```
Sub InvoegenBladzijdeNr_Word()  
    ActiveDocument.Sections(1).Footers(1).PageNumbers.Add  
End Sub
```

Word heeft geen `ActiveSheet` maar wel een `ActiveDocument` met daarin een of meer secties. Iedere sectie kan verschillende voetteksten hebben waar je een paginanummer aan kan toevoegen.

■ Voorbeeld 1.3 Kopiëren van de huidige selectie naar cel C1 in Excel

Het volgende voorbeeld betreft het kopiëren van cellen in Excel of tekst in Word. De via het toetsenbord opgenomen macro (zie hoofdstuk 2) ziet er in Excel als volgt uit (we kopiëren de huidige cel naar de cel C1):

```
Selection.Copy  
Range("C1").Select  
ActiveSheet.Paste
```

De cel of cellen die geselecteerd waren op het moment dat de macro start, worden gekopieerd naar het klembord door de instructie `Selection.Copy`. Daarna wordt de cel C1 geselecteerd met de instructie `Range("C1").Select`. Ten slotte wordt de inhoud van het klembord geplakt met `ActiveSheet.Paste`.

■ Voorbeeld 1.4 Het geselecteerde woord twee woorden naar rechts kopiëren in Word

De volgende Word-macro kopieert het woord dat geselecteerd is in de tekst twee woorden naar rechts. Merk de sterke gelijkenis op met de Excel-macro.

```
Selection.Copy  
Selection.MoveRight Unit:=wdWord, Count:=3  
Selection.Paste
```

Met de eerste instructie wordt het geselecteerde woord gekopieerd naar het klembord. De tweede instructie verplaatst de selectie drie woorden naar rechts. De laatste instructie plakt de inhoud van het klembord. Merk op dat het verschil in beide macro's zit in het gebruik van de typische Excel- of Word-objecten (zoals ActiveSheet).

Zoals je kan zien, vertonen de Excel- en Word-macro's erg veel overkomst. VBA is dan ook een programmeeromgeving die je in de verschillende applicaties kan gebruiken.

1.3 Waarom Excel VBA?

Veel problemen kunnen met een macro vaak gemakkelijker of sneller opgelost worden dan met de hand. Soms echter is een macro wat veel van het goede en ben je beter af met de ingebouwde functies. We geven een voorbeeld van elk.

1.3.1 Elegante oplossing met een Excel-macro

■ Voorbeeld 1.5 Verlagen van de prijzen voor producten met meer dan 100 eenheden in voorraad

Stel dat je in een werkboek de voorraad en de prijzen bijhoudt van alle producten die je verkoopt (zie figuur 1.2). Op een bepaald moment besluit je de prijzen van alle producten waarvan er nog meer dan 100 stuks in voorraad zijn, te verlagen met 5%.

Als de voorraad > 100 wordt de prijs verminderd met 5%

	A	B	C	D
1	product	voorraad	prijs	aangepaste prijs
2	AA-101	150	77	73.15
3	AA-206	64	42	42
4	B-166	155	3	2.85

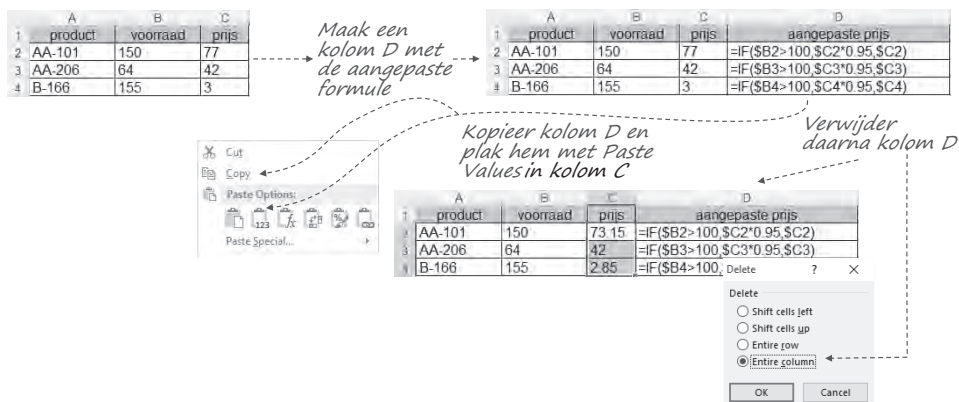
Maar eigenlijk wil je geen bijkomende kolom en moet de prijs meteen aangepast worden in kolom C.

