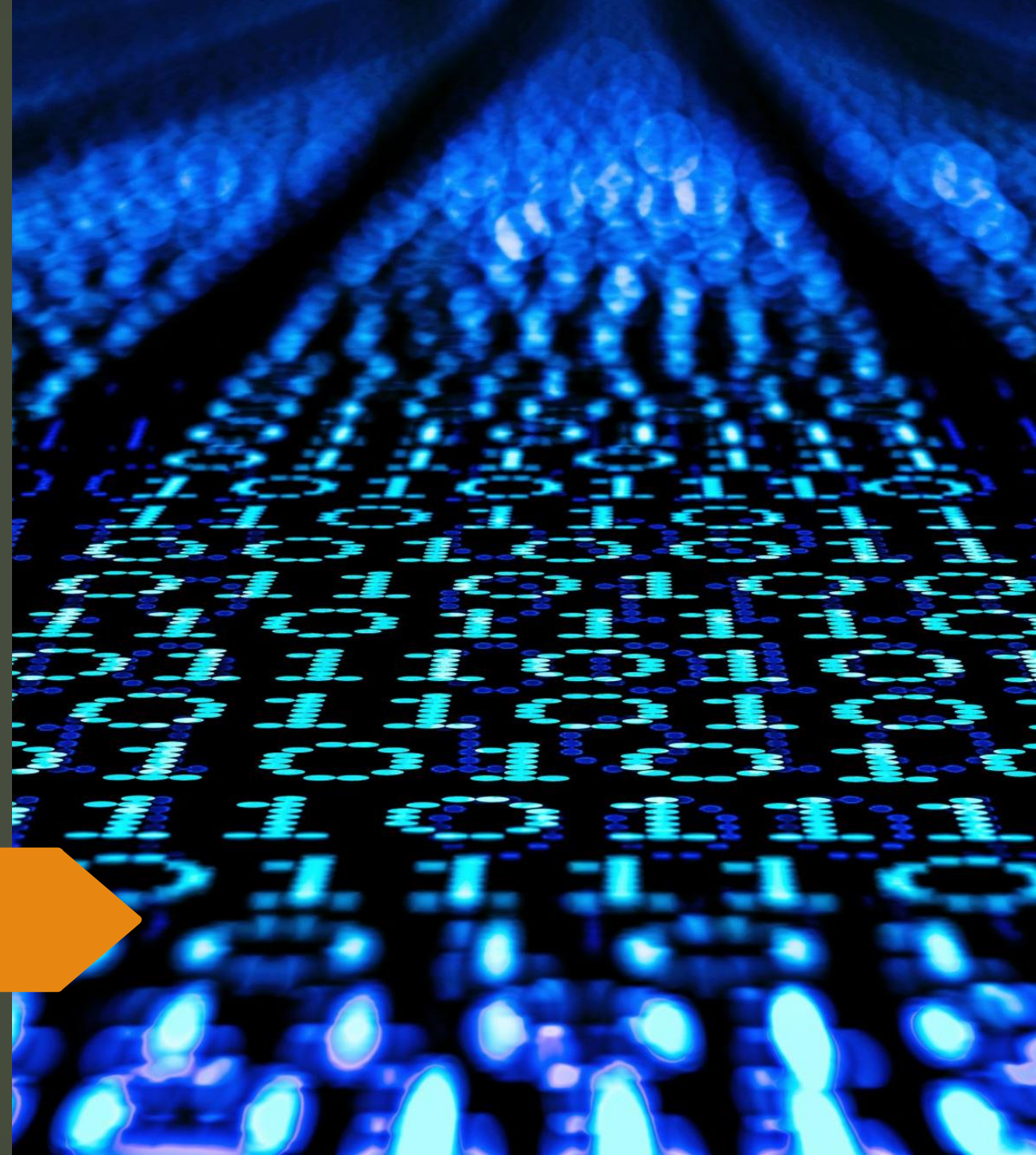


Programování

Složitost algoritmů





Složitost algoritmů

- Každý navržený algoritmus má svou složitost
- Není tím myšleno, jak komplikovaný návrh je, ale jak je náročný pro jeho vyřešení procesorem
- **Časová složitost**
 - Doba výpočtu řešení pro konkrétní množství zpracovaných dat
- **Paměťová složitost**
 - Velikost paměti potřebná k řešení algoritmu

