ALGORITMY

Přesný postup, který je potřeba k vykonání určité činnosti

Co je to algoritmus

- předpis, jak řešit určitý problém (přesně určená konečná posloupnost pravidel, jejichž realizace nám umožní pro přípustné hodnoty vstupních dat nalézt po konečném počtu kroků správná výstupní data).
- Navrhování a vymýšlení algoritmů je základem programování

Příklady algoritmů

nenumerické

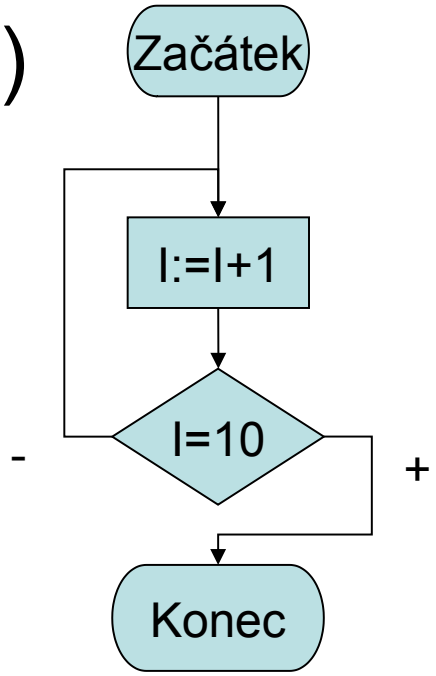
- praní prádla,
- telefonování,
- zavěšení obrazu na stěnu
- Recept na Sachrův dort
- použití hasicího přístroje,
- údržbu bot,
- vyhledání knihy v knihovně.

numerické

- sečtení nebo násobení dvou čísel
- Postup při výpočtu lineární, nebo kvadratické rovnice.

Vlastnosti algoritmu

1. **Musí mít začátek a konec** (po vykonání konečného počtu kroků musí dojít od začátku do konce)



Vlastnosti algoritmu

2. Musí být věcně správný

- pozor na správný zápis vzorce – závorky
- Výrazy např. s odmocninami – funkce

3. Musí být jednoznačný

- Nejčastější chyba
- Je nutné zvážit všechny možnosti, které mohou v dané situaci nastat
- Nejčastější: - výrazy se zlomky – ošetřit 0 ve jmenovateli
  - odmocniny – výraz pod odmocninou >=0
  - různé další funkce - definiční obory

Vlastnosti algoritmu

4. **Obecnost**

   algoritmus co nejobecnější

5. **Opakovatelnost**

   možnost algoritmus kdykoli zopakovat tak, aby se při stejných podmínkách choval stejně – při stejných vstupních hodnotách musí dávat stejný výsledek (pozor, aby se se stejnou proměnnou nepracovalo v jiné části programu, která by ji na vstupu mohla změnit)

Vlastnosti algoritmu

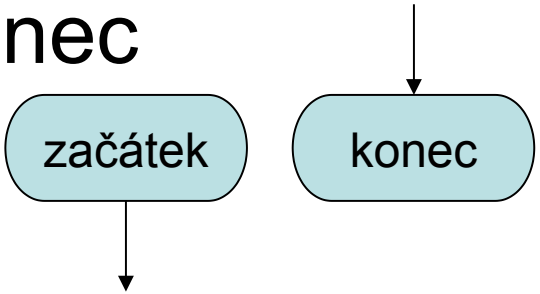
6. **Srozumitelnost**

metody zápisu algoritmu:

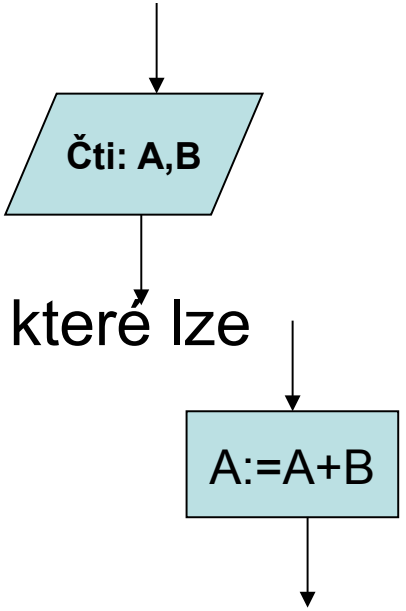
- Slovní vyjádření
- Matematický zápis
- Rozhodovací tabulky
- Vývojové diagramy
- Počítačové programy

