17. Algoritmizácia a programovanie

Základné termíny

Problém - stav, v ktorom jestvuje rozdiel medzi tým, čo v danom momente máme a tým, čo chceme dosiahnuť.

Riešenie problému - odstraňovanie rozdielu medzi aktuálnym stavom a tým, čo chceme dosiahnuť.

Algoritmus - návod na vykonanie činnosti, ktorý nás od (meniteľných) vstupných údajov privedie v konečnom čase k výsledku.

Vlastnosti – požiadavky na algoritmus:

- **elementárnosť** – postup je zložený z jednoduchých krokov, ktoré sú pre vykonávateľa (počítač, nemysliace zariadenie, človeka) zrozumiteľné,
- **determinovanosť** – postup je zostavený tak, že v každom momente jeho vykonávania je jednoznačne určené, aká činnosť má nasledovať, alebo či sa už postup skončil,
- **rezultatívnosť** – výpočet dáva po konečnom počte krokov výsledok,
- **konečnosť** – výpočet (činnosť vykonávaná podľa algoritmu) vždy skončí po vykonaní konečného počtu krokov,
- **hromadnosť** – algoritmus je použiteľný na celú triedu prípustných vstupných údajov,
- **efektívnosť** – výpočet sa uskutočňuje v čo najkratšom čase a s využitím čo najmenšieho množstva prostriedkov (časových i pamäťových).

Spôsoby zápisu algoritmov:

1) Graficky orientované
   - vývojové diagramy,
   - štruktúrogramy,
   - obrázkové jazyky.

2) Textovo orientované
   - programovacie jazyky,
   - slovný zápis algoritmu v prirodzenom jazyku,
   - rozhodovacie tabuľky.

Vývojový diagram - postupnosť činností popisovaná prostredníctvom grafických značiek a textu v nich, pričom tok výpočtu je znázornený šípkami.

Premenná - objekt (môžeme ju považovať za nejakú pamäť alebo miesto v pamäti) slúžiaci počas behu algoritmu na odkladanie údajov.

Algoritmické štruktúry:

1) **Sekvencia** – vykonávanie príkazov v takom poradí v akom sú zapísané.
2) **Vetvenie** – možnosť rozhodnúť sa a vykonať príkazy na základe pravdivosti skúmaného znaku.
3) **Cyklus** – zápis umožňujúci opakovanie.

Druhy cyklov:

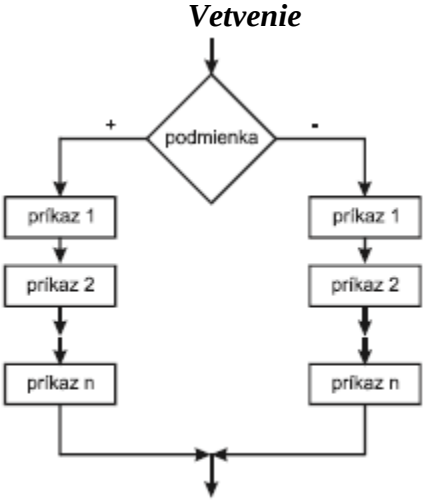
a) cyklus so známym počtom opakovaní **(for)**
b) cyklus s podmienkou na začiatku **(while – do)**
c) cyklus s podmienkou na konci **(repeat – until)**

Schémy - algoritmické štruktúry

Sekvencia

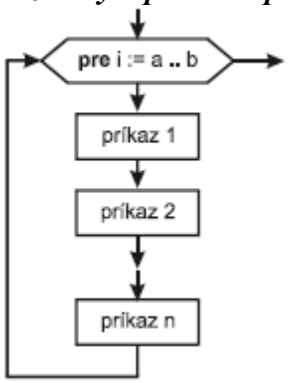
Realizujú sa jednotlivé príkazy P1 až Pn, a to v poradí, v akom sú zapísané

Vetvenie



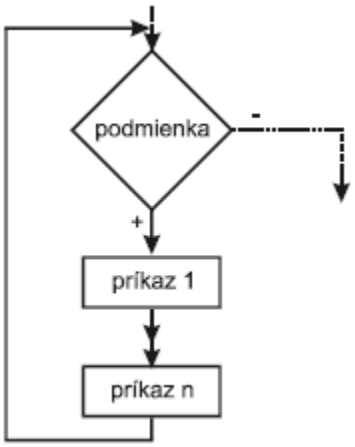
Ak je podmienka splnená, realizujú sa príkazy pod vetvou +, ak nie, pod vetvou-.

Cyklus so známym počtom opakovaní (for)



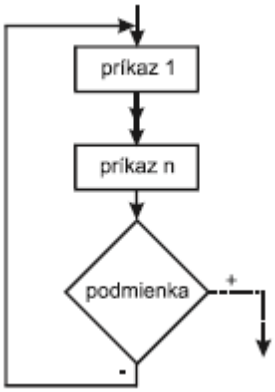
Predpokladom využitia takéhoto cyklu je, že počet opakovaní sa dá vyjadriť pred jeho odštartovaním a operácie v tele naň nemajú žiaden vplyv. Cyklus so známym počtom opakovaní používa premennú, hovorí sa jej riadiaca premenná, ktorá si „pamätá“ koľkokrát cyklus prebehol. Po skončení tela cyklu sa automaticky (ma to na starosti cyklus, nie my) pripočíta hodnota 1 a skočí sa na začiatok cyklu.

Cyklus s podmienkou na začiatku (while – do)



Tento typ cyklu má podmienku, ktorá sa stará o ukončenie cyklu, umiestnenú pred telom. Ak je podmienka splnená, vykoná sa telo cyklu a opäť sa otestuje. Ak „vstupná“ podmienka nie je splnená už pri prvom vstupe do cyklu, nemusí sa tento vykonať vôbec.

Cyklus s podmienkou na konci (repeat – until)



Tento cyklus na prvý pohľad vyzerá oproti cyklu s podmienkou na začiatku ako opačný – najprv sa vykoná telo cyklu a až potom sa zisťuje splnenie podmienky. Ak je podmienka cyklu splnená, vykonávanie cyklu sa ukončí, v opačnom prípade sa pokračuje opätovným vykonávaním tela cyklu. Dôsledkom takéhoto riadenia je, že cyklus vždy prebehne minimálne raz.

Program - algoritmus zapísaný v programovacom jazyku.

Programovacie jazyky - umelé jazyky, pomocou ktorých môžeme jednoducho a pritom jednoznačne vyjadriť algoritmus, a ktoré dokáže naše zariadenie interpretovať

Strojový kód - jazyk zrozumiteľný procesoru počítača.

Zdrojový kód - súbor obsahujúci algoritmus zapísaný v programovacom jazyku

Prekladač – prekladá zdrojový kód do strojového kódu. Prekladač môže byť samostatnou aplikáciou alebo len jej časťou. V praxi rozlišujeme dva typy prekladačov: kompilátor a interpreter.

Interpreter - prekladá zdrojový kód po každom spustení programu príkaz po príkaze (interpreter príkaz prečíta, preloží a procesor ho vykoná).

Kompilátor - pracuje tak, že zo zdrojového kódu vytvorí aplikáciu – spustiteľný súbor obsahujúci príkazy, ktoré dokáže procesor po jeho spustení okamžite vykonávať.

Programovanie - proces tvorby programu

Fázy programovania:

1) algoritmizácia,
2) prepísanie algoritmu do inštrukcií programovacieho jazyka,
3) ladenie a testovanie programu

Objektovo orientované programovacie jazyky - jazyky vyššej úrovne, vychádzajú z faktu, že objektový pohľad na programovanie je veľmi podobný pohľadu človeka na svet