Príklady makier v Exceli

Stručný pohľad na premenné

Premenné môžete považovať za pamäťové kontajnery, ktoré môžete použiť vo vašich procedúrach. Existujú rôzne typy premenných, pričom každá je zodpovedná za uchovávanie špecifického typu údajov. Niektoré bežné typy premenných sú:

- ✓ **String:** Uchováva textové údaje.
- ✓ **Integer:** Uchováva numerické údaje v rozsahu od –32,768 to 32,767
- ✓ **Long:** Uchováva numerické údaje v rozsahu od –2,147,483,648 to 2,147,483,647
- ✓ **Double:** Uchováva numerické údaje s pohyblivou desatinnou čiarkou.
- ✓ **Variant:** Uchováva ľubovoľný typ údajov.
- ✓ **Boolean:** Uchováva binárne údaje, ktoré vrátia hodnotu True alebo False.
- ✓ **Object:** Uchováva skutočný objekt z objektového režimu v aplikácii Excel.

Termín používaný pre vytvorenie premennej v makre je deklarovanie premennej. Urobíte to tak, že zadáte "Dim" (skratka pre dimension), názov vašej premennej a potom jej typ. Napríklad:

- Dim MyText as String
- Dim MyNumber as Integer
- Dim MyWorksheet as Worksheet

Príklady makier

Makro 1: Vytvorenie nového súboru

Možno budete občas potrebovať vytvoriť nový zošit automatickým spôsobom. Napríklad môžete potrebovať skopírovať údaje z tabuľky a vložiť ich do nového zošitu. Nasledujúce makro kopíruje rozsah buniek z aktívneho hárku a vkladá údaje do nového zošitu.

```
Sub Makro1 ()
        ‘Krok 1 Skopírovať údaje
         Sheets(“Example 1”).Range(“B4:C15”).Copy 'Treba zadefinovať vlastné názvy a rozsahy
        ‘Krok 2 Vytvoriť nový zošit
         Workbooks.Add
        ‘Krok 3 Prilepiť údaje
         ActiveSheet.Paste Destination:=Range(“A1”)
        ‘Krok 4 Vypnúť upozornenia aplikácie
        Application.DisplayAlerts = False
        ‘Krok 5 Uložiť novovytvorený zošit
        ActiveWorkbook.SaveAs _Filename:=”C:\Temp\MyNewBook.xlsx”
        ‘Krok 6 Znova zapnúť upozornenia aplikácie
         Application.DisplayAlerts = True
End Sub
```

Tu je, ako funguje toto makro:

1. V Kroku 1 jednoducho skopírujeme údaje, ktoré sa nachádzajú v rozsahu buniek od B4 do C15. Dôležité je si všimnúť, že špecifikujete nielen hárok, ale aj rozsah pomocou názvu. Toto je osvedčená prax pri práci s viacerými otvorenými zošitmi naraz.
2. Používame metódu Add objektu Workbook na vytvorenie nového zošitu. To je ekvivalent manuálneho kliknutia na Súbor ➜ Nový ➜ Prázdny dokument v paneli nástrojov programu Excel.
3. V tomto Kroku používate metódu Paste na odoslanie skopírovaných údajov do bunky A1 nového zošitu. Venujte pozornosť skutočnosti, že kód odkazuje na objekt ActiveSheet. Keď pridáte zošit, nový zošit okamžite získava pozornosť, stáva sa aktívnym zošitom. Ide o rovnaké správanie, ktoré by ste videli pri manuálnom pridaní zošitu.
4. V Kroku 4 kódu nastavujeme metódu DisplayAlerts na False, čo efektívne vypína upozornenia programu Excel. Robíme to preto, že v ďalšom kroku kódu ukladáme novovytvorený zošit. Toto makro môžeme spustiť viackrát a v takom prípade sa program Excel pokúsi súbor uložiť viackrát. Čo sa stane, keď sa pokúsite uložiť zošit viackrát? Excel vás upozorní, že s týmto názvom už existuje súbor, a potom sa opýta, či chcete prejsť na predchádzajúci existujúci súbor. Pretože vašim cieľom je automatizovať vytváranie nového zošitu, chcete toto upozornenie potlačiť.
5. V Kroku 5 uložíme súbor pomocou metódy SaveAs. Všimnite si, že zadávame celú cestu k umiestneniu uloženia vrátane konečného názvu súboru.
6. Pretože sme v Kroku 4 vypli upozornenia aplikácie, musíme ich znova zapnúť. Ak to neurobíme, Excel bude pokračovať v potláčaní všetkých upozornení počas aktuálnej relácie.

Makro 2: Ochrana zošit pred uzatvorením

Niekedy je potrebné poslať váš zošit do sveta s určitými chránenými hárkami. Ak zistíte, že neustále chránite a odstraňujete ochranu listov pred distribúciou vašich zošitov, toto makro vám môže pomôcť.

Tento kód je spustený udalosťou BeforeClose zošitu. Keď sa pokúšate zatvoriť zošit, táto udalosť sa aktivuje, spúšťajúc kód v nej. Makro automaticky chráni určený hárok zadaným heslom a potom uloží zošit.