Figuur 1.2 Excel-werkblad met producten, voorraad en prijzen

In het voorbeeld van figuur 1.2 zou dit dus betekenen dat je de prijzen moet aanpassen van de producten AA-101 en B-166. Dit probleem is met de hand als volgt op te lossen, maar het is vrij omslachtig:

- Maak een nieuwe kolom, bijvoorbeeld kolom D, met de aangepaste prijs. Gebruik hiervoor de volgende formule in cel D2: `=IF($B2>100;$C2*0,95;$C2)`. Vervang het woordje IF door ALS in de Nederlandstalige versie van Excel.
- Kopieer deze formule naar het gebied D3:D4. Zie figuur 1.3.
- Knip het gebied D2:D4 op het klembord.
- Plak deze gegevens in het bereik C2:C4. Hiervoor moet je wel het menu Home ► Paste ► Values (Start ► Plakken ► Waarden) gebruiken. Als je gewoon zou plakken, krijg je namelijk een kringverwijzing met foutboodschap. In cel D2 staat immers een

formule met een verwijzing naar cel C2. Deze formule (die verwijst naar cel C2) zou je dan plakken in dezelfde cel C2. Daarom moet je het resultaat of de waarde (value) van de formule plakken in cel C2, en niet de formule zelf.



Figuur 1.3 Aanpassing met de hand van de voorraad met zichtbare formule

Tip

Je kan de formules van een werkblad zichtbaar maken via het menu File ► Options ► Advanced ► Display Options for this worksheet ► Show formulas in cells instead of their calculated results (Bestand ► Opties ► Geavanceerd ► Weergaveopties voor deze werkmap ► Formules weergeven in cellen in plaats van de berekende resultaten).

De handmatige oplossing voor dit probleem is als gezegd vrij omslachtig, vergt heel wat formules (en daardoor ook rekenkracht en geheugenruimte) en is foutgevoelig. Bovendien veronderstelt deze oplossing dat je nog een vrije kolom hebt in je rekenblad, liefst in de buurt van kolom C.

Tip

Op de website www.aandeslagmetexcelvba.nl kan je een filmpje bekijken waarin je de bovenstaande stappen een voor een ziet uitvoeren, vergezeld van enig commentaar.

De oplossing met een macro is echter zeer eenvoudig en bovendien goed leesbaar. Het uitvoeren van de macro kan dan bijvoorbeeld gebeuren door het indrukken van een toetsencombinatie.

```
Sub verlagen_prijzen()  
    Dim c As Range
```

```
For Each c In ActiveSheet.Range("C2:C4")
    If c.Offset(0, -1).Value > 100 Then
        c.Value = c.Value * 0.95
    End If
Next c
End Sub
```

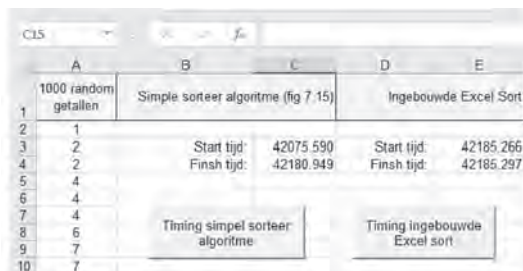
De macro zal iedere cel in het gebied C2:C4 onderzoeken. Als de cel links van de cel, aangegeven met de uitdrukking `offset(0,-1)`, groter is dan 100, wordt de waarde in de cel verminderd met 5%.

