Algoritmizace

- Algoritmus je přesný návod či postup, kterým lze vyřešit daný typ úlohy
- Zapisuje se tak teoretický princip řešení problému

Vlastnosti algoritmů

- **Elementárnost**
  - Algoritmus se skládá z konečného počtu jednoduchých kroků
- **Konečnost (finitnost)**
  - Každý algoritmus musí skončit v konečném počtu kroků. Tento počet může být libovolně velký, ale pro každý jednotlivý postup konečný
  - Postupy, po tuto podmínku nesplňují, se mohou nazývat výpočetní metody.
- **Univerzálnost**
  - Algoritmus neřeší jeden konkrétní problém, ale obecnou třídu obdobných problémů
  - př. neřeší 3*7 ale jak spočítat součin dvou celých čísel
- **Determinovanost**
  - Algoritmus je determinovaný, pokud za stejných podmínek poskytuje stejný výstup
  - tato vlastnost je požadována u velké části úloh
  - existují ale také algoritmy, kde je vyžadována náhodnost (míchání karet)
  - Je obtížné dosáhnout náhodnost na počítači
- **Determinismus**
  - Každý krok algoritmu musí být jednoznačně a přesně definován
  - Přirozené jazyky neposkytují takovou přesnost definice a proto jsou lepší programovací, kde každý příkaz má jeden jasný význam
  - Některé algoritmy jsou determinované ale ne deterministické (řadící algoritmus quick sort s náhodnou volbou pivota)
- **Výstup**
  - Algoritmus má alespoň jeden výstup, která je v požadované vztahu k zadaným vstupům, a tím tvoří odpověď na problém, který řeší

Druhy algoritmů

- **Rekurzivní algoritmy**
  - algoritmy, které volají samy sebe
  - Fibonacci
- **Pravděpodobnostní algoritmy**
  - provádějí některá rozhodnutí pseudonáhodně
- **Genetické algoritmy**
  - pracují na základě napodobování biologických evolučních procesů
  - mutace a křížení
- **Heuristický algoritmus**
  - neklade si nalézt přesné řešení, ale pouze vhodné přiblížení
  - používá se kde zdroje nepostačují k využití exaktních algoritmů
- další...

Paradigmata návrhů algoritmů

- **Rozděl a panuj**
  - Algoritmy typu rozděl a panuj dělí problém na menší podproblémy, na něž se rekurzivně aplikují, po čemž se dílčí řešení vhodným způsobem sloučí
  - např. binární vyhledávání, quick sort
- **Hladový algoritmus**
  - Přímočarý postup k řešení určité třídy optimalizačních úloh
  - Zpracovává se množina V složená z n údajů. Úkolem je najít podmnožinu W množiny V, která vyhovuje určitým podmínkám a přitom optimalizuje předepsanou účelovou funkci
  - Jakákoliv množina vyhovující pomínkám se nazývá **Přípustné řešení**
  - Algoritmus bude procházet prvky a rozhodovat, zda vyhovuje nebo ne
  - např. cesta grafu
- Dynamické programování
  - používá se v případech kdy lze optimální řešení složit z řešení jednodušších
  - Protože se požadavky na řešení jednodušších podproblémů můžou mnohokrát opakovat, je nutné zvolit správné pořadí a výsledky si pamatovat pro opakované použití
  - Opírá se o princip optimality
    - Optimální posloupnost rozhodnutí má tu vlastnost, že ať je počáteční stav a rozhodnutí jakékoliv, musí být všechna následující rozhodnutí optimální vzhledem k výsledkům rozhodnutí prvního.
  - např. grafové úlohy
- Broadforce
  - metoda hrubé síly

Známé algoritmy

- Eratosthenovo síto
  - nalezení prvočísel
- Euklidův algoritmus
  - Největší společný dělitel
- Dijkstrův algoritmus
  - hledání nejkratší cesty

Časová složitost

- časová složitost nebo také Asymptotická složitost je nástroj pro porovnání efektivity algoritmů
- Zapisuje se pomocí Landauovy notace nebo známá jako big O notation
  - O(f(N))
- Abychom zvolili nejlepší algoritmus, musí být časová složitost co nejlepší
- časové složitosti
  - O(N) - Linearní
    - S větším počtem vstupů se čas linearně zvětšuje
  - O(1)
    - Čas se s roustoucímy vstupy neprodlužuje
  - O(N na druhou) - exponencialně

    - O(log N)

Paměťová složitost

- Jedná se o množství paměti, kterou potřebujeme při provádění výpočtu na vstupem může to být např.
    - maximální počet bitů nutných pro uložení všech dat v každé jednotlivé konfiguraci
    - maximální počet paměťových buněk použitých během výpočtu
    - Paměťová složitost mnohdy je mnohem menší než časová
        - insertion sort má paměťovou složitost O(n) ale časovou má kvadratickou
    - Orientační typické hodnoty pro běžný pc
        - O(N)
            - až 100 000 000
        - O(n log n)
            - až 1 000 000
        - O(N na druhou)
            - až 10 000
        - O(N na třetí)
            - až 1000
        - 2 na O(n)
            - 20 - 30