



11. Visual Basic pro aplikace

11.1 Makrorekordér

Visual Basic
for
Applications

Součástí MS Excelu je také programovací jazyk Visual Basic for Applications, který je zjednodušenou verzí jazyka Visual Basic. Pro tento programovací jazyk se používá zkratka VBA. Již v kancelářském balíku MS Office 97 došlo ke sjednocení jazyka Visual Basic for Applications mezi jednotlivými programy. Lze proto používat ve Wordu, Excelu, PowerPointu, Accessu i Outlooku stejné příkazy.

Jednoduchou aplikací Visual Basicu je záznam časté sekvence stisků kláves, např. často zadávané příkazy z nabídky Excelu. Při záznamu sekvence nemusíme ani znát příkazy Visual Basicu, Excel totiž nabízí pro tvorbu jednoduchých programů – maker makrorekordér.

Makrorekordér

Makrorekordér zaznamenává postup práce v Excelu, abychom jej mohli později opakovaně provádět. Řadu sekvencí obsahuje Excel přímo v menu, v panelech nástrojů nebo pod klávesovými zkratkami. Pomocí maker můžeme zaznamenat a později provádět své vlastní postupy.



DEM-11-1



Základy

Makra budeme demonstrovat na jednoduché výchozí tabulce Excelu na listu *Základy* (viz obr. 11-1).

OBR. 11-1: VÝCHOZÍ TABULKA

	A	B	C	D	E	F	G
1	11	1200	130		Praha	Brno	Praha
2	210	2200	23				
3	3100	320	3300				

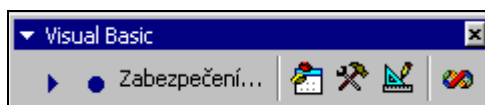
Naším prvním úkolem bude vytvořit makro, které zobrazí číslo s přesností na dvě desetinná místa³⁷. Před záznamem makra je vhodné vyzkoušet si postup, který budeme chtít zaznamenat. Kurzor přemístíme do buňky A1 a zadáme příkaz FORMÁT, BUŇKY. V kartě **Číslo** vybereme první číselný formát 0,00.

Kurzor přemístíme do buňky A2. Zobrazíme panel nástrojů *Visual Basic*:

- buď příkazem ZOBRAZIT, PANELE NÁSTROJŮ, VISUAL BASIC
- nebo klepnutím pravým tlačítkem myši do libovolného panelu nástrojů a výběrem volby VISUAL BASIC z místní nabídky.

Objeví se panel nástrojů *Visual Basic* (viz obr. 11-2).

OBR. 11-2: PANEL NÁSTROJŮ VISUAL BASIC



Záznam makra



- Spustíme makrorekordér:
- buď příkazem NÁSTROJE, MAKRO, ZÁZNAM NOVÉHO MAKRA
 - nebo klepnutím do tlačítka **Záznam makra** z panelu nástrojů *Visual Basic*.

Makra bude možné později spouštět několika způsoby:

- příkazem NÁSTROJE, MAKRO, MAKRA a výběrem makra,
- klávesovou zkratkou,
- z panelu nástrojů *Visual Basic* tlačítkem **Spustit makro**,
- z uživatelského panelu nástrojů.



V dialogovém okně **Záznam makra** se připravujeme na první tři způsoby spouštění makra, tzn. příkazem z menu, z panelu nástrojů *Visual Basic* nebo klávesovou zkratkou:

³⁷ Na rozdíl od tlačítka *Styl oddělovače* v panelu nástrojů *Formát* nechceme za číslem vynechat mezeru o šířce měnové jednotky.



– *Název makra* se bude objevovat v abecedním seznamu maker po příkazu NÁSTROJE, MAKRO, MAKRA. Tímto příkazem bude možné makro také upravit nebo odstranit. Název makra nesmí obsahovat mezery, proto místo mezer budeme psát podtržítka. Naše makra budeme předznačovat písmeny tak, aby se správně řadila v abecedním seznamu.

– Jestliže chceme spouštět makro kombinací kláves, klepneme do pole *Klávesová zkratka* a stiskneme písmeno nebo písmeno s klávesou **Shift**. Makro budeme moci později vyvolat kombinací **Ctrl** a zvoleného písmene, popř. kombinací kláves **Ctrl** **Shift** a zvoleného písmene. Abychom nepředefinovali užitečné kombinace Excelu (např. **Ctrl** **A**, **Ctrl** **C**), budeme využívat kombinace s klávesou **Shift**. Pokud bychom omylem využili dvakrát stejnou kombinaci, kombinace kláves bude spouštět dříve připravené makro.³⁸

– Makro uložíme do aktuálního sešitu (*tento sešit*), k němuž se vytvoří kód ve Visual Basicu. Prohlížení kódu si popíšeme v následujících odstavcích. Kód Visual Basicu můžeme také uložit do *nového sešitu*, spouštět bude možné makra ze všech otevřených sešitů. Často používaná makra můžeme uložit do *osobního sešitu maker*, který bude uložen do adresáře Program Files\Microsoft Office\Office\XLStart pod názvem *Personal.xls* a bude načten do Excelu vždy po spuštění Excelu.

– *Popis* pomůže při orientaci v seznamu maker.

Ukončením práce s dialogovým oknem **Záznam makra** tlačítkem **OK** zahájíme vlastní záznam makra. Ve stavovém řádku okna Excelu se vypisuje hlášení *Záznam*. Zadáme příkaz FORMÁT, BUŇKY a na kartě **Číslo** vybereme *číslo* na dvě desetinná místa. *Záznam makra* ukončíme:

- příkazem NÁSTROJE, MAKRO, ZASTAVIT ZÁZNAM
- nebo klepnutím do tlačítka **Zastavit záznam**.

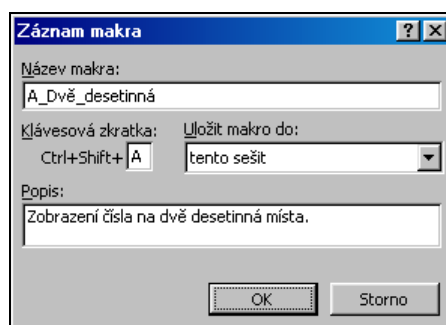
Makro se zaznamenalo ve formě kódu jazyka Visual Basic, který zobrazíme:

- příkazem NÁSTROJE, MAKRO, EDITOR JAZYKA VISUAL BASIC
- nebo kombinací kláves (levá klávesa) **Alt** **F11**
- nebo klepnutím do tlačítka **Editor jazyka Visual Basic**.

Editor Visual Basicu se skládá z několika částí zobrazených v samostatných oknech. (viz obr. 11-4, v němž jsou uvedeny všechny části editoru)³⁹:

- Prohlížeč projektu (*Project Explorer*): Ke každému sešitu se makra zaznamenávají do samostatného projektu. Samostatné projekty si vytváří některé doplňky Excelu (např. doplněk *Analytické nástroje* obsahuje projekt *funcres*).
- Okno vlastností (*Properties Window*): Jednotlivé objekty Visual Basicu mají své vlastnosti. Vlastnosti jsou významné zejména při tvorbě uživatelských formulářů, jejichž použití budeme demonstrovat později (viz kap. 11.6).
- Okno kódu (*Code*): K objektu může být připojen kód. Makra jsou zaznamenávána do tzv. modulů. Poklepáním na *Modules* v prohlížeči projektů a poklepáním na *Module1* zobrazíme námi vytvořené makro (viz obr. 11-4).
- Okamžité okno (*Immediate Window*): V okně můžeme zadat nebo vložit řádek kódu a pomocí klávesy **Enter** jej spustit.
- Okno místních položek (*Locals Window*): Automaticky zobrazuje všechny proměnné deklarované v aktuální proceduře a jejich hodnoty.⁴⁰
- Okno kukátek (*Watch Window*): Umožňuje průběžně zkoumat hodnoty vybraných proměnných.

OBR. 11-3: DIALOGOVÉ OKNO
ZÁZNAM MAKRA



Konec
záznamu



³⁸ Platnost klávesové zkratky skončí odstraněním makra či změnou jeho klávesové zkratky příkazem NÁSTROJE, MAKRO, MAKRA, MOŽNOSTI.

³⁹ Jednotlivé části editoru zobrazíme příkazem VIEW.

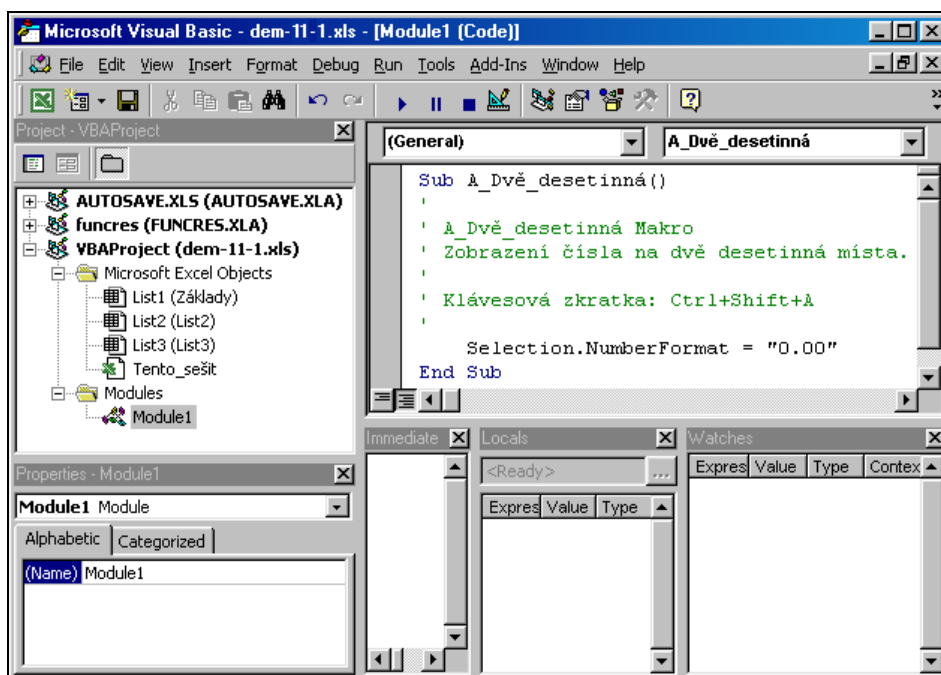
⁴⁰ Pojmy deklarace, proměnná, procedura budou vysvětleny v dalším výkladu.



OBR. 11-4: EDITOR VISUAL BASICU



DEM-11-1
Makro A
Dvě_Desetinná



Námi připravené makro je uloženo v *Module1*:

Sub

– Makro začíná slovem *Sub*, za nímž je uveden název makra. Závorky za názvem makra obecně umožňují doplnit parametry makra, které bychom využili při volání maker navzájem. (Parametry modifikují průběh makra, využijeme je později v uživatelských funkcích.)

Komentář

– Dalších šest řádků obsahuje komentáře. Řádek komentářů má dokumentační význam, neovlivňuje chod makra, pouze zpřehledňuje makro čtenáři. Obdobně lze komentáře dopsat i na konec příkazových řádků za apostrof. Excel do komentáře automaticky vložil název makra, jeho popis a klávesovou zkratku.

Selection

– Slovo *Selection* vypovídá, že operaci provádíme s aktuálním vybraným rozsahem. (Obecně můžeme makro aplikovat na změnu formátu zobrazení nejen jedné buňky, ale celého předem označeného rozsahu.) Slovo *NumberFormat* určuje měněnou vlastnost označeného rozsahu – formát zobrazení čísla. Za rovnítkem je uveden jeden z číselných formátů. Příkaz makra je nejlépe „čist“ z konce: „Nechť 0.00 je číselný formát vybraného rozsahu.“

End Sub

– Makro je ukončeno slovy *End Sub*.