Veel problemen, zoals het genoemde, kunnen vaak makkelijker of sneller opgelost worden met een macro. Soms kan het probleem alleen maar met een macro tot een goed einde gebracht worden. Neem bijvoorbeeld de zeer handige functie van Conditional Format (Voorwaardelijke opmaak) onder het menu Home (Start). Afhankelijk van een instelbare voorwaarde kan je cellen een andere tekstkleur, achtergrond of rand geven. Het is echter niet mogelijk om via deze functie het lettertype of de lettergrootte van de cel aan te passen. Als je dat toch zou willen realiseren, ben je verplicht om een macro te gebruiken.

1.3.2 Niet zo elegante oplossing met een Excel-macro

■ Voorbeeld 1.6 Sorteren van getallen

Stel dat je een spreadsheet hebt met 1000 getallen en je wilt deze sorteren van klein naar groot. Dan zou je in de verleiding kunnen komen om snel een macro te schrijven. In hoofdstuk 7 bespreken we zelfs enkele sorteeralgoritmen en geven we de VBA-code. Je zou dan bijvoorbeeld het simpel sorteeralgoritme kunnen gebruiken van figuur 7.15. Dit is een perfecte oplossing voor je probleem, alleen ... duurt het op mijn PC ongeveer 105 seconden om deze 1000 getallen te sorteren, terwijl de ingebouwde Excel-sorteerfunctie maar 31 milliseconden nodig heeft om de klus te klaren (zie figuur 1.4). Uiteraard kan je sorteerprogramma verbeterd worden (zie hoofdstuk 7 voor enkele efficiëntere technieken) of kan je enkele optimalisatietrucjes van Excel toepassen, maar de prestatie evenaren van de ingebouwde sorteerfunctie zal niet lukken.



	A	B	C	D	E
1	1000 random getallen	Simple sorteer algoritme (fig 7.15)		Ingebouwde Excel Sort	
2	1				
3	2	Start tijd:	42075.590	Start tijd:	42185.266
4	2	Finish tijd:	42180.949	Finish tijd:	42185.297
5	4				
6	4				
7	4				
8	6	Timing simpel sorteer algoritme		Timing ingebouwde Excel sort	
9	7				
10	7				

Figuur 1.4 Benodigde verwerkingstijd voor het simpel sorteeralgoritme (zie figuur 7.15) en de ingebouwde Excel-sortering. De tijd wordt gemeten met de Timer-functie (nauwkeurig tot 1/256 s) en geeft het aantal seconden (42075,590 in de figuur) sedert middernacht.

```
Sub sorteren_elegant()  
    Range("E3").Value = Timer  
    Range("A2:A1001").Sort Key1:=Range("A1"), order1:=xlAscending  
    Range("E4").Value = Timer  
End Sub  
Sub sorteren_niet_zo_elegant()  
    Range("C3").Value = Timer  
    Call simpelSortAlgorime  
    'verwijst naar de code van hoofdstuk 7, figuur 7.15  
    Range("C4").Value = Timer  
End Sub
```

Met de instructie `Range("E3").Value = Timer` wordt het tijdstip (aantal seconden sedert middernacht) genoteerd in cel E3. In de eerste macro worden dan de 1000 getallen in het gebied A2:A1001 via de ingebouwde sorteerfunctie van Excel gesorteerd; in de tweede macro wordt een eigen sorteeralgoritme opgeroepen (zie hoofdstuk 7). Daarna wordt het tijdstip opnieuw genoteerd.

Soms kan je niet buiten het schrijven van je eigen sorteeralgoritme, omdat de ingebouwde sorteerfunctie niet voldoet aan jouw vereisten. Stel dat je bijvoorbeeld namen alfabetisch wenst te sorteren. In Excel zal de naam Van Morgen voor de naam VanAvond gesorteerd worden, omdat een spatie als 'kleiner' dan een letter of ander leesteken wordt beschouwd. Beide namen zullen onder de letter V komen te staan, wat prima is in België, maar niet in Nederland (Van Morgen wordt in Nederland gesorteerd onder de letter M; Morgen, van). Wens je de Nederlandse of Belgische sorteer- volgorde te volgen, dan dien je ofwel een kolom bij te maken met daarin de namen geschreven volgens de landconventie en daarop te sorteren, ofwel een eigen sorteer- macro te schrijven.

