



# **Programovanie v jazyku Python**

Zložitosť algoritmov, triedenie a vyhľadávanie  
prednáška 3

Katedra kybernetiky a umelej inteligencie  
Technická univerzita v Košiciach  
Ing. Ján Magyar

# Algoritmizácia

- popis riešenia problému ako postupnosť jednoznačných krokov
- každý problém má niekoľko riešení - ako vybrať to najlepšie?
- výber vhodného algoritmu môže byť rozdiel medzi riešiteľným/neriešiteľným problémom
- väčšina problémov sa dá namapovať do problému, ktorý je možné vyriešiť niektorým z klasických algoritmov

# Analýza algoritmů - počet operací

```
def exp1(a, b):  
    ans = 1  
    while (b > 0):  
        ans *= a  
        b -= 1  
    return ans
```

# Analýza algoritmov - asymptotická notácia

- vyjadruje horný limit zložitosti algoritmu ak vstup je stále väčší a väčší
- najčastejšie sa používa Big-O notácia
- zanedbávajú sa konštantné časti algoritmu, do úvahy sa berie iba časť ktorá je závislá od veľkosti vstupných parametrov
- $f(x) \in O(n^2)$

# Výpočet mocnín

```
def exp2(a, b):  
    if b == 1:  
        return a  
    if (b % 2) == 0:  
        return exp2(a * a, b / 2)  
    else:  
        return a * exp2(a, b - 1)
```

# Fibonacciho čísla

```
def fib(a):  
    if a == 0 or a == 1:  
        return 1  
    else:  
        return fib(a - 1) + fib(a - 2)
```

# Vyhľadávanie

Predpokladajme, že máme zoradený zoznam celých čísel od najmenšieho po najväčšie. Aký je najmenej zložitý spôsob nájdania ľubovoľného prvku v zozname?

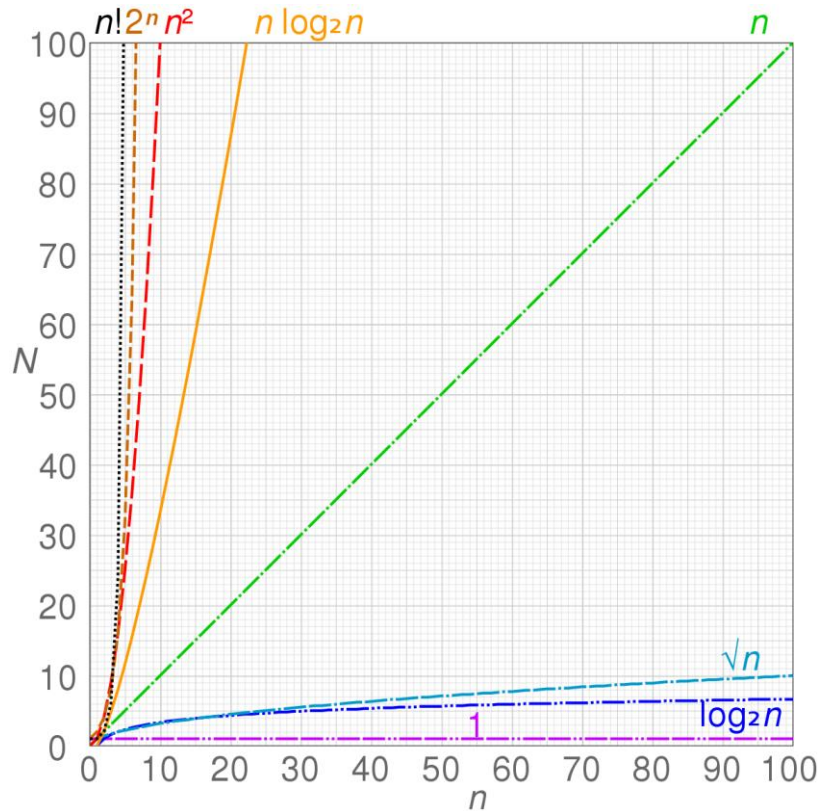
# Zoradenie zoznamu

Majme nezoradený zoznam celých čísel. Aký je intuitívny spôsob ich zoradenia?

