Algoritmizácia a programovanie

Základy algoritmizácie

Počítač je prostriedkom pre automatizované riešenie úloh založených na informačných procesoch. Proces vykonávania operácií sa nazýva výpočtový proces. Riešenie úlohy je často formulované tak, že postupnosť operácií nie je celkom jasná. Preto sa robí algoritmizácia úlohy = zostavenie algoritmu.
Algoritmus je návod na prácu pre počítač = postupnost krokov, ktoré vedú k riešeniu danej úlohy.
Algoritmizácia úlohy prebieha v troch etapách:

- formulácia- slovné zadanie úlohy
- analýza- úloha sa zovšeobecňuje, určujú sa podmienky postupu
- zostavenie algoritmu- presné vyjadrenie logiky a postupu riešenia

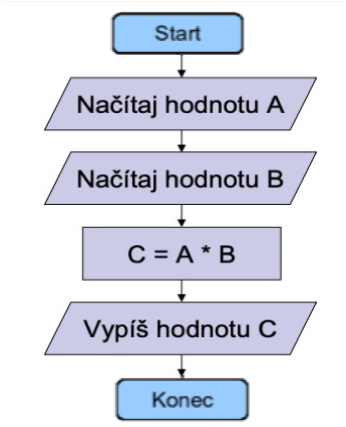
Pojem a vlastnosti algoritmu :

Algoritmus je presný a jednoznačný systém pravidiel určujúci postup riešenia úlohy, a ktorý nás od vstupných údajov privedie v konečnom čase k výsledku. Každý algoritmus musí mať nasledovné vlastnosti:

- elementárnosť- postup je zložený z krokov, ktoré sú pre realizátora elementárne a zrozumiteľné
- determinovanosť- pre každý krok algoritmu je jednoznačne určený nasledujúci krok
- hromadnosť- algoritmus by mal byť použiteľný pre riešenie každej úlohy daného typu so všetkými možnými vstupnými údajmi
- rezultatívnosť- algoritmus dáva pre rovnaké vstupné údaje vždy rovnaké výsledky
- konečnosť- postup skončí vždy v určitom čase a po vykonaní konečného počtu činností
- efektívnosť- postup sa uskutočňuje v čo najkratšom čase a s využitím čo najmenšieho počtu prostriedkov

Zápis algoritmov

- slovný zápis- pre vyjadrenie algor. je nám blízky, dobre sa nám v ňom uvažuje, ale ako prostriedok analýzy zložit. úloh je nevýhodný pretože je neprehľadný, nedostatočne zvýrazňuje zmeny postupu, môžu sa v ňom vyskytnúť nepresnosti
- pomocou vývojových diagramov- vývojový diagram úlohy je bloková grafická reprezentácia postupnosti operácií, kt. má realizovať úlohu v súlade s príslušným algoritmom.
- Pomocou programovacieho jazyka

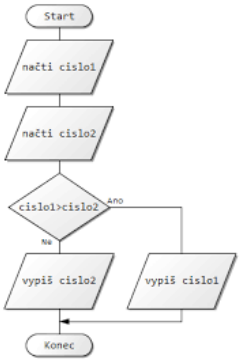


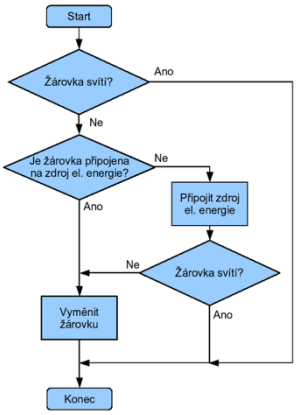
Vytvoriť algoritmus na výpočet súčinu dvoch čísel vývojovým diagramom

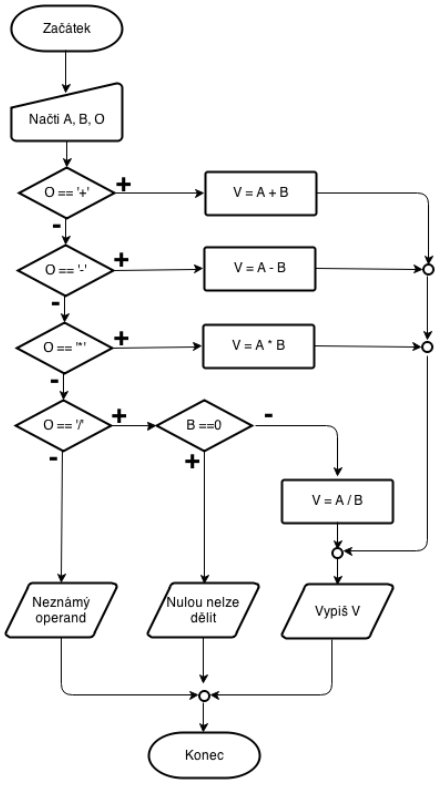
- vstupné podmienky: existencia dvoch čísel (činiteľov)
- postup – vytvoríme
- výsledkom bude súčin vstupných hodnôt

