# Základy forenzních databází

Tereza Uhlíková

verze 2.0

# Kdo jsem

Tereza Uhlíková

Ústav analytické chemie

skupina teoretické spektroskopie

místnost A277

https://web.vscht.cz/~uhlikovt/

tereza.uhlikova@vscht.cz

# 1. lekce

- Co je to databáze
- Proč, kdo, kdy, jak ...
- Něco málo z historie
- CAP problém

*Dobrodružství kriminalistiky - televizní seriál*
*Dějiny psané Římem - Vojtěch Zamarovský*

# 2. lekce

- Základy informatiky
- Software
  - informace
  - záznam informace
  - číselné soustavy
  - písmenné kódy
- Hardware
  - Alan Turing, John von Neumann a počítač
  - historie vývoje počítače & super počítač
  - procesor a datová uložiště
- Architektura databází

*Alan Turing - Enigma*
*Contact film z roku 1997; seti@home*

# 3. lekce

- Algoritmus
  - vlastnosti
  - zápis
  - struktura
- Datové typy

*Arthur C. Clarke - Devět miliard božích jmen*

# 4. lekce

- Výroková logika
- Databáze
  - DB a SŘBD
  - databázové modely
- Relační model
  - návrh tabulky

*Aghata Christie - Hercules Poirot*

# 5. lekce

- ERA model
- Klíče, integrita a kardinalita
- Normalizace databáze

# 6. lekce

- Datové struktury
- Ukládání
- Složitost
- Řazení
- Přenos dat

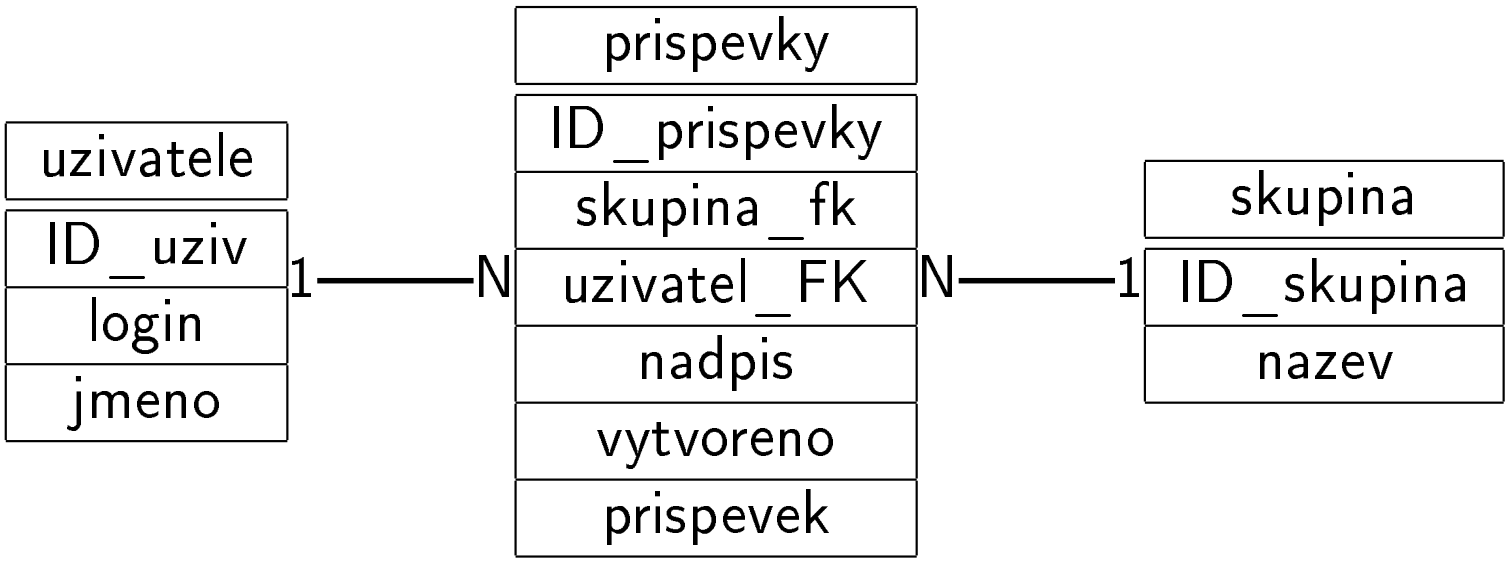
# Co bude dnes

- Vyhledávání
  - sekvenční
  - binární
  - hashing
- Jak google pracuje
- Neuronové sítě a Strojové učení

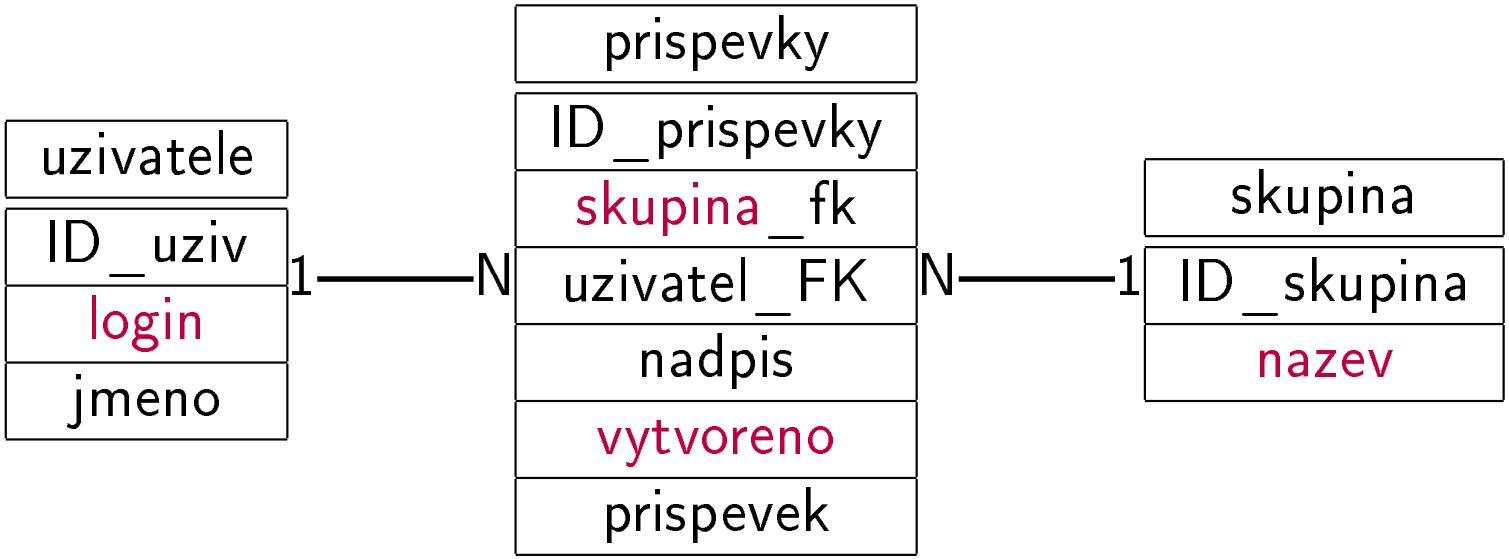
# Zahřívací příklad

Vytvořte databázi (aplikaci), kde registrovaní uživatelé mohou vkládat příspěvky do různých diskusních skupin.
Uživatelé se budou přihlašovat pomocí loginu,
diskusní skupiny budeme vypisovat seřazené podle názvu a příspěvky v nich potom podle datumu vložení.