Vývojové diagramy - značky

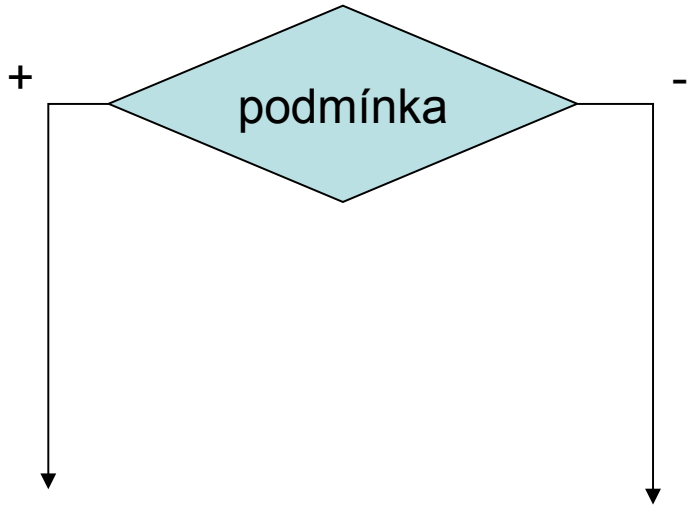
- **Mezní značky** – začátek, konec



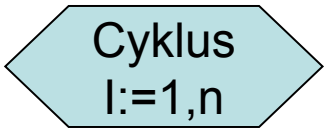
- **Sekvenční bloky** –
  - **vstup** (načtení dat)
  - **výstup** (zobrazení výstupních dat)
  - **zpracování** (jedna nebo více instrukcí, které lze vykonat najednou)



- **Větvení** – na základě nějaké podmínky
  - Je-li podmínka splněna: + větev
  - Není-li podmínka splněna: - větev

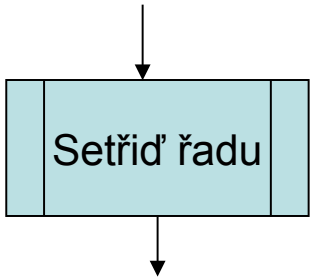


- **Příprava** – přípravná fáze programu např. pro zahájení cyklu o známém počtu opakování; konec cyklu



- **Spojka** – spojení dvou částí VD, které nabylo možno nakreslit souvisle, označíme stejnými čísly

- **Podprogram** – samostatná část programu, která obsahuje větší množství kroků (opakuje-li se stejná část programu na více místech)



Počítačový program

- Je to algoritmus přepsaný v jazyce, kterému počítač díky vývojovému prostředí rozumí a umí z něho díky překladači vytvořit = vygenerovat strojový kód
  - **Zdrojový kód** – vlastní zápis programu v programovacím jazyce
  - **Strojový kód** – sled instrukcí pro procesor

Programovací jazyk

- slouží pro zápis algoritmu pro počítač
- vyhovuje vždy určitým pravopisným (syntaktickým) pravidlům s předem domluveným významem (sémantikou)
- Rozdělení:
  - Počítačově orientované (strojově závislé)
    - Strojový kód
    - Jazyk symbolických adres
    - autokódy
  - Makroprogramovací (strojově nezávislé)
    - Vyšší programovací jazyky

Počítačově orientované – Strojový kód

- vyjádřen číselně s přímými odkazy na příslušné adresy v paměti počítače
- nejnižším programovacím jazykem, jehož syntaxe a sémantika je přímo přizpůsobena danému typu počítače. Program napsaný v tomto jazyce je pak možné použít jenom pro tento typ a je nepřenositelný na jiné typy počítačů.
- **Instrukce**: jsou zapsány čísly (zapsanými ve dvojkové či šestnáctkové soustavě) - pro člověka velmi nesrozumitelný.
    - operační znak (kód – jaká činnost se má vykonat)
    - operand – adresa -vstupní hodnota, se kterou bude operace vykonávána

Př. Pro vytvoření součtu (čemuž odpovídá operační znak 18) dvou hodnot uložených na adresách označených pořadovými čísly 0117 a 0128 a uložení výsledku do buňky s adresou 0252 může být zápis této instrukce např. následující:

Operační znak	***Adresa 1.operandu***	***Adresa 2.operandu***	***Adresa 3.operandu***
18	*0252*	*0117*	*0128*
10 010	*0000 01111 1100*	*000 0 0111 0101*	*0000 1000 0000*

Počítačově orientované – **jazyk symbolických adres assembly language**

- Instrukce a odkazy nahrazuje definovanými symboly a symbolickými adresami ve formě zkratek
- Převod do strojového kódu – překladač **(assembler)**
- Jeden příkaz assembleru = jedna instrukce strojového kódu

Příklad *- fragment programu v assembleru pro PC*

```
ExcessOfMemory label near
  mov bx, di
  add bx, dx
  mov word ptr _heapbase@ + 2, bx
  mov word ptr _brklvl@ + 2, bx
  mov ax, _psp@
  sub bx, ax ; BX = Number of paragraphs to keep
  mov es, ax ; ES = Program Segment Prefix address
  mov ah, 04Ah
  push di ; preserve DI
  int 021h ; this call clobbers SI,DI,BP !!!!!!
  pop di ; restore DI
```

Autokódy

- Byly to jazyky na půl cesty mezi jazyky nižší a vyšší úrovně. Byly sice vázány na určitý typ počítače, ale zbavily se již pravidla, že jeden příkaz jazyka = jedna strojová instrukce
- syntaxe a sémantika přizpůsobena zápisu matematických informací

  1příkaz = 3-4instrukce stroj.kódu

Příklad

M = z+0.5	*součet hodnoty proměnné z a konstanty 0,5 a uložení výsledku do pomocné buňky M*
*N = x*x*	*umocnění hodnoty proměnné x a uložení výsledku do pomocné proměnné N*
*N = N*N*	*umocnění hodnoty proměnné N a zpětné uložení výsledku (x4) do téže proměnné N*

Makroprogramovací jazyky - Vyšší programovací jazyky

- Propracovaná syntaxe - zkratky a pravidla, kterým rozumí překladač a umí je převést do strojového kódu
- Zahrnují v sobě i programové konstrukce – struktury a objekty
- Jeden příkaz tohoto jazyka odpovídá 4 až 10 a někdy i více strojovým instrukcím
- **Vývoj programovacích jazyků**
  - **FORTRAN, ALGOL** – pro matematické výpočty a vztahy
  - **COBOL, RPG** – ekonomické účely a hromadné zpracování dat
  - **D-BASE, FOX PRO** –databázové jazyky
  - **BASIC –** pro výukové účely (Visual Basic – MS Office - makroinstrukce)
- **Současnost - nejrozšířenější:**
  - **PASCAL** – strukturované programování pro výuku
  - **a jeho nadstavba DELPHI –** objekt.orient.programování
  - **C a C++ -** v komerční oblasti pro systémový a aplikační software
  - **JavaScript** – síťově orientovaný jazyk pro web
  - **PHP** – skriptovací jazyk pro web

Strukturované programování

- ***3 základní typy jazykových konstrukcí***
  - **Sekvence**
  - **Podmínky**
  - **Cykly**
- Program řešíme shora dolů, rozpracováváme detaily
- Používá jednoduché struktury - srozumitelné

Objektově orientované programování

Objekty – komponenty (předem vytvořené v prostředí)

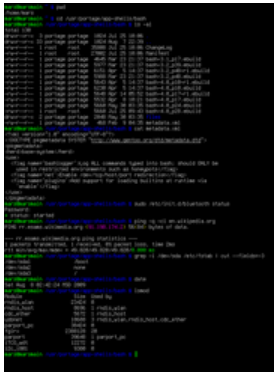
- **Události** – tvoříme funkční konstrukce pro jednotlivé objekty
- **Vlastnosti** – nastavujeme hodnoty (pro různé objekty jsou různé vlastnosti)

3 základní principy objektového programování

- **zapouzdření** vzájemné svázání datových položek a metod
- **dědičnost** možnost vytvářet objekty vycházející z jiných objektů - hierarchie
  obecný objekt – zpřesněním se vytváří jeho následníci
- **polymorfismus**

Rozdělení dle vývojového prostředí

- Konzolové aplikace – komunikují s uživatelem prostřednictvím textového rozhraní – příkazový řádek pro psaní zdrojového kódu (Pascal)



- Programy s grafickým uživatelským rozhraním – uživatelsky přitažlivější (Delphi)