Kombinací kláves (levý) **Alt** **F11** se vrátíme do Excelu a pokusíme se makro aplikovat. Makro můžeme spustit zatím následujícími způsoby:

– kombinací kláves **Ctrl** **Shift** **A**,

– příkazem **NÁSTROJE, MAKRO, MAKRA**, výběrem makra a klepnutím do tlačítka **Spustit**,

– klepnutím do tlačítka **Spustit makro** a stejným postupem jako v předchozím případě.

Vyzkoušíme nejprve kombinaci kláves **Ctrl** **Shift** **A** na buňce B3. Také buňka B3 je nyní zobrazena s přesností na dvě desetinná místa. Označíme nesouvislý rozsah A3, B1:B2 (v kombinaci s klávesou **Ctrl**) a zadáme příkaz **NÁSTROJE, MAKRO, MAKRA** a klepneme do tlačítka **Spustit**. Vidíme, že makro můžeme aplikovat i na předem označený rozsah buněk.

Analogicky připravíme makro, které naopak zobrazí číslo s přesností na jednotky:

– Kurzor přichystáme do buňky A2.

– Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.

– *Název makra*: B_Celé_číslo.

– *Klávesová zkratka*: **Ctrl** **Shift** **B**.

– *Uložit makro do*: tento sešit.

– *Popis*: Zobrazení na celé číslo.

– Klepneme do tlačítka **OK**.

– Zadáme příkaz **FORMÁT, BUŇKY**. V kartě **Číslo** vybereme první číselný formát 0. (*Desetinná místa*: 0.) Klepneme do tlačítka **OK**.



– Klepneme do tlačítka *Zastavit záznam*.

OBR. 11-5: MAKRO B_CELÉ_ČÍSLO



DEM-11-1
Makro B
Celé číslo

```
Sub B_Celé_číslo()  
'  
' B_Celé_číslo Makro  
' Zobrazení na celé číslo.  
'  
' Klávesová zkratka: Ctrl+Shift+B  
'  
    Selection.NumberFormat = "0"  
End Sub
```

Fungování makra ověříme např. na buňce A3.

Oběma makry jsme měnili *vlastnost* buňky. V jednom makru můžeme měnit i více vlastností najednou. Chceme např. připravit makro, které buňku či označený rozsah vycentruje vodorovně i svisle a změní orientaci psaní textu:

- Kurzor přichystáme do buňky E1, kde je text *Praha* (viz obr. 11-1).
- Klepneme do tlačítka *Záznam makra* v panelu nástrojů *Visual Basic*.
- *Název makra*: C_Centrování.
- *Klávesová zkratka*: .
- *Uložit makro do*: tento sešit.
- *Popis*: Vodorovně i svisle centrování.
- Klepneme do tlačítka *OK*.
- Zadáme příkaz FORMÁT, BUŇKY a v kartě *Zarovnání* zvolíme *Vodorovně: na střed*, *Svisle: Na střed*, *Orientace: svisle*. Klepneme do tlačítka *OK*.
- Klepneme do tlačítka *Zastavit záznam*.



OBR. 11-6: MAKRO C_CENTROVÁNÍ



DEM-11-1
Makro C
Centrování

```
Sub C_Centrování()  
'  
' C_Centrování Makro  
' Vodorovně i svisle centrování  
'  
' Klávesová zkratka: Ctrl+Shift+C  
'  
    With Selection  
        .HorizontalAlignment = xlCenter  
        .VerticalAlignment = xlCenter  
        .WrapText = False  
        .Orientation = xlVertical  
        .AddIndent = False  
        .ShrinkToFit = False  
        .MergeCells = False  
    End With  
End Sub
```

With
End With

Makrorekordér zahrnul do makra nastavení čtyř vlastností, které bylo možné v kartě *Zarovnání* dialogového okna **Formát buněk** změnit. Všechny vlastnosti se vztahují k aktuálnímu výběru. Tělo makra by proto mohlo obsahovat příkazy:

```
Selection.HorizontalAlignment = xlCenter  
Selection.VerticalAlignment = xlCenter  
Selection.WrapText = False  
Selection.Orientation = xlVertical  
Selection.AddIndent = False  
Selection.ShrinkToFit = False  
Selection.MergeCells = False
```

Dvojice příkazů *With* a *End With* umožňuje zpřehlednit kód makra zjednodušením zápisu změn vlastností, které se vztahují ke stejnému výběru. Navíc je makro s využití dvojice *With* a *End With* rychlejší, neboť makro pracuje s aktuálním výběrem pouze jednou.



11.2 Editace makra

Editace makra

Změnu vlastnosti *WrapText*, *AddIndent*, *ShrinkToFit* a *MergeCells* jsme do makra *C_Centrování* nechťeli zaznamenávat. Můžeme je dodatečně odstranit editací makra. V editoru Visual Basicu označíme řádky s těmito vlastnostmi a klávesou **Delete** jej smažeme. Makro můžeme tedy běžně editovat. Vrátime se zpět do Excelu na list *Základy*. Označíme rozsah F1:G1 a stisknutím kombinace kláves **Ctrl** **Shift** **C** otestujeme fungování makra. Na rozsahu E1:G1 zrušíme provedené formátování příkazem FORMÁT, BUŇKY v kartě **Zarovnání**.

Další makro nebude měnit vlastnost buňky či rozsahu, ale aktuálního okna – skryje mřížku sešitu (přesněji skryje či zobrazí mřížku podle aktuálního stavu):

– Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.

– *Název makra*: D_Mřížka.

– *Klávesová zkratka*: **Ctrl** **Shift** **D**.

– *Uložit makro do*: tento sešit.

– *Popis*: Skryje nebo zobrazí mřížku.

– Klepneme do tlačítka **OK**.

– Zadáme příkaz NÁSTROJE, MOŽNOSTI a v kartě **Zobrazení** zrušíme zaškrtnutí pole *Mřížky* a klepneme do tlačítka **OK**.

– Klepneme do tlačítka **Zastavit záznam**.



ActiveWindow

Tělo makra obsahuje jediný příkaz: `ActiveWindow.DisplayGridlines = False`. My však chceme makro zobecnit. Chceme, aby se stav pole *Mřížky* změnil na opačný. Tělo makra proto upravíme dle obr. 11-7.

OBR. 11-7: MAKRO D_MŘÍŽKA



DEM-11-1
Makro D
Mřížka

```
Sub D_Mřížka()  
'  
' D_Mřížka Makro  
' Skryje nebo zobrazí mřížku.  
'  
' Klávesová zkratka: Ctrl+Shift+D  
'  
  
With ActiveWindow  
    .DisplayGridlines = Not .DisplayGridlines  
End With  
End Sub
```

Dvojici *With* a *End With* jsme využili, abychom v řádku změny mřížky nemuseli opakovat, že se vlastnost *DisplayGridlines* vztahuje k aktuálnímu oknu. Pomocí slova *Not* můžeme změnit stav vlastnosti na opak. Dále jsme omezili počet mezer na začátku řádku. Každou úroveň makra oddělujeme jen jednou mezerou.

Umístění maker

v modulech

Pokud sešit s makry uložíme, opět otevřeme a doplňujeme další makra, zaznamenávají se makra do dalšího modulu (*Module2*). Odtud je můžeme přesunout (kombinacemi kláves **Ctrl** **X** a **Ctrl** **V**) do původního modulu. Při přesunu se nezachová klávesová zkratka. V případě potřeby ji musíme definovat příkazem NÁSTROJE, MAKRO, MAKRA a klepnutím do tlačítka **Možnosti**.

11.3 Databázová makra

Často při vypisování tabulky opakujeme pod sebou stejné hodnoty. Následující makro zajistí zkopírování hodnoty z buňky z předchozího řádku. Použijeme nepřímý způsob: do aktuální buňky napíšeme vzorec odkazující se na buňku nad aktuální buňkou. Vzorec potom zkopírujeme do stejné buňky se současnou záměnou vzorce za hodnotu:

– Přesuneme kurzor do buňky E2.

– Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.

– *Název makra*: E_Dtto.

– *Klávesová zkratka*: **Ctrl** **Shift** **E**.

– *Uložit makro do*: tento sešit.

– *Popis*: Kopírování buňky nad aktivní buňkou.





- Klepneme do tlačítka **OK**.
- Do buňky E2 zapíšeme rovnítko zahajující vzorec a klepneme do buňky E1.
- Vzorec $=E1$ ukončíme kombinací **Ctrl** **Enter** (nikoliv **Enter** přesunující kurzor do E3)⁴¹.
- Stiskneme kombinaci kláves **Ctrl** **C**, čímž označíme buňku E2 jako zdroj kopírování.
- Zadáme příkaz ÚPRAVY, VLOŽIT JINAK, v sekci Vložit zaškrtneme volbu *Hodnoty* a klepneme do tlačítka **OK**.
- Klávesou **Esc** zrušíme označení zdroje kopírování.
- Klepneme do tlačítka **Zastavit záznam**.



OBR. 11-8: MAKRO E_DTTO



DEM-11-1
Makro E
Dtto

```
Sub E_Dtto()  
'  
' E_Dtto Makro  
' Kopírování buňky nad aktivní buňkou.  
'  
' Klávesová zkratka: Ctrl+Shift+E  
'  
Selection.FormulaR1C1 = "=R[-1]C"  
Selection.Copy  
Selection.PasteSpecial Paste:=xlValues, Operation:=xlNone, _  
    SkipBlanks:=False, Transpose:=False  
Application.CutCopyMode = False  
End Sub
```

Metoda a její
argumenty

V úvodu makra (viz obr. 11-8) Excel vybranou buňku pracovně označí jako *R1C1* (row 1, column 1) a vloží do ní vzorec $=R[-1]C$ (odkaz na předchozí řádek, stejný sloupec).

Slovo *Selection* v druhém řádku těla makra *Selection.Copy* je stejné jako v předcházejících makrech, značí výběr. Výběru však není přisouzena vlastnost (jako *NumberFormat* a další), ale s výběrem je provedena akce, která se v terminologii Visual Basicu nazývá *metoda*. Metoda *Selection.Copy* zkopíruje buňku či označený rozsah do schránky. Další metoda *Selection.PasteSpecial* vloží do aktuální buňky obsah schránky speciálním způsobem, který se v Excelu upřesňuje v dialogovém okně, v makru je upřesněn tzv. *argumenty* oddělenými navzájem čárkami. Argumenty jsou podobné vlastnostem, jsou však umístěny až za názvem metody. Vlastnosti jsou atributy objektu, argumenty jsou upřesňující instrukce pro provedení metody.

Pokud je vytvořený řádek delší než 80 znaků, makrorekordér vloží mezeru a podtržítka za vhodné slovo a pokračuje na dalším řádku. Pomocí mezery a podtržítka můžeme řádky rozdělit za účelem větší přehlednosti sami při editaci makra.

Poslední příkaz v makru *Application.CutCopyMode = False* upřesňuje vlastnost označení zdroje kopírování. (Zastupuje funkci klávesy **Esc** při běžném kopírování.⁴²) Makro otestujeme na rozsahu F2:G2.