# Užití indexů v databáziče



# Užití indexů v databáziče



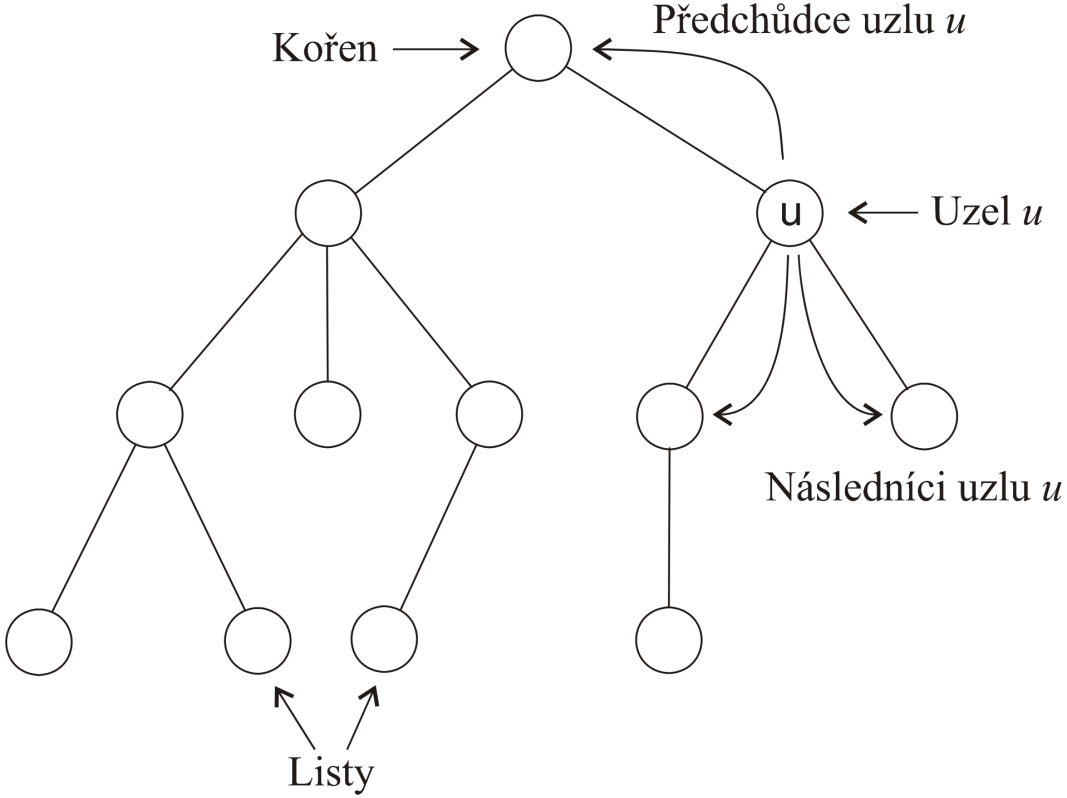

vytvoreno - vypisování několika nejnovějších příspěvků nezávisle na skupině

(skupina, vytvoreno) - výpis ve skupinách ...

