MQL4 COURSE

By Coders' guru
www.forex-tsd.com

-5
Smyčky & Rozhodnutí
Part 1

Vítejte v páté lekci mého kurzu MQL4.

Předchozí lekci si můžete stáhnout z tohoto odkazu:
http://forex-tsd.com /attachment.php?attachmentid=399
http://forex-tsd.com /attachment.php?attachmentid=372
http://forex-tsd.com /attachment.php?attachmentid=469
http://forex-tsd.com /attachment.php?attachmentid=481
Nezapomeňte se nejprve zalogovat.

Běžná kontrola toku programu, který jste zapsali v jazyce MQL4 (stejně jako v jiných jazycích), se provádí seshora dolů, příkaz po příkazu.

Příkaz je kódovaný řádek, který počítači něco sděluje. Např.:

Print("Hello World"); return 0;

Středník na konci sdělení je rozhodující částí syntaxe, ale velmi lehce se na něj zapomíná, z čehož vyplývá 90% chyb.

Provádění seshora dolů však není jediná možnost a vyskytují se zde dvě výjimky.

Říká se jim „smyčky" a „rozhodnutí".

Programy, které zapisujete, se rozhodují – jako člověk – jak zareagovat na změnu okolností. V těchto případech tok řízení skáče z jedné části programu do druhé. Příkazy způsobující takovýto skok se nazývají kontrolními příkazy. Jedná se o kontrolu funkcí smyček a rozhodnutí.

SMYČKY

Smyčky způsobují, že určitá sekce vašeho programu bude opakována po určitou dobu. Toto opakování pokračuje, pokud je nějaká podmínka je true a končí, pokud se změní na false. Po skončení smyčky je řízení předáno dalšímu příkazu následujícímu po sekci smyčky.

V MQL 4 existují dva druhy smyček:

Smyčka for

Smyčka for je považována za nejjednodušší smyčku, protože veškeré její ovládací elementy jsou seskupeny na jednom místě. Smyčka for provede sekci kódu v pevně stanoveném počtu opakování.

Příklad:

```
int j;
for(j=0; j<15; j++)
        Print(j);
```

Jak to funguje?

Sdělení **for** sestává z klíčového slova následovaného kulatými závorkami, které obsahují tři výrazy oddělené středníky:

for(j= 0;j< 15;j++)

Tyto tři výrazy jsou inicializační výraz, první výraz a přírůstkový výraz:

j= 0 - initializační výraz

j<15 - testovací výraz

j++ - přírůstkový výraz

Tělem smyčky je kód, který má být proveden pevným počtem opakování: Print(j);

V našem případě je smyčka opakována 15x.

Poznámka: Příkaz **for** není v tomto případě oddělen středníkem. To proto, že příkaz **for** a tělo smyčky jsou dohromady považovány za příkaz.

Inicializační výraz:

Inicializační výraz je proveden pouze jednou při prvním spuštění smyčky. Jeho účelem je poskytnout proměnné ve smyčce inicializační hodnotu (v našem případě 0).

Proměnnou smyčky můžete deklarovat zvenčí smyčky (předem), jako v tomto příkladě:

```
int j;
```

Nebo můžete provést deklaraci uvnitř smyčky prostřednictvím kulatých závorek:

```
for(int j=0; j<15; j++)
```

Předchozí dva kódové řádky se shodují, kromě **rozsahu** každé proměnné (o deklaraci proměnných a rozsahů se dozvíte více v lekci „Proměnné“).
Metoda vnějšího deklarování provádí každý řádek v kódovém bloku, aby se o proměnné vědělo, zatímco vnitřní deklarování provede toto pouze prostřednictvím příkazu **for** ve smyčce.

Ve smyčce **for** můžete použít více než jeden inicializační výraz, výrazy oddělujete čárkou (,), jako zde:

```
int i;
int j;
for(i=0 ,j=0;i<15;i++)
  Print(i);
```

Testovací výraz

Testovací výraz je vždy vztažným výrazem používajícím porovnávací operátory (ohledně porovnávacích operátorů nahlédněte do předchozí lekce).