Před tvorbou dalšího makra připravíme do tabulky náznak jednoduchého seznamu, který chceme později databázově zpracovávat (viz obr. 11-9). V seznamu jsou uvedeni studenti, kteří již splnili podmínky pro udělení zápočtu z předmětu PM_351. Pro rychlost nejsou vyplňovány údaje shodné s předchozím řádkem. Jejich vyplnění je však pro další databázové zpracování (např. třídění) nutné. Vyplnění zajistíme makrem.

Seznam zkopírujeme z rozsahu A5:C9 do rozsahu E5:G9, abychom ve zkopírovaném seznamu mohli provádět testy, po nichž se budeme kopírováním vracet k výchozímu stavu.

⁴¹ Při ukončení klávesou **Enter** vrátíme kurzor o řádek výše. Zápis makra v editoru Visual Basicu potom musíme upravit: odstraníme nadbytečný řádek *Range("E2").Select*. Dále místo příkazu *Selection.FormulaR1C1* figuruje v zápisu příkaz *ActiveCell.FormulaR1C1*. Tato záměna by znemožnila aplikovat makro na blok buněk.

⁴² Kopírování jsme nemohli ukončit klávesou **Enter**, aby se nepřesunul kurzor do dalšího řádku. Při ukončení **Ctrl** **Enter** zůstane zdroj kopírování označen.



DEM-11-1



Základy

OBR. 11-9: SEZNAM STUDENTŮ, KTERÍ SPLNILI PODMÍNKY PRO ZÁPOČET

	A	B	C	D	E	F	G
5	PM_351	00001	Novák		PM_351	00001	Novák
6			Dvořáková				Dvořáková
7			Fiala				Fiala
8		00002	Nosek			00002	Nosek
9			Novotná				Novotná

Připravíme makro:

- Přesuneme kurzor do buňky E5.
- Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.
- *Název makra*: F_Dtto_souhrnné.
- *Klávesová zkratka*: **Ctrl** **Shift** **F**.
- *Uložit makro do*: tento sešit.
- *Popis*: Příprava seznamu. Poté klepneme do tlačítka **OK**.
- Příkazem ÚPRAVY, PŘEJÍT NA, klepnutím do tlačítka **Jinak** a volbou *Aktuální oblast* nebo kombinací kláves **Ctrl** ***** označíme obdélníkový rozsah buněk, který obsahuje aktivní buňku obklopenou na sebe navazujícími vyplněnými buňkami (včetně prázdných buněk v obdélníku).
- Zadáme příkaz ÚPRAVY, PŘEJÍT NA, klepneme do tlačítka **Jinak** a zvolíme možnost *Prázdné buňky*. Poté klepneme do tlačítka **OK**, čímž označíme pouze prázdné buňky v dříve označeném rozsahu.
- Zapišeme =, čímž začneme psát vzorec do jedné z buněk označeného rozsahu (do buňky F6), vytyčíme zde odkaz na buňku ve stejném sloupci v předchozím řádku.
- Stisknutím kombinace **Ctrl** **Enter** připravený vzorec vložíme do všech označených buněk.
- Nechceme, aby v buňkách byly vzorce, ale hodnoty (stejně jako v předchozím makru).
 - Klepnutím do klávesy **End** a šipky nahoru zrušíme označení rozsahu a přesuneme kurzor na první řádek seznamu, klepnutím do klávesy **End** a šipky vlevo přesuneme kurzor do prvního sloupce prvního řádku seznamu.
 - Kombinací kláves **Ctrl** ***** označíme celý seznam.
 - Kombinací kláves **Ctrl** **C** celý seznam vybereme jako zdroj kopírování.
 - Zadáme příkaz ÚPRAVY, VLOŽIT JINAK, *Vložit hodnoty* a klepneme do tlačítka **OK**.
 - Klávesou **Esc** zrušíme označení zdroje kopírování.
- Klepneme do tlačítka **Zastavit záznam**.

OBR. 11-10: MAKRO F_DTTTO_SOUHRNNÉ



DEM-11-1

Makro F

Dtto_souhrnné

```
Sub F_Dtto_souhrnné()  
'  
' F_Dtto_souhrnné Makro  
' Příprava seznamu.  
'  
' Klávesová zkratka: Ctrl+Shift+F  
'  
Selection.CurrentRegion.Select  
Selection.SpecialCells(xlCellTypeBlanks).Select  
Selection.FormulaR1C1 = "=R[-1]C"  
Selection.End(xlUp).Select  
Selection.End(xlToLeft).Select  
Selection.CurrentRegion.Select  
Selection.Copy  
Selection.PasteSpecial Paste:=xlValues, Operation:=xlNone, _  
    SkipBlanks:=False, Transpose:=False  
Application.CutCopyMode = False  
End Sub
```

Z rozsahu A5:C9 zkopírujeme výchozí stav seznamu do rozsahu E5:G9, kurzor umístíme do buňky E5 a spustíme makro, abychom otestovali jeho funkčnost.



11.4 Základní příkazy Visual Basicu

Vytvoříme makro, které změní barvu textu označeného rozsahu na červenou:

- Přesuneme kurzor do buňky A1.
- Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.
- *Název makra*: G1_Barvy.
- *Klávesová zkratka*: nevyplníme.
- *Uložit makro do*: tento sešit.
- *Popis*: Barvy.
- V panelu nástrojů *Formát* zvolíme červenou barvu písma tlačítkem **Barva písma (Červená)**.
- Klepneme do tlačítka **Zastavit záznam**.

OBR. 11-11: VÝCHOZÍ KÓD MAKRA G1_BARVY (ČERVENÁ BARVA)

```
Sub G1_Barvy ()  
'  
' G1_Barvy Makro  
' Barvy  
'  
Selection.Font.ColorIndex = 3  
End Sub
```

V makru se označenému rozsahu přisoudí vlastnost barvy 3 z palety barev. (Červená barva je uvedena jako třetí v paletě.)

My však chceme, aby barva byla závislá na hodnotě čísla. Čísla menší než 100 budou červená, čísla větší nebo rovna 100 budou zelená. V okně makra zkopírujeme kód makra jako běžný text a vložíme jej za výchozí kód. Druhé makro přejmenujeme na *G2_Barvy* (v příkazu *Sub* i v komentáři) a upravíme jej dle obr. 11-12. (Příkaz napsaný kurzívou jsme již použili).

OBR. 11-12: MAKRO BARVY S ČERVENOU A ZELENOU BARVOU

```
Sub G2_Barvy ()  
'  
' G2_Barvy Makro  
' Barvy  
'  
If Selection < 100 Then  
    Selection.Font.ColorIndex = 3  
Else  
    Selection.Font.ColorIndex = 4  
End If  
End Sub
```

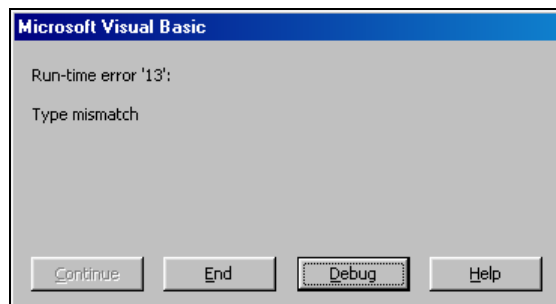
DEM-11-1
Makro G2
Barvy

If...Then...Else V makru jsme využili podmíněný příkaz *If...Then...Else...End If*, který umožňuje větvit chod makra. (Část *Else* není povinná). Nové makro můžeme aplikovat pouze na jedinou buňku. Označíme-li před spuštěním makra více buněk, nebude možné provést příkaz *If Selection...* a makro ohlásí chybu (viz obr. 11-13).

Dialogové okno oznamuje pouze číslo chyby. Klepnutím do tlačítka **Help** se zobrazí popis chyby. Tlačítkem **Debug** můžeme přejít do okna s kódem makra, kde bude zvýrazněn žlutým podkladem chybný příkaz. Tlačítkem **End** předčasně ukončíme makro.

Visual Basic umožňuje další členění v makru podle hodnoty. Příkazem *Select Case* budeme demonstrovat rozvětvení barev do tří pásem. Nejdříve zkopírujeme makro G2 pod jeho původní umístění, novou verzi přejmenujeme na G3.

OBR. 11-13: DIALOGOVÉ OKNO CHYBY MAKRA





OBR. 11-14: MAKRO BARVY S ČERVENOU, ZELENOU A MODROU BARVOU



DEM-11-1
Makro G3
Barvy

```
Sub G3_Barvy ()  
'  
' G3_Barvy Makro  
' Barvy  
'  
  
Select Case Selection  
Case Is < 100  
    Selection.Font.ColorIndex = 3  
Case 100 To 999  
    Selection.Font.ColorIndex = 4  
Case Is > 999  
    Selection.Font.ColorIndex = 5  
End Select  
End Sub
```

Select Case

Úvodní příkaz *Select Case* definuje výraz, podle kterého je makro rozvětveno. Jednotlivé větve *Case* obsahují podmínky pro hodnotu vybraného výrazu. Poslední větev může mít tvar *Case Else* (ostatní případy). Větvení musí být ukončeno příkazem *End Select*.

Makro zatím mění barvu pouze jedné buňky. Chceme, aby se kurzor po změně barvy jedné buňky přemístil do následující buňky, v níž budeme moci znovu makro spustit. Připravíme si pomocné makro:

- Kurzor umístíme do buňky A1.
- Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.
- *Název makra*: GX_Pomoc.
- *Klávesová zkratka*: nevyplníme.
- *Uložit makro do*: tento sešit.
- *Popis*: nezměníme.
- Stiskneme kurzorovou šipku dolů.
- Klepneme do tlačítka **Zastavit záznam**.

Nové makro obsahuje jediný příkaz:

```
Range("A2").Select
```

Příkaz umístí kurzor vždy do buňky A2. My však chceme provést přemístění kurzoru o pozici dolů relativně vzhledem k poloze kurzoru.

Relativní odkazy

Pokusíme se pomocné makro připravit ještě jednou:

- Kurzor umístíme do buňky A1.
- Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.
- *Název makra*: GX_Pomoc.
- *Klávesová zkratka*: nevyplníme.
- *Uložit makro do*: tento sešit.
- *Popis*: nezměníme.
- V dialogovém okně odpovíme na dotaz *Makro s názvem GX_Pomoc již existuje. Chcete existující makro nahradit?* klepnutím do tlačítka **Ano**.
- V panelu nástrojů **Zastavit záznam** klepneme do tlačítka **Relativní odkaz**.
- Stiskneme kurzorovou šipku dolů.
- Klepneme do tlačítka **Zastavit záznam**.

Upravené makro obsahuje jediný příkaz:

```
ActiveCell.Offset(1, 0).Range("A1").Select
```

Selection. Offset

ActiveCell.Offset(1,0) nastaví levý horní roh budoucího aktivního rozsahu o jeden řádek níže. *Range* definuje rozsah vzhledem k nastavenému levému rohu. Slovo *ActiveCell* můžeme nahradit slovem *Selection*. Pro úplnost dodejme, že kdybychom na buňku B2 aplikovali příkaz:

- *Selection.Offset(1, 0).Range("A1").Select*
aktivní by se stala buňka B3
- *Selection.Offset(0, 1).Range("A1").Select*
aktivní by se stala buňka C2
- *Selection.Offset(-1, 0).Range("A1").Select*
aktivní by se stala buňka B1
- *Selection.Offset(1, 0).Range("A1.B2").Select*



aktivním by se stal rozsah B3:C4
– Selection.Offset(5, 0).Range("A1").Select
aktivní by se stala buňka B7
– Selection.Offset(1, 0).Range("A5").Select
aktivní by se stala také buňka B7
– Selection.Offset(1, 1).Range("A1").Select
aktivní by se stala buňka C3
– Selection.Offset(1, 0).Range("B1").Select
aktivní by se stala také buňka C3

Zkopírujeme makro G3 jako makro G4 a příkaz Selection.Offset(1, 0).Range("A1").Select přesuneme za příkaz End Select. Makro GX_Pomoc smažeme.