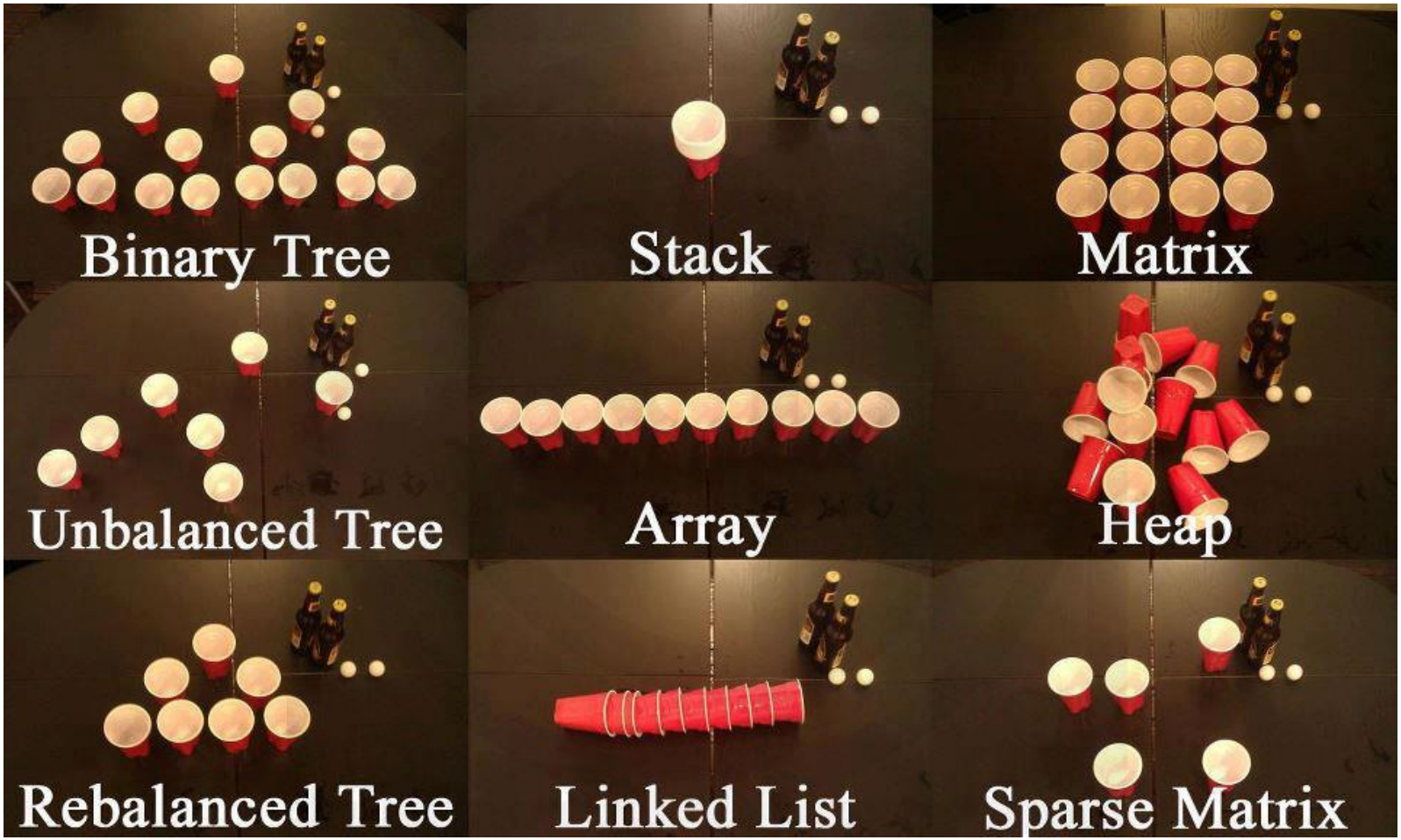
# Lineární a stromová datová struktura

# Lineární a stromová datová struktura

| klíč (index) | 5 | 6 | 12 | 13 | 14 | 28 |
|---|---|---|---|---|---|---|
| hodnota | a | r | 23 | b | x | 54 |

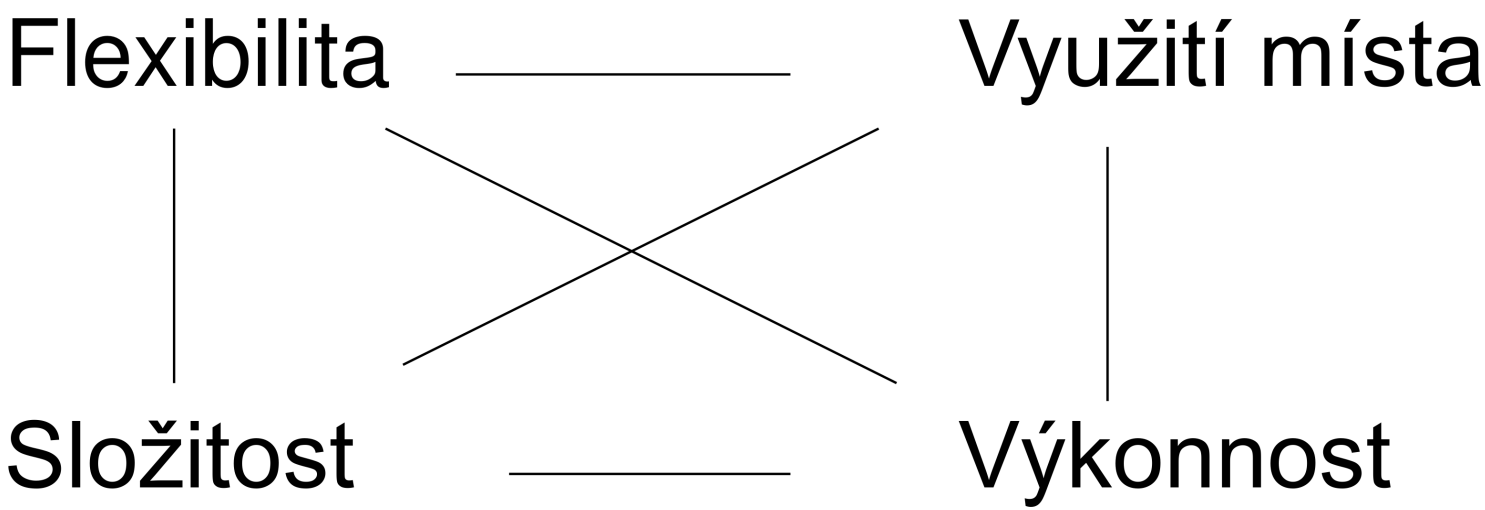


# Porovnání


Binary Tree
Stack
Matrix
Unbalanced Tree
Array
Heap
Rebalanced Tree
Linked List
Sparse Matrix

# Vyváženost

# Vyváženost



# Složitost $O$

# Složitost $O$

závislost doby výpočtu na počtu zpracovávaných údajů $n$ - časová složitost
závislost velikosti paměti potřebné pro výpočet na počtu zpracovávaných údajů $n$ - paměťová složitost

$$\log_2(n)$$
$$n$$
$$n \log_2(n)$$
$$n^2$$
$$2^n$$

# Pojmy ve vyhledávání

- vyhledávání

# Pojmy ve vyhledávání

- vyhledávání - operace získávání dílčích informací z velkého objemu dat (V souboru dat chceme najít konkrétní datovou položku.)
- efektivita vyhledávacího algoritmu

# Pojmy ve vyhledávání

- vyhledávání - operace získávání dílčích informací z velkého objemu dat (V souboru dat chceme najít konkrétní datovou položku.)
- efektivita vyhledávacího algoritmu - najít požadovanou datovou položku s co nejmenším počtem porovnání nebo opakování = složitost
- pomáháme si klíčem (indexem) -

# Pojmy ve vyhledávání

- vyhledávání - operace získávání dílčích informací z velkého objemu dat (V souboru dat chceme najít konkrétní datovou položku.)
- efektivita vyhledávacího algoritmu - najít požadovanou datovou položku s co nejmenším počtem porovnání nebo opakování = složitost
- pomáháme si klíčem (indexem) - množina hodnot, která jednoznačně identifikuje záznam (primární, cizí, unikátní, sekundární)
- ukončení vyhledávání

# Pojmy ve vyhledávání

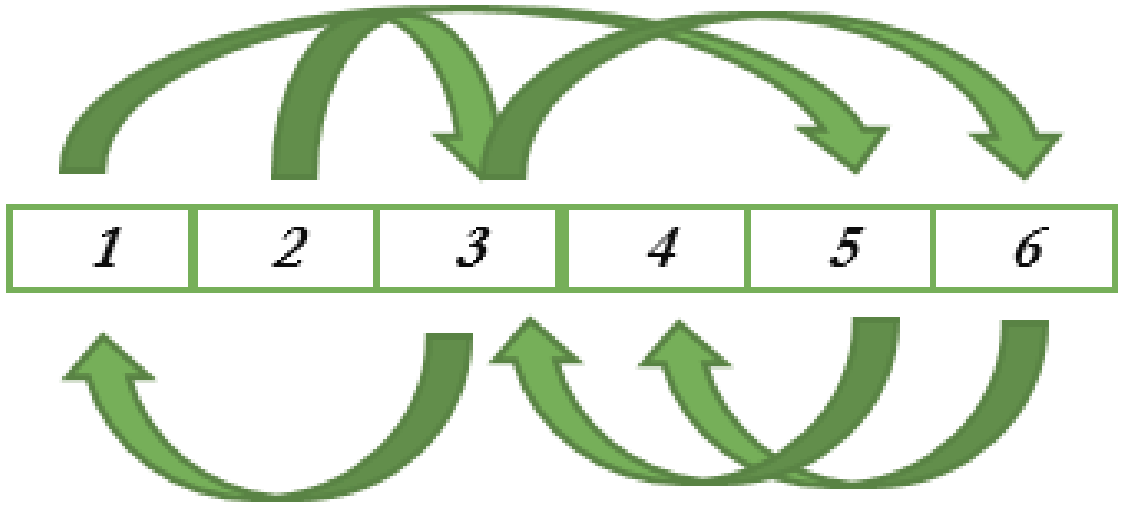
- vyhledávání - operace získávání dílčích informací z velkého objemu dat (V souboru dat chceme najít konkrétní datovou položku.)
- efektivita vyhledávacího algoritmu - najít požadovanou datovou položku s co nejmenším počtem porovnání nebo opakování = složitost
- pomáháme si klíčem (indexem) - množina hodnot, která jednoznačně identifikuje záznam (primární, cizí, unikátní, sekundární)
- ukončení vyhledávání - definuje se ekvivalence = vyhledávaný klíč a klíč právě porovnávané položky

