ONE USE OF MS EXCEL IN TEACHING MATHEMATICS
JEDNO VYUŽITÍ MS EXCEL VE VÝUCE MATEMATIKY

Lukáš Honzík, Jan Frank

Abstract

The spreadsheet MS Excel is a well-known part of MS Office. Pupils are taught to work with it in computer science lessons, and so it is not a completely strange environment for them. Besides working with cells and patterns, charting and other issues that are generally manageable at the secondary school, it also offers the possibility of programming so-called macros in the VBA programming language. Macros are primarily, of course, used to make it easier to work with the spreadsheet, but with a little effort, it can also be used to demonstrate the run of selected mathematical algorithms. This may contribute to a better pupils' understanding of the taught topics.

Key words: *MS Excel, macros, Visual Basic for Applications, algorithmization*

Abstrakt

Tabulkový procesor MS Excel je dobře známou součástí programového balíku MS Office. Práci s ním se žáci učí v hodinách výpočetní techniky, a tak ani pro ně není úplně cizím prostředím. Kromě práce s buňkami a vzorci, tvorby grafů a dalších záležitostí, které jsou vesměs zvládnutelné již na 2. stupni základní školy, ale nabízí i možnost programování tzv. maker v programovacím jazyku VBA. Primární využití maker samozřejmě spočívá v ulehčení práce při využívání zmíněného tabulkového procesoru, ale při troše snahy je lze vhodně použít i pro demonstraci běhu vybraných matematických algoritmů, což může přispět k lepšímu pochopení dané látky ze strany žáků.

Klíčová slova: *MS Excel, makra, Visual Basic for Applications, algoritmizace*

ÚVOD

Používání programů počítačové algebry nebo šířeji matematického softwaru je v současné době vcelku rozšířené, a to nejen ve sféře vědecké, ale též v oblasti vzdělávání, kde se takový software dá jako dobrý pomocník využít téměř na všech stupních škol. V této souvislosti dnes hovoříme o nasazení tzv. kognitivních technologií do výuky matematiky. Z typických představitelů tohoto softwaru lze jmenovat například program Wolfram Mathematica nebo vědomostní a výpočetní engine Wolfram|Alpha, případně můžeme též zmínit některé matematické aplikace pro chytré telefony a tablety. V geometrii jsou pak celosvětově známy programy GeoGebra a Cabri.

Ve valné většině ale použití programů počítačové algebry ve výuce může svádět k jejich užívání jako tzv. černé skříňky, tedy zařízení, které dostane určitá vstupní data a uživatel následně bez jakékoliv znalosti dalších procesů uvnitř černé skříňky obdrží výsledek. Zároveň přiznejme, že v některých případech může potenciální uživatele odrazovat i cena, kterou je nutné za užívání licence zmíněných programů zaplatit.

V následujícím textu představíme jedno zajímavé využití programu MS Excel od společnosti Microsoft. Ten je sice především tabulkovým procesorem, ale s použitím programovacího jazyku Visual Basic for Applications (neboli VBA) lze prostřednictvím maker jeho možnosti podstatně rozšířit. Mezi tyto možnosti spadá kromě jiného i

„krokování“ a názorné představení některých procesů či matematických algoritmů. Ty jsou na rozdíl od klasického počítání tužkou na papíře v podstatě interaktivní a změna vstupních údajů se okamžitě projeví ve všech vypsaných krocích algoritmu i ve výsledku.

Nutnou podmínkou pro takovéto užívání Excelu je alespoň základní znalost algoritmizace a schopnost zapsat daný algoritmus ve zmíněném programovacím jazyce. To však není nikterak složité. Vše podstatné může zájemce dohledat v oficiální nápovědě programu či na internetu. Lze doporučit kupříkladu webové stránky *Jak na Excel* na adrese https://office.lasakovi.com/excel/vba/, kde je možné nalézt teoretický úvod do VBA, jakož i názorné ilustrační příklady pro lepší pochopení. (*MS Excel VBA Manual*, Lasák)

1 TESTOVÁNÍ PRVOČÍSELNOSTI

Nejprve se podívejme na záležitost, která je vcelku dobře přístupná již na druhém stupni základní školy, a sice na rozhodnutí, zda je zadané přirozené číslo prvočíslem, nebo číslem složeným. Obecně, především pro větší čísla, je sice vhodné použít některý z efektivnějších algoritmů (například Pollardův rho algoritmus, Pollardův $p - 1$ algoritmus a podobně), žáci se ale ve škole setkají především s jednoduššími postupy, mezi něž patří Eratosthenovo síto a hrubá síla – oba postupy jsou více méně experimentální, tudíž nejsou příliš efektivní, ale zato jsou dosti názorné.