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
        'Krok 1: Ochrana hárku pomocou hesla
         Sheets("Sheet1").Protect Password:="RED"
        'Krok 2: Uloženie zošita
        ActiveWorkbook.Save
End Sub
```

1. V Kroku 1 explicitne špecifikujeme, ktorý hárok chceme chrániť - v tomto prípade Hárok1. Taktiež nastavujeme heslo, Password:=RED. Týmto definujeme heslo potrebné na odstránenie ochrany. Tento argument hesla je úplne voliteľný. Ak ho úplne vynecháte, hárok bude stále chránený, ale na odstránenie ochrany heslo nebudete potrebovať. Taktiež si uvedomte, že heslá v Exceli sú citlivé na veľkosť písmen, takže si treba dávať pozor na presné heslo a veľkosť písmen, ktoré používate.
2. V Kroku 2 hovoríme Excelu, aby uložil zošit. Ak neuložíme zošit, ochrana hárka, ktorú sme práve aplikovali, nebude mať účinnosť pri ďalšom otvorení zošitu.

Makro 3: Vytvorenie zálohy aktuálneho zošitu s dnešným dátumom

Mali by ste vedieť, že vytváranie záloh vášho súboru je veľmi dôležité. Teraz môžete nechať makro túto úlohu urobiť za vás. Toto jednoduché makro uloží váš zošit do nového súboru s dnešným dátumom ako súčasťou názvu.

Tajomstvom tohto makra je poskladať nový názov súboru. Nový názov súboru má tri časti: cestu, dnešný dátum a pôvodný názov súboru.

Cestu zachytíme pomocou vlastnosti Path objektu ThisWorkbook. Dnešný dátum získame pomocou funkcie Date.

Uvidíte, že formátujeme dátum (Format(Date, "mm-dd-yy")). To preto, že funkcia Date prednastavene vracia formát mm/dd/yyyy. Používame pomlčky namiesto lomiek, pretože lomky by spôsobili, že uloženie súboru by zlyhalo. (Windows neumožňuje lomky v názvoch súborov.)

Poslednou časťou nového názvu súboru je pôvodný názov súboru. Používame vlastnosť Name objektu ThisWorkbook na zachytenie tohto názvu:

```
Sub Makro3 ()
    'Krok 1: Uložiť súbor s novým názvom
        ThisWorkbook.SaveCopyAs _Filename:=ThisWorkbook.Path & "\" & _
    Format(Date, "mm-dd-yy") & " " & _
    ThisWorkbook.Name
End Sub
```

V jedinom Kroku makro vytvára nový názov súboru a používa metódu SaveCopyAs na uloženie súboru.

Makro 4: Vytvorenie nových zošitov pre každý hárok zvlášť

Mnohí analytici v Exceli potrebujú analyzovať svoje zošity a rozdeliť ich do samostatných zošitov pre každý hárok. Inými slovami, potrebujú vytvoriť nový zošit pre každý zo svojich existujúcich hárkov. Viete si predstaviť, aký by to bol otravný úkon, ak by ste to mali robiť manuálne. Nasledujúce makro pomáha automatizovať túto úlohu.

V tomto makre opakujete hárky, každý hárok kopírujete a potom kópiu odosielate do nového zošitu, ktorý sa vytvára na mieste. Dôležité je si všimnúť, že novovytvorené zošity sa ukladajú do rovnakého adresára ako váš pôvodný zošit, s rovnakým názvom ako skopírovaný hárok (wb.SaveAs ThisWorkbook.Path & "" & ws.Name).

```
Sub Makro4 ()
    'Krok 1: Deklarácia všetkých premenných.
        Dim ws As Worksheet
        Dim wb As Workbook
    'Krok 2: Začnite opakovaným prechádzaním cez hárky.
        For Each ws In ThisWorkbook.Worksheets
    'Krok 3: Vytvorenie a uloženie nového zošita
        Set wb = Workbooks.Add
        wb.SaveAs ThisWorkbook.Path & "\" & ws.Name
    'Krok 4: Skopírovanie cieľového hárku do nového zošita
        ws.Copy Before:=wb.Worksheets(1)
        wb.Close SaveChanges:=True
    'Krok 5: Opäť prejde na ďalší hárok.
```