# Pojmy ve vyhledávání

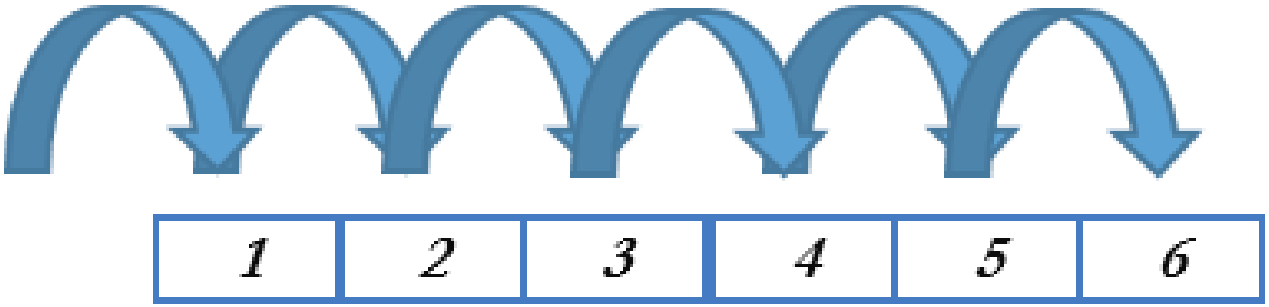
- vyhledávání - operace získávání dílčích informací z velkého objemu dat (V souboru dat chceme najít konkrétní datovou položku.)
- efektivita vyhledávacího algoritmu - najít požadovanou datovou položku s co nejmenším počtem porovnání nebo opakování = složitost
- pomáháme si klíčem (indexem) - množina hodnot, která jednoznačně identifikuje záznam (primární, cizí, unikátní, sekundární)
- ukončení vyhledávání - definuje se ekvivalence = vyhledávaný klíč a klíč právě porovnávané položky

vyhledávací tabulka symbolů - datová struktura položek s klíči (např. index v knize, hodnota funkce na gridu)

# Vyhledávání náhodné



# Sekvenční v lineární datové struktuře



# Sekvenční vyhledávání v lineární datové struktuře

- jednoduchá implementace / algoritmus
- funguje u neuspořádaných i uspořádaných seznamů
- minimální dodatečná paměť
- použít pro různé sekvenční datové struktury
- nejvhodnější při hledání jednoho prvku

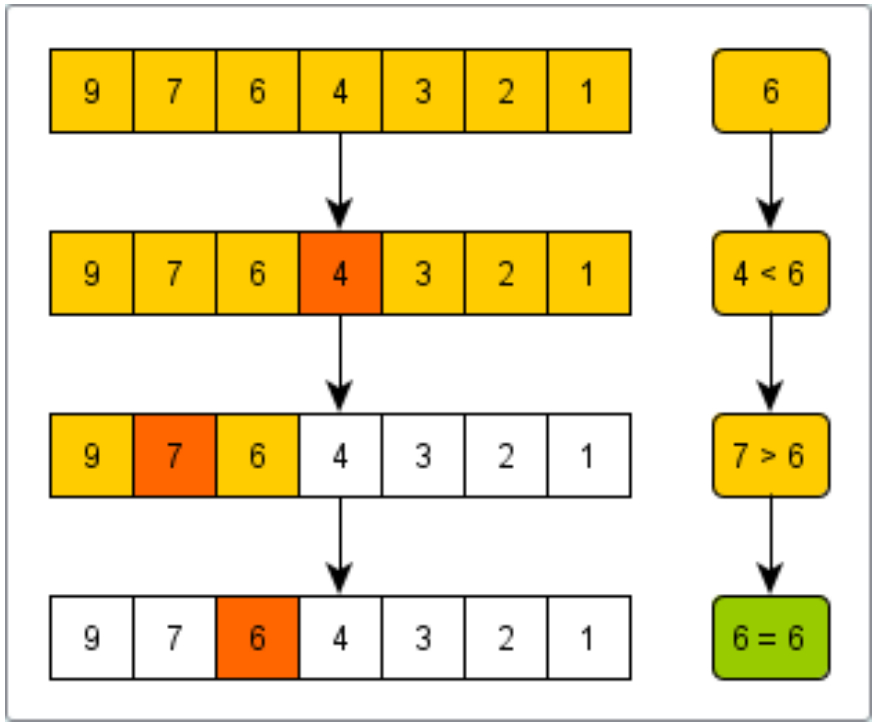
- u velkých seznamů nebo polí má nejhorší dobu provádění.
- Složitost $O(n)$
- při vyhledávání v uspořádaných polích nebo seznamech není efektivní = není přirozený alg.

zarážky - vložení vyhledávaného prvku za poslední prvek tabulky

# Binární vyhledávání v lineární datové struktuře

# Binární vyhledávání v lineární datové struktuře

pracují se setříděnou tabulkou symbolů - „rozděl a panuj“ a rekurze
Soubor položek se rozdělí na dvě části, určí se ke které části vyhledávaný klíč náleží, a pak se na tuto část soustředíme.




složitost $O(\log_2 N)$.