1.4 Instellen van de Excel VBA werkomgeving

Excel VBA is een krachtige programmeertaal. Naast de vele voordelen dat dit oplevert, zijn er ook wat problemen en gevaren.

Programmeren van Excel-macro's wordt meestal afgeschilderd als een moeilijke klus. Dit is ten dele waar. Werken met Excel VBA veronderstelt namelijk dat je twee vaardigheden beheerst. Je moet goed vertrouwd zijn met het gebruik van Excel (zonder de programmeertaal). Dit wil zeggen dat concepten zoals gebied, rij, kolom, formule, functie, en absolute en relatieve celreferentie bekend zijn. Tussenvoegen van rijen en kolommen, verplaatsen en kopiëren, invoeren van functies en formules, opmaken van cellen, aanmaken van grafieken, enzovoort, mag evenmin een probleem zijn. Excel VBA is namelijk bedoeld om deze handmatige activiteiten te automatiseren. Je hoeft geen expert te zijn, maar hoe meer je weet van Excel (zonder programmeertaal), hoe meer voordeel je kan halen uit het gebruik van Excel VBA.

Naast de kennis van Excel als rekenblad is ook enige kennis van programmeren vereist. Een programma is een verzameling van instructies die door een computer uitgevoerd kunnen worden. Een programma in Excel VBA zal dus instructies bevatten die door Excel uitgevoerd worden. Programmeren is dan de kunde (of kunst, volgens sommigen) van het samenstellen van zo'n lijst instructies. Elk programma, ongeacht de programmeertaal, heeft een aantal gemeenschappelijke kenmerken. Iedere programmeertaal kent bijvoorbeeld elementaire gegevensstructuren zoals variabelen. Daarnaast kent een programmeertaal ook een aantal controlestructuren zoals een herhaling of selectie. Veel handboeken voor Excel VBA veronderstellen dat de lezer deze kennis bezit, met andere woorden dat de lezer al kan programmeren in een andere programmeertaal dan Excel VBA. Veelal is deze veronderstelling niet juist. Een typische Excel-gebruiker is meestal geen doorgewinterde programmeur en hoeft dat ook niet te zijn. Dit boek onderscheidt zich van de andere doordat er evenveel aandacht geschonken wordt aan de programmeerkennis en de kennis van Excel (VBA).

1.4.1 Keuze van de pakkettaal

Je moet onderscheid maken tussen Excel VBA en Excel als rekenpakket. De instructies in Excel VBA zijn Engelstalig en je kan geen andere taal kiezen. Je zal dus Engelstalige instructies schrijven zoals IF ... THEN ... ELSE, DO WHILE ... LOOP, enz. Ook de objecten die je zal gebruiken zoals Range, Worksheet, Chart, ... hebben enkel een Engelstalige benaming.

Voor het rekenbladgedeelte van Excel kan je wel een eigen taal instellen. Dit doe je met File ► Options ► Language (Bestand ► Opties ► Taal). Je kan zowel de bewerkingstaal (onder andere voor spellingscontrole en toetsenbordindeling) als de gebruikersinterfacetaal (onder andere de taal van de menu's en help) instellen. De keuze van de bewerkingstaal is bepalend voor de schrijfwijze van de functies. In het Nederlands bijvoorbeeld gebruik je functies als: SOM, GEMIDDELD en ASELECT. In de Engelstalige versie wordt dat dan: SUM, AVERAGE en RAND. Gebruik je de verkeerde benaming, dan wordt een foutboodschap getoond (#NAAM? of #NAME?). Veelal wordt de taal van het besturingssysteem (bijvoorbeeld Windows 10) overgenomen. Deze systeeminstellingen bepalen eveneens het symbool dat gebruikt wordt voor het decimaalteken, het lijtscheidingsteken en de datumnotatie. In het Nederlands is dit een komma, een puntkomma en een opmaak als dd-mm-jjjj. In het Engels gebruikt men een punt als decimaalteken en een komma als lijtscheidingsteken. De datumopmaak is veelal als mm-dd, yyyy. In Windows 10 kan je deze instellingen veranderen in Start ► Settings ► Time & Language (Start ► Instellingen ► Tijd en Taal).