Základné činnosti algor. jazyka považujeme:

- príkaz vstupu- umožňuje zadať konkrétne hodnoty údajov, kt. má spracovať. Tieto hodnoty sa uložia do premenných
- príkaz výstupu- umožňuje získať výsledky algoritmu alebo iné oznamy.
- priraďovací príkaz - zmena hodnôt premenných je počas vykonávania algor. možná dvomi spôsobmi: príkazom vstupu aleba priradením novej hodnoty
- podmienka- je v programovacích jazykoch chápaná ako logický výraz







Základné Algoritmické konštrukcie

Z hľadiska postupu vykonania ich delíme na:

- **sekvencie-** postupnosť príkazov: vyplní sa v poradí, v akom sú príkazy pod sebou zapísané
- **vetvenie-** v závislosti od splnenia podmienky sa postup vetví na 2 cesty (alebo postupne aj viac) a robí sa to zistením, či je nejaká podmienka splnená, alebo nie, vetvenie je jednoduché = jedna podmienka a 2 cesty, alebo je podmienok za sebou viac = zložené vetvenie
- **cyklus-** pri opakovaní činnosti musíme vedieť, čo sa má opakovať a dokedy sa to má opakovať
    - činnosť, kt. sa má opakovať nazývame telom cyklu
    - podmienkou, kt. určuje dokedy sa bude opakovať, nazývame podmienka cyklu
    - cyklus so známym počtom opakovaní- telo cyklu sa opakuje vopred známy počet krát
    - cyklus s podmienkou na začiatku- najčastejšie sa volí cyklus, kedy sa telo cyklu opakuje, pokiaľ platí podmienka cyklu
    - cyklus s podmienkou na konci- je opačný cyklus s podmienkou na začiatku- najskôr sa vykoná telo cyklu. Potom sa zisťuje splnenie podmienky cyklu.

Programovanie

Programovanie je proces vytvárania algoritmu a jeho zápisu v tvare programu. Program je konkrétna formulácia algoritmu založená na konkrétnej reprezentácií a štruktúre údajov.

Vlastnosti programu:

- správnosť a spoľahlivosť,
- pružnosť a modifikovanosť,
- prispôsobivosť,
- odolnosť voči vonkajším chybám,
- zdokumentovateľnosť,
- zrozumiteľnosť,

Programovacie jazyky

- Strojový kód = jazyk zrozumiteľný procesoru počítača (1 a 0).
- Pre programátora je však veľmi zložité komunikovať s počítačom v tomto jazyku. Preto boli vyvinuté programovacie jazyky, ktoré sú pre človeka zrozumiteľnejšie a prekladače, ktoré programy zapísané v týchto jazykoch prekladajú do strojového kódu.
- Zdrojový kód = súbor zapísaný v programovacom jazyku, ktorý môže byť do strojového kódu preložený.
- Prekladač = preloží program zo zdrojového do strojového kódu. Pre prekladač je zdrojovým (vstupným) súborom, z ktorého sa vygeneruje cieľový (výstupný) súbor v strojovom kóde – spustiteľná aplikácia.
- Kompilátor je počítačový program, ktorý číta program vo vysokej úrovni jazyka, ktorý sa nazýva zdrojový jazyk, a prekladá ho do iného jazyka na nízkej úrovni.

Rozdelenie jazykov

Podľa úrovne:

1. nižšie programovacie jazyky = neponúkajú programátorovi veľa funkcií a sú si veľmi blízke so strojovým kódom, aj keď sa zapisujú o niečo čitateľnejšie.
   - iba pár základných príkazov len pre daný procesor
   - sú najrýchlejšie,
   - patrí sem jazyk symbolických adries Assembler a strojový kód (vidíme ak si .exe súbor pozrieme v text editore)
   - ponúkajú ale aj funkcie , ktoré vyššie nemajú
   - používané sú iba v situáciách, kedy je potrebný najvyšší možný výkon, ako napríklad pri ovládačoch grafických kariet. Ich preklad do strojového kódu sa v angličtine nazýva „assemble“, a preto sa nazývajú **Assemblery**.
2. vyššie programovacie jazyky (väčšina jazykov) = zrozumiteľnejšie človeku
   - je potrebné preložiť omnoho zložitejšími procesmi
   - nie sú závislé na type procesora
   - obsahujú syntax podobnú ANJ
   - v podstate všetky súčasné