# Binární vyhledávání v lineární datové struktuře

operace

1. uspořádání
2. 
   - if <je co dělit> <rozděl prohledávanou tabulku na dvě poloviny>
   - else <ukonči vyhledávání jako neúspěšné>
   - if <dělící prvek je shodný s hledaným> <ukonči vyhledávání a vrať polohu dělícího prvku>
   - if <dělící prvek je menší než hledaný> <proveď binární vyhledávání nad pravou polovinou tabulky>
   - else <proveď binární vyhledávání nad levou polovinou tabulky>

# Binární vyhledávání v lineární datové struktuře

- jednoduchá implementace
- velmi efektivní $O(\log_2 N)$
- funguje pouze na uspořádaných seznamech nebo polích

- před vyhledáváním musí být celé pole seřazeno
- vyžaduje přístup k jednotlivým prvkům v seznamu
- udržování indexů nebo ukazatelů na interval vyhledávání může být náročné na další paměť

# Porovnání sekvenčního a binárního vyhledávání

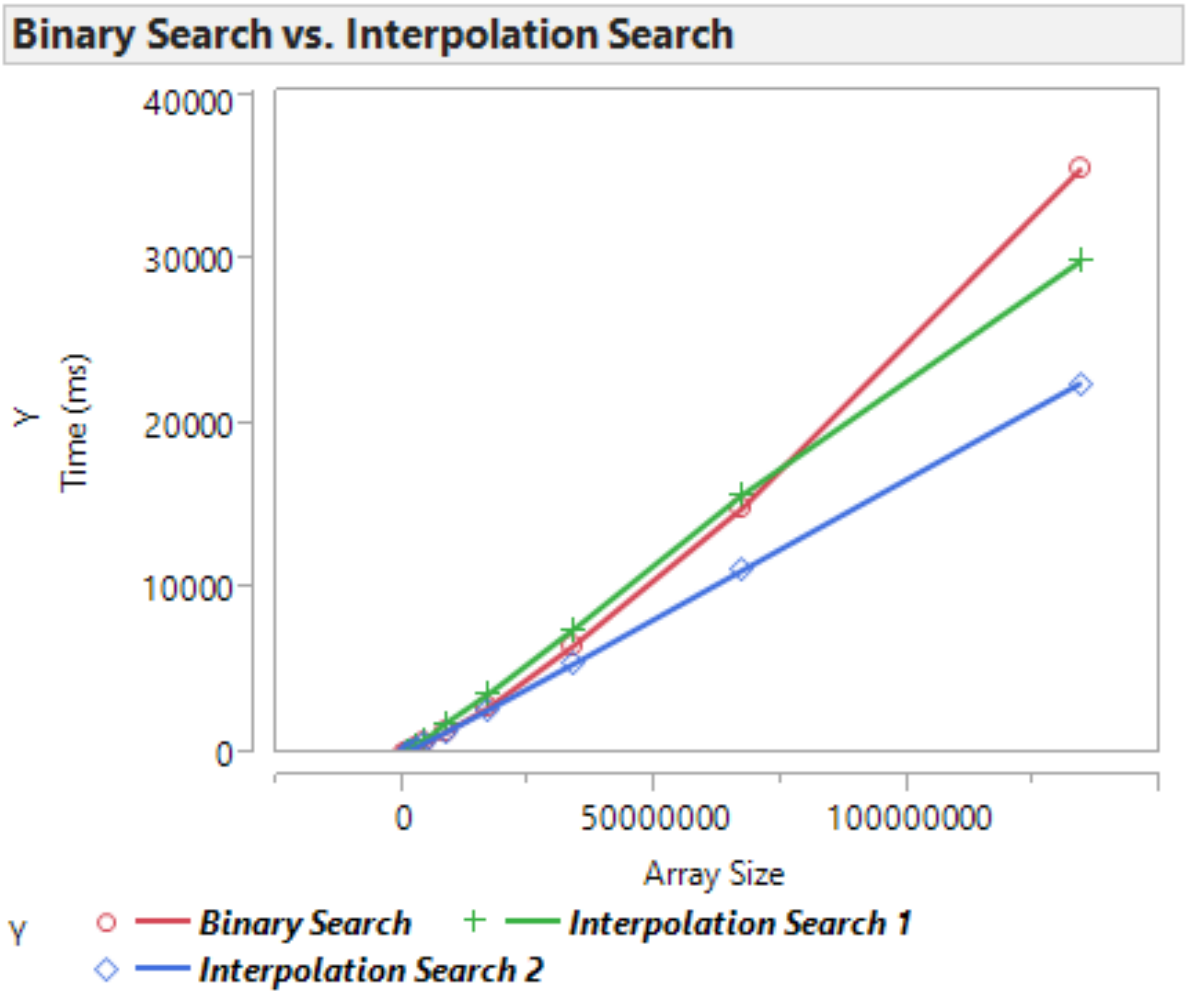
předpoklady - jeden záznam pro danou hodnotu klíče
1 operace = 1/1000 vteřiny

| počet záznamů | průměrný případ pro sekvenční vyhledávání | | nejhorší případ pro binární vyhledávání | |
|---|---|---|---|---|
| | operací | čas (vteřiny) | operací | čas (vteřiny) |
| 31 | 16 | 0.016 | 5 | 0.005 |
| 119 | 60 | 0.060 | 7 | 0.007 |
| 1 999 | 1 000 | 1 | 11 | 0.011 |
| 9 999 | 5 000 | 5 | 14 | 0.014 |
| 999 999 | 500 000 | 500 | 20 | 0.020 |