Pro pořádek uveďme, že Eratosthenovo síto v podstatě není algoritmem rozhodujícím, zda zadané číslo náleží mezi prvočísla, nebo čísla složená, jak bychom si takový algoritmus představovali (tedy že na vstupu algoritmu je jedno číslo a výstupem je příslušná informace), nýbrž jde o postup užívaný pro nalezení všech prvočísel menších než předem zadaná mez.

Určování prvočíselnosti pomocí hrubé síly vychází ze skutečnosti, že pokud má být zadané přirozené číslo prvočíslem, nesmí být dělitelné žádným přirozeným číslem, počínaje číslem 2 a konče hodnotou $n - 1$. Je tedy nasnadě provést pokus o dělení beze zbytku všemi čísly z této množiny. Tento způsob můžeme trochu zefektivnit tím, že nebudeme testovat dělitelnost sudými čísly (pokud není zadané číslo dělitelné číslem 2, nemůže být dělitelné ani žádným dalším sudým číslem). Zároveň je možné snížit horní mez, takže stačí testovat dělitelnost pouze přirozenými čísly i splňujícími řetězec nerovností

$$2 \leq i \leq \sqrt{n}.$$

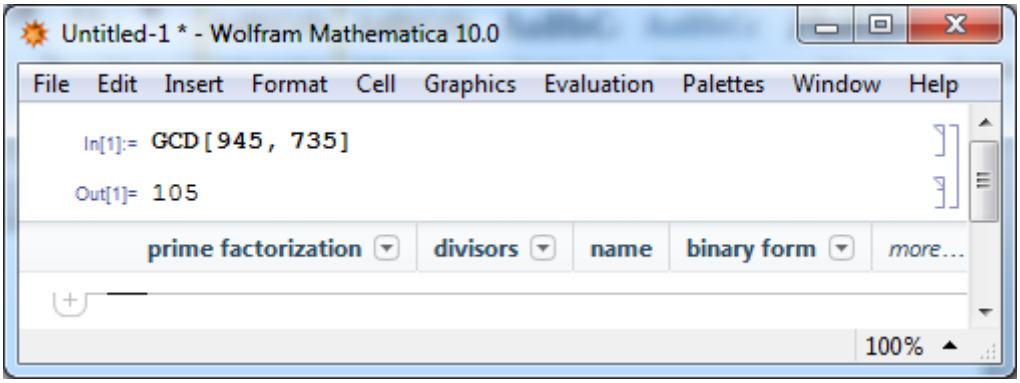
Úplné optimalizace bychom dosáhli ve chvíli, pokud bychom testovali dělitelnost čísla n pouze prvočísly, která jsou menší než hodnota $\sqrt{n}$. To je však ze zřejmých důvodů především u větších čísel hůře realizovatelné, neboť o každém z potenciálních dělitelů bychom nejprve museli rozhodnout, zda se jedná o prvočíslo, a teprve posléze bychom jím zkoušeli dělit číslo n. (Hefler, Rais)

V případě naší ukázky tedy proveďme částečnou optimalizaci. Hlídána je možnost, zda není zadané číslo rovno právě číslu 2 (v takovém případě je rovnou prohlášeno, že jde o prvočíslo), následně testujeme dělitelnost číslem 2 a dále pouze lichými čísly, a to až do hodnoty $\sqrt{n}$. Zápis algoritmu v programovacím jazyku VBA by pak vypadal takto:

```
Sub rozklad()
	Dim n As Integer, odmocnina As Integer
	Dim radka As Byte, i As Byte
	Dim prvočíslo As Boolean
	Range("B3").Select
	Range(Selection, ActiveCell.SpecialCells(xlLastCell)).Select
	Selection.ClearContents
	Selection.Interior.ColorIndex = x1None
	Range("B3").Select
	n = Cells(1, 2)
	prvocislo = True
	radka = 3
	If n > 2 Then
		If n Mod 2 = 0 Then
			prvocislo = False
			Cells(radka, 2) = "Testování 2: OK"
			Cells(radka, 2).Interior.ColorIndex = 4
			i = 4
		Else
			Cells(radka, 2) = "Testování 2: KO"
			Cells(radka, 2).Interior.ColorIndex = 3
			odmocnina = Fix(Sqr(n))
			i = 3
			Do While (i <= odmocnina) And (prvocislo)
				radka = radka + 1
				If n Mod i = 0 Then
					prvocislo = False
					Cells(radka, 2) = "Testování " & i & ": OK"
					Cells(radka, 2).Interior.ColorIndex = 4
				Else
					Cells(radka, 2) = "Testování " & i & ": KO"
					Cells(radka, 2).Interior.ColorIndex = 3
				End If
				i = i + 2
			Loop
		End If
		radka = radka + 2
		If prvocislo Then
			Cells(radka, 2) = "Číslo " & n & " je prvočíslo."
		Else
			Cells(radka, 2) = "Číslo " & n & " není prvočíslo. Lze jej dělit např.
		číslem " & i - 2 & "."
		End If
	ElseIf n = 2 Then
		Cells(radka, 2) = "Číslo 2 je prvočíslo."
	Else
		Cells(radka, 2) = "Toto číslo nelze na prvočíslenost testovat."
	End If
End Sub
```