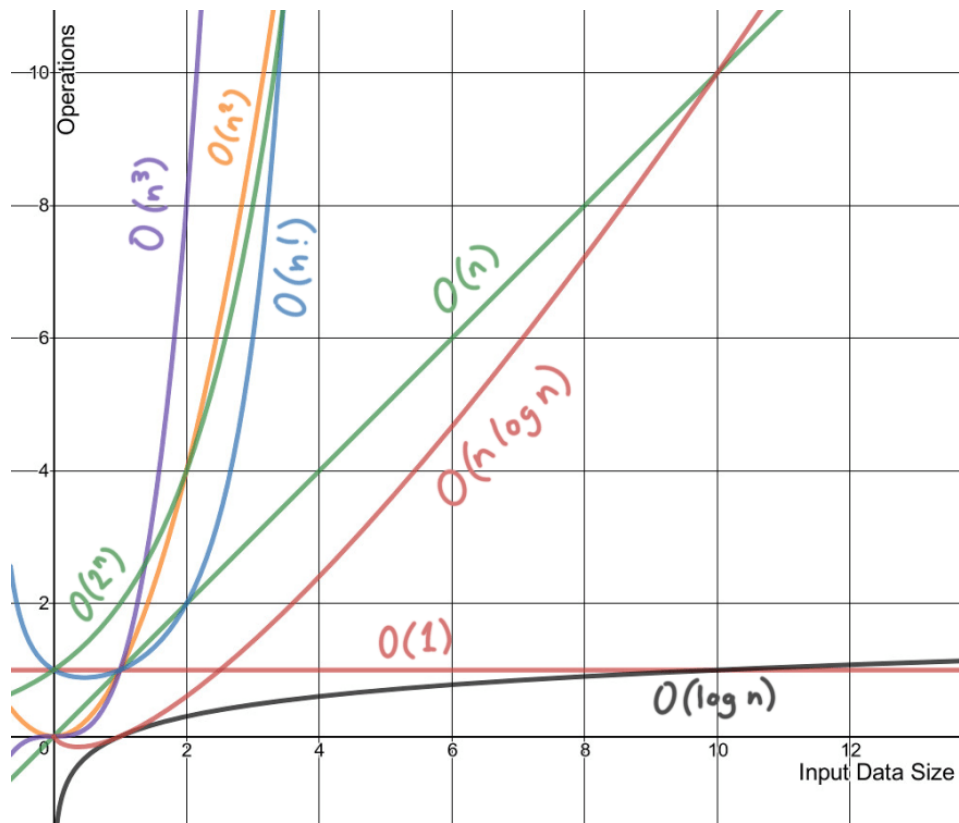
2, 8, 6, 1, 3, 9, 5, 4, 7



# Časová komplexita algoritmov



# Časová komplexita algoritmů



# Časová komplexita algoritmov

predpokladáme frekvenciu 1GHz (jedna operácia za nanosekundu)

	$n = 1000$	$n = 1,000,000,000$
$O(\log n)$	10 ns	10 ms
$O(n)$	1 $\mu$ s	1 s
$O(n^2)$	1 ms	16 minút
$O(2^n)$	$10^{284}$ rokov	...

# Základné triediace algoritmy

- Selection sort
- Bubble sort
- Insertion sort
- Quick sort
- Merge sort
- Bucket sort
- Radix sort

# Selection sort

- rozdelí zoznam na dve časti
  - zoradený začiatok
  - nezoradená zvyšná časť
- zoradí zoznam opakovaným výberom najmenšieho prvku z nezoradenej časti a pridá ho na začiatok zoznamu
- algoritmus končí ak dĺžka nezoradenej časti je jeden
  
- <https://www.youtube.com/watch?v=xWBP4lzkoyM>

# Bubble sort

- najjednoduchší algoritmus triedenia
  - prechádza zoznamom po dvojiciach a vymení ich ak druhý prvok je menší ako prvý
  - algoritmus končí ak počas prechodu nie je potrebné vymeniť prvky dvojice
- 
- <https://www.youtube.com/watch?v=nmhjrI-aW5o>

# Insertion sort

- podobne ako selection sort delí zoznam na dve časti
  - zotriedený začiatok
  - nezotriedená zvyšná časť
- vždy zoberie prvý prvok nezoradenej časti a pridá ho na správne miesto v zoradenom začiatku
- algoritmus končí ak dĺžka zvyšnej časti je 0
  
- <https://www.youtube.com/watch?v=OGzPmgsI-pQ>

# Quick sort

- algoritmus typu rozdeľuj a panuj
  - vyberie sa prvok (tzv. pivot) a zoznam sa usporiada tak, aby hodnoty nižšie ako pivot boli vľavo a hodnoty vyššie ako pivot vpravo
  - následne sa vykoná quick sort algoritmus pre ľavú partíciu a potom pre pravú (rekurzia)
- 
- <https://www.youtube.com/watch?v=PgBzjlCcFvc>



# Merge sort

- algoritmus typu rozdeľuj a panuj
  - rozdelí zoznam rekurzívne na polovice až kým nemáme zoznamy s jedným prvkom
  - následne menšie zoznamy spojí tak že postupne zoberie najmenšie hodnoty (buduje zoradené zoznamy)
- 
- <https://www.youtube.com/watch?v=JSceec-wEyw>

# Bucket sort

- ideálna metóda pre vstup s uniformnou distribúciou z istého intervalu
- používa hašovanie, vytvorí niekoľko bucketov
- najprv pridelí členy zoznamu do príslušných bucketov
- zotriedi zoznamy bucketov pomocou insertion sort
- urobí sa konkatenácia bucketov

# Radix sort

- zoradenie na základe číslíc
- pre každé desatinné miesto sprava zoradí zoznam (použije sa ľubovoľná metóda)

# Zložitosť triediacich algoritmov

algoritmus	best case	average case	worst case
selection sort	$n^2$	$n^2$	$n^2$
bubble sort	$n$	$n^2$	$n^2$
insertion sort	$n$	$n^2$	$n^2$
heap sort	$n \log(n)$	$n \log(n)$	$n \log(n)$
quick sort	$n \log(n)$	$n \log(n)$	$n^2$
merge sort	$n \log(n)$	$n \log(n)$	$n \log(n)$
bucket sort	$n+k$	$n+k$	$n^2$
radix sort	$nk$	$nk$	$nk$

# Zhrnutie

- spôsoby analýzy algoritmov
- big-O notácia
- skupiny zložitosti algoritmov
- porovnávanie zložitosti algoritmov
- binárne vyhľadávanie
- základné algoritmy triedenia