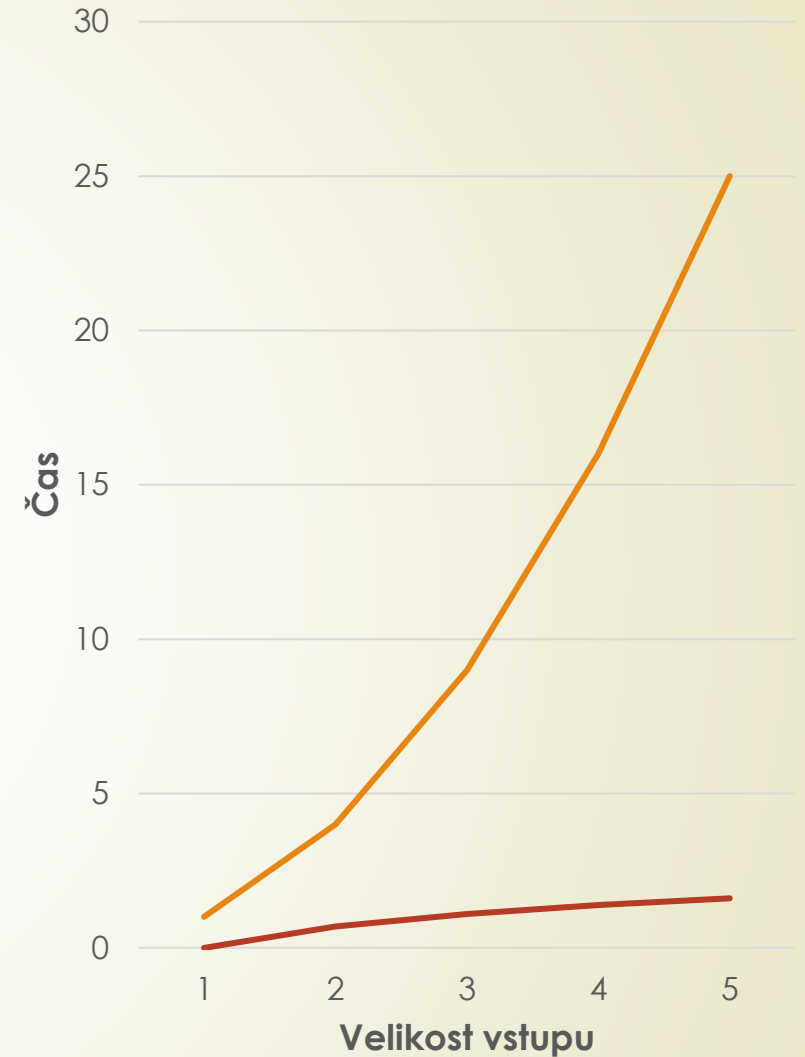


Asymptotická složitost

- Skutečná složitost (náročnost) je odvislá od mnoha faktorů
 - Implementace, výkon procesoru, přenosová rychlost pamětí, ...
- **Odhad složitosti**
- = popis růstu složitosti s rostoucí velikostí vstupních dat bez určení konkrétní hodnoty
- Zajímá nás složitost blížící se k nekonečně velkému vstupu
- Konstantní hodnoty tak lze ignorovat

Porovnání dvou algoritmů

- ▶ Jak lze porovnat rychlost dvou algoritmů, řešící stejný problém?
- ▶ Každému algoritmu náleží spojitá neklesající matematická funkce
- ▶ Jedná se o závislost velikosti vstupních dat a času, který je potřebný k dosažení řešení
- ▶ Čím „pomaleji“ funkce roste, tím je algoritmus rychlejší





Výpočet složitosti algoritmu


- ▶ Jedná se o součet a následné zjednodušování elementárních operací vzhledem k velikosti vstupu
- ▶ Elementární operace
- ▶ Aritmetické operace, porovnání, přesun hodnoty v paměti
- ▶ V kódu procházíme jednotlivé kroky a ohodnocujeme potřebným počtem operací
- ▶ Pro výpočet řešíme vždy nejhorší možnou náročnost

Výpočet složitosti

- příklad

- Jednotlivé elementární operace jsou ohodnoceny jako 1
- Cyklus se opakuje N
- V nejhorším případě je složitost:
 $1 + 1 + 1 + 1 + N + N + 1 + 1$
 $= 2N + 6$
- Jednotlivé konstanty můžeme odstranit a celková složitost je tak N
- Zapisujeme $\Theta(n)$

```
public int Suma(int[] pole, int n)
{
    if(pole.Length < n) 1
        n = pole.Length; 1
    int result = 0; 1
    for(int i = 0; i <= n ; i++) 1
    {
        result += pole[i]; 1
    }
    return result; 1
}
```



Přehled základních složitostí algoritmů

Notace		Název	Příklady algoritmů
$O(1)$		Konstantní	Získání prvku z pole s konstantním indexem
$O(\log n)$		Logaritmická	Vyhledávání binárním hledáním, třídění QuickSort
$O(n)$		Lineární	Prohledávání pole, výpis všech prvků v poli
$O(n \log n)$		Lineárně-logaritmická	MergeSort, HeapSort, Quicksort v nejhorším případě
$O(n^2)$		Kvadratická	BubbleSort, SelectionSort, InsertionSort
$O(n^3)$		Kubická	Násobení matic
$O(2^n)$		Exponenciální	Hledání všech možností, řešení problému obchodního cestujícího
$O(n!)$		Faktoriální	Řešení problému obchodního cestujícího