2 NEJVĚTŠÍ SPOLEČNÝ DĚLITEL

Jako druhý v pořadí představme proces zjištění největšího společného dělitele dvou přirozených čísel, který bývá v matematickém softwaru reprezentován většinou příkazem GCD (greatest common divisor).



Obr. 1 Vstup a výstup příkazu GCD v programu Wolfram Mathematica

Zjištění největšího společného dělitele lze samozřejmě provést více způsoby – podle definice pomocí množin dělitelů, rozkladem zadaných čísel na prvočísla nebo Euklidovým algoritmem, který je vcelku univerzální a efektivní. Připomeňme, jak algoritmus funguje.

Mějme dvě přirozená čísla a a b a předpokládejme, že $a \geq b$ (pokud by tomu tak nebylo, provedli bychom přeznačení čísel). Potom existují dvě přirozená čísla q a Z_1 taková, že platí rovnost

$$a = q \cdot b + Z_1,$$

kde $0 \leq Z_1 < b$.

Pokud je $Z_1 = 0$, nalezli jsme největší společný dělitel čísel a a b, který je roven číslu b. Pokud naopak $Z_1 \neq 0$, použijeme čísla b a Z_1, pro která existuje dvojice přirozených čísel q_1 a Z_2. Pro ně platí rovnost

$$b = q_1 \cdot Z_1 + Z_2,$$

kde $0 \leq Z_2 < Z_1$.

Pokud je $Z_2 = 0$, nalezli jsme největší společný dělitel čísel a a b, který je roven číslu Z_1. Pokud naopak $Z_2 \neq 0$, pokračujeme v dalším zápisu s čísly Z_1 a Z_2. Další takovýto krok tedy existuje vždy, je-li daný zbytek Z_i nenulový.

Zároveň platí, že posloupnost čísel b, Z_1, Z_2, … je klesající a že jde o čísla přirozená. Je tedy zřejmé, že se jedná o posloupnost konečnou a její poslední člen Z_n je roven nule. Předposlední (nenulový) člen Z_{n-1} je hledaným největším společným dělitelem čísel a a b. (Rais, Drábek)

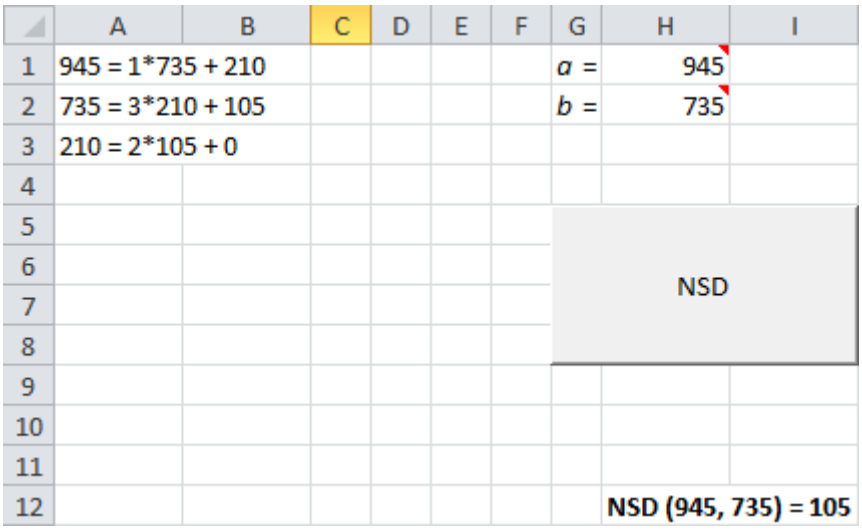
Tento algoritmus zapsaný v makru programovacího jazyku VBA vypadá například takto:

```
Sub NSD()
    Dim a As Integer, b As Integer, delitel As Integer, x As Integer, y As Integer, q As Integer, Z As Integer
    Dim radek As Byte
    Dim konec As Boolean
    Range("A1").Select
```

```
        Range(Selection, Selection.End(xlDown)).Select
        Selection.ClearContents
        Cells(12, 8).ClearContents
        konec = False
        radek = 1
        delitel = 1
        If Cells(1, 8) >= 1 And Cells(1, 8) <= 32767 And Cells(2, 8) >= 1 And Cells(2, 8)
<= 32767 Then
                If Cells(1, 8) > Cells(2, 8) Then
                        a = Cells(1, 8)
                        b = Cells(2, 8)
                Else
                        b = Cells(1, 8)
                        a = Cells(2, 8)
                End If
                x = a
                y = b
                Do Until konec
                        q = x \ y
                        Z = x Mod y
                        Cells(radek, 1) = x & " = " & q & "*" & y & " + " & Z
                        If Z = 0 Then
                                konec = True
                                delitel = y
                        End If
                        radek = radek + 1
                        x = y
                        y = Z
                Loop
                Cells(12, 8) = "NSD(" & a & ", " & b & ") = " & delitel
        Else
                Cells(5, 1) = "Největší společný dělitel nelze určit."
        End If
End Sub
```

Uživatel do příslušného listu a do příslušných buněk (buňky H1 a H2) zadá dvě přirozená čísla a stiskne tlačítko, které spustí výše uvedené makro.

Makro poté provede nejprve vymazání starých hodnot a výpočtů, které v listu mohly zůstat z předchozího užívání (buňka H12 a sloupec A), a kontrolu, zda jsou zadaná čísla ve správném rozsahu (pokud ne, podá o tom informaci). Dále je určeno, které ze zadaných čísel je větší (druhá podmínka If … End If), a vstoupí do cyklu s podmínkou na začátku (cyklus Do Until … Loop), v němž jsou počítány koeficienty q, q_1, q_2, ... a zbytky Z_1, Z_2, … Po každém výpočtu je kontrolováno, zda je aktuální zbytek nulový (třetí podmínka If … End If). Pokud ne, probíhá cyklus znovu, pokud ano, cyklus je ukončen. Zároveň s tím je při každém průběhu cyklu vypsána aktuální rovnost do příslušné buňky ve sloupci A. Po skončení cyklu je do buňky H12 zapsán výsledek.



Obr. 2 Výpis zjištění největšího společného dělitele čísel 945 a 735 pomocí makra

3 BUBBLESORT

V některých případech uživatel potřebuje setřídit určité množství hodnot podle velikosti. Pro tyto potřeby bývají užívány různé řadicí algoritmy, kterých je nepřeberné množství. Mezi ty nejjednodušší a nejlépe pochopitelné patří bubblesort, s nímž se žáci a studenti někdy seznamují již při úvodu do programování.

Algoritmus funguje takto: Z množiny n neseřazených hodnot je vybrána ta největší a je přesunuta na poslední (n.) pozici. V dalším kroku je z množiny prvních $n - 1$ neseřazených hodnot vybrána opět největší a je přesunuta na předposlední ($n - 1$.) pozici. Po celkem $n - 1$ opakováních je množina seřazena od nejmenší hodnoty k největší. (Neckář, *Wikipedia: Otevřená encyklopedie: Bublinkové řazení*)

Zápis algoritmu v makru vypadá takto:

```
Sub Bubblesort()
    Dim i As Byte, j As Byte, pom As Byte
    Dim test As Boolean
    Range("A2").Select
    Range(Selection, Selection.End(xlToRight)).Select
    Range(Selection, Selection.End(xlDown)).Select
    Selection.ClearContents
    Selection.Interior.ColorIndex = x1None
    Range("A1").Select
    test = True
    For i = 1 To 10
        If (Cells(1, i) < 0) Or (Cells(1, i) > 255) Then
            test = False
        End If
    Next i
    If test Then
        For i = 1 To 9
            For j = 1 To 10
                Cells(i + 1, j) = Cells(i, j)
            Next j
            For j = 1 To 10 - i
```

```
                    If Cells(i + 1, j) > Cells(i + 1, j + 1) Then
                        pom = Cells(i + 1, j + 1)
                        Cells(i + 1, j + 1) = Cells(i + 1, j)
                        Cells(i + 1, j) = pom
                    End If
                Next j
                For j = 10 - i To 9
                    Cells(i + 1, j + 1).Interior.ColorIndex = 4
                Next j
            Next i
            Cells(10, 1).Interior.ColorIndex = 4
        Else
            Cells(3, 1) = "Řazení nebylo provedeno."
        End If
End Sub
```

Makro po spuštění vstoupí do cyklu s pevným počtem opakování (vnější cyklus For … Next i). Při každém jeho opakování nejprve zkopíruje hodnoty z předchozího řádku do řádku následujícího (první vnitřní cyklus For … Next j), následně pak projde daný řádek zleva (druhý vnitřní cyklus For … Next j), přičemž porovnává dvě sousední hodnoty. Pokud je levá hodnota větší než pravá, hodnoty v buňkách jsou prohozeny (podmínka If … End If). Po každém proběhnutí vnějšího cyklu „probublá“ další z nejvyšších hodnot na svou pozici, kde je zabarvením buňky označena jako seřazená. V jednotlivých řádkách se tak postupně objevuje hledaná uspořádaná množina. Po ukončení algoritmu jsou v poslední řádce vypsány seřazené hodnoty.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	9	82	171	157	74	191	173	11	188	72						
2	9	82	157	74	171	173	11	188	72	191						
3	9	82	74	157	171	11	173	72	188	191						
4	9	74	82	157	11	171	72	173	188	191						
5	9	74	82	11	157	72	171	173	188	191						
6	9	74	11	82	72	157	171	173	188	191						
7	9	11	74	72	82	157	171	173	188	191						
8	9	11	72	74	82	157	171	173	188	191						
9	9	11	72	74	82	157	171	173	188	191						
10	9	11	72	74	82	157	171	173	188	191						

Generátor čísel

Spustit bubblesort

Obrázek 3 – Výpisy v průběhu řazení bubblesortem

Poznamenejme ještě, že tlačítko Generátor čísel slouží ke smazání starých hodnot, které mohly v listu zůstat z předchozího řazení, a k vyplnění buněk A1 až J1 náhodně vygenerovanými čísly (zde v rozmezí 0 až 199; generování je provedeno samostatným makrem). Uživatel této možnosti může využít nebo může do příslušných buněk vyplnit čísla vlastní.

4 DALŠÍ ALGORITMY

Algoritmy uvedené v předchozím textu jsou samozřejmě jen jednoduchými zástupci celé řady algoritmů, které je možné v prostředí VBA připravit a využít je pro názorné demonstrace způsobu provedení daných výpočtů. Nicméně i tak ukazují, že i s nástrojem, jakým je Excel, lze žákům a studentům přiblížit základy algoritmizace.

Vyzkoušet nejen výše popsané algoritmy, ale i některé další, můžete na webové adrese http://home.zcu.cz/~honzikl/vba/, kde jsou k dispozici v podobě souborů pro aplikaci Excel s podporou maker. Tyto soubory navíc vždy obsahují i krátkou dokumentaci k danému algoritmu, v níž je popsán způsob práce algoritmu, vstupní a výstupní data a zápis algoritmu s komentáři jednotlivých kroků.

Literatura

1. DRÁBEK, J. Texty přednášek k předmětu KMT/ELA.
2. HEFLER, S. (2015). *Prvočísla a faktorizace celých čísel*. Plzeň, 2015. Diplomová práce. Západočeská univerzita v Plzni. Vedoucí práce doc. RNDr. Jaroslav Hora, CSc.
3. LASÁK, P. *Jak na Excel* [online]. 2004-2021 [cit. 2021-06-05]. Dostupné z: http://office.lasakovi.com/excel/
4. *MS Excel VBA Manual*. MTC Training Solutions Limited.
5. NECKÁŘ, J. Bubble sort. *Algoritmy* [online]. 2016 [cit. 2021-06-09]. Dostupné z: https://www.algoritmy.net/article/3/Bubble-sort
6. RAIS, M. (2011) Prvočíslený rozklad a jeho užití. Plzeň, 2011. Diplomová práce. Západočeská univerzita v Plzni. Vedoucí práce doc. PaedDr. Jana Coufalová, CSc.
7. *Wikipedie: Otevřená encyklopedie: Bublinkové řazení* [online]. c2018 [cit. 2021-06-10]. Dostupné z: https://cs.wikipedia.org/wiki/Bublinkov%C3%A9_%C5%99azen%C3%AD

Kontakty

PhDr. Lukáš Honzík, Ph.D.
Západočeská univerzita v Plzni, Fakulta pedagogická
Klatovská tř. 51, 306 14 Plzeň
Tel: +420 377 636 285
E-mail: honzikl@kmt.zcu.cz

Mgr. Jan Frank, Ph.D.
Západočeská univerzita v Plzni, Fakulta pedagogická
Klatovská tř. 51, 306 14 Plzeň
Tel: +420 377 636 452
E-mail: frankjan@kmt.zcu.cz