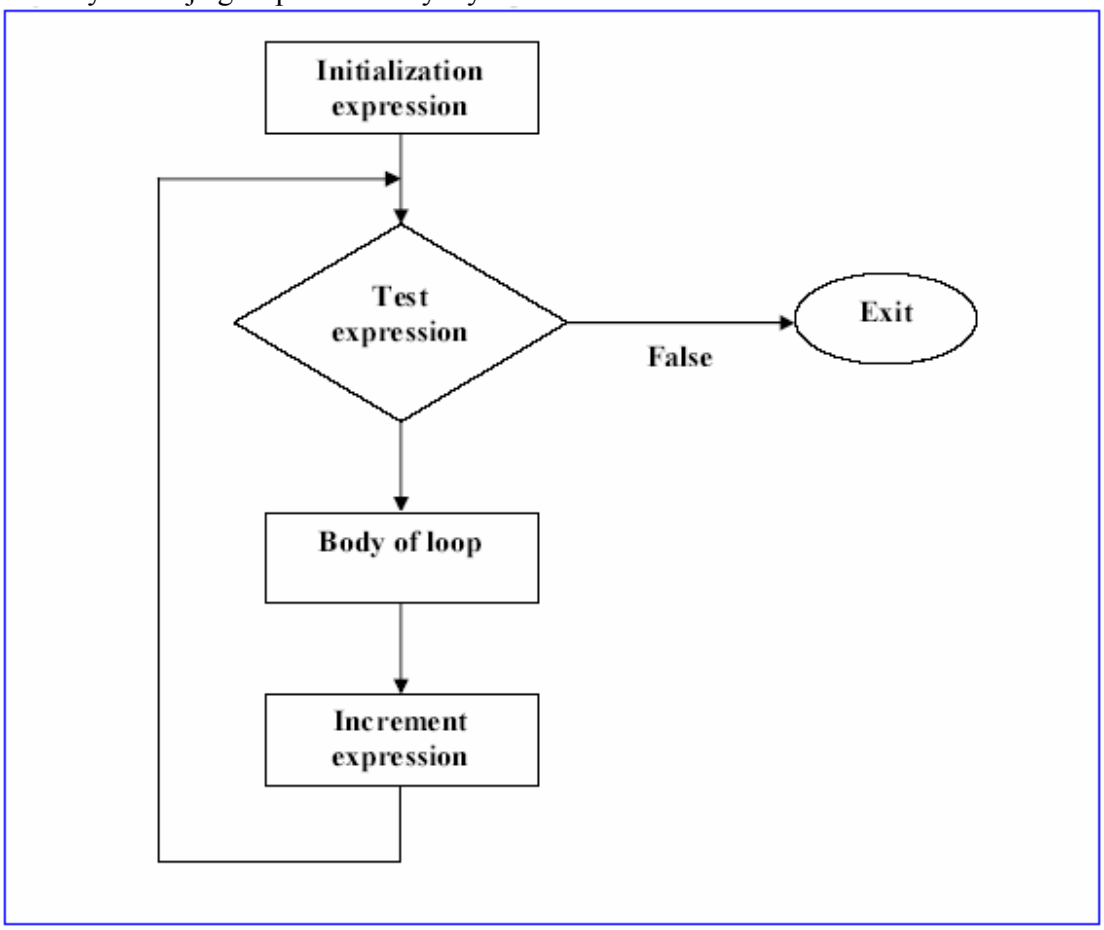
Ty vyhodnocují smyčku při každém jejím provedení, aby určily, zda má smyčka pokračovat či nikoliv. Pokračovat bude, pokud výsledná hodnota bude true, dosáhne-li hodnoty false, bude smyčka ukončena.

V našem případě **tělo** smyčky pokračuje v tisku **i(print)(i)),** zatímco v případě $j<15$ je hodnota true.
Např. j= 0,1,2,3,4,5,6,7,8,9,10,11,12,13 a 14.
Když **j** dosáhne hodnoty **15**, smyčka je zastavena a kontrola je předána dalšímu příkazu smyčky.

