

VBA jako nástroj tvorby vlastních funkcí pro zkvalitnění výstupu v MS Excel

Ing. Petr Dydowicz, Ph.D.¹

e-mail: dydowicz@skolskykomplex.cz

¹ I. Německé zemské gymnasium, základní škola a mateřská škola, o. p. s., Brno

Klíčová slova

Visual Basic for Application (VBA), MS Office, MS Excel, funkce SUMA, vlastní funkce.

1 Popis problému

V prostředí MS Excel se data, čas od času, importují [3] ze souboru s jiným datovým formátem (txt, csv) do výchozího formátu sešitu aplikace MS Excel (xlsx). Může však nastat situace, kdy díky nekonzistentnímu importu dat vznikne problém, který může mít pro uživatele fatální následky. Problém spočívá v tom, že některá data mohou mít v importovaném souboru jiný datový formát, než je očekávaný na výstupu. Interní funkce MS Excel pro matematické operace v některých případech ignorují hodnoty buněk, které nejsou číselné a do výpočtu funkce, prostřednictvím svých argumentů, nevstupují. Běžný uživatel pak v domněnku, že došlo ke korektnímu zpracování dat, může získat nerelevantní výsledky. Interní funkce sice ignorují buňky, jejichž datový typ není numerický, ale na tento fakt uživatele neupozorní.

Tento článek si proto klade za cíl o tomto nežádoucím efektu některých matematických interních funkcí nejen informovat, ale rovněž poukázat na způsob, jak jej eliminovat. K tomu bude využit nástroj Visual Basic for Application (dále jen VBA), který je interním nástrojem balíku MS Office. Výstupem článku bude návrh, rozbor a vytvoření ukázkové funkce v prostředí VBA, která by uživatele na nekonzistenci vstupních dat upozornila a tím zamezila nežádoucím výsledkům. V článku jsou použity ukázky běžných metod matematických operací pro hromadné zpracování dat v prostředí MS Excel, jejich porovnání včetně opatření na eliminaci nežádoucího efektu.

2 Popis prostředí

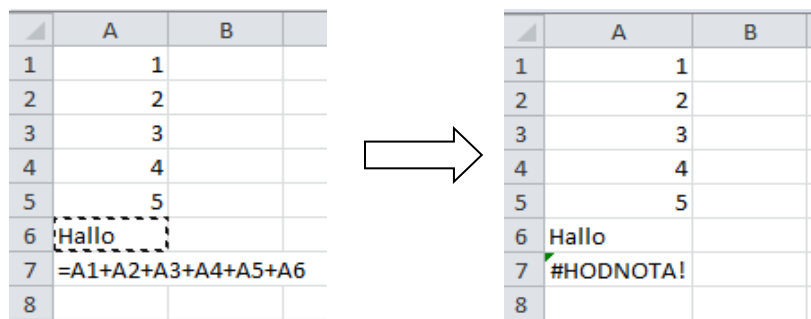
Současná funkcionalita, kterou poskytuje tabulkový procesor MS Excel, umožňují zdokonalovat výpočetní metody formou vlastní tvorby funkcí prostřednictvím nástroje VBA. Množina interních funkcí, která je implementována v prostředí MS Excel, má pro využití v praxi zajisté obrovský potenciál. Otázkou však zůstává, zdali filosofie zpracování dat prostřednictvím těchto (interních) funkcí je shodná s filosofií konkrétního uživatele. Pokud se liší, je možné prostřednictvím nástroje VBA vytvořit vlastní funkce tak, aby splňovaly veškeré požadavky na očekávaný výstupní formát zpracovávaných dat. A to je důvod, proč tento článek vznikl. Cílem článku je tedy nejen poukázat na často opomíjené možnosti MS Excel s použitím VBA, ale taky ukázat příkladem, jakým způsobem lze eliminovat nedostatky, vyplývající z očekávaných výstupů interních funkcí s využitím vlastních funkcí. Pro dosažení tohoto cíle byla použita metoda programování vlastních funkcí, umístěna v modulu sešitu XLSM. Příklad vlastní funkce, který bude následovat v textu tohoto článku, lze pojmut jako element z množiny vlastních funkcí, kterou lze použít v libovolném sešitu (nezávisle na původním sešitu XLSM, ve které tato funkce vznikla) prostřednictvím tzv. doplňku sešitu. VBA je programovacím jazykem implementovaným do prostředí Microsoft office a slouží k tvorbě jednoduchých aplikací a to jak na úrovni desktopových aplikací (formuláře, ovládací prvky ActiveX), tak jej lze využít pro tvorbu vlastních funkcí či jednoduchých maker (bez znalosti programování).

2.1 Výstupy funkcionalit MS Excel

V této kapitole chci představit různé způsoby matematického zpracování elementárních hodnot umístěných v buňkách, přičemž chci poukázat na odlišné výsledky. Jako příklad uvedu použití vzorců a funkcí pro sčítání hodnot v množině buněk, kde některé buňky jsou nenumerní (nebo jsou numerické, ale jsou formátovány jako datum), např. text nebo logická hodnota PRAVDA či NEPRAVDA.

Příklad vzorce

Vzorec je tvořen odkazy na jednotlivé buňky, a matematickými operátory jsou znaky pro sčítání. Buňky na adresách A1 až A5 obsahují numerické hodnoty, buňka na adrese A6 obsahuje textovou hodnotu. V buňce A7 je umístěn vzorec pro součet buněk A1 až A6. Výsledkem vzorce výraz #HODNOTA, k níž dojde, když parametr funkce je chybného datového typu nebo se vzorec pokouší provést operaci pomocí chybných dat [1].



The diagram shows two spreadsheets. The left spreadsheet has columns A and B. Row 1: A=1, B=; Row 2: A=2, B=; Row 3: A=3, B=; Row 4: A=4, B=; Row 5: A=5, B=; Row 6: A=Hallo, B=; Row 7: A>=A1+A2+A3+A4+A5+A6, B=; Row 8: empty. An arrow points to the right spreadsheet. The right spreadsheet has the same data for rows 1-5, but row 6: A=Hallo, B=; row 7: A=#HODNOTA!, B=; row 8: empty.

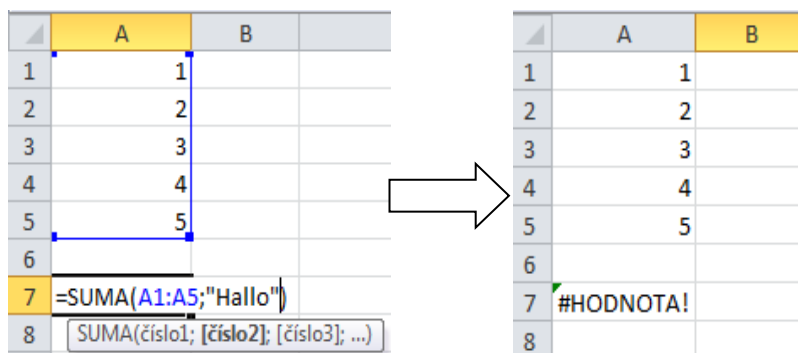
Vlevo: Vzorec v buňce A7, vpravo výstup vzorce v buňce A7. Zdroj: vlastní zpracování autora

Možné příčiny:

- Nejméně jedna buňka ve vzorci obsahuje text a vzorec provádí s těmito buňkami matematický výpočet pomocí standardních aritmetických operátorů (+, -, * a /). Například vzorec =A1+B1, kde buňka A1 obsahuje řetězec Ahoj a buňka B1 obsahuje číslo 3, vrátí chybu #HODNOTA!.
- Vzorec obsahující matematickou funkci, například SUMA, SOUČIN nebo PRŮMĚR, obsahuje jako argument textový řetězec místo čísla. Například vzorec SOUČIN(3;"Ahoj") vrátí chybu #HODNOTA!, protože funkce SOUČIN vyžaduje argumenty numerického typu.
- Sešit používá datové propojení, které není dostupné [5].

Příklad funkce A

Tato funkce má 2 argumenty. Prvním argumentem je rozsah buněk A1 až A5, druhým argumentem je textová konstanta „Hallo“. Funkce je umístěna v buňce A7. Funkce vrací chybovou zprávu, chyba vznikla nekompatibilitou vstupních hodnot. S tímto výsledkem lze jen souhlasit, nelze sčítat hodnoty nekompatibilních typů.



The diagram shows two spreadsheets. The left spreadsheet has columns A and B. Row 1: A=1, B=; Row 2: A=2, B=; Row 3: A=3, B=; Row 4: A=4, B=; Row 5: A=5, B=; Row 6: empty; Row 7: A>=SUMA(A1:A5;"Hallo"), B=; Row 8: A=SUMA(číslo1; [číslo2]; [číslo3]; ...), B=. A blue selection box highlights cells A1:A5. An arrow points to the right spreadsheet. The right spreadsheet has the same data for rows 1-5, but row 6: empty; row 7: A=#HODNOTA!, B=; row 8: empty.

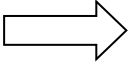
Vlevo: Funkce SUMA v buňce A7, vpravo výstup funkce SUMA v buňce A7. Zdroj: vlastní zpracování autora

Příklad funkce B

Jediným argumentem funkce je rozsah buněk A1 až A6 se stejnými hodnotami jako u předchozího příkladu. Funkce je umístěna v buňce A7. Funkce vrátila číselnou hodnotu 15, což je hodnota, která odpovídá součtu hodnot buněk A1 až A5. Výsledek je numerický, přičemž jedna z buněk definovaného rozsahu A1 až A6 byla nenumernická – textová. Došlo k situaci, ve které buňka, která neobsahovala numerickou hodnotu, byla ve výpočtu funkce eliminována. V literatuře a koneckonců i v online nápovědě se uvádí, že výstup funkce nebo vzorce #HODNOTA!, vznikne tehdy, pokud při matematických operacích jsou vstupní hodnoty datově

nekompatibilní – obsahují text. Pokud je ale argumentem funkce odkaz na buňku, která obsahuje text, pak je funkce na rozdíl od vzorce vyhodnocena „bezchybně“ – buňka s textovou hodnotou nevstupuje do výpočtu. Naskytá se však otázka, proč vzorec =A1+A2+A3+A4+A5+A6 (buňky A1 až A5 jsou číselné, buňka A6 je textová) je ukončen chybou #HODNOTA!, ale funkce =SUMA(A1;A2;A3;A4;A5;A6) (totéž jako =SUMA(A1:A6)) s hodnotami argumentů shodnými s buňkami výše uvedeného vzorce, vrací číselnou hodnotu. Nutno podotknout, že funkce kontrola chyb v možnostech vzorce (nastavení pravidel MS Excel) je trvale aktivní.

	A	B
1	1	
2	2	
3	3	
4	4	
5	5	
6	Hallo	
7	=SUMA(A1:A6)	
8		

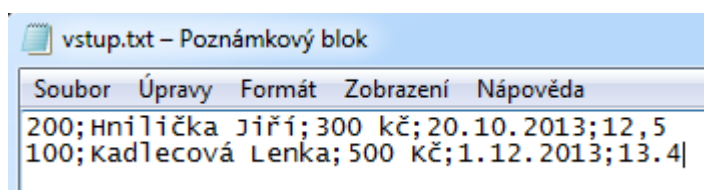


	A	B
1	1	
2	2	
3	3	
4	4	
5	5	
6	Hallo	
7		15
8		

Vlevo: Funkce SUMA v buňce A7, vpravo výstup funkce SUMA v buňce A7. Zdroj: vlastní zpracování autora

Pokud porovnáme výstupy funkce A a B, zjistíme, že jediným rozdílem je existence druhého argumentu ve funkci A. Hodnotou argumentu není odkaz na buňku, ale textová konstanta „Hallo“. Ve funkci B je text „Hello“ hodnotou buňky A6, tedy buňky, která vstupuje do výpočtu funkce SUMA a to jako prvek rozsahu (pole) buněk A1 až A6. Tento rozsah je jediným argumentem funkce SUMA.

Otázkou zůstává, zdali je výsledek řešení funkce B z pohledu zpracování většího množství dat logicky správný, či nikoliv. Pokusím se namodelovat situaci, ke které může dojít kdekoliv, kde se hromadně zpracovávají data v prostředí MS Excel. Uživatel používá aplikaci v prostředí MS Excel pro hromadné zpracování dat. Jedná se o tabulku na jediném listu, která má řádově stovky sloupců a řádků. Data jsou pravidelně obnovována a to exportem z externího souboru nekompatibilního typu (txt). Jedná se o konstantní počet dat, to znamená, že v čase dochází pouze ke změně hodnot na konkrétních buňkách. Na konstantní adrese jsou uloženy výpočtové funkce, např. SUMA nebo PRŮMĚR. Po načtení dat (export z externího souboru) dojde k přepočítání funkcí. Jelikož se jedná o list s velkým objemem vstupních dat, žádná vstupní, vizuální kontrola hodnot v buňkách po exportu se neprovádí. Může dojít k situaci, kdy některá z buněk, které vstupují do výpočtu např. funkcí SUMA prostřednictvím argumentu funkce, nebude numerického datového typu. Jak už bylo uvedeno výše, běžná interní funkce SUMA bude tuto buňku eliminovat, protože se nejedná o číslo. Výstupem funkce SUMA je pak hodnota, která sice vychází ze součtu hodnot prostřednictvím argumentů funkce, ale pouze číselných a to i tehdy, kdy chybou lidského faktoru došlo k zadání špatného typu vstupních dat.



Příklad struktury vstupních dat z formátu txt (VSTUP.TXT). Zdroj: vlastní zpracování autora

Na obrázku je vidět struktura věty textového souboru pro export do tabulky MS Excel. První položka je číselná, druhá položka je textová, 3. Položka je opět číselná, ale za předpokladu, že symbol měny odpovídá formátovacím pravidlům MS Excel pro formátování měny. Čtvrtá položka je typu datum, poslední položka je číselná s desetinnou částí. Všimněte si, že hodnota třetí položky, konkrétně symbol měny se v první větě liší od symbolu ve druhé větě. Rozdíl je pouze ve velikosti prvního písmene názvu měny. Korektní název, který MS Excel akceptuje je „Kč“ (bez uvozovek).

E2		fx		13.4.2013	
	A	B	C	D	E
1	200	Hnilička Jiří	300 Kč	20.10.2013	12,5
2	100	Kadlecová Lenka	500 Kč	1.12.2013	13.4

Data v tabulce po provedeném importu ze souboru VSTUP.TXT. Zdroj: vlastní zpracování autora

Na obrázku tabulky je vidět, jak následný export „dopadl“. V buňce C1 je textová hodnota, zatímco v buňce C2 je numerická hodnota s formátem měny. Jestliže nebylo použito na těchto buňkách zarovnávání, pak přirozené zarovnávání textu je doleva a pro číslo doprava. Za povšimnutí stojí i hodnota v buňce E2. Její hodnota je 13.4, v tomto případě oddělovačem celého čísla a desetinných míst není čárka, ale tečka. Systém tuto hodnotu chápe jako datum s formátem den a měsíc. Skutečnou hodnotu lze vidět v řádku vzorců – 13. 4. 2013.

Tyto chyby vznikly už při vytváření vstupního souboru. Pokud neexistuje v aplikaci kontrolní mechanismus, který striktně kontroluje datové typy buněk po exportu do tabulky, uživatel se pak vystavuje možné situaci, kdy může získat zkreslené výsledky výstupem matematických funkcí, např. SUMA.

Jak již bylo psáno výše, jestliže bude použita funkce SUMA, s argumenty odkazujícími na buňky C1 a C2 (=SUMA(C1;C2) nebo =SUMA(C1:C2)), pak funkce vrátí (v obou verzích) číselnou hodnotu 500. Jestliže sečtu pomocí funkce SUMA buňky E1 a E2, dostanu výsledek 41389,5. Výsledek je součtem číselné hodnoty 12,5 (buňka C1) a hodnoty 41377, což je celočíselná hodnota odpovídající datu 13. 4. 2013.

3 Návrh, popis a realizace řešení

Pokud neexistuje kontrolní mechanismus pro zjištění nevhodných datových typů hodnot buněk, je vhodné vytvořit vlastní funkci, které bude provádět součet stejně jako funkce SUMA, ale s tím rozdílem, že vstoupí-li prostřednictvím argumentu funkce do výpočtu buňka s hodnotou nenumernickou, výpočet bude přerušen a uživatel aplikace bude na tento fakt upozorněn systémovým hlášením a v buňce, ve které bude umístěna tato funkce, se zobrazí text #MISHMASH.

Funkce bude vytvořena v prostředí VBA, bude umístěna v samostatném modulu jako veřejná funkce, bude se jmenovat SUMEX a stejně jako interní funkce suma bude mít neomezený počet argumentů. Funkce bude dostupná uživateli v katalogu funkcí v kategorii vlastní.

```

1 Function SUMEX(zobraz As Boolean, ParamArray pole() As Variant)
2   Dim rozsah, prvek
3   Dim popis As String
4   SUMEX = 0
5   popis = ""
6   For Each rozsah In pole()
7     If IsObject(rozsah) Then
8       For Each prvek In rozsah
9         If (VarType(prvek) Like "[7,8]") Or (VarType(prvek) = 11) Then
10          popis = popis & prvek.Address & chyba(VarType(prvek)) & prvek.Value & vbCrLf
11        Else: SUMEX = SUMEX + prvek
12        End If
13      Next prvek
14    ElseIf (VarType(rozsah) Like "[7,8]") Or (VarType(rozsah) = 11) Then
15      popis = popis & " argument funkce " & chyba(VarType(rozsah)) & rozsah & vbCrLf
16    Else: SUMEX = SUMEX + rozsah
17    End If
18  Next rozsah
19  If Len(Trim(popis)) > 0 Then
20    If zobraz Then popis = MsgBox(popis, 16)
21    SUMEX = "#MISHMASH"
22  End If
23 End Function

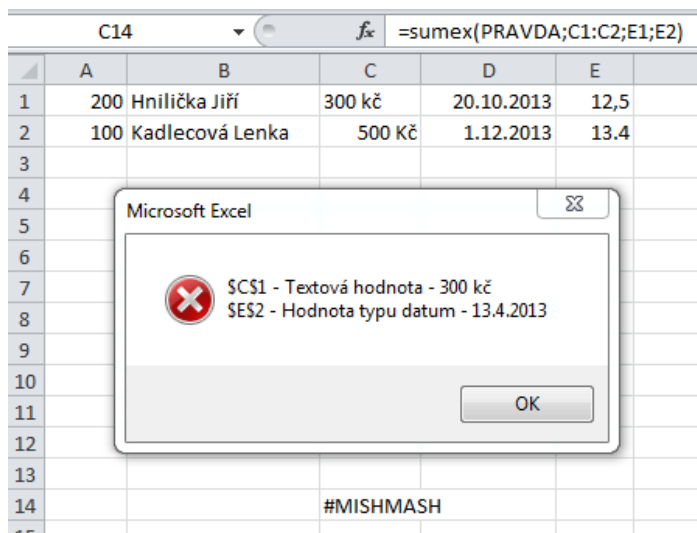
```

Kód funkce SUMEX s doplněnými čísly řádků (vlevo). Zdroj: vlastní zpracování autora

3.1 Popis funkce

Funkce má 2 argumenty. 1. Argument je typu Boolean (logický datový typ). Jestliže je tento argument uživatelem nastaven na hodnotu PRAVDA, pak se v případě chybného vstupu aktivuje dialogové okno MSGBOX se seznamem všech zjištěných chyb a to v pořadí:

- Adresa buňky, ve které byla nalezena nekompatibilita datových typů
- Popis chyby (Hodnota typu datum, Textová hodnota, Logická hodnota)
- Hodnota buňky



Hlášení funkce SUMEX, pokud je první argument funkce nastaven na TRUE. Zdroj: vlastní zpracování autora

Pokud uživatel zadá do prvního argumentu hodnotu NEPRAVDA, pak se dialog nezobrazí, ale v buňce, ve které je umístěna funkce SUMEX se zobrazí text: #MISHMASH.

Druhý argument funkce SUMEX je typu ParamArray, tedy uživatel může vkládat neomezený počet argumentů funkce. Jedná se tedy o dynamické pole argumentů POLE() typu Variant. Jednotlivými prvky POLE() pak mohou být objekty typu Range (odkaz na buňku nebo rozsah buněk) nebo číselné konstanty.

Návratový datový typ funkce není deklarován, tedy je implicitně typu Variant. Je to logické, jelikož očekávaným výstupem funkce SUMEX je číselná hodnota nebo, v případě chyby, text #MISHMASH. Vlastní tělo funkce začíná deklarací proměnných a u některých, včetně identifikátoru funkce, probíhá prvotní inicializace na výchozí hodnotu[6].

Jelikož každý prvek POLE() může být objektem (buňka nebo rozsah buněk) nebo konstantou, je potřeba provést prvotní test na každý prvek ParamArray POLE(). Pro test každého prvku by vybraný cyklus typu For Each ...Next [2] u kterého se nedefinuje počáteční ani koncová hodnota čítače cyklu (řádek 6). Pro každý prvek se pak provede analýza funkcí IsObject (nebo IsArray), kde se zjišťuje, zdali se jedná o objekt (buňka, seznam buněk), či nikoliv (konstanta). Pokud funkce IsObject (IsArray) [4] s parametrem testovaného prvku POLE() vrátí hodnotu TRUE, znamená to, že testovaný prvek je objektem. Následně se spustí druhá analýza (řádek 8), jelikož prvkem POLE() může být rozsah buněk, tedy opět Array [6].

Na řádku 7 se provádí test, zdali hodnota buňky je datového typu text, datum nebo logická. Pokud je podmínka splněna, do proměnné popis se kumulativně uloží ji dříve zmiňované hodnoty pro dialogové chybové hlášení (řádek 10). V případě, že podmínka není splněna (nejedná se o text, logiku nebo datum), pak dojde k přičtení hodnoty buňky k identifikátoru funkce.

Jestliže prvek POLE() není objektem, jedná se tedy o konstantu, kterou uživatel uložil přímo jako argument funkce, pak je zde proveden stejný test na datový typ (řádek 14), jako u předchozího kódu. Je na povážení, zdali test dělat, či nedělat, ovšem může se stát, že uživatel omylem vloží do argumentu funkce textovou hodnotu, pak by ovšem výsledek mohl být zkrácen.

Po ukončení prvotního cyklu (řádek 19) je proveden test na délku proměnné, do které se, v případě výskytu chyby, ukládá výše zmiňovaný text s využitím uživatelské, privátní funkce CHYBA (řádek 10 a 15). Pokud k žádné chybě nedošlo má proměnná nulovou délku a funkce SUMEX vrací číselnou hodnotu.

```

Private Function chyba(cislo As Byte) As String
    Select Case cislo
        Case 7: chyba = " - Hodnota typu datum - "
        Case 8: chyba = " - Textová hodnota - "
        Case 11: chyba = " - Logická hodnota - "
    End Select
End Function

```

Kód privátní funkce Chyba, která vrací popis chyby. Zdroj: vlastní zpracování autora

Test na datový typ byl proveden funkcí VBA – VarType (řádek 9 a 14). Pro tento účel není možné použít funkci IsNumeric, jelikož tato funkce vrací hodnotu TRUE i v případě, že se jedná o datum nebo text, ovšem složeným zřetěžením číslic.

4 Závěr

Tento příspěvek názorně ukazuje, jakým způsobem lze vytvořit vlastní funkci, pokud nejsme spokojeni s obdobné interní funkcí. Článek však popisuje pouhý zlomek toho, co prostředí VBA umožňuje a jak silný nástroj pro tvorbu vlastních aplikací nabízí. Cíl tohoto článku byl naplněn, bylo demonstrováno, jak lze pomocí programovací metody – tvorba vlastní funkce vytvořit plnohodnotnou funkci, která, na rozdíl od interní, vrací logický, správný výsledek a to i v případě, že argumenty funkce ukazují na buňky, jejíž datový typ není číselný. Přínos tohoto článku vidím hlavně v možnostech tvorby vlastních funkcí, které lze používat kdekoliv v praktickém životě tam, kde dochází k hromadnému zpracování dat a tak minimalizovat možná rizika spojená s nerelevantními výstupy.

Obdobným způsobem lze vytvářet další vlastní funkce, jejíž funkcionalita není součástí interních funkcí MS Excel nebo jednoduše vstupně-výstupní formát stávajících interních funkcí neodpovídají požadavkům uživatele.

Citace

- [1] BARILLA, Jiří; SIMR, Pavel; SÝKOROVÁ, Květuše. *Microsoft Excel 2016*. Brno: Computer Press, 2016. 456 s. ISBN 978-80-251-4838-9.
- [2] KRÁL, Martin. *Excel VBA. Výukový kurz*. Brno: Computer Press, a.s., 2010. 504 s. ISBN 978-80-251-2358-4.
- [3] LAURENČÍK, Marek. *Excel pokročilé nástroje*. Praha: Grada, 2016. 224 s. ISBN 978-80-247-5570-0.
- [4] WALKENBACH, John. *Microsoft Office Excel 2007. Programování ve VBA*. Brno: Computer Press, a.s., 2008. 912 s. ISBN 978-80-251-2011-8.
- [5] Online Help MS Excel 2010
- [6] Internetové zdroje: <https://office.lasakovi.com/excel/vba-funkce-vzorce/Uzivatelске-funkce-vcraceni-chyb-Excel/>