# Další algoritmy vyhledávání

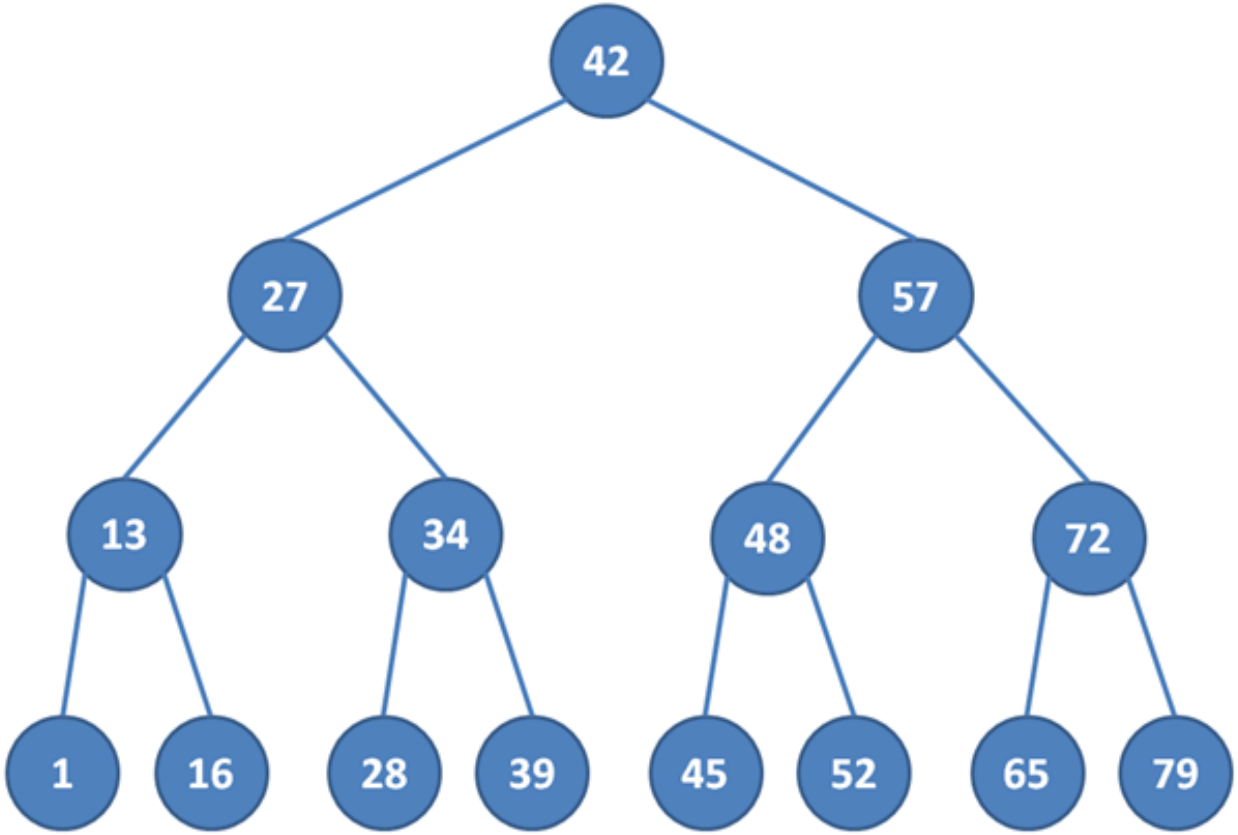
interpolační
skokové
exponenciální
ternární ...

# Interpolační vyhledávání - porovnání



# Binární vyhledávací strom (BST)

# Binární vyhledávací strom (BST)



# Vyhledávání v binárním stromě

Postup pro vyhledání prvku 48
Začneme v kořeni, kde je číslo 42.

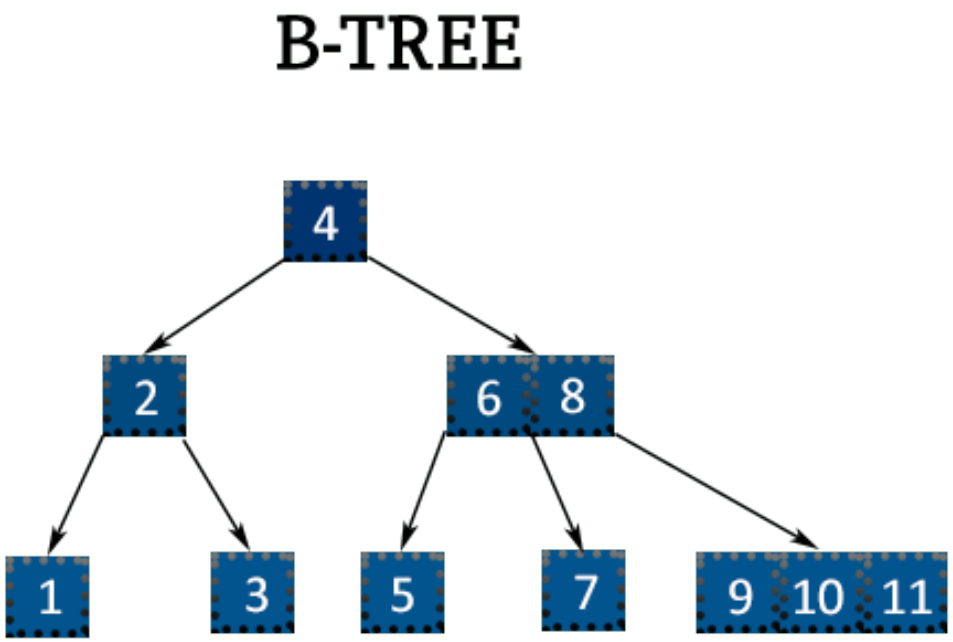
1. Je 48 rovno 42? => NE.
2. Jelikož 48 je větší než 42, potřebujeme přejít do pravé větve, ve které jsou větší čísla.
3. Je 48 rovno 57? => NE, je 48 větší než 57 => Ne, jdi doleva.
4. Je 48 rovno 48? => ANO. Máme nalezeno, otevřeme si šampáňo!

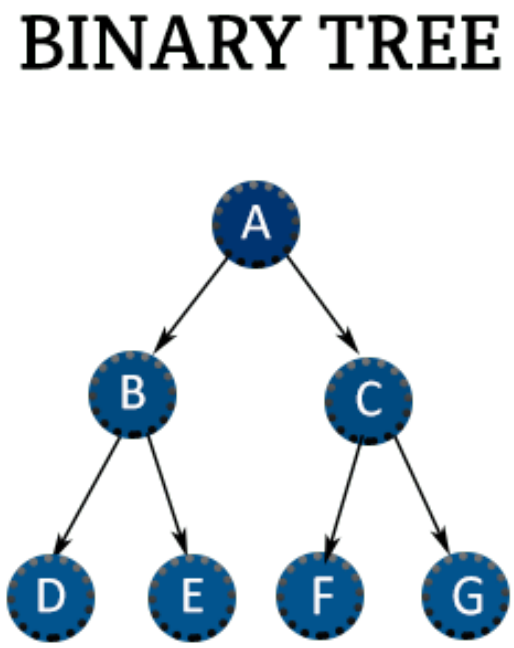
složitost $O(\log_2 N)$

# Další stromy

ALV-strom - binární vyhledávací strom s jednou podmínkou navíc: V každém vrcholu stromu se hloubka jeho levého a pravého podstromu liší nejvýše o jedna.