Přírůstkové výrazy:

Přírůstkové výrazy mění hodnotu proměnné smyčky (v našem případě **j**) přičtením hodnoty o hodnotu 1.
Proveden je jako poslední z kroků smyčky, po inicializaci proměnné smyčky, ověření testovacího výrazu a provedení těla smyčky.

Obr. 1 vyobrazuje graf průběhu smyčky **for:**



Obr. 1 – Graf průběhu smyčky

I u inicializačních výrazů můžete použít více než jeden přírůstkový výraz ve smyčce **for,** tyto opět oddělíte čárkami (,):

```
int i;
int j;
for(i=0 ,j=0;i<15,i<;i++,j++)
  Print(i);
```

Můžete však použít pouze testovací výraz.

Další poznámka o přírůstkovém výrazu: Nejen, že může zvýšit hodnotu proměnné smyčky, může také provést jinou operaci, jako v tomto případě snížení hodnoty proměnné smyčky:

```
int i;
for(i=15;i>0,i<;i--)
  Print(i);
```

Ve výše uvedeném případě bude provedena inicializace **i** na hodnotu **15,** a spuštěna smyčka. Pokaždé sníží hodnotu **i** o 1 a zkontroluje výraz **(i>0)**.
Program bude probíhat takto: 15,14,13,12,11,10,9,8,7,6,5,4,3,2,1.

Vícenásobné příkazy v těle smyčky:

V předchozích případech jsme používali pouze jednoduché příkazy v těle smyčky, ne vždy se však jedná pouze o tento případ.V těle smyčky můžete použít vícenásobné příkazy oddělené svorkami:

```
for(int i=1;i<=15;i++)
{
  Print(i);
  PlaySound("alert.wav");
}
```

Ve výše uvedeném kódu obsahuje tělo smyčky dva příkazy. Nejprve je programem proveden první, poté druhý, a to při každém provedení smyčky. Nezapomeňte na konci každého příkazu vložit středník.

Příkaz „Break“:

Pokud se ve smyčce **for** vyskytuje klíčové slovo (a to i ve smyčkách **while** a příkazu **switch)**, provádění smyčky bude přerušeno a řízení předáno následujícímu výrazu za sekcí smyčky.

Příklad:

```
for(int i=0;i<15;i++)
{
  if((i==10)
    break;
  Print(i);
}
```

Ve výše uvedeném příkladu bude smyčka prováděna až do momentu, kdy **i** dosáhne hodnoty **10,** v tom momentě klíčové slovo break přeruší smyčku. Kód tak vytvoří tyto hodnoty: 0,1,2,3,4,5,6,7,8,9.

Příkaz „continue“:

Zatímco příkaz break vás vynese ven ze smyčky, příkaz continue vás vrátí zpět na začátek smyčky (svorky).

Příklad:

```
for(int i=0;i<15; i++)
{
  if(i==10) continue;
  Print(i)
}
```

V tomto případě bude smyčka prováděna až do momentu, kdy **i** dosáhne hodnoty **10**, po dosažení této hodnoty příkaz přenese operaci na začátek smyčky bez vytištění **i** v desátém případě. Kód vytvoří hodnoty 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14.

Poslední poznámka:

Ve smyčce **for** můžete některé nebo i všechny výrazy vynechat**,** pokud si to přejete, např.:

```
for(;;)
```

Tato smyčka odpovídá výrazu **while** s testovací hodnotou trvale nastavenou na true.

Smyčku **while** vám představíme nyní.

Smyčka „while“

Smyčka **for** se obvykle používá v případě, že je vám známo, kolikrát bude smyčka vykonávána. Co se však stane, pokud nevíte, kolikrát budete chtít smyčku vykonat?
Pro tento účel je vytvořena smyčka **while.**

Obě výše uvedené smyčky mají své testovací výrazy. V této smyčce se však nevyskytuje inicializační a přírůstkový výraz.

Zde je příklad:

```
int i=0;
while(i<15)
{
  Print(i);
  i++;
}
```

V příkladě zaznamenáte toto:

- Proměnná smyčky byla deklarována a inicializována před smyčkou samotnou, nemůžete ji deklarovat nebo inicializovat uvnitř svorek jako tomu bylo u smyčky **for.**

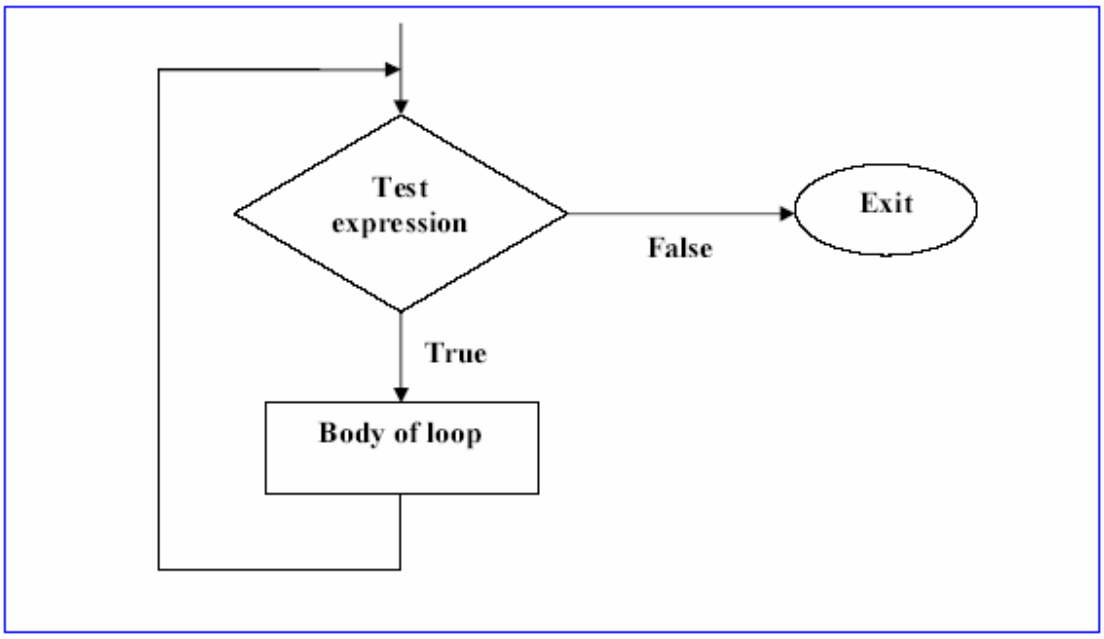
- Příkaz **i++** zde není přírůstkovým výrazem, jak by se mohlo zdát, tělo smyčky totiž musí obsahovat některá sdělení, která mění hodnotu proměnných, jinak by smyčka nemohla být nikdy ukončena.

Jak tento příklad funguje?

Příkaz **while** obsahuje pouze testovací výraz, který bude proveden u každé smyčky, pokud bude hodnota true, smyčka bude pokračovat, pokud false, zastaví se a předá řízení příkazu následujícímu po sekci smyčky.

V příkladě bude smyčka prováděna dokud **i** dosáhne hodnoty **16,** pak bude hodnota **i<15=false** a smyčka je ukončena.

Obr. 2 vyobrazuje graf smyčky **while.**



Obr. 2 – průběhový graf smyčky loop

Jak jsem vám již dříve sdělil, smyčka **while** a smyčka **for** vypadají obdobně:

1. U obou můžete použít sdělení **break** a **continue**
2. Můžete použít jenoduché nebo vícenásobné příkazy v těle smyčky u obou, v případě vícenásobných příkazů nezapomeňte na svorky.
3. Stejná funkce, jako u výrazu **for(;;)** platí i pro výraz **while (true)**

Doufám, že se vám lekce líbila.
Uvítám jakékoliv náměty a dotazy.

S pozdravem

Coders´ Guru
24-10-2005