```
        Next ws
End Sub
```

Nie všetky platné názvy hárkov sa dajú preložiť na platné názvy súborov. Windows má špecifické pravidlá, ktoré vám bránia v pomenovaní súborov určitými znakmi. Tieto znaky nemôžete použiť pri pomenovaní súboru: spätné lomítko (), lomítko (/), dvojbodka (:), hviezdička (*), otáznik (?), vertikálna čiara (|), dvojitá úvodzovka (“), väčšie než (>), a menšie než (<). Problém spočíva v tom, že môžete použiť niektoré z týchto obmedzených znakov vo svojich názvoch hárkov; konkrétne dvojitú úvodzovku, vertikálnu čiaru (|), väčšie než (>), a menšie než (<).
Keď spúšťate toto makro a snažíte sa pomenovať novovytvorené súbory podľa názvu hárka, môže dôjsť k chybe. Napríklad makro vyvolá chybu pri vytváraní nového súboru z hárka s názvom May| Revenue (kvôli znaku vertikálnej čiary). Skrátka povedané, vyhnite sa pomenovávaniu svojich hárkov týmito obmedzenými znakmi.

1. V Kroku 1 sú deklarované dve objektové premenné. Premenná ws vytvára pamäťový kontajner pre každý hárok, ktorý makro prechádza. Premenná wb vytvára kontajner pre nové zošity, ktoré vytvárame.
2. V Kroku 2 makro začína opakovaným prechádzaním hárkov. Použitie objektu ThisWorkbook zabezpečuje, že aktívny hárok, ktorý sa kopíruje, pochádza zo zošitu, v ktorom je uložený kód, a nie z nového zošitu, ktorý sa vytvára.
3. V Kroku 3 vytvárame nový zošit a uložíme ho. Tento nový zošit ukladáme do rovnakého adresára ako pôvodný zošit (ThisWorkbook). Názov súboru je nastavený na rovnaký názov ako aktuálne aktívny hárok.
4. Kroku 4 kopíruje práve aktívny hárok a pomocou parametra Before ho posiela do nového zošitu ako prvý hárok.
5. Kroku 5 sa opäť vracia k ďalšiemu hárku. Po vyhodnotení všetkých hárkov makro končí.

Makro 5: Vytvorenie obsahu z pracovných hárkov

Mimo triedenia hárkov je vytvorenie obsahu pre hárky v zošite jedným z najčastejšie požadovaných Excelových makier. Často pracujeme so súbormi, ktoré majú viac hárkov, ako je možné jednoducho vidieť alebo navigovať. Obsah určite pomáha. Nasledujúce makro nielenže vytvorí zoznam názvov z hárkov v zošite, ale pridá aj hypertextové odkazy, takže môžete ľahko preskočiť na hárok jednoduchým kliknutím.

```
Sub Makro5 ()
    ‘Krok 1: Deklarácia premenných
        Dim i As Long
    ‘Krok 2: Vymazanie predošlých obsahov ak existujú
        On Error Resume Next
        Application.DisplayAlerts = False
        Sheets(“Table Of Contents”).Delete
        Application.DisplayAlerts = True
        On Error GoTo 0
    ‘Krok 3: Pridanie nového obsahu do prvého hárku
        ThisWorkbook.Sheets.Add _
        Before:=ThisWorkbook.Worksheets(1)
        ActiveSheet.Name = “Table Of Contents”
```

```
    'Krok 4: Spustí sa počítadlo premennej i
        For i = 1 To Sheets.Count
    'Krok 5: Výber nasledovného vhodného riadku
        ActiveSheet.Cells(i, 1).Select
    'Krok 6: Pridanie názvu hárku a prepojenia
        ActiveSheet.Hyperlinks.Add _
        Anchor:=ActiveSheet.Cells(i, 1), _
        Address:="", _
        SubAddress:="'" & Sheets(i).Name & "'!A1", _
        TextToDisplay:=Sheets(i).Name
    'Krok 7: Vráť sa späť a zvýš premennú i
        Next i
End Sub
```

1. Kroku 1 deklaruje celočíselnú premennú s názvom i, ktorá slúži ako počítadlo počas opakovania makra cez hárky. V tomto postupe používame počítadlo (našu premennú i). Hlavným dôvodom je, že musíme nielen sledovať hárky, ale musíme aj zabezpečiť, aby sme zaznamenali každý názov hárka na novom riadku v obsahu. Myšlienka spočíva v tom, že keď počítadlo postupuje cez hárky, slúži aj na posunutie kurzora nadol v obsahu, takže každý nový záznam ide na nový riadok.
2. V Kroku 2 sa v podstate pokúšame odstrániť akýkoľvek predchádzajúci hárok s názvom Obsah. Pretože nemusí existovať žiadny hárok s obsahom na odstránenie, musíme začať Krokom 2 s obsluhou chýb On Error Resume Next. To hovorí Excelu, aby pokračoval v makre, ak sa tu vyskytne chyba. Potom odstránime hárok s obsahom pomocou metódy DisplayAlerts, ktorá efektívne vypne upozornenia programu Excel, takže nemusíme potvrdiť odstránenie. Napokon obnovíme obsluhu chýb, aby sme chyby znova zachytili, vložením On Error GoTo 0.
3. V Kroku 3 pridáme nový hárok do zošitu pomocou argumentu Before na umiestnenie nového hárka ako prvého. Potom pomenujeme hárok Obsah. Ako sme už uviedli v tejto časti, keď pridáte nový hárok, automaticky sa stáva aktívnym hárkom. Keďže tento nový hárok má zameranie počas celého postupu, všetky odkazy na ActiveSheet v tomto kóde sa vzťahujú na hárok s obsahom.
4. Kroku 4 spúšťa i odpočítava od 1 a končí ho na maximálnom počte všetkých hárkov v zošite. Opäť, namiesto opakovania cez kolekciu hárkov, ako sme to urobili v predchádzajúcich makrách, jednoducho používame i ako indexové číslo, ktoré môžeme odovzdať objektu Sheets. Keď sa dosiahne maximálny počet, makro končí.
5. Krok 5 vyberá príslušný riadok v hárku s obsahom. Inými slovami, ak je počítadlo i na 1, vyberie prvý riadok v hárku s obsahom. Ak je počítadlo i na 2, vyberie druhý riadok a tak ďalej. Toto dokážeme pomocou položky Cells. Položka Cells poskytuje veľmi užitočný spôsob výberu rozsahov cez kód. Vyžaduje len relatívne pozície riadkov a stĺpcov ako parametre. Takže Cells(1,1) zodpovedá riadku 1, stĺpca 1 (alebo bunka A1). Cells(5, 3) zodpovedá riadku 5, stĺpca 3 (alebo bunka C5). Číselné parametre v položke Cells sú obzvlášť užitočné, keď chcete prechádzať sériou riadkov alebo stĺpcov pomocou inkrementujúceho indexového čísla.
6. Krok 6 používa metódu Hyperlinks.Add na pridanie názvu hárka a hypertextových odkazov do vybranej bunky. Tento krok poskytuje metóde Hyperlinks.Add parametre, ktoré potrebuje na vytvorenie hypertextových odkazov.

7. Posledný krok v makre sa vráti k zvýšeniu počítadla i na ďalší počet. Keď sa počítadlo i dostane na číslo, ktoré sa rovná počtu hárkov v zošite, makro končí.

Makro 6: Vyznačenie aktívneho riadku a stĺpca

Keď sa pozriete na tabuľku s číslami, bolo by pekné, ak by Excel automaticky zvýraznil riadok a stĺpec, na ktorom práve ste. Tento efekt vám poskytuje vizuálnu pomôcku pre orientáciu hore a dole po stĺpci, ako aj vľavo a vpravo po riadku. Nasledujúce makro umožňuje dosiahnutie tohto efektu jednoduchým dvojitým kliknutím. Keď je makro nastavené, Excel zvýrazní riadok a stĺpec pre aktívnu bunku, čo výrazne zlepšuje vašu schopnosť zobraziť a upravovať veľký rozsah.

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, Cancel As Boolean)
        ‘Krok 1: Deklarácia premennej
                Dim strRange As String
        ‘Krok2: Vytvoriť reťazec rozsahu.
                strRange = Target.Cells.Address & “,” & _
                Target.Cells.EntireColumn.Address & “,” & _
                Target.Cells.EntireRow.Address
        ‘Krok 3: Preniesť reťazec rozsahu do rozsahu.
                 Range(strRange).Select
End Sub
```

1. Najprv deklarujeme objekt nazvaný strRange. Týmto vytvárame pamäťový kontajner, ktorý môžeme použiť na vytvorenie reťazca rozsahu.
2. Reťazec rozsahu nie je nič iné ako adresa rozsahu. "A1" je reťazec rozsahu, ktorý ukazuje na bunku A1. "A1:G5" je tiež reťazec rozsahu; tento ukazuje na rozsah buniek zahŕňajúci bunky od A1 po G5. V kroku 2 vytvárame reťazec rozsahu, ktorý zahŕňa dvojkliknutú bunku (v tomto makre nazývanú Target), celý aktívny riadok a celý aktívny stĺpec. Vlastnosť Address pre tieto tri rozsahy sú zachytené a poskladané do premennej strRange.
3. V kroku 3 používame premennú strRange ako adresu pre príkaz Range.Select. Toto je riadok kódu, ktorý nakoniec zvýrazní dvojkliknutý výber.

Makro 7: Vymazanie prázdnych riadkov

Keď pracujte s Excelom dostatočne dlho, zistíte, že prázdne riadky často môžu spôsobovať problémy na mnohých úrovniach. Mohli by spôsobiť problémy so vzorcami, predstavovať riziko pri kopírovaní a vkladaní a niekedy spôsobovať zvláštne správanie v kontingenčných tabuľkách. Ak zistíte, že manuálne vyhľadávate a odstraňujete prázdne riadky vo svojich údajoch, toto makro môže pomôcť automatizovať túto úlohu.

V tomto makre používame vlastnosť UsedRange objektu Activesheet na definovanie rozsahu, s ktorým pracujeme. Vlastnosť UsedRange nám poskytuje rozsah, ktorý zahŕňa bunky, do ktorých boli zadané údaje. Potom zriaďujeme počítadlo, ktoré začína na poslednom riadku použitého rozsahu, aby sme skontrolovali, či je celý riadok prázdny. Ak je celý riadok skutočne prázdny, odstránime riadok. Tú istú operáciu odstránenia vykonávame v každej iterácii cyklu, pričom každým razom zvyšujeme počítadlo na predchádzajúci riadok.

```
Sub Makro7 ()
    ‘Krok1: Deklarovanie premenných.
        Dim MyRange As Range
        Dim iCounter As Long
    ‘Krok 2: Definovanie cieľového rozsahu.
        Set MyRange = ActiveSheet.UsedRange
    ‘Krok 3: Začať opačné opakovanie cez rozsah.
        For iCounter = MyRange.Rows.Count To 1 Krok -1
    ‘Krok 4: Ak je celý riadok prázdny, vymaže sa.
    If Application.CountA(Rows(iCounter).EntireRow) = 0 Then
    Rows(iCounter).Delete
    End If
    ‘Krok 5: Zvýšenie počítadla nadol.
    Next iCounter
End Sub
```