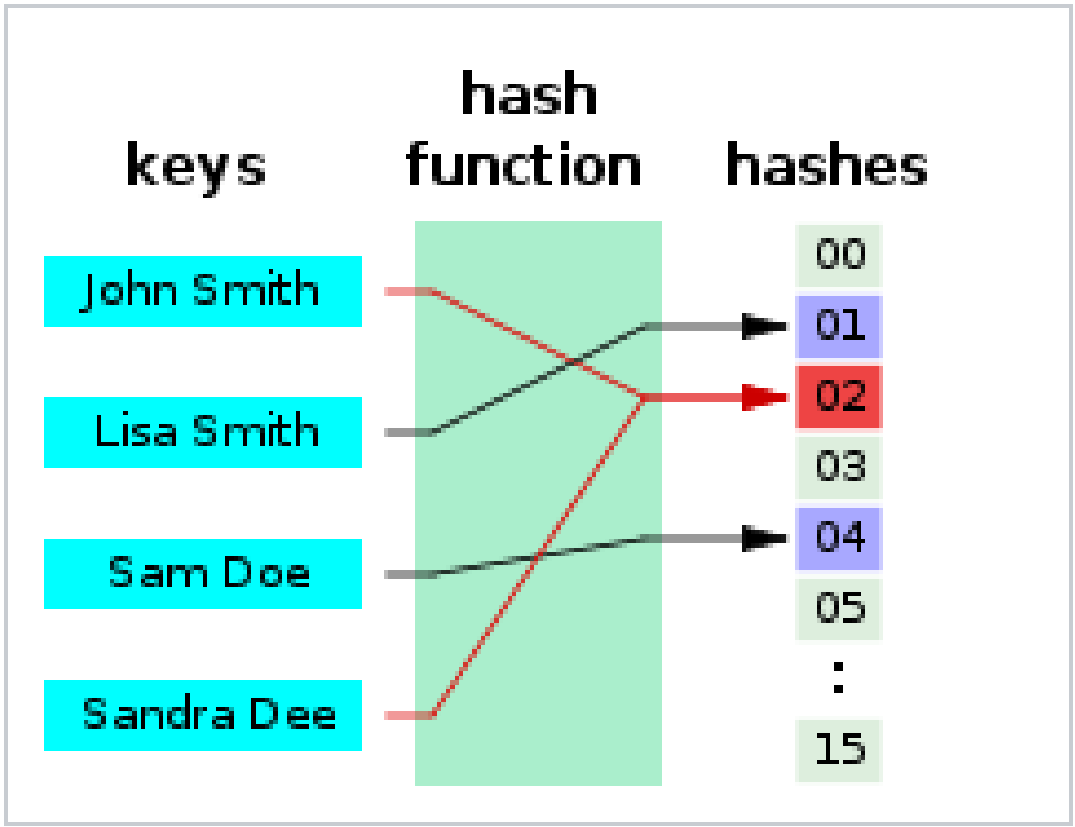
B-stromy - vícečetný uzel, v uzlu mohou být celé další struktury, které jsou do stromu navěšené podle nějaké jejich vlastnosti (ID, jméno uživatele)



# Hešovací tabulky kam se podíváš

tabulky s přímým přístupem, homogení datové pole,
hash (heš) - otisk; pomocí funkce dostaneme jednoznačný otisk; $O(1)$



# Hešovací funkce

je matematická funkce (resp. algoritmus), která
převádí posloupnost vstupních dat = bitů $k$ (klíč) na posloupnost bitů (relativně) malé pevné délky $a$ (adresa),

$$|a| < |k|$$

vlastnosti:

- Nemůže být příliš složitá, aby se tíha prohledávání množiny údajů zbytečně nepřenášela na časově náročný výpočet $f(\mathrm{K})$.
- Hešh musí být pro dva odlišné vyhledávací klíče s co největší pravděpodobností odlišný.
- Použití funkce $f(\mathrm{K})$ by mělo zajistit rovnoměrné a náhodné rozmístění prvků v tabulce.

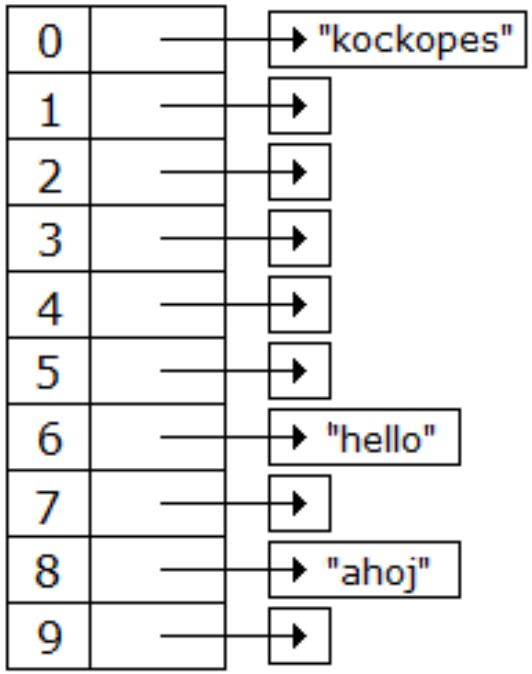
# Příklad hešovací funkce

součin ASCII hodnot znaků v řetězci modulo N

"hello" $\rightarrow$ (104*101*108*108*111) mod 10 = 6

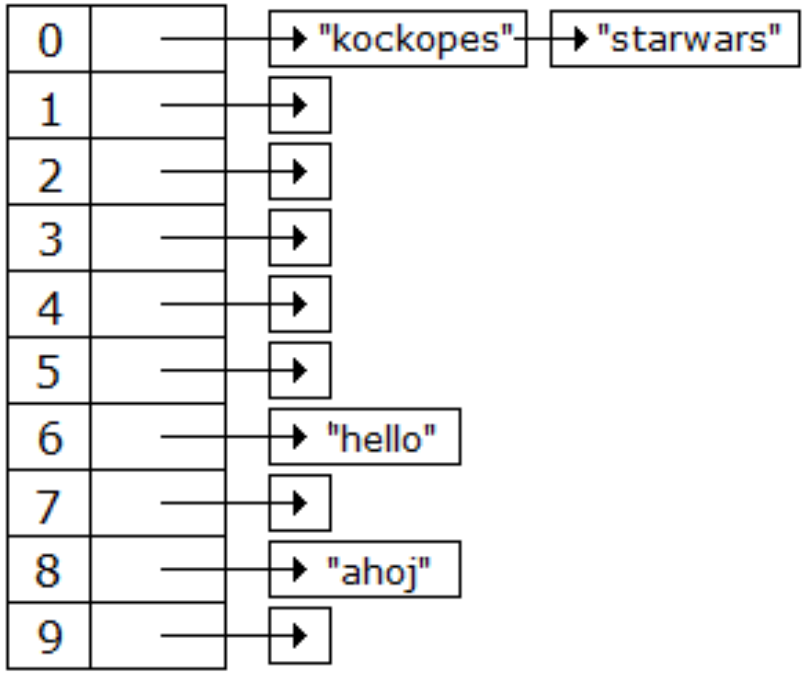
"ahoj" $\rightarrow$ (97*104*111*106) mod 10 = 8