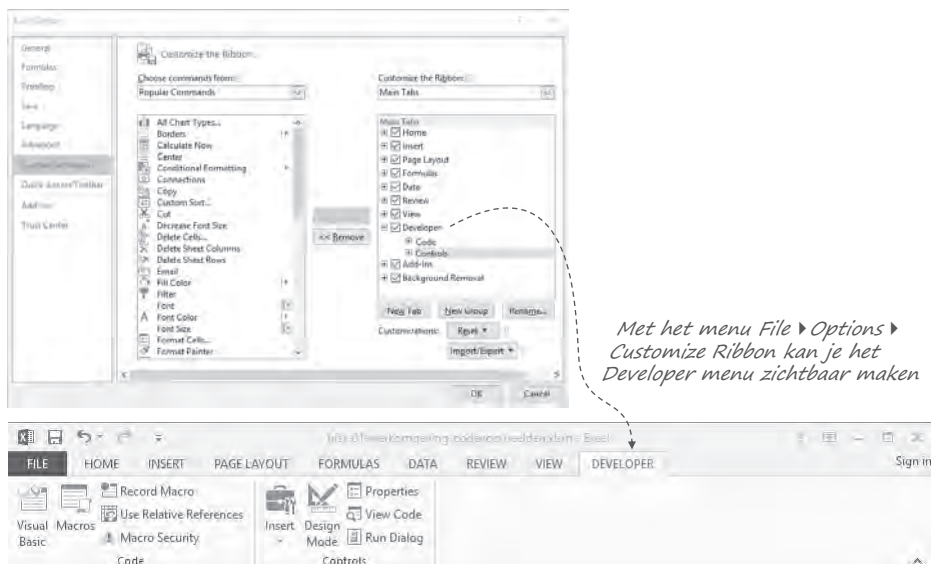
In dit boek gaan we uit van een Engelstalige (UK) installatie. Dit is gedaan omdat de Excel VBA-taal sowieso Engels is. Ook heb je meer kans om relevante hulp op het internet te vinden met Engelstalige zoektermen. Het is daarom handiger als je weet dat Excel over een 'Pivot table' beschikt dan dat je de Nederlandse term Draaitabel kent. In dit boek gebruiken we dus de Engelstalige benamingen van de menu's en functies maar krijg je, waar nodig, bij het eerste gebruik het Nederlandstalige equivalent tussen haakjes.

Tip

Onthoud dat we in de voorbeelden een punt gebruiken als decimaal-teken, een komma als scheidingsteken en telkens de Engelstalige menu's en functies tonen.

1.4.2 Zichtbaar maken van het menu Developer (Ontwikkelaars)

Microsoft gaat er eigenlijk van uit dat een modale Excel-gebruiker geen VBA-programma's zal schrijven. Het menu om dit makkelijk te realiseren is namelijk niet standaard geactiveerd. Je kan het menu Developer (Ontwikkelaars) zichtbaar maken via **File ► Options ► Customize Ribbon** (Bestand ► Opties ► Lint aanpassen). Plaats een vinkje in het vak aan de rechterkant naast Developer. In dit boek gebruiken we enkel de opties **Code** (Programmacode) en **Controls** (Besturingselementen). Vink de beide andere opties (**XML** en **Add-ins**) aan en klik op de knop **Remove** (Verwijderen) om ze uit het menu te halen (zie figuur 1.5).