1. Makro najprv deklaruje dve premenné. Prvá premenná je objektová premenná nazývaná MyRange. Táto objektová premenná definuje náš cieľový rozsah. Druhá premenná je celočíselná premenná typu Long s názvom iCounter. Táto premenná slúži ako inkrementálne počítadlo.
2. V kroku 2 makro vyplní premennú MyRange vlastnosťou UsedRange objektu ActiveSheet. Vlastnosť UsedRange nám poskytuje rozsah, ktorý zahŕňa bunky, do ktorých boli zadané údaje. Treba poznamenať, že ak by sme chceli špecifikovať skutočný rozsah alebo pomenovaný rozsah, mohli by sme jednoducho zadať jeho názov - Range("MôjPomenovanýRozsah").
3. V tomto kroku makro nastavuje parametre prírastkového počítadla tak, aby začalo na maximálnom počte pre rozsah (MyRange.Rows.Count) a skončilo na 1 (prvý riadok zvoleného rozsahu). Treba poznamenať, že používame kvalifikátor kroku-1. Pretože špecifikujeme krok -1, Excel vie, že budeme inkrementovať počítadlo späť, pohybujúc sa o jeden prírastok pri každej iterácii. Celkovo krok 3 hovorí Excelu, aby začal na poslednom riadku zvoleného rozsahu a pohyboval sa späť, až kým sa nedostane na prvý riadok rozsahu. Pri práci s rozsahom môžete explicitne volať konkrétny riadok v rozsahu, prenášaním čísla indexu riadku do kolekcie Rows rozsahu. Napríklad Range("D6:D17"). Rows(5) ukazuje na piaty riadok v rozsahu D6:D17.
4. V kroku 4 makro používa premennú iCounter ako indexové číslo pre kolekciu Rows objektu MyRange. Tým sa presne určuje, s ktorým presným riadkom pracujeme v aktuálnom cykle. Makro skontroluje, či sú bunky v tomto riadku prázdne. Ak áno, makro odstráni celý riadok.
5. V kroku 5 sa makro opäť vráti k zvýšeniu počítadla nadol.

Makro 8: Vymazanie prázdnych stĺpcov

Podobne ako prázdne riadky, aj prázdne stĺpce majú potenciál spôsobiť nepredvídané chyby. Ak zistíte, že manuálne vyhľadávate a odstraňujete prázdne stĺpce vo svojich údajoch, toto makro môže automatizovať túto úlohu.

V tomto makre používame vlastnosť UsedRange objektu ActiveSheet na definovanie rozsahu, s ktorým pracujeme. Vlastnosť UsedRange nám poskytuje rozsah, ktorý zahŕňa bunky, do ktorých boli zadané údaje. Potom zriaďujeme počítadlo, ktoré začína na poslednom stĺpci

použitého rozsahu, aby sme skontrolovali, či je celý stĺpec prázdny. Ak je celý stĺpec skutočne prázdny, odstránime stĺpec. Tú istú operáciu odstránenia vykonávame v každej iterácii cyklu, pričom každým razom zvyšujeme počítadlo na predchádzajúci stĺpec.

```
Sub Makro8 ()
    'Krok1: Deklarovanie premenných
        Dim MyRange As Range
        Dim iCounter As Long
    'Krok 2: Definovanie cieľového rozsahu
        Set MyRange = ActiveSheet.UsedRange
    'Krok 3: Začať opačné opakovanie cez rozsah.
        For iCounter = MyRange.Columns.Count To 1 Krok -1
    'Krok 4: Ak je delý stĺpec prázdny tak sa vymaže
         If Application.CountA(Columns(iCounter).EntireColumn) = 0 Then
        Columns(iCounter).Delete
        End If
    'Krok 5: Zvýšenie počítadla nadol.
        Next iCounter
End Sub
```

1. V kroku 1 sa najprv deklarujú dve premenné. Prvá premenná je objektová premenná nazývaná MyRange. Ide o objektovú premennú, ktorá definuje cieľový rozsah. Druhá premenná je celočíselná premenná typu Long s názvom iCounter. Táto premenná slúži ako prírastkové počítadlo.
2. V kroku 2 sa premenná MyRange naplní vlastnosťou UsedRange objektu ActiveSheet. Vlastnosť UsedRange nám poskytuje rozsah, ktorý zahŕňa bunky, do ktorých boli zadané údaje. Treba poznamenať, že ak by sme chceli špecifikovať skutočný rozsah alebo pomenovaný rozsah, mohli by sme jednoducho zadať jeho názov - Range("MôjPomenovanýRozsah").
3. V tomto kroku makro nastavuje parametre inkrementálneho počítadla tak, aby začalo na maximálnom počte pre rozsah (MyRange.Columns.Count) a skončilo na 1 (prvý stĺpec zvoleného rozsahu). Treba poznamenať, že používame kvalifikátor kroku-1. Pretože špecifikujeme krok -1, Excel vie, že budeme inkrementovať počítadlo späť, pohybujúc sa o jeden prírastok pri každej iterácii. Celkovo krok 3 hovorí Excelu, že chceme začať na poslednom stĺpci zvoleného rozsahu a pohybovať sa späť, až kým sa nedostane na prvý stĺpec rozsahu.
4. Pri práci s rozsahom môžete explicitne volať konkrétny stĺpec v rozsahu, prenášaním čísla indexu stĺpca do kolekcie Columns rozsahu. Napríklad Range("A1:D17").Columns(2) ukazuje na druhý stĺpec v rozsahu (stĺpec B). V kroku 4 makro používa premennú iCounter ako indexové číslo pre kolekciu Columns objektu MyRange. Tým sa presne určuje, s ktorým stĺpcom pracujeme v aktuálnom cykle. Makro skontroluje, či sú všetky bunky v tomto stĺpci prázdne. Ak áno, makro odstráni celý stĺpec.
5. V kroku 5 makro sa opäť vráti k zvýšeniu počítadla nadol.

Makro 9: Odstránenie medzier zo všetkých buniek v rozsahu

Častým problémom pri importovaní údajov z iných zdrojov sú medzery na začiatku alebo na konci buniek. To znamená, že hodnoty, ktoré sú importované, obsahujú medzery na začiatku alebo na konci bunky. Toto samozrejme komplikuje použitie funkcií ako VLOOKUP alebo triedenie. Tu je makro, ktoré umožňuje ľahko vyhľadávať a odstraňovať nadbytočné medzery vo vašich bunkách. V tomto makre prechádzame cez cieľový rozsah a prechádzame každou bunkou v tomto rozsahu prostredníctvom funkcie Trim.

```
Sub Makro9 ()
    'Krok 1:Deklarovanie premenných
        Dim MyRange As Range
        Dim MyCell As Range
    'Krok 2: Chceme uložiť zošit pred ostránením medzier?
        Select Case MsgBox("Can't Undo this action. " & _
        "Save Workbook First?", vbYesNoCancel)
        Case Is = vbYes
        ThisWorkbook.Save
        Case Is = vbCancel
        Exit Sub
        End Select
    'Krok 3: Definovanie cieľového rozsahu
        Set MyRange = Selection
    'Krok 4: Začať opakované prechádzanie rozsahom
        For Each MyCell In MyRange
    'Krok 5: Odstránenie medzier
        If Not IsEmpty(MyCell) Then
        MyCell = Trim(MyCell)
        End If
    'Krok 6: Získať ďalšiu bunku v rozsahu
        Next MyCell
End Sub
```

1. Krokom 1 sa deklarujú dve objektové premenné typu Range, jedna nazvaná MyRange na uchovanie celého cieľového rozsahu a druhá nazvaná MyCell na uchovanie každej bunky v rozsahu, keď makro prechádza cez ne postupne.
2. Pri spustení makra sa vymaže zásobník na vrátenie späť. Zmeny, ktoré makro vykoná, nie je možné vrátiť späť. Pretože reálne meníme údaje, musíme si pred spustením makra dať možnosť uložiť pracovný zošit. Krok 2 toto robí. Tu zobrazíme správu, ktorá sa pýta, či chceme najprv uložiť pracovný zošit. Potom nám dá tri možnosti: Áno, Nie a Zrušiť. Kliknutím na Áno uložíme pracovný zošit a pokračujeme s makrom. Kliknutím na Zrušiť ukončíme postup bez spustenia makra. Kliknutím na Nie spustíme makro bez uloženia pracovného zošitu.
3. Krok 3 naplní premennú MyRange cieľovým rozsahom. V tomto príklade používame vybraný rozsah - rozsah, ktorý bol vybraný na tabuľke. Môžete ľahko nastaviť premennú MyRange na konkrétny rozsah, ako je Range("A1:Z100"). Ak je váš cieľový rozsah pomenovaným rozsahom, jednoducho zadajte jeho názov - Range("MôjPomenovanýRozsah").

4. Krok 4 začne opakované prechádzanie každou bunkou v cieľovom rozsahu, pričom každú bunku aktivuje.
5. Po aktivácii bunky makro používa funkciu IsEmpty na overenie, či bunka nie je prázdna. To robíme pre úsporu výkonu tým, že preskočíme bunku, ak nie je nič v nej. Potom prechádzame hodnotu tejto bunky funkciou Trim. Funkcia Trim je natívnou funkciou programu Excel, ktorá odstráni vedúce a koncové medzery.
6. Krok 6 sa opäť vráti, aby získal ďalšiu bunku. Po aktivácii všetkých buniek v cieľovom rozsahu končí makro.

Makro 10: Skopírovanie vyfiltrovaných riadkov do nového zošita

Často, keď pracujete s množinou údajov, ktorá je filtrovaná automaticky, chcete extrahovať filtrované riadky do nového zošitu. Samozrejme, môžete manuálne kopírovať filtrované riadky, otvoriť nový zošit, vložiť riadky a potom formátovať novo vložené údaje, aby sa všetky stĺpce zmestili. Ale ak to robíte dostatočne často, možno by ste chceli mať makro, ktorý proces zrýchli. Toto makro zachytí rozsah autofiltrovania, otvorí nový zošit a potom vloží údaje.

```
Sub Makro10 ()
    'Krok 1: Skontroluj autofilter – ukonči ak neexistuje
        If ActiveSheet.AutoFilterMode = False Then
        Exit Sub
        End If
    'Krok 2: Skopíruj rozsah filtrovania do nového zošita
        ActiveSheet.AutoFilter.Range.Copy
        Workbooks.Add.Worksheets(1).Paste
    'Krok 3: Nastavenie automatickej šírky stĺpcov
        Cells.EntireColumn.AutoFit
End Sub
```

1. Krok 1 používa vlastnosť AutoFilterMode na overenie, či je na databáze aplikované autofiltrovanie. Ak nie je, ukončíme postup.
2. Každý objekt AutoFilter má vlastnosť Range. Táto vlastnosť Range nám vráti riadky, na ktoré sa aplikuje autofiltrovanie, čo znamená, že vráti len riadky, ktoré sú zobrazené vo filtrovanom súbore údajov. V kroku 2 používame metódu Copy na zachytenie týchto riadkov a potom vložíme tieto riadky do nového zošitu. Treba poznamenať, že používame Workbooks.Add.Worksheets(1). Týmto sa hovorí Excelu, aby vložil údaje do prvého listu novovytvoreného zošitu.
3. Krok 3 jednoducho hovorí Excelu, aby prispôsobil šírku stĺpcov tak, aby sa automaticky prispôsobila údajom, ktoré sme práve vložili.

Makro 11: Vytvorenie zoznamu kontingenčných tabuliek

Keď váš zošit obsahuje viacero kontingenčných tabuliek, často je užitočné mať súhrnný prehľad, ktorý popisuje základné informácie o kontingenčných tabuľkách. S takýmto súhrnom môžete rýchlo vidieť dôležité informácie, ako je umiestnenie každej kontingenčných tabuľky, umiestnenie zdrojových údajov každej kontingenčných tabuľky a index kontingenčnej vyrovnávacej pamäte, ktorý každá kontingenčná tabuľka používa.

Keď vytvoríte objektovú premennú kontingenčnej tabuľky, odhalíte všetky vlastnosti kontingenčnejtabuľky - vlastnosti ako je jej názov, umiestnenie, index vyrovnávacej pamäte a

podobne. V tomto makre prechádzame každou kontingenčnou tabuľkou v zošite a extrahujeme špecifické vlastnosti do nového listu. Pretože každý objekt kontingenčnej tabuľky je zvyčajne zvlášť na hárku musíme najprv prejsť hárkami v zošite a potom prejsť kontingenčnými tabuľkami na každom hárku. Venujte chvíľu podrobnému prechodu krokmi tohto makra.

```
Sub Makro11 ()
    'Krok 1: Deklarovanie premenných
        Dim ws As Worksheet
        Dim pt As PivotTable
        Dim MyCell As Range
    'Krok 2: Pridanie nového hárku s hlavičkou
        Worksheets.Add
        Range("A1:F1") = Array("Pivot Name", "Worksheet", _
        "Location", "Cache Index", _
        "Source Data Location", _
        "Row Count")
    'Krok 3: Začneme kurzorom na bunke A2 nastavením ukotvenia na tejto pozícii
        Set MyCell = ActiveSheet.Range("A2")
    'Krok 4: Prechádzame každý hárok v zošite.
        For Each ws In Worksheets
    'Krok 5: Prechádzame cez každú kontingenčú tabuľku
        For Each pt In ws.PivotTables
        MyCell.Offset(0, 0) = pt.Name
        MyCell.Offset(0, 1) = pt.Parent.Name
        MyRange.Offset(0, 2) = pt.TableRange2.Address
        MyRange.Offset(0, 3) = pt.CacheIndex
        MyRange.Offset(0, 4) = Application.ConvertFormula _
        (pt.PivotCache.SourceData, xlR1C1, xlA1)
        MyRange.Offset(0, 5) = pt.PivotCache.RecordCount
    'Krok 6: Posunieme kurzor o jeden riadok nižšie a nastavíme nové ukotvenie
        Set MyRange = MyRange.Offset(1, 0)
    'Krok 7: Prechádzanie cez všetky kontingenčné tabuľky a hárky
        Next pt
        Next ws
    'Krok 8: Nastavanie automatickej šírky stĺpcov
        ActiveSheet.Cells.EntireColumn.AutoFit
End Sub
```

1. Krok 1 deklaruje objekt s názvom ws. Týmto sa vytvára pamäťový kontajner pre každý hárok, cez ktorý prechádzame. Potom deklarujeme objekt s názvom pt, ktorý prechádza každú kontingenčnú tabuľku, cez ktorú prechádzame. Nakoniec vytvárame rozsahovú premennú s názvom MyCell. Táto premenná slúži ako náš kurzor, keď vyplňujeme súhrn inventára.
2. Krok 2 vytvára nový hárok a pridáva stĺpcové záhlavia, ktoré sú od bunky A1 po F1. Treba poznamenať, že môžeme pridávať stĺpcové záhlavia pomocou jednoduchého

poľa, ktoré obsahuje naše záhlavia. Tento nový hárok zostáva aktívnym hárkom od tohto okamihu.

3. Tak, ako by ste manuálne umiestnili kurzor do bunky, ak by ste začali písať údaje, v kroku 3 sa kurzor MyCell umiestni do bunky A2 aktívneho hárku. Toto je naša kotva, ktorá nám umožňuje odtiaľto navigovať. Počas makra vidíte použitie vlastnosti Offset. Vlastnosť Offset nám umožňuje posunúť kurzor o x-tý počet riadkov a x-tý počet stĺpcov od kotvy. Napríklad Range(A2).Offset(0,1) by posunulo kurzor o jednu bunku doprava. Ak by sme chceli posunúť kurzor o jednu bunku nadol, zadali by sme Range(A2).Offset(1, 0). V makre sa naviguje pomocou Offset na MyCell. Napríklad MyCell.Offset(0,4) by posunulo kurzor o štyri stĺpce doprava od kotvovej bunky. Po umiestnení kurzora môžeme vkladať údaje.
4. Krok 4 spúšťa opakovanie a hovorí Excelu, že chceme vyhodnotiť všetky hárky v tomto zošite.
5. Krok 5 prechádza všetky kontingenčné tabuľky v každom hárku. Pre každú kontingenčnú tabuľku, ktorú nájde, extrahuje príslušnú vlastnosť a vyplní tabuľku na základe pozície kurzora (pozri krok 3). Používame šesť vlastností kontingenčnej tabuľky: Name, Parent.Range, TableRange2.Address, CacheIndex, PivotCache.SourceData a PivotCache.Recordcount. Vlastnosť Name vráti názov kontingenčnej tabuľky. Vlastnosť Parent.Range nám poskytuje hárok, kde sa kontingenčná tabuľka nachádza. Vlastnosť TableRange2.Address vráti rozsah, v ktorom sa nachádza objekt kontingenčnej tabuľky. Vlastnosť CacheIndex vráti indexové číslo pivot cache pre kontingenčnú tabuľku. Pivot cache je pamäťový kontajner, ktorý uchováva všetky údaje o kontingenčnej tabuľke. Pri vytvorení novej kontingenčnej tabuľky Excel urobí snímku zdrojových údajov a vytvorí pivot cache. Pri každom obnovení kontingenčnej tabuľky sa Excel vráti k zdrojovým údajom a urobí ďalšiu snímku, čím obnoví pivot cache. Každý pivot cache má vlastnosť SourceData, ktorá identifikuje umiestnenie údajov, ktoré sa použijú na vytvorenie pivot cache. Vlastnosť PivotCache.SourceData nám hovorí, na ktorý rozsah sa obrátime, keď obnovíme kontingenčnú tabuľku. Môžete tiež získať počet záznamov zdrojových údajov pomocou vlastnosti PivotCache.Recordcount.
6. Pri každom stretnutí s novou kontingenčnou tabuľkou posúva makro kurzor MyCell o riadok nadol, efektívne začínajúc nový riadok pre každú kontingenčnú tabuľku.
7. Krok 7 hovorí Excelu, aby sa vrátil a iteroval cez všetky kontingenčné tabuľky a všetky hárky. Po vyhodnotení všetkých kontingenčných tabuliek sa presunieme na ďalší hárok. Po vyhodnotení všetkých hárkov sa makro presunie na posledný Krok.
8. Krok 8 zakončuje formátovaním, nastavuje veľkosť stĺpcov podľa údajov.

Makro 12: Funkcia na výpočet sumy podľa farby buniek

```
Function SumByFontColor(rng As Range, fontColorCell As Range) As Double
        Dim cell As Range
        Dim sum As Double
        Dim targetColor As Long
    Definuje farbu z bunky – dať vlastnú bunku s podfarbením
        targetColor = fontColorCell.Font.Color
        sum = 0
        For Each cell In rng
```

```
        'Kontroluje či sa farba zhoduje s cieľovou bunkou
                If cell.Font.Color = targetColor Then
                If IsNumeric(cell.Value) Then ' Kontroluje či bunka obsahuje číslo
                sum = sum + cell.Value
                End If
                End If
                Next cell
                SumByFontColor = sum
End Function
```

Makro 13: Funkcia na výpočet počtu buniek podľa farby

```
Function CountByFontColor(rng As Range, fontColorCell As Range) As Long
                Dim cell As Range
                Dim count As Long
                Dim targetColor As Long
        ' Definuje farbu z bunky – dať vlastnú bunku s podfarbením
                targetColor = fontColorCell.Font.Color
                count = 0
                For Each cell In rng
        'Kontroluje či sa farba zhoduje s cieľovou bunkou
                If cell.Font.Color = targetColor Then
                count = count + 1
                End If
                Next cell
                CountByFontColor = count
End Function
```

Makro 14: Zvýraznenie rozdielov medzi stĺpcami

```
Sub columnDifference ()
        Range("H7:H8,I7:I8").Select 'Treba zadefinovať vlastné porovnávané rozsahy
        Selection.ColumnDifferences(ActiveCell ).Select
        Selection.Style = "Bad"
End Sub
```

Makro 15: Zvýraznenie rozdielov medzi riadkami

```
Sub rowDifference ()
        Range("H7:H8,I7:I8").Select 'Treba zadefinovať vlastné porovnávané rozsahy
        Selection.RowDifferences(ActiveCell ).Select
        Selection.Style = "Bad"
End Sub
```