Kapitola 3

Úvod do skriptování

Používání Octave nebo Matlabu jako kalkulačky je sice zajímavé, možnosti těchto programů tím však nejsou ani zdaleka vyčerpány. Jejich síla spočívá v možnosti skriptování, tj. využití textových souborů obsahujících posloupnosti příkazů a definice funkcí. Velmi jednoduchý skript je uveden v následujícím výpisu.

```
1  % reseni kvadraticke rovnice
2  % koeficienty kvadraticke rovnice
3  a=2;b=-1;c=4;
4  %diskriminant
5  D=b^2-4*a*c;
6  % reseni
7  x1=(-b+sqrt(D))/2/a;
8  x2=(-b-sqrt(D))/2/a;
9  % vypis reseni
10 x1
11 x2
```

Řádky začínající znakem procenta (%) jsou během zpracování skriptu programem ignorovány. Slouží k zápisu komentářů autora či uživatele skriptu. Na třetím řádku uložíme do proměnných `a`, `b` a `c` číselné hodnoty. Příkazy jsou zakončeny středníkem, hodnoty tak nebudou v průběhu činnosti skriptu vypisovány na obrazovku. Čtvrtý řádek je opět komentář, který odkazuje na výpočet diskriminantu na řádku následujícím. Šestý řádek je další komentář. Sedmý a osmý řádek obsahují vzorce pro výpočet kořenů kvadratické rovnice. Závěrečné dva řádky vypíší výsledek na obrazovku.

Pokud uložíme tento skript do souboru `skript1.m`, spustíme jej napsáním jména `skript1` (bez přípony!) do příkazového okna. Co skript dělá, je ukázáno v následujícím výpisu.

```
octave:1> skript1
x1 =  0.25000 + 1.39194i
x2 =  0.25000 - 1.39194i
octave:2>
```

Tento výpis uvidíte jen v případě, že skript se nachází v adresáři, který Matlab/Octave prochází při hledání jména skriptu. Jinak se vypíše chybové hlášení, že uvedený příkaz není definován. Pokud se vám chybové hlášení objeví, nevěšte hlavu. Máte dvě možnosti, jak to napravit. Zaprvé, můžeme použít příkaz `cd` (change directory). Máte–li skript uložený v adresáři třeba (v linuxu např. `/home/ja/skripty`, ve Windows `D:\skripty`), napíšete příkaz `cd /home/ja/skripty` (`cd D:\skripty`). Tím se přepnete do daného adresáře, což si můžete ověřit příkazem `pwd` (path to working directory). Tento příkaz vypíše na obrazovku jméno adresáře, ve kterém se právě nacházíte. Druhý, daleko efektivnější způsob je využití nastavení cesty k adresáři, kde jsou uloženy skripty do seznamu adresářů, které Octave/Matlab automaticky prohledává. V případě Matlabu se přidá adresář se skripty do `PATH` pomocí položky `Set Path` v menu `File`. V Octave se adresář se skripty přidá do prohledávaných adresářů příkazem `addpath(jméno_adresáře)`. Pokud chcete nastavení uchovat, uložte nový seznam adresářů příkazem `savepath`.

3.1 První skript

Zkusme nyní uvedený skript upravit. Určitě by bylo vhodnější, kdybychom mohli hodnotu parametrů kvadratické rovnice zadat až po spuštění skriptu. K tomu slouží funkce `input(`*'řetězec'*`)`, viz výpis níže:

```
1  % reseni kvadraticke rovnice
2  a=input('zadej a: ');
3  b=input('zadej b: ');
4  c=input('zadej c: ');
5  %diskriminant
6  D=b^2-4*a*c;
7  % reseni
8  x1=(-b+sqrt(D))/2/a;
9  x2=(-b-sqrt(D))/2/a;
10 % vypis reseni
11 disp(x1);
12 disp(x2);
```

Při běhu programu jsme nejdříve vyzváni k zadání konstant `a`, `b`, `c` (řádky 2 až 4). Kořeny kvadratické rovnice jsou pak vypsány na obrazovku prostřednictvím příkazu `disp(`*proměnná*`)` na řádcích 11 a 12.

3.2 Funkce jako skript

Skript, který jsme zatím vytvořili je sice funkční, ale má svá omezení. Mnohem častější používání skriptů jsou funkce a procedury, které zpracovávají vstupní hodnoty a vrací hodnoty výstupní. Ukažme si, jak by vypadala funkce, která řeší kvadratickou rovnici.

```
1 function [x1,x2]=KvadrRce(a,b,c)
2 % funkce resi kvadratickou rovnici
3 % [x1,x2]=KvadrRce(a,b,c),
4 % kde a,b,c jsou parametry kv.rce
5 %
6   D=b^2-4*a*c; % diskriminant
7 % reseni
8   x1=(-b+sqrt(D))/2/a;
9   x2=(-b-sqrt(D))/2/a;
```

Uvedený skript uložíme do souboru pod názvem `KvadrRce.m`. V souboru s koncovkou `m` může být uvedena jedna nebo vvíce funkcí[1] a jméno souboru se **musí** shodovat se jménem první funkce. Je třeba dát pozor, neboť Matlab i Octave rozlišují mezi velkými a malými písmeny.