Figuur 1.5 Het menu Developer zonder de opties XML en Add-ins (Invoegtoepassingen)

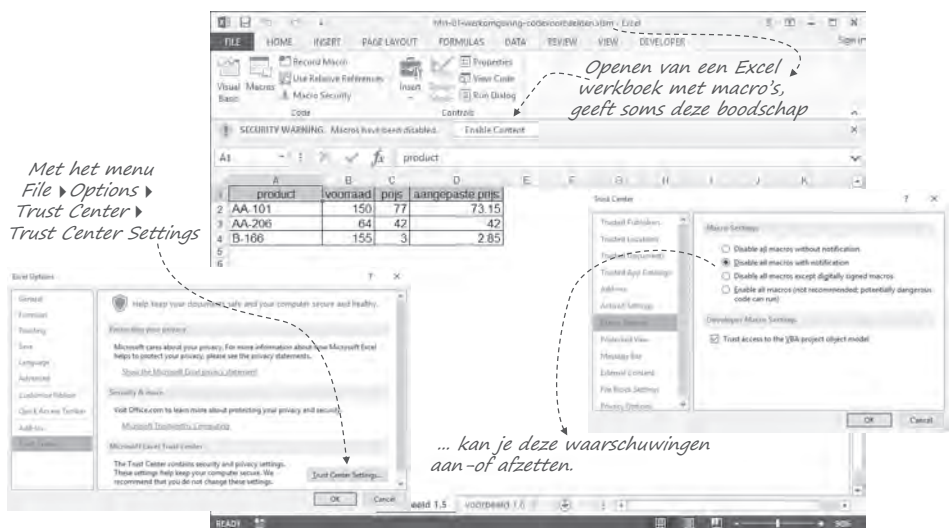
1.4.3 Aanpassen Macro Security (Macrobeveiliging)

Het inbouwen van een programmeertaal in een applicatiepakket zoals Excel en Word zet ook de deur open voor allerlei vormen van misbruik. Men kan nuttige macro's schrijven in Excel VBA, maar evengoed kan men macro's schrijven die gevaarlijk zijn, bijvoorbeeld omdat ze documenten verwijderen of ongewenste e-mail versturen. Met het krachtiger worden van de VBA-programmeeromgeving is het aantal Word- en Excel-virussen dan ook toegenomen.

Tip

In juli 1996 werd het eerste Excel-virus, LAROUX genaamd, ontdekt. Dit virus was weliswaar niet destructief (het beschadigde bijvoorbeeld geen bestanden) maar verspreidde zichzelf wel en besmette alle documenten van Excel versie 5. Sindsdien zijn er tientallen virussen verschenen waarvan sommige niet zo onschadelijk waren.

Microsoft heeft deze bedreiging ingezien en vanaf Excel 97 is er de mogelijkheid om het openen van spreadsheets met (verborgen) macro's te verhinderen. Standaard staat dit ingesteld op Disable all macros with notification (met de verwarrende Nederlandse vertaling 'Alle macro's zonder melding uitschakelen'). Dit wil zeggen dat je een bericht krijgt bij het openen van een werkboek dat een macro bevat. Je dient dan expliciet de macro's beschikbaar te maken via de knop Enable (Inhoud inschakelen) (zie figuur 1.6).



Figuur 1.6 Waarschuwing bij het (voor de eerste maal) openen van een werkboek met macro's bij de instelling Disable all macros with notification

Deze instelling wordt onthouden, zodat je dit de volgende keren niet meer hoeft te doen. Jouw Excel-document wordt dan bij de Trusted Documents (Vertrouwde documenten) geplaatst. Het kan ook zijn dat je heel strikte instellingen hebt op jouw pc: Disable all macros without notification (Alle macro's uitschakelen zonder melding). Je Excel map wordt dan geopend zonder de macro's en zonder dat je een melding krijgt. Je kan het niveau van beveiliging instellen via het menu Developer ► Security (Ontwikkelaars ► Macrobeveiliging).

De makkelijkste manier wellicht om de voorbeelden uit dit boek uit te proberen is echter om een aparte map aan te maken op je systeem en deze als Trusted Locations (Vertrouwde locaties) aan te merken. Je kan dit doen via Developer ► Security ► Trusted locations (Ontwikkelaars ► Macrobeveiliging ► Vertrouwde locaties).

Excel-bestanden met macro's kunnen ook alleen opgeslagen worden als *.xlsm bestanden (macro-enabled workbook). Normale Excel-workbooks sla je op onder de extensie *.xlsx.

1.4.4 Aanpassen van enkele werkmapopties

Standaard is Excel zo ingesteld dat als je een gegeven in een cel typt en dan op de Enter-toets drukt, de cursor één cel lager wordt geplaatst. Meestal vergemakkelijkt dit de gegevensinvoer, maar bij het opnemen van een macro (zie hoofdstuk 2) zal deze instelling je parten kunnen spelen. Het verplaatsen van de cursor wordt immers ook opgenomen. Je kan deze instelling uitschakelen via het menu File ► Options ► Advanced ► Move selection after ENTER (Bestand ► Opties ► Geavanceerd ► Selectie verplaatsen nadat ENTER is ingedrukt).

1.4.5 Instellen van de VBA-editor

De Visual Basic Editor is een specifieke omgeving waarin je makkelijk kan programmeren. Deze editor heeft zelf ook enkele instellingen. Je kan deze veranderen in de editor zelf (dus niet in Excel met het menu File ► Options). Start eerst de editor met Developer ► Visual Basic (Ontwikkelaars ► Visual Basic) en klik dan op het menu Tools ► Options (Extra ► Opties).

Op de tab Editor schakel je bij voorkeur de optie Require variable declaration (Variabelen declareren vereist) in. Hierdoor ben je verplicht om alle variabelen die je zal gebruiken expliciet te benoemen. In het begin geeft deze verplichting wel wat meerwerk, maar uiteindelijk zal het je vele uren spuurwerk besparen door moeilijk te vinden fouten te voorkomen, zoals het verkeerd spellen van een naam. Deze optie staat standaard niet aan.

Tip

Aangezien je bij het lezen van dit boek waarschijnlijk regelmatig een macro zal schrijven of een van de voorbeeldmacro's zal openen, wordt aanbevolen de instellingen aan te passen zoals hier beschreven. Op de website www.aandeslagmetexcelvba.nl vind je een filmpje waarin dit stap voor stap wordt getoond.

Dit boek is bestemd voor studenten in het hoger onderwijs en professionals die programma's of macro's willen schrijven in Excel VBA. Het veronderstelt geen voorkennis of ervaring met programmeren.

Het boek combineert het beste van twee werelden. Klassieke programmeerconcepten worden uitgebreid besproken en voorzien van voorbeelden, maar ook de kleinste details van de taal worden toegelicht. Bovendien werk je in een omgeving die je wellicht al gebruikt: Microsoft Excel.

Er wordt gewerkt aan de hand van een groot aantal oefeningen die stap voor stap worden opgebouwd en die resulteren in een uitgewerkte code. Je haalt het meeste uit dit boek door de voorbeelden en oefeningen na te maken op de computer. Neem het boek bij de hand en begin!

Op **www.aandeslagmetexcelvba.nl** vind je de uitwerking van alle voorbeelden en oefeningen uit het boek, net als extra toetsen per hoofdstuk.

Je kan dit boek gebruiken in combinatie met de meest gangbare versies van Microsoft Excel (Excel 2010, 2013, 2016, 2019 en Office 365).



9 789024 402052

www.aandeslagmetexcelvba.nl
www.boomhogeronderwijs.nl

42 7 5255989438 7
678 83 492760005011
37447254697 820501
02307076476315 088
65692177846 510073
02469383 405440655
33 938163160 500 8
681 7065848693236
1027 4 132960205
396182 96081056788
482739 9342 5148 4
740679 566998397
3950 2 21 56589403
89993442874815398
18076341611 631 96
345988881802075 74
77007178308980541
7 44 246 3 728447
572129005982744 21
159 114871 1 8079
7286 3939198 1054
209079163596 86902
149934 45 92 01805
26092691723366602
30471 966396085509
024 1321 193998003
0902867 2615476402
011420488 26571974
812720459223390 9
342515661272 25249
4140885 8 4169 3
03760 170 02768 92
42904 99 977444 19
481672121984 99425
03114 3981 8764717
810 70 32090864869
311977646141 95373
8597 8837535571 2
53497 619351046700
38551552684 04 6