"kockopes" $\rightarrow$ (107*111*99*107*111*112*101*115) mod 10 = 0



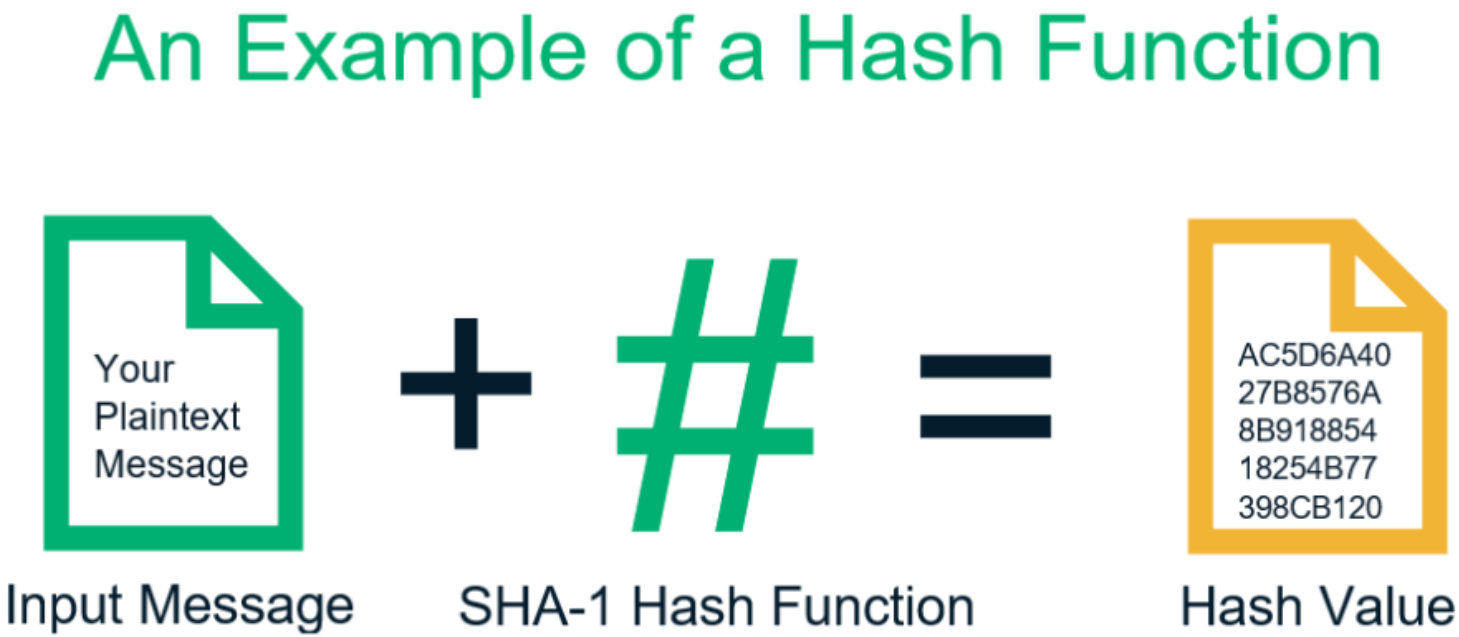
# Hashing

Jeden hash pro dva klíče - řetězení




V ideálním případě je binární vyhledávací strom pomalejší, pro případ nejhorší je strom rychlejší.

# Hash v kryptografii



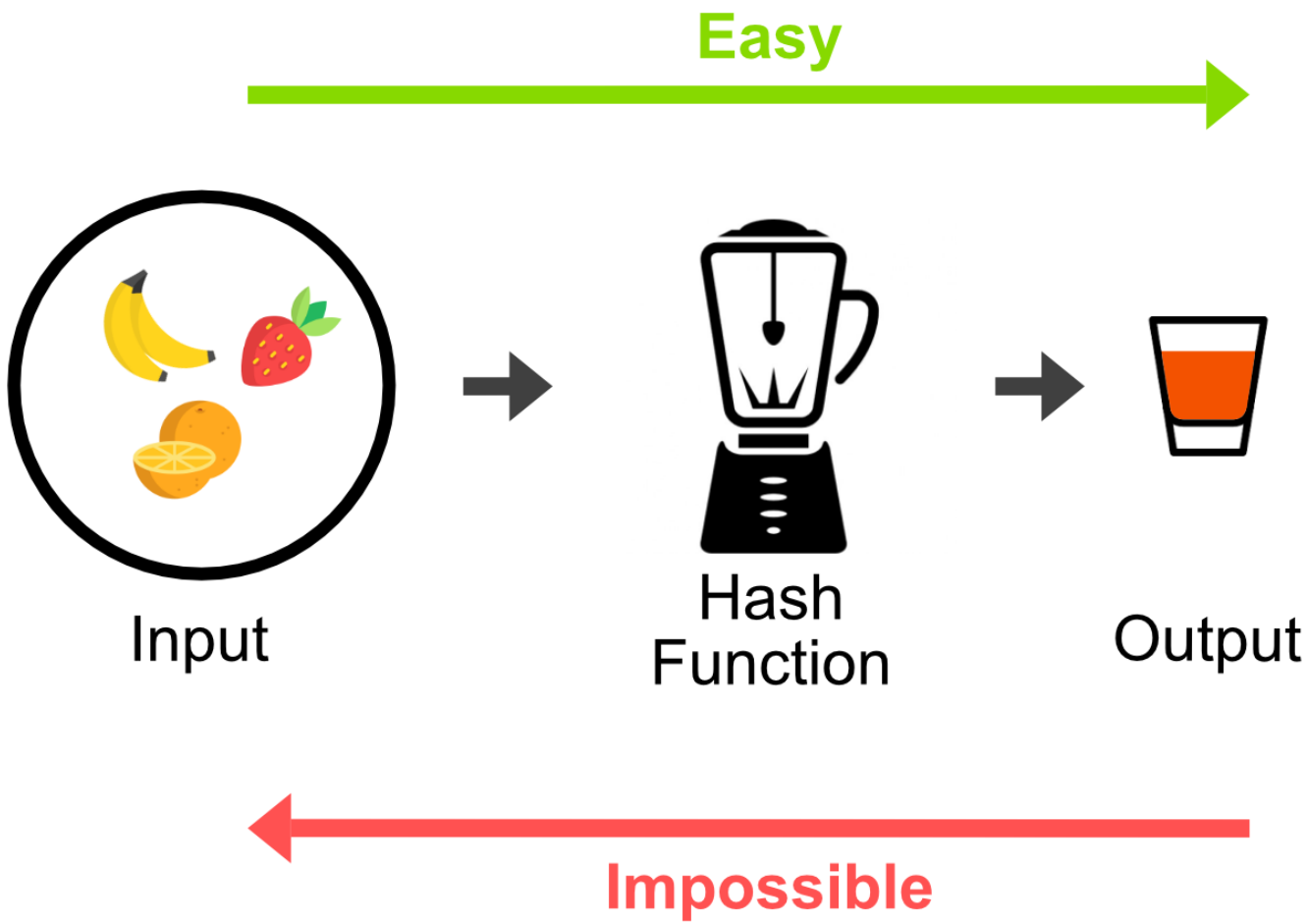
# Hash v kryptografