18. Rekurze, základní schéma, základní vlastnosti, souvislost rekurze a iterace, vlastnosti implementace, efektivita. (A4B36ALG)

Základní schéma

1. inicializační algoritmus - rekurzivní funkce většinou potřebuje nějakou inicializační hodnotu, se kterou zahájí výpočet. To většinou zajišťuje jiná funkce, která není rekurzivní a která danou rekurzivní funkci volá a předává hodnotu, se kterou rekurzivní výpočet začíná
2. kontrola, zda vstupní parametr odpovídá stanoveným podmínkám. Pokud hodnota parametru odpovídá, rekurzivní funkce provede výpočet a vrátí hodnotu
3. rekurzivní funkce redefinuje řešení problému tak, že jej nějakým způsobem zmenší, zjednoduší nebo rozloží na dílčí podproblémy
4. volání funkcí, které řeší daný podproblém - tady nastává přímé nebo nepřímé volání sebe sama
5. sestavení výsledku
6. vrácení vypočtené hodnoty

Základní vlastnosti

Rekurzivní volání funkce je tedy takové volání, ke kterému dojde ve chvíli, kdy funkce samotná ještě nedokončila svou činnost.
Dělení rekurze 1:

- Lineární - v každé iteraci dojde k pouze jednomu zavolání sebe sama
- Stromová - v každé iteraci dochází k více dělení na podprogramy

Dělení rekuzre 2:

- Přímá - v dáné rekurzivní funkci dojde k volání téže funkce

- Nepřímá - rekurzivní volání, ke kterému dochází přes další funkci (např. A volá B a B volá A)

vé volání, ke kterému dojde ve chvíli, kdy funkce samotná ještě nedokončila svou činnost. Volání může probíhat přímo nebo nepřímo. Každá rekurze se nechá přepsat na nerekurzivní tvar například pomocí zásobníku

Souvislost rekurze a iterace

Rekurzí lze dosáhnout zjednodušení řešené úlohy, ale je nutné brát v úvahu následující úskalí:

- každé další volání rekurzívní funkce prohlubuje zapouzdření a tedy zabírá paměťový prostor a procesorový čas
- v případě chybně ošetřeného ukončení rekurze dochází k vučerpání veškérých volných prostředků a k havárii programu
- použití může vést ke zvýšení složitosti výpočtu

Příklad výpočtu Faktoriálu pomocí rekurze:

```
function Faktorial ( integer X )
   if X < 0 then return "Chybny argument" end if
   if X = 0 then return 1 end if
   return Faktorial(X-1) * X
end function
```

a pomocí iterace:

```
function Faktorial (integer X)
     integer nfact
     if X < 0 then return "Chybny argument" end if
     nfact = 1
     for i = 1 to X do
           nfact = nfact * i
     end for
     return nfact
end function
```

Vlastnosti implementace

aa

Efektivita

aa