Do...Loop

Makro G4 dále upravíme tak, aby se změna barev provedla od aktivní buňky v celém sloupci až do první prázdné buňky. Využijeme k tomu příkaz Do...Loop. Příkazy uvedené mezi slovy Do a Loop se provádějí cyklicky. Opakování lze omezit uvedením podmínky za slovem Do. Podmínka začíná slovem Until. V našem případě jako podmínku zadáme, že aktivní buňkou se po skončení kola opakování stala prázdná buňka (dvojitě uvozovky bez mezery).

Until

OBR. 11-15: MAKRO ODLIŠUJÍCÍ BARVY V JEDNOM SLOUPCI



DEM-11-1
Makro G4
Barvy

```
Sub G4_Barvy()  
'  
' G4_Barvy Makro  
' Barvy  
'  
Do Until Selection = ""  
    Select Case Selection  
        Case Is < 100  
            Selection.Font.ColorIndex = 3  
        Case 100 To 999  
            Selection.Font.ColorIndex = 4  
        Case Is > 999  
            Selection.Font.ColorIndex = 5  
    End Select  
    Selection.Offset(1, 0).Range("A1").Select  
Loop  
End Sub
```

Po skončení makra musíme ručně přesunout kurzor na začátek dalšího sloupce a makro znovu spustit. V další verzi makra po skončení cyklu barvení sloupce přesuneme kurzor doprava a zpět o buňku nahoru, stiskneme klávesu **End** a šipku nahoru (tj. skok na horní buňku seznamu v dalším sloupci). Cyklus barvení sloupce budeme opakovat, dokud nenarazíme na sloupec začínající prázdnou buňkou. Dopíšeme makro podle obr. 11-16.

OBR. 11-16: MAKRO ODLIŠUJÍCÍ BARVY V CELÉ SOUVISLÉ TABULCE



DEM-11-1
Makro G5
Barvy

```
Sub G5_Barvy()  
'  
' G5_Barvy Makro  
' Barvy  
'  
Do Until Selection = ""  
    Do Until Selection = ""  
        Select Case Selection  
            Case Is < 100  
                Selection.Font.ColorIndex = 3  
            Case 100 To 999  
                Selection.Font.ColorIndex = 4  
            Case Is > 999  
                Selection.Font.ColorIndex = 5  
        End Select  
        Selection.Offset(1, 0).Range("A1").Select  
    Loop  
End Sub
```



```
Selection.Offset(-1, 1).Range("A1").Select  
Selection.End(xlUp).Select  
Loop  
End Sub
```

V poslední verzi makra:

- Zařadíme na úvod barvení dotaz, jakou barvu použijeme. Uživatel zadá kód pro nejmenší čísla, další kódy barev budou vždy o jedničku vyšší. Základní barvy uživateli nabídneme do dialogového okna.
- Pro snadné0 testování barev se bude makro opakovat, dokud uživatel nezadá barvu 0.
- Po zadání barvy 0 ukončí cyklus zadávání barev, vrátí se na první buňku seznamu a vypíše se zpráva *Hotovo!*

OBR. 11-17: MAKRO S DIALOGOVÝMI OKNY



DEM-11-1
Makro G6
Barvy

```
Sub G6_Barvy()  
'  
' G6_Barvy Makro  
' Barvy  
'  
Do  
B = InputBox("Zadejte kód barvy (3-červená, 4-zelená, 5-modrá, _  
6-žlutá, 7-fialová, 0-konec):", "Kód výchozí barvy")  
If B = 0 Then Exit Do  
Do Until Selection = ""  
Do Until Selection = ""  
Select Case Selection  
Case Is < 100  
Selection.Font.ColorIndex = B  
Case 100 To 999  
Selection.Font.ColorIndex = B + 1  
Case Is > 999  
Selection.Font.ColorIndex = B + 2  
End Select  
Selection.Offset(1, 0).Range("A1").Select  
Loop  
Selection.Offset(-1, 1).Range("A1").Select  
Selection.End(xlUp).Select  
Loop  
Selection.Offset(0, -1).Range("A1").Select  
Selection.End(xlToLeft).Select  
Loop  
MsgBox " Hotovo!"  
End Sub
```

Proměnná

InputBox

Exit Do

MsgBox

V makrech můžeme používat *proměnné*. Např. v našem makru (viz obr. 11-17) zavedeme proměnnou *B* (barva). Do proměnné můžeme přiřadit hodnotu, např. *B = 3*. Místo přiřazení hodnoty můžeme příkazem *InputBox* zobrazit dialogové okno s výzvou pro uživatele k zadání hodnoty proměnné. Druhý argument je titulek okna. Při zadání nulové hodnoty proměnné *B* je ukončen cyklus *Do...Loop* příkazem *Exit Do*.

Na konec makra zobrazíme příkazem *MsgBox* zprávu pro uživatele. Mezery jsou doplněny do zprávy tak, aby zpráva byla v dialogovém okně vycentrována.

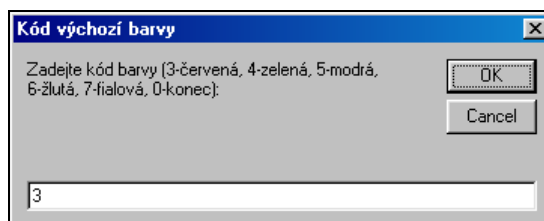
11.5 Ovládací prvky



DEM-11-1
Průsečíky



Další soustava maker nebude s předchozími makry souviset. Budeme graficky zkoumat průběh dvou polynomických funkcí znázorňujících např. výkony a náklady, abychom zjistili bod či body zvratu.⁴³ Další list sešitu nazveme *Průsečíky*. Úvodem vyplníme list až od 15. řádku dle obr. 11-19⁴⁴. Písmo v řádcích 15 a 16 zadáme modře, v řádcích 17 a 18 fialově.

OBR. 11-18: UŽIVATELSKÉ DIALOGOVÉ OKNO
KÓD VÝCHOZÍ BARVY

OBR. 11-19: DEFINICE POLYNOMICKÝCH FUNKCÍ

	A	B	C	D	E	F	G	H
15	1. křivka	x^3		x^2		x^1		x^0
16		0		1		110		1500
17	2. křivka	x^3		x^2		x^1		x^0
18		0		-3		350		0

Od 20. řádku přichystáme podklady pro grafické znázornění dvou polynomických funkcí (na rozdíl od obr. 11-20 neměníme šířky sloupců):

- Vyplníme řádek 20 a přichystáme sloupec A.
- Koeficient z buňky B21 bude později sloužit k volbě vhodného měřítka grafu.
- Do buňky C21 nachystáme vzorec pro upravenou hodnotu x:
C21: =A21*\$B\$21
a zkopírujeme jej do celého sloupce C.
- Do buněk D21 a E21 chceme přichystat hodnoty polynomických funkcí. Využijeme k tomu uživatelsky definované funkce (viz dále).
- Do modulu maker Visual Basicu doplníme na konec funkci dle obr. 11-21:

OBR. 11-20: PODKLADY PRO GRAF

	A	B	C	D	E
20	x	k	x^1	y_1	y_2
21	-10	10	-100	500	-65000
22	-9		-90	-300	-55800
23	-8		-80	-900	-47200
	...				
40	9		90	19500	7200
41	10		100	22500	5000

OBR. 11-21: FUNKCE POLYNOM



DEM-11-1
Funkce
Polynom

```
Function Polynom(A3, A2, A1, A0, X)
'
' Polynom Funkce
' Výpočet hodnoty polynomické funkce.
'
Polynom = A3 * X ^ 3 + A2 * X ^ 2 + A1 * X + A0
End Function
```

- Na rozdíl od maker, která jsou zaznamenávána ve formě procedur, nezačíná kód funkce slovem *Sub*, ale slovem *Function* a končí slovy *End Function*.
- Funkce má pět argumentů. Je vhodné nazvat je výstižně, neboť názvy se uplatní v dialogovém okně pro vkládání funkce při aplikaci funkce. (Uživatelská funkce bude zařazena do skupiny *vlastní*.)
- V kódu funkce musí být jejímu názvu (*Polynom*) přiřazena hodnota.
- V buňkách D21 a E21 aplikujeme uživatelskou funkci (viz obr. 11-22):
D21: =Polynom(\$B\$16;\$D\$16;\$F\$16;\$H\$16;C21)
E21: =Polynom(\$B\$18;\$D\$18;\$F\$18;\$H\$18;C21)
- Buňky D21 a E21 zkopírujeme do řádků 22 – 41.
Z připravených podkladů nakreslíme graf.

⁴³ Zobecníme tak řešení úlohy *Grafická analýza bodu zvratu* ilustrující příkaz HLEDAT ŘEŠENÍ z kapitoly 7.6.

⁴⁴ Excel umožňuje formátovat část buňky. Lze proto např. do buňky B15 zapsat x^3 , označit v řádku vzorců exponent a příkazem FORMÁT, BUŇKY, na kartě **Písmo** volbou **Horní index** zapsat exponent v horní polovině řádku. Pro vypsání 2. znaku obsahu buňky horním indexem lze nadefinovat jednoduché makro, např. *X_Mocnina* obsahující příkaz `Selection.Characters(Start:=2, Length:=1).Font.Superscript = True`.



- Nejdříve změníme šířku sloupců pro lepší názornost. Kurzor přesuneme do sloupce A, příkazem FORMÁT, SLOUPEC, ŠÍŘKA zadáme šířku sloupce 6,86. Označíme sloupec B – J, příkazem FORMÁT, SLOUPEC, ŠÍŘKA změníme jejich šířku na 5,86.

OBR. 11-22: VKLÁDÁNÍ FUNKCE POLYNOM

Polynom

A3	\$B\$16	= 0
A2	\$D\$16	= 1
A1	\$F\$16	= 110
A0	\$H\$16	= 1500
X	C21	= -100

Klepnutím na tlačítko Nápověda získáte popis funkce a jejích argumentů.

X

Výsledek = 500

OK Storno

- Označíme rozsah C20:E41.
- Zadáme příkaz VLOŽIT, GRAF.
- V prvním okně průvodce grafem vybereme *XY bodový graf* a zvolíme poslední podtyp.
- Ve druhém okně průvodce grafem potvrdíme oblast dat C20:E41 použitou pro graf. (Řady tvoří sloupce.)
- Ve třetím okně průvodce grafem ponecháme nevyplněné názvy grafu, osy x a osy y.
- Ve čtvrtém okně průvodce grafem zadáme umístění grafu jako objekt do listu *Průsečíky*.
- Objekt grafu přesuneme do rozsahu A1:J14 a upravíme velikost.
- V grafu poklepáním postupně na obě osy zmenšíme velikost písma jejich popisu na 8. Obdobně zmenšíme velikost písma legendy.
- Potlačíme ohraničení zobrazované oblasti. Plochu zobrazované oblasti změníme ze šedé na žádnou.
- Odstraníme zobrazování mřížky.
- Křivka y_1 je v grafu zobrazena modrou barvou, křivka y_2 je zobrazena fialovou barvou, což odpovídá barvám písma v tabulce. Křivku y_2 navíc zobrazíme čárkovaně (viz obr. 11-24).

Polynomické funkce zobrazené v grafu můžeme měnit změnou koeficientů. Pro funkce druhého stupně mohou existovat dva průsečíky (pro výkony a náklady se jedná o body zvratu). Předpokládejme, že funkce tržeb má průběh jako křivka y_2 (je rostoucí) a funkce nákladů je klesající jako křivka y_1 . Pokusme se vypočítat hodnoty průsečíků těchto funkcí. Požadujeme, aby platilo $y_1 = y_2$, tj. $y_2 - y_1 = 0$. Hodnoty průsečíků (x) budeme počítat do buněk L15 a L17. Příslušné rozdíly $y_2 - y_1$ budeme počítat do buněk L16 a L18. Budeme tedy počítat, jaké bude L15 (popř. L17), aby L16 (L18) bylo nulové:

- L16: =Polynom(\$B\$16;\$D\$16;\$F\$16;\$H\$16;L15)-Polynom(\$B\$18;\$D\$18;\$F\$18;\$H\$18;L15)
- L18: =Polynom(\$B\$16;\$D\$16;\$F\$16;\$H\$16;L17)-Polynom(\$B\$18;\$D\$18;\$F\$18;\$H\$18;L17)

Dále již připravujeme makro, které bude hledat obě řešení:

- Kurzor umístíme do buňky K1 (abychom do makra zaznamenali přechod na buňku L15).
- Klepneme do tlačítka **Záznam makra** v panelu nástrojů *Visual Basic*.
- *Název makra*: H_Průsečíky.
- *Klávesová zkratka*: **Ctrl** **Shift** **H**.
- *Uložit makro do*: tento sešit.
- *Popis*: Průsečíky křivek.
- Klepneme do tlačítka **OK**.
- Zkontrolujeme, zda není zatlačeno tlačítko **Relativní odkaz** v panelu nástrojů *Zastavit záznam*.
- Klepneme do buňky L15 a zapíšeme sem hodnotu -1 000 000 (výchozí velmi nízká hodnota) a odešleme.



- Z buňky L16 zadáme příkaz NÁSTROJE, HLEDÁNÍ ŘEŠENÍ. V poli *Nastavená buňka* ponecháme L16, do pole *Cílová hodnota* zapíšeme 0, do pole *Měněná buňka* zapíšeme L15 a klepneme do tlačítka **OK**. Tlačítkem **OK** potvrdíme nalezené řešení.
- Klepneme do buňky L17 a zapíšeme sem hodnotu 1 000 000 (výchozí velmi vysoká hodnota je nutná pro nalezení extrému) a odešleme.
- Z buňky L18 zadáme příkaz NÁSTROJE, HLEDÁNÍ ŘEŠENÍ. V poli *Nastavená buňka* ponecháme L18, do pole *Cílová hodnota* zapíšeme 0 (výchozí nízká hodnota je nutná pro nalezení druhého extrému), do pole *Měněná buňka* zapíšeme L17 a klepneme do tlačítka **OK**. Tlačítkem **OK** potvrdíme řešení.
- Klepneme do tlačítka **Zastavit záznam**. (Upravený kód makra je uveden v obr. 11-28).

Makra jsme dosud vyvolávali kombinacemi kláves či z menu. Excel umožňuje také spouštění makra pomocí tlačítka umístěného v listu. Do listu můžeme umisťovat i další ovládací prvky. Nejprve zobrazíme panel nástrojů *Formuláře* (viz obr. 11-23). Do našeho listu budeme postupně doplňovat ovládací prvky dle obr. 11-24.

OBR. 11-23: PANEL NÁSTROJŮ FORMULÁŘE

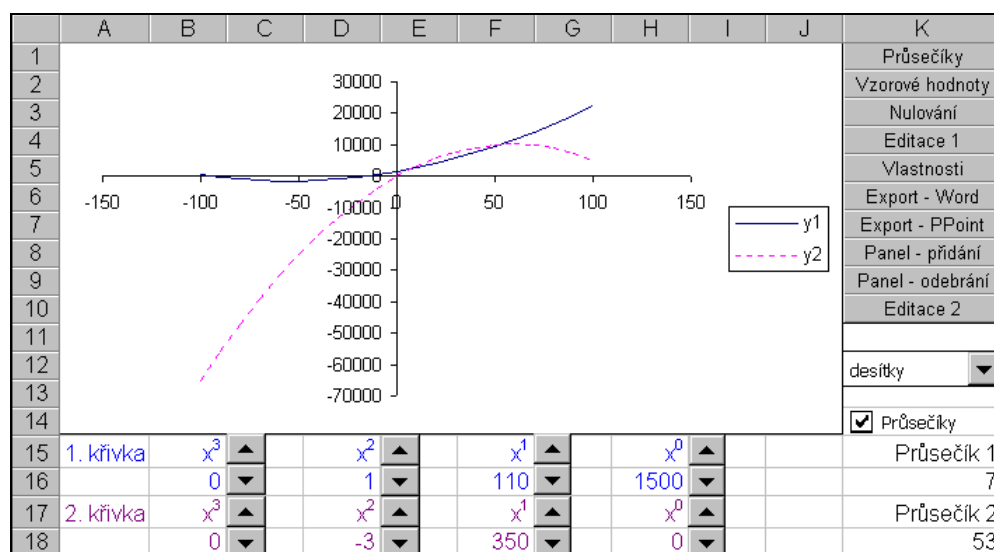


Vytvoření
tlačítka v listu



Rozšíříme sloupec *K* na šířku 12,86. Klepneme do tlačítka **Tlačítko**. V buňce K1 vytyčíme tlačítko o výšce řádku a šířce sloupce *K*. Ze seznamu maker vybereme makro *H_Průsečíky*. Klepnutím do názvu tlačítka opravíme název tlačítka na *Průsečíky*. Klepneme stranou mimo tlačítko. Dalším klepnutím do tlačítka již vyvoláme funkci tlačítka. Případné úpravy či odstranění tlačítka můžeme provést z místní nabídky po klepnutí pravým tlačítkem myši do tlačítka. Takto upravíme velikost písma v tlačítku na 8.

OBR. 11-24: LIST PRŮSEČÍKY S OVLÁDACÍMI PRVKY



Číselník

Úpravou koeficientů polynomů můžeme snadno sledovat vývoj funkcí a jejich průsečíky, které můžeme makrem *Průsečíky* vypočítat. (V případě jediného průsečíku budou oba průsečíky po výpočtu shodné. Více průsečíků pro jednoduchost nepředpokládáme, abychom nemuseli stanovovat další vhodný výchozí odhad *x*.) Abychom nemuseli koeficienty přepisovat z klávesnice, doplníme vedle nich číselníky. Klepneme do tlačítka **Číselník** z panelu nástrojů *Formuláře* a vytyčíme jej v levé polovině buněk C15:C16.



Vlastnosti



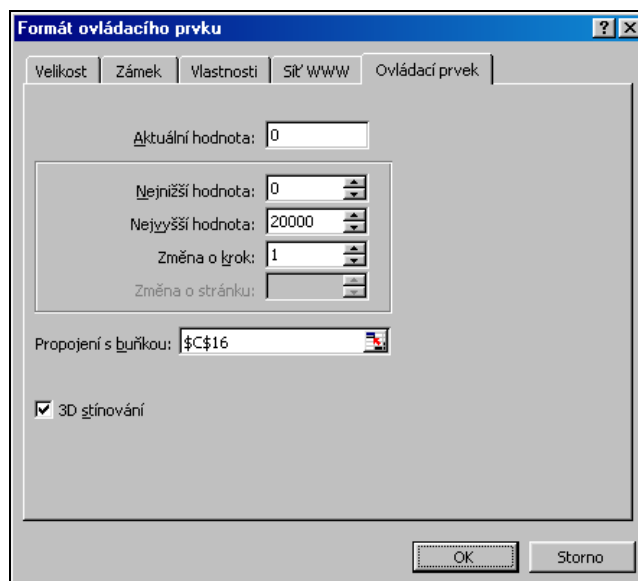
Klepnutím do tlačítka **Vlastnosti** vyvoláme dialogové okno **Formát ovládacího prvku**, v němž klepneme do karty **Ovládací prvek** (viz obr. 11-25). *Změna o krok* definuje, o kolik se zvětší či zmenší hodnota po klepnutí do číselníku. Propojení určuje buňku, kterou takto měníme. Minimální hodnota bohužel nemůže být záporná. Proto se odkážeme na hodnotu buňky \$C\$16. Do buňky B16 zapíšeme vzorec = C16-10000. Tím nepřímo ovlivníme buňku



B16, kde budeme moci (dle hodnot v polích *Minimální hodnota* a *Maximální hodnota*) nastavovat hodnoty v intervalu $<-10000;10000>$.

Do buňky C16 zapíšeme výchozí hodnotu 10000. Buňku C16 skryjeme zadáním barvy písma bílá. Pozor! Nebudeme již moci přímo vkládat hodnoty do buňky B16, abychom nepřepsali vzorec. Po skončení všech úprav by proto bylo vhodné list ochránit, ochranu uvolnit jen pro buňky, které povolíme měnit (např. C16). Analogicky připravíme číselníky, vzorce a výchozí transformované (o 10000 zvýšené) hodnoty pro ostatní koeficienty obou křivek. Koeficienty polynomů můžeme nyní měnit klepáním do číselníků.

OBR. 11-25: DIALOGOVÉ OKNO FORMÁT OVLÁDACÍHO PRVKU PRO OBJEKT ČÍSELNÍK



Za účelem snadného předvádění makra doplníme další dvě jednoduchá makra s vlastními tlačítky. V makru *I_Vzor* vložíme do skrytých buněk hodnoty, pro něž jsme makro připravovali. Tvorba makra je jednoduchá. Klepneme na příslušnou buňku a zapíšeme do ní příslušnou hodnotu (i kdyby zde již byla). Vkládané hodnoty jsou patrné z obr. 11-26. Obdobně doplníme makro *J_Nulování* pro nulování koeficientů polynomů.

OBR. 11-26: MAKRA DEFINUJÍCÍ VZOROVÉ A NULOVÉ HODNOTY KOEFICIENTŮ



DEM-11-1
Makro I
Vzor
Makro J
Nulování

```
Sub I_Vzor ()  
'  
' I_Vzor Makro  
' Vzorové hodnoty.  
'  
' Klávesová zkratka: Ctrl+Shift+I  
'  
  
Range("C16").Select  
Selection.FormulaR1C1 = "10000"  
Range("E16").Select  
Selection.FormulaR1C1 = "10001"  
Range("G16").Select  
Selection.FormulaR1C1 = "10110"  
Range("I16").Select  
Selection.FormulaR1C1 = "11500"  
Range("C18").Select  
Selection.FormulaR1C1 = "10000"  
Range("E18").Select  
Selection.FormulaR1C1 = "9997"  
Range("G18").Select  
Selection.FormulaR1C1 = "10350"  
Range("I18").Select  
Selection.FormulaR1C1 = "10000"  
End Sub
```

```
Sub J_Nulování ()  
'  
' J_Nulování Makro  
' Nulování hodnot.  
'  
' Klávesová zkratka: Ctrl+Shift+J  
'  
  
Range("C16").Select  
Selection.FormulaR1C1 = "10000"  
Range("E16").Select  
Selection.FormulaR1C1 = "10000"  
Range("G16").Select  
Selection.FormulaR1C1 = "10000"  
Range("I16").Select  
Selection.FormulaR1C1 = "10000"  
Range("C18").Select  
Selection.FormulaR1C1 = "10000"  
Range("E18").Select  
Selection.FormulaR1C1 = "10000"  
Range("G18").Select  
Selection.FormulaR1C1 = "10000"  
Range("I18").Select  
Selection.FormulaR1C1 = "10000"  
End Sub
```



Pole se
seznamem



Před umístěním dalšího ovládacího prvku připravíme do listu jednoduchou tabulku desítkových řádů (viz obr. 11-27). Do buňky L12 zapíšeme aktuální výběr pro měřítka grafu: 2 (desítky), které jsme uvedli v buňce B21. Buňku B21 provážíme s buňkou L12 vzorcem v buňce B21: =INDEX(K21:K24;L12).

Provázání buňky B21 na buňku L12, v níž je pořadí desítkového řádku, jsme přichystali proto, abychom mohli řád měřítka vybírat ze seznamu. Klepneme do tlačítka **Pole se seznamem** z panelu nástrojů *Formuláře* a vytyčíme umístění seznamu do buňky K12.

Klepneme do tlačítka **Vlastnosti**. V kartě **Ovládací prvek** vyplníme:

- *Vstupní oblast*: \$J\$21:\$J\$24
- *Propojení s buňkou*: \$L\$12
- *Počet řádků*: Ponecháme 8 (maximální počet řádků, které se nabídnou po vyklopení seznamu).

Klepneme mimo seznam, potom do šipky seznamu a vybereme desítky.

Výsledek hledání průsečíků zobrazíme přehledněji v zaokrouhlené podobě. Na uživateli ponecháme, zda výsledky hledání zobrazí či ne. Přepínač rozhodnutí umístíme do buňky L14, kam pro začátek zapíšeme hodnotu *Pravda*. Dále zapíšeme do tabulky vzorce:

- K15: =KDYŽ(\$L\$14;"Průsečík 1";"")
- K16: =KDYŽ(\$L\$14;L15;"")
- K17: =KDYŽ(\$L\$14;"Průsečík 2";"")
- K18: =KDYŽ(\$L\$14;L17;"")

Dříve připraveným makrem *B_Celé_číslo* upravíme zobrazování buněk K16 a K18.

Pro přepínání buňky L14 připravíme ovládací prvek *Zaškrtnuté políčko*:

Zaškrtnuté
políčko



- Klepneme do tlačítka **Zaškrtnuté políčko** z panelu nástrojů *Formuláře* a vytyčíme umístění políčka do buňky K14.
- Do políčka zapíšeme text *Průsečíky*.
- Klepneme do tlačítka **Vlastnosti**. V kartě **Ovládací prvek** vyplníme:
 - *Hodnota*: Zaškrtnuto
 - *Propojení s buňkou*: \$L\$14.

Klepneme mimo políčko, potom do políčka a můžeme testovat jeho funkčnost.

Do makra *H_Průsečíky* doplníme na začátek potlačení zobrazení průsečíků při výpočtu. Po dokončení makra budou hodnoty průsečíků vždy zobrazeny. Ukončení makra signalizujeme také akusticky (příkaz *Beep*).

Beep

OBR. 11-28: MAKRO H_PRŮSEČÍKY BEZ ZOBRAZOVÁNÍ PRŮSEČÍKŮ BĚHEM VÝPOČTŮ



DEM-11-1
Makro H
Průsečíky

```
Sub H_Průsečíky ()
'
' H_Průsečíky Makro
' Průsečíky křivek
'
' Klávesová zkratka: Ctrl+Shift+H
'
Range("L14").Select
Selection.FormulaR1C1 = "FALSE"
Range("L15").Select
Selection.FormulaR1C1 = "-1000000"
Range("L16").Select
Range("L16").GoalSeek Goal:=0, ChangingCell:=Range("L15")
Range("L17").Select
Selection.FormulaR1C1 = "1000000"
Range("L18").Select
Range("L18").GoalSeek Goal:=0, ChangingCell:=Range("L17")
Range("L14").Select
Selection.FormulaR1C1 = "TRUE"
Range("K14").Select
Beep
End Sub
```



Dopíšeme ještě jednoduché makro pro skrytí či zobrazení pomocného sloupce *L*. Makro budeme spouštět kombinací kláves **Ctrl** **Shift** **K**, nepřiradíme jej žádnému tlačítku.

OBR. 11-29: MAKRO SKRYTÍ ČI ZOBRAZENÍ SLOUPCE *L*

DEM-11-1
Makro K
Sloupec

```
Sub K_Sloupec ()  
'  
' K_Sloupec Makro  
' Skrytí či zobrazení pomocného sloupce  
'  
' Klávesová zkratka: Ctrl+Shift+K  
'  
Columns("L").Select  
With Selection.EntireColumn  
    .Hidden = Not .Hidden  
End With  
End Sub
```

Pomocí makra nejprve vybereme sloupec *L*. V případě, že je zobrazen, skryjeme jej, jinak jej naopak zobrazíme.

11.6 Dialogové okno a uživatelský formulář

Podívejme se podrobněji na možnosti komunikace zprostředkované Visual Basicem. Připravíme makro, které zobrazí uživateli vlastnosti sešitu, s nímž pracuje.

OBR. 11-30: MAKRO *L1_VLASTNOSTI*

DEM-11-1
Makro L1
Vlastnosti

```
Sub L1_Vlastnosti ()  
T1 = ActiveWorkbook.BuiltinDocumentProperties.Item("Title").Value  
T = "Název: " & T1  
T2 = ActiveWorkbook.BuiltinDocumentProperties.Item("Subject").Value  
T = T & Chr(13) & "Předmět: " & T2  
T3 = ActiveWorkbook.BuiltinDocumentProperties.Item("Author").Value  
T = T & Chr(13) & "Autor: " & T3  
T4 = ActiveWorkbook.BuiltinDocumentProperties.Item("Comments").Value  
T = T & Chr(13) & "Komentář: " & T4  
X = MsgBox(T, 4, "Rekapitulace vlastností")  
If X = 7 Then MsgBox ("Zadejte příkaz Soubor, Vlastnosti.")  
End Sub
```

Poznámky k makru:

- V procedurách Visual Basicu pracujeme s objekty, které mají vlastnosti. Např. objekt *ActiveWorkbook* (aktivní sešit) je upřesňován vlastností *BuiltinDocumentProperties* (vlastnosti sešitu), která obsahuje kolekci dílčích vlastností.⁴⁵ Jednou z dílčích vlastností je její hodnota. Při zápisu kódu pomáhá Visual Basic nabídkou seznamu vlastností k objektu, např. když zapíšeme objekt *ActiveWindow* a tečku, objeví se seznam vlastností zapsaného objektu. Po výběru vlastnosti zapíšeme vlastnost do kódu stisknutím klávesy **Tab**.
- Do proměnných *T1*, *T2*, *T3* a *T4* zjišťujeme hodnoty (*Value*)⁴⁶ jednotlivých položek (*Item*) vlastností (*BuiltinDocumentProperties*) aktivního sešitu (*ActiveWorkbook*).
- V proměnné *T* textově slučujeme načtené hodnoty a přidáváme k nim komentář. Přičítáním znaku s ASCII kódem 13 - *Chr(13)* zajišťujeme přechod na nový řádek mezi vlastnostmi.
- Funkce *MsgBox* vrací hodnotu zvolenou v dialogovém okně. Má několik parametrů:
 - první (je povinný): text uváděný v dialogovém okně,
 - druhý: typ okna (např. 4 značí, že okno bude obsahovat tlačítka *Ano* a *Ne*),
 - třetí: nadpis titulku okna.
- Pokud za slovem *MsgBox* použijeme jediný parametr zobrazovaného textu, nemusíme přiřazovat výslednou hodnotu žádné proměnné.

⁴⁵ Dílčí vlastnosti můžeme pojmenovat Excelem připravenými názvy (*Title*, *Subject* a další) nebo očíslovat (1 – 9).

⁴⁶ Zápis vlastnosti *Value* je nepovinný. Místo *ActiveWorkbook.BuiltinDocumentProperties.Item("Subject").Value* můžeme zapsat *ActiveWorkbook.BuiltinDocumentProperties.Item("Subject")*.



MsgBox

- Výsledkem funkce *MsgBox* je kód uživatelem stisknutého tlačítka (např. kód 7 zastupuje klepnutí do tlačítka *Ne*). V našem případě, pokud klepneme do tlačítka *Ne*, objeví se dialogové okno s návodem, jak změnit vlastnosti sešitu.
- Makro přiřadíme k tlačítku v listu *Průsečíky*.

Makro *L2_Vlastnosti* vychází z předchozího makra. Obecně zobrazuje všechny vlastnosti sešitu.

OBR. 11-31: MAKRO L2_VLASTNOSTI



DEM-11-1
Makro L2
Vlastnosti

```
Sub L2_Vlastnosti ()
    T = ""
    For I = 1 To 9
        With ActiveWorkbook.BuiltinDocumentProperties.Item(I)
            T = T & Chr(13) & .Name & ": " & .Value
        End With
    Next I
    X = MsgBox(T, 4, "Rekapitulace vlastností")
    If X = 7 Then MsgBox ("Zadejte příkaz Soubor, Vlastnosti.")
End Sub
```

Poznámky k makru:

- Proměnné *T* je zpočátku přiřazena prázdná hodnota.
- Excel sleduje devět vlastností sešitu. Kromě názvů (*Title*, *Subject* atd.) jsou identifikovány jednoznačně indexy. V cyklu, který se prochází devětkrát, je k proměnné *T* vždy přičten znak odřádkování – *Chr(13)*, název vlastnosti a po dvojtečce a mezeře hodnota vlastnosti aktuálního sešitu.

Další makro povede uživatele při editaci koeficientů křivek. Postupně budeme vyplňovat šest buněk. Z předchozího výkladu víme, že vyplňujeme nepřímo sousední buňky hodnotami o 10000 zvětšenými.

OBR. 11-32: MAKRO M_EDITACE1



DEM-11-1
Makro M
Editace1

```
Sub M_Editace1 ()
    A13 = InputBox("Zadejte A3", "1. křivka", "0")
    ActiveSheet.Cells(16, 3).Value = A13 + 10000
    A12 = InputBox("Zadejte A2", "1. křivka", "1")
    ActiveSheet.Cells(16, 5).Value = A12 + 10000
    A11 = InputBox("Zadejte A1", "1. křivka", "110")
    ActiveSheet.Cells(16, 7).Value = A11 + 10000
    A10 = InputBox("Zadejte A0", "1. křivka", "1500")
    ActiveSheet.Cells(16, 9).Value = A10 + 10000
    A23 = InputBox("Zadejte A3", "2. křivka", "0")
    ActiveSheet.Cells(18, 3).Value = A23 + 10000
    A22 = InputBox("Zadejte A2", "2. křivka", "-3")
    ActiveSheet.Cells(18, 5).Value = A22 + 10000
    A21 = InputBox("Zadejte A1", "2. křivka", "350")
    ActiveSheet.Cells(18, 7).Value = A21 + 10000
    A20 = InputBox("Zadejte A0", "2. křivka", "0")
    ActiveSheet.Cells(18, 9).Value = A20 + 10000
End Sub
```

Poznámky k makru:

InputBox

- Výsledkem funkce *InputBox* je zobrazení dialogového okna, v němž uživatel musí zadat jednu hodnotu. Funkce má tři parametry:
 - první: doprovodný text v dialogovém okně,
 - druhý: nadpis titulků okna,
 - třetí: uživateli nabídnutá hodnota, kterou může změnit (výchozí hodnota).
- Po zadání hodnoty uživatelem se hodnota uloží do proměnné např. *A13*, *A22* apod., které je přiřazena funkce *InputBox*.



Uživatelský
formulář

– Hodnota proměnné je vložena na aktivním listu do stanovené buňky. (Např. buňka C16 je vyjádřena indexy 16, 3. Jednou z vlastností buňky je její hodnota.)⁴⁷

Vyplňování hodnot po jedné v několika dialogových oknech je nepřehledné. Připravíme proto pro editaci uživatelský formulář. V editoru Visual Basicu zadáme příkaz VIEW, USERFORM.

V levém dolním rohu editoru jsou zobrazeny vlastnosti aktuálního objektu.⁴⁸ Formulář jako celek má také své vlastnosti, z nichž upravíme:

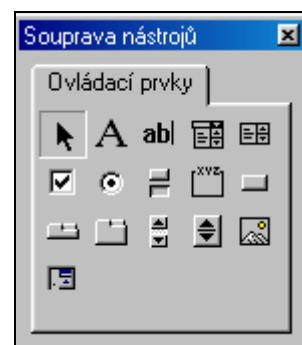
- (*Name*): Zadání_hodnot (Vlastnost *Name* nesmí obsahovat mezery. Název se projeví v okně projektu – viz obr. 11-34.)
- *Caption*: Zadání hodnot (Titulek uživatelského formuláře může obsahovat mezery.)

Souprava
nástrojů
Popis



Pro tvorbu uživatelského formuláře se zobrazil speciální panel nástrojů *Souprava nástrojů* (viz obr. 11-33). Klepneme do tlačítka **Popis**. V uživatelském formuláři vytyčíme umístění prvního textového popisu *1. křivka* (dle obr. 11-34). Klepnutím do rámečku textového popisu můžeme upravit text (neboli vlastnost⁴⁹ *Caption*). Formát textu lze upravit prostřednictvím vlastnosti *Font*. (Klepneme do tlačítka se třemi tečkami na konci řádku vlastnosti *Font*. V dialogovém okně **Písmo** vybereme písmo *Tahoma* o velikosti 10.) Levé horní rohy objektů formuláře se automaticky přichycují do mřížky⁵⁰.

OBR. 11-33:
SOUPRAVA NÁSTROJŮ



Textové pole

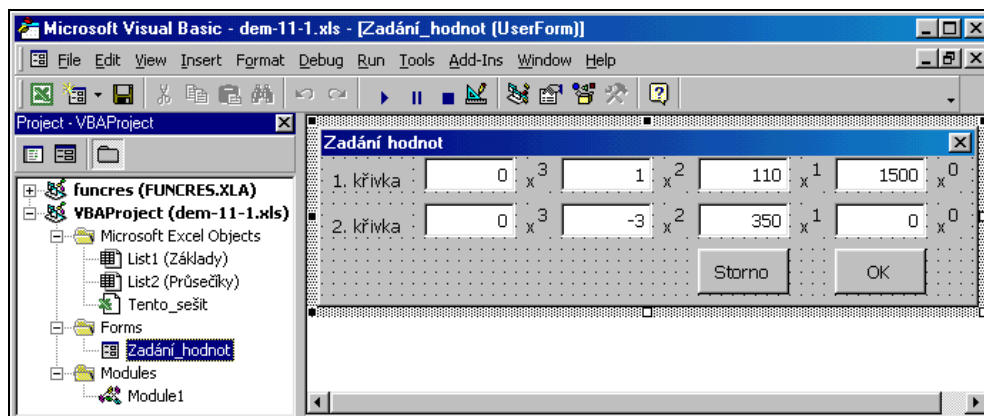


Dále klepneme v *Soupravě nástrojů* do tlačítka **Textové pole**. Do políčka vyplníme výchozí nabízenou hodnotu 0. Ve vlastnostech upřesníme název pole a jeho zarovnání doprava:

- (*Name*): A13
- *TextAlign*: 3 – fmTextAlignRight.

Popis x^3 připravíme jako dva samostatné popisy. Obdobně doplníme další ovládací prvky dle obr. 11-34.⁵¹

OBR. 11-34: UŽIVATELSKÝ FORMULÁŘ



Příkazové
tlačítko



Objekty **Storno** a **OK** vytyčíme po klepnutí do tlačítka **Příkazové tlačítko** ze *Soupravy nástrojů*. Poklepáním na tlačítko v režimu návrhu formuláře můžeme doplnit kód ve Visual Basicu, který se provede po klepnutí do příslušného tlačítka. Takto doplníme kód:

– k tlačítku **Storno**:

```
Private Sub Storno_Click() 52
    Unload Me
End Sub
```

⁴⁷ Mohli bychom také použít dříve uvedenou dvojici příkazů *Range.Select* a *Selection.FormulaRIC1*.

⁴⁸ Pokud okno vlastností chybí, zadáme příkaz VIEW, PROPERTIES WINDOW nebo stiskneme klávesu **F4**.

⁴⁹ Pokud nemáme zobrazeno okno vlastností, tak je vyvoláme z místní nabídky příkazem PROPERTIES.

⁵⁰ Přichycování do mřížky lze vypnout příkazem TOOLS, OPTIONS v kartě **General** volbou *Align Controls to Grid*.

⁵¹ Pořadí vyplňování ovládacích prvků lze ovlivnit vlastností *TabIndex*, kde jsou ovládací prvky číslovány od 0.

⁵² Tlačítko jsme změnou vlastnosti *Name* nazvali *Storno*. Obdobně druhé tlačítko *OK*.



– k tlačítku **OK**:

```
Private Sub OK_Click()  
    ActiveSheet.Cells(16, 3).Value = A13 + 10000  
    ActiveSheet.Cells(16, 5).Value = A12 + 10000  
    ActiveSheet.Cells(16, 7).Value = A11 + 10000  
    ActiveSheet.Cells(16, 9).Value = A10 + 10000  
    ActiveSheet.Cells(18, 3).Value = A23 + 10000  
    ActiveSheet.Cells(18, 5).Value = A22 + 10000  
    ActiveSheet.Cells(18, 7).Value = A21 + 10000  
    ActiveSheet.Cells(18, 9).Value = A20 + 10000  
    Unload Me  
End Sub
```

Unload

Příkaz *Unload Me* zruší zobrazení formuláře, ať už uživatel klepne do tlačítka **Storno** či **OK**. Klepne-li do tlačítka **OK**, umístí se navíc hodnoty z proměnných A13, A12,... A20 do příslušných buněk tabulky.

Na závěr tažením za rohové úchyty nastavíme vhodný tvar formuláře. Formulář můžeme otestovat již z prostředí Visual Basicu příkazem RUN, RUN SUB/USERFORM nebo stisknutím klávesy **F5**. (Obdobně jsme mohli testovat i předchozí makra či přesněji procedury.)

Připravíme nyní ještě tlačítko **Editace 2** do listu *Průsečíky* a k němu připojíme jednoduché makro, které zobrazí formulář.

OBR. 11-35: MAKRO N_EDITACE2



DEM-11-1
Makro N
Editace2

Load

```
Sub N_Editace2()  
    Load Zadání_hodnot  
    Zadání_hodnot.Show  
End Sub
```

Příkaz *Load* načte formulář. Metoda *Název formuláře.Show* zobrazí formulář. Zavření objektu formuláře a ukončení procedury zajistí klepnutí do tlačítka **Storno** ve formuláři, kterému je přiřazena výše uvedená procedura s příkazem *Unload Me*.

11.7 Export do Wordu a PowerPointu. Panel nástrojů

*Export
do Wordu*

Dosud kód Visual Basicu prováděl operace pouze v rámci Excelu. Kód Visual Basicu může zprostředkovat komunikaci mezi různými programy. Předvedme si komunikaci na příkladu vytvoření dokumentu Wordu s výsledky řešení úlohy o průsečících.

*Využití dalších
knihoven*

Protože budeme využívat i metod Wordu musíme nejprve v editoru Visual Basicu zadat příkaz **TOOLS, REFERENCES** a přidat k dostupným knihovnám i *Microsoft Word 9.0 Object Library* jejím zaškrtnutím v seznamu.

Dim

Dosud jsme proměnné definovali jejich prvním použitím (např. proměnnou *T* v makru *L2_Vlastnosti*). V některých případech je však nutné proměnnou deklarovat. V našem případě budeme deklarovat příkazem *Dim* objektovou proměnnou *ObjektWordu*, zastupující aplikaci Wordu (viz obr. 11-36). Díky slovu *New* spuštěním procedury otevřeme Word. (Pokud již byl spuštěn, spustí se znovu v dalším okně.)⁵³

Set

Příkazem *Set DokumentWordu = ObjektWordu.Documents.Add* přidáme nový dokument v právě otevřeném Wordu. Na nový dokument se budeme moci odkazovat proměnnou *DokumentWordu*.

Activate

V další části procedury uložíme do proměnných *T1* a *T2* řešení úlohy s průsečíky a do proměnné *T3* načteme autora z vlastností sešitu.

Dalšími příkazy zobrazíme a zaktivníme Word (část *With ObjektWordu – End With*).

Příkazy v části vymezené příkazy *With DokumentWordu – End With* pracují s prázdným dokumentem Wordu. Zpočátku je v něm přichystán jediný odstavec (paragraph) k němuž jsou přidávány další odstavce. V rámci Wordu můžeme upřesňovat vlastnosti dílčích objektů Wordu: odstavců (paragraphs) a slov (words).

⁵³ Použije-li se klíčové slovo *New* při deklarování objektové proměnné, vytvoří se nový exemplář objektu při prvním odkazu na tento objekt, takže není nutno použít příkaz *Set* pro přiřazení odkazu na objekt.



DEM-11-1
Makro O
Export do
Wordu

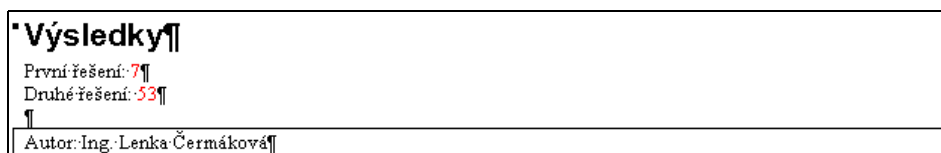
OBR. 11-36: MAKRO O_EXPORT_DO_WORDU

```
Sub O_Export_do_Wordu()  
    Dim ObjektWordu As New Word.Application  
    Set DokumentWordu = ObjektWordu.Documents.Add  
    T1 = ActiveSheet.Cells(16, 11).Text  
    T2 = ActiveSheet.Cells(18, 11).Text  
    T3 = ActiveWorkbook.BuiltinDocumentProperties.Item("Author")  
    With ObjektWordu  
        .Visible = True  
        .Activate  
    End With  
    With DokumentWordu  
        With .Paragraphs(1).Range  
            .Text = "Výsledky" & Chr(13)  
            .Style = "Nadpis 1"  
        End With  
        With .Paragraphs.Add.Range  
            .Text = "První řešení: " & T1 & Chr(13)  
            .Style = "Normální"  
        End With  
        With .Paragraphs.Add.Range  
            .Text = "Druhé řešení: " & T2 & Chr(13) & Chr(13)  
            .Style = "Normální"  
        End With  
        With .Paragraphs.Add.Range  
            .Text = "Autor: " & T3  
            .Style = "Normální"  
        End With  
        .Words(6).Font.ColorIndex = wdRed  
        .Words(11).Font.ColorIndex = wdRed  
        .Paragraphs(5).Borders.OutsideLineStyle = wdLineStyleSingle  
    End With  
End Sub
```

Poznámky k tvorbě dokumentu Wordu Visual Basicem:

- Odstavce a slova jsou číslována pořadovými čísly. Odkazovat se lze jen na již vytvořené odstavce či slova. Hned zpočátku můžeme upřesnit vlastnosti prvního odstavce, zadáme tak jeho text a styl odstavce. Text ukončíme odesláním zastoupeným funkcí *Chr(13)* generující znak odeslání.
- Další odstavce musíme již přidávat. Využíváme k tomu metodu *Paragraphs.Add*. Současně s přidáním odstavce definujeme text a styl přidaného odstavce.
- Na závěr celé komunikace změním barvu 6. a 11. slova dokumentu, tj. hodnot výsledků. Do počtu slov se počítají i znaky ukončení odstavce a dvojtečky.
- Pátý odstavec orámujeme čarou. Jako odstavec je počítáno každé ukončení odstavce, tj. i prázdný odstavec za druhým řešením.

OBR. 11-37: DOKUMENT WORDU GENEROVANÝ EXCELEM



Export do
PowerPointu

Obdobně v další proceduře připravíme export výsledků do prezentace PowerPointu (viz obr. 11-38).

Protože budeme využívat i metod PowerPointu musíme nejprve v editoru Visual Basicu zadat příkaz **TOOLS, REFERENCES** a přidat k dostupným knihovnám i *Microsoft PowerPoint 9.0 Object Library* jejím zaškrtnutím v seznamu.



DEM-11-1
Makro P
Export do
PowerPointu

OBR. 11-38: MAKRO P_EXPORT_DO_POWERPOINTU

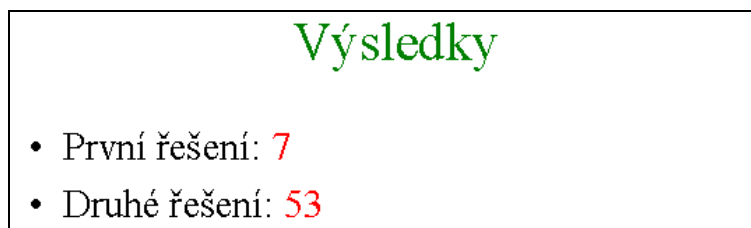
```
Sub P_Export_do_PowerPointu()  
    Dim ObjektPowerPointu As New PowerPoint.Application  
    Set Presentace = ObjektPowerPointu.Presentations.Add  
    T1 = "První řešení: " & ActiveSheet.Cells(16, 11).Text & Chr(13)  
    T2 = "Druhé řešení: " & ActiveSheet.Cells(18, 11).Text & Chr(13)  
    With ObjektPowerPointu  
        .Visible = True  
        .Activate  
    End With  
    Set NovySnimek = Presentace.Slides.Add(1, ppLayoutText)  
    With NovySnimek.Shapes.Placeholders(1).TextFrame.TextRange  
        .Text = "Výsledky"  
        .Words(1).Font.Color.RGB = RGB(0, 128, 0)  
    End With  
    With NovySnimek.Shapes.Placeholders(2).TextFrame.TextRange  
        .Text = T1  
        .InsertAfter(T2).IndentLevel = 1  
        .Words(4).Font.Color.RGB = RGB(255, 0, 0)  
        .Words(8).Font.Color.RGB = RGB(255, 0, 0)  
    End With  
End Sub
```

Poznámky k makru:

- Deklarací připravíme objektovou proměnnou zastupující novou aplikaci PowerPointu.
- V rámci aplikace PowerPointu otevřeme příkazem *Set* novou prezentaci.
- Do proměnných *T1* a *T2* přichystáme dílčí texty obsahující důležité buňky z otevřeného sešitu Excelu. Do proměnných zapíšeme také znaky ukončující odstavec.
- Dalšími příkazy procedury zobrazíme a zaktivizujeme PowerPoint.
- Do prezentace přidáme nový snímek (číslo 1) typu seznamu (*ppLayoutText*). Tato operace neměla obdobu ve Wordu, neboť dokument Wordu se nečlení na snímky.
- Do prvního objektu v rámci nového snímku zapíšeme text *Výsledky*, kterému přiřadíme tmavě zelenou barvu. (Funkcí *RGB* mícháme červenou, zelenou a modrou barvu. Podíl každé barvy je vyjádřen číslem z intervalu <0;255>.)
- Do druhého objektu nového snímku zapíšeme text z proměnné *T1*. Tento text je vzhledem k typu objektu vypisován s odrazkou v první (nejvyšší) úrovni. Do objektu doplníme text z proměnné *T2* také v úrovni 1. (*IndentLevel* může měnit úroveň odražení.)
- V textu v druhém objektu změníme barvu 4. a 8. slova (číselných výsledků) na červenou.

RGB

OBR. 11-39: SNÍMEK POWERPOINTU GENEROVANÝ EXCELEM



Poslední dvě makra budou sloužit k přidání a odebrání vlastního panelu nástrojů. Do panelu nástrojů umístíme dvě tlačítka na spouštění předchozích exportních maker.

OBR. 11-40: MAKRO R_PŘIDÁNÍ PANELU NÁSTROJŮ



DEM-11-1
Makro R
Přidání panelu
nástrojů

```
Sub R_Přidání_panelu_nástrojů()  
    Dim Panel As CommandBar  
    Dim Tlacitko As CommandBarButton  
    Set Panel = CommandBars.Add  
    Panel.Name = "Exporty"  
    With Panel.Controls
```



```
Set Tlacitko = .Add(msoControlButton)
With Tlacitko
    .Style = msoButtonIconAndCaption
    .Caption = "do Wordu"
    .FaceId = 42
    .TooltipText = "Export do Wordu"
    .Visible = True
    .OnAction = "O_Export_do_Wordu"
End With
Set Tlacitko = .Add(msoControlButton)
With Tlacitko
    .Style = msoButtonIconAndCaption
    .Caption = "do PowerPointu"
    .FaceId = 267
    .TooltipText = "Export do PowerPointu"
    .Visible = True
    .OnAction = "P_Export_do_PowerPointu"
End With
End With
Panel.Visible = True
End Sub
```

Poznámky k makru:

- V úvodu deklarujeme objektové proměnné zastupující panel nástrojů (*CommandBar*) a jednotlivé tlačítka (*CommandBarButton*).
- Přidáme nový panel nástrojů, který nazveme *Exporty*. (Panel nástrojů *Exporty* nesmí před přidáním existovat ani v nezobrazené podobě, jinak dojde k chybě makra.)
- Přidáme postupně dvě tlačítka, pro něž upřesňujeme styl (*msoButtonIconAndCaption* zastupuje tlačítko a textový nadpis), textový nadpis (*Caption*), číslo ikony (*FaceId*), text do nápovědné žluté bubliny (*TooltipText*), viditelnost a makro, které se spustí po klepnutí do tlačítka.

Přidání panelu
nástrojů a
tlačítek

OBR. 11-41: NĚKTERÉ Z DOSTUPNÝCH IKON TLAČÍTEK





Odebrání
panelu
nástrojů

- Dalším makrem odstraníme panel nástrojů *Exporty*. Poznámky k makru:
- Deklarujeme dvě objektové proměnné (*Panel1* a *Panel2*) zastupující panely nástrojů.
 - Postupně prohledáváme všechny panely nástrojů. Řídící proměnnou je *Panel1*.
 - Jestliže jsme našli panel nástrojů nazvaný *Exporty*, předčasně ukončíme prohledávací cyklus. Vyhledaný objekt našeho panelu je zastoupen proměnnou *Panel2*.
 - Pokud existuje náš panel nástrojů *Exporty*, odstraníme jej.

OBR. 11-42: MAKRO S_ODSTRANĚNÍ PANELU NÁSTROJŮ



DEM-11-1
Makro S
Odstranění
panelu nástrojů

```
Sub S_Odstranění_panelu_nástrojů()
    Dim Panel1 As CommandBar
    Dim Panel2 As CommandBar
    For Each Panel1 In Application.CommandBars
        If Panel1.Name = "Exporty" Then
            Set Panel2 = Panel1
            Exit For
        End If
    Next Panel1
    If Not Panel2 Is Nothing Then Panel2.Delete
End Sub
```

Ladění

- Závěrem se stručně zmíníme o ladění kódu. Tvorbu kódu doprovázejí většinou chyby:
- syntaktické (např. neuzavřené závorky nebo uvozovky): Chybu odhalí již editor Visual Basicu a upozorní nás na ni při editaci červeným zvýrazněním chybné části kódu.
 - chyby kompilace (např. neukončení příkazu *Select Case* slovy *End Select*): Na chybu jsme upozorněni až při spuštění makra (např. varovnou zprávou *Compile Error: Select Case Without End Select*). Excel přejde do editace kódu a označí pravděpodobný řádek chyby.
 - chyby běhu (např. se odkazujeme na název rozsahu, který neexistuje): Chyby se projeví až při běhu makra.
 - logické chyby (např. chybný výpočet): Excel samozřejmě neodhalí věcné chyby.

Krokování

Složitější chyby odhalíme až tzv. krokováním. Zkusme krokovat makro *G6_Barvy*. Kurzor umístíme do listu *Základy* do buňky *A1*. Zadáme příkaz NÁSTROJE, MAKRO, MAKRA. Vybereme makro *G6_Barvy* a klepneme do tlačítka **Krokovat s vnořením**. Z kódu se provede jen jeden řádek a makro se zastaví. Zobrazení přejde do editoru Visual Basicu, kde provedený řádek je žlutě podbarven. V okně **Locals** můžeme sledovat hodnoty proměnných (v našem případě *B*). Další řádek kódu provedeme klávesou **F8**. V průběhu krokování můžeme nahlížet do Excelu na změny v listu *Základy*.

11.8 Microsoft Script Editor

Microsoft
Script Editor

Příkazem NÁSTROJE, MAKRO, MICROSOFT SCRIPT EDITOR, kombinací kláves **[Alt] [Shift] [F11]**, nebo klepnutím do tlačítka **Microsoft Script Editor** v panelu nástrojů *Visual Basic* se spustí aplikace sloužící k ladění www stránek.

Shrnutí

1. Součástí Excelu je programovací jazyk *Visual Basic for Applications*, který je zjednodušenou verzí jazyka Visual Basic. *Makrorekordér* zaznamenává postup práce Excelu ve formě maker, což jsou procedury Visual Basicu.
2. Makra můžeme *spouštět* příkazem z menu NÁSTROJE, MAKRO, MAKRA, zadanou kombinací kláves, tlačítkem z panelu nástrojů či tlačítkem z listu.
3. *Procedura* jazyka Visual Basic začíná slovem *Sub* a končí slovy *End Sub*.
4. *Komentáře* začínají ve Visual Basicu apostrofem. Nemají vliv na průběh kódu.
5. Dvojice příkazů *With* a *End With* umožňuje zpřehlednit kód makra zjednodušením zápisu vlastností. Běh procedur a funkcí Visual Basicu je možné ovlivnit příkazy *If...Then...Else*, *Select Case ... End Select*, *Do...Loop*.
6. *Komunikaci s uživatelem* zajišťují příkazy *MsgBox* (zobrazení zprávy), *InputBox* (zadání hodnoty uživatelem) a uživatelské formuláře. Přímou do listu Excelu můžeme doplnit *ovládací prvky*: tlačítko, číselník, pole se seznamem, zaškrtnávací políčko a další.
7. Visual Basic umožňuje realizovat spolupráci Excelu s různými programy. Pomocí Visual Basicu můžeme připravit *vlastní panel nástrojů*.