Skript, který obsahuje funkci, začíná klíčovým slovem `function` (1. řádek). Následuje seznam návratových hodnot (v hranatých závorkách) a jméno funkce se vstupními parametry. Řádky začínající znakem % značí komentáře (2 až 5). Je velmi vhodné psát úvodní komentáře, resp. popis co funkce dělá, co jsou vstupní a výstupní hodnoty. Tento komentář (mezi definicí funkce a prvním příkazem) se zobrazí po zavolání funkce `help`.

```
octave:1> help KvadrRce
funkce resi kvadratickou rovnici
[x1,x2]=KvadrRce(a,b,c),
kde a,b,c jsou parametry kv.rce

KvadrRce is the user-defined function from the file
/home/standa/texty/vyuka/matlab/2/KvadrRce.m
```

Na řádku 6 se provádí výpočet diskriminantu, kořeny kvadratické rovnice jsou počítány na řádcích 8 a 9. Funkce se jednoduše zavolá příkazem `[koren1, koren2]=KvadrRce(1,2,3)`

```
octave:2> [koren1, koren2]=KvadrRce(1,2,3)
koren1 = -1.0000 + 1.4142i
koren2 = -1.0000 - 1.4142i
octave:3>
```

[1] Ve skriptu mohou být deklarovány i další funkce – ty jsou však pouze lokální, tj. mohou je využít jen funkce deklarované ve stejném souboru.

Do proměnných `koren1` a `koren2` se zapíší hodnoty kořenů rovnice. Není–li příkaz zakončen středníkem, vypíší se výstupní hodnoty na obrazovku.

3.3 Jak psát skripty

Pro psaní skriptů existuje několik doporučení. Shrňme je do následujícího stručného výčtu:

1. jméno skriptu a (první) funkce se MUSÍ shodovat, včetně velikosti písmen
2. ve skriptu lze deklarovat více funkcí - kromě první z nich jsou však pouze lokální, tj. mohou je volat pouze funkce v daném skriptu
3. **zdrojový kód opatřte výstižnými komentáři** !!!!!!!!!
4. pište strukturovaně, tj. využívejte odsazení. Oceníte to zvláště v případě cyklů, větvení apod.

Bod 1) je povinný, zbylé jsou následování hodná doporučení, která přispívají k přehlednosti.

Ukažme si ještě, jak by vypadal skript, ve kterém je deklarováno více funkcí:

```
1  function [x1,x2]=KvadrRce(a,b,c)
2  % funkce resi kvadratickou rovnici
3  % [x1,x2]=KvadrRce(a,b,c),
4  % kde a,b,c jsou parametry kv.rce
5  %
6    D=Diskriminant(a,b,c);
7  % reseni
8    x1=(-b+sqrt(D))/2/a;
9    x2=(-b-sqrt(D))/2/a;
10
11 function d=Diskriminant(a,b,c)
12   d=b^2-4*a*c; % diskriminant
```

Použití uvedené funkce je stejné, jako v předchozím případě. Pokud však budeme chtít použít pouze funkci `Diskriminant`, Octave/Matlab nahlásí chybu, že uvedená funkce není deklarována. To je způsobeno tím, že „zvenčí“ jsou přístupné pouze první funkce ve skriptu (jejichž jméno se shoduje s jménem skriptu).

Funkce zapsaná ve skriptu může volat jiné skripty (funkce). Pokud nechceme, resp. nepotřebujeme, definovat nějakou funkci jako skript, můžeme využít tzv. `inline` funkce:

```
octave:1> fx=inline('cos(x)*sin(x)');
octave:2> fx(pi/6)
```

```
ans =  0.43301
octave:3> fxy=inline('cos(x)*sin(y)');
octave:4> fxy(pi/6,pi/3)
ans =  0.75000
octave:5>
```

3.4 Cykly a podmínky

Při psaní skriptů se neobejdeme bez cyklů a podmínek. Octave i Matlab disponují cykly `for` a `while` a podmínkami `if` a `switch`. Ukažme si, jak se jednotlivé cykly a podmínky definují a používají. Začněme podmínkou `if`.

3.4.1 Podmínka if

Deklarace podmínky `if` je následující:
`if` (*podmínka*)
   příkazy
`else`
   příkazy
`end`

Podmínkou může být test na rovnost či nerovnost hodnot proměnných. Matlab umožňuje definici podmínky `není rovno` pomocí zápisu ∼=, Octave umožňuje také zápis `!=`. Pro test rovnosti se používají dvě znaménka je rovno (==). Samotné znaménko *rovná se* (=) je určeno **výhradně** přiřazovacímu příkazu!!! Příkaz `if-then-else` může mít libovolný počet větví `elseif`, což je užitečné při komplikovaných podmínkách.

Doplňme nyní náš skript o testování zadávaných vstupních hodnot, což je právě jeden z případů užití podmínky `if`:

```
function [x1,x2]=KvadrRce1(a,b,c)
% funkce resi kvadratickou rovnici
% [x1,x2]=KvadrRce(a,b,c),
% kde a,b,c jsou parametry kv.rce
%
D=b^2-4*a*c; % diskriminant
% reseni
if (a~=0) % a je ruzne od nuly
  x1=(-b+sqrt(D))/2/a;
  x2=(-b-sqrt(D))/2/a;
else
  disp(‘‘Rovnice neni kvadraticka!!’’);
  x1=c/b;x2=0.0;
end
```