Podľa spôsobu prekladu a spustenia:

a) kompilované programovacie jazyky (napr. Pascal, C) = pred spustením sú najprv kompletne preložené kompilátorom
   - nie po príkaze ale celý naraz a vytvorí spustiteľný .exe súbor
   - výsledkom je väčšia rýchlosť, ale tiež väčšiu náročnosť na správne zapísaný kód
   - Kompilované do binárnej formy = pred spustením najskôr preložiť za pomoci kompilátoru.
     - sú rýchlejšie, ako interpretované, keďže sa preklad vykoná hneď po napísaní programu
     - nevýhodou je neprenosnosť na iné platformy = treba pre každú platformu zvlášť.
   - Kompilované do bajtového kódu = preklad je prenesiteľný
     - použitie ako výkonné webové aplikácie a iné situácie,
     - ak programátor nevie, na ktorej platforme bude program spúšťaný

b) interpretované programovacie jazyky (napr. BASIC, Perl, Python, shell)
   - zdrojový kód sa v nich do strojového prekladá až pri behu programu, čo v niektorých prípadoch značne znižuje ich výkon
   - preklad je po riadkoch = príkazoch
   - potrebujú mať na počítači nainštalovaný ich interpreter, aby bolo možné spúšťať programy v nich napísané.
   - výhodou je, že jeden program je spustiteľný na rôznych platformách bez akýchkoľvek komplikácii. Najznámejšie sú Javascript, PHP a Python.

Vyššie programovacie jazyky sa ďalej delia podľa prístupu k údajom a funkciám

- štruktúrované - jeho hlavnou ideou je snaha o maximálnu zrozumiteľnosť a prehľadnosť programu.
- objektovo orientované (OOP) - pracujú s najrôznejšími objektami, ktoré obsahujú premenné a funkcie, tie majú svoje vlastnosti a máme ich potom vždy k dispozícii.

Základné príkazy prog.jazyka

príkazy sú vety jazyka, ktoré prikazujú procesoru vykonať stanovené činnosti. Základné príkazy:

- *vstupu*
- *výstupu*
- *priradenia*

1. *premenné* - sú objekty, ktoré obsahujú počas realizácie algoritmu konkrétnu hodnotu presne stanoveného typu *číslo, text, pole, 1/0
2. *konštanty* - sú objekty, kt. nadobúdajú počas celej realizácie algor. jedinú konkrétnu hodnotu príslušného typu
3. *výrazy* - sú predpisy, kt. obsahujú konštanty, premenné a spôsob ich spracovania pomocou operácií a funkcií.

Základné etapy riešenia úlohy na počítači

Tvorba programu to nie je iba jeho zápis v danom programovacom jazyku. Etapy tvorby programu tvoria **životný cyklus programu**:

1. etapa: špecifikácia úlohy
2. etapa: návrh algoritmu
3. etapa: implementácia programu
4. etapa: realizácia programu
5. etapa: spracovanie dokumentácie

1**. Špecifikácia úlohy =** pomocou špecifikácie požiadaviek na program určujeme, **čo** má program robiť; nemá robiť o nič viac, ale ani o nič menej ako sa od neho požaduje - má mať tieto **vlastnosti**:

a) *správnosť* - ak nebude správna špecifikácia úlohy, nebudeme riešiť úlohu, ktorú máme riešiť
b) *úplnosť* - program má robiť iba to, čo sa od neho požaduje
c) *jednoznačnosť* - aby nedošlo k nesprávnemu pochopeniu zadania úlohy,
d) *testovateľnosť*

2. **Návrh algoritmu =** jej úlohou je navrhnúť postup riešenia úlohy a vytvoriť logickú štruktúru programu - poznáme dva **základné spôsoby návrhu programu**:

1. ***zhora nadol*** - označovaný ako TOP-DOWN
    - postup od "všeobecného ku konkrétnemu"
    - postupné zjemňovanie až po úroveň konštrukcií, ktoré možno vyjadriť v programovacom jazyku
2. ***zdola nahor*** - označovaný ako DOWN-TOP = opačne

3. **Implementácia programu =** začína návrhom programu a končí zápisom programu v danom programovacom jazyku, zvyčajne sa nerealizuje v jednom kroku, väčšinou je to proces, ktorý sa realizuje v

4. **Realizácia programu** - cieľom je získať výstupné hodnoty

Preklad

Napísaním programu v danom programovacom jazyku práca na riešení úlohy nekončí. Programom predpisujeme počítaču, **čo** má robiť. Aby však mohol pracovať podľa programu, musí mu rozumieť. Program tak musí byť napísaný vo vhodnom jazyku. Takým jazykom, ktorému počítač rozumie, je ***strojový jazyk*** počítača = jazyk **nižšej úrovne**.

Ak je program napísaný v inom jazyku, potrebujeme, aby ho "niekto" *preložil* z daného programovacieho jazyka do strojového jazyka. PYTHON je programovací jazyk **vyššej úrovne**. Umožňuje zapisovať programy v tvare, ktorý je bližší užívateľovi ako počítaču. Počítače však nerozumejú vyšším programovacím jazykom ***priamo*****,** a preto programy musia byť najskôr prevedené - *preložené* do jazyka počítača. Tento preklad programu realizuje **prekladač** - **kompilátor**.

Prekladač preloží zdrojový program napísaný vo vyššom programovacom jazyku do strojového jazyka počítača.

Ladenie

Ďalej nastáva etapa *ladenia programu* ("odvšivovanie"), t. j. odstraňovanie chýb v navrhnutom programe. Nikto z nás totiž nie je schopný pracovať bez chýb. Niektoré z nich odhalí prekladač, na niektorých nám "spadne" program. Najhoršie sú však tie, pri ktorých program síce funguje, ale robí niečo celkom iné, ako sme očakávali. Zistiť v programe chybu nie je vždy jednoduché.

Chyby v programe môžu byť:

1. **syntaktické** - zistí ich sám prekladač (kompilátor) lebo prekladač môže preložiť len syntakticky správny program napr. ele namiesto else alebo A > A+1 namiesto A = A + 1
2. **sémantické -** napr. či premenné na pravej strane priraďovacieho príkazu majú priradené hodnoty alebo či premenná je iba jedného typu
3. **run-time errors -** chyby v priebehu programu **-** napr. snaha deliť nulou
4. **logické** - program môže byť vykonaný ("zbehne"), môže vytlačiť výsledky, ale tieto nezodpovedajú špecifikácii úlohy (vstupnej a výstupnej podmienke)

Pri ladení potrebujeme poznať hodnoty premenných v určitých bodoch programu. Preto je ladenie spojené ***s testovaním*** - t. j. realizáciou programu pre konkrétne vstupné údaje. Ladiace programy umožňujú:

- **sledovať beh programu po jednotlivých príkazoch**
- **zobraziť v každom okamihu hodnoty jednotlivých premenných - ďalšie užitočné funkcie**

5**. Spracovanie dokumentácie**

Dokumentácia programu je materiál, ktorý obsahuje informácie o tvorbe, používaní a údržbe programu. Jej tvorba je neoddeliteľnou súčasťou tvorby vlastného programu.

Spracovanie dokumentácie programu je najmenej obľúbenou činnosťou pri programovaní. Program je však hotový až vtedy, keď je s odladeným programom dodaná i užívateľská a programátorská (alebo inými slovami riešiteľská či technická) dokumentácia.

1. **užívateľská dokumentácia** - odpovedá na otázky, čo program robí a ako sa s ním zaobchádza; má dve formy:

a) ***manuál*** – má slúžiť na pohotové vyhľadanie informácie pri rutinnej praxi s programom
   - musí byť zostavený prehľadne, aby a v ňom dobre hľadalo

- obsahuje terminologický slovník a obsah, krížové odkazy

b) ***učebnica*** - má za úlohu naučiť užívateľa pracovať s programom, preto musí byť napísaná príťažlivo, aby sa dala ľahko čítať - je to vlastne návod na použitie programu:
   - ako ho spustiť a ukončiť
   - ako ho ovládať
   - ako interpretovať výsledky

2. **programátorská dokumentácia** - je podkladom pre opravy alebo úpravy programu autorom alebo inými osobami, ktoré majú k dispozícii zdrojový súbor programu; zodpovedá na otázky **ako** je program urobený a **prečo** je tak urobený .

Udržiavanie a aktualizácia programu

Bežné programy z praxe je potrebné udržovať a aktualizovať. Napríklad program na účtovníctvo je potrebné prepracovať pri zmene nariadenia o spôsobe účtovania (zmenilo sa vlastne zadanie úlohy). Alebo sa zmenilo používané technické vybavenie, či bola odhalená dlho skrytá logická chyba, ktorá sa prejavila u užívateľa. Uvádza sa, že údržba programového systému vyžaduje dvojnásobok práce ako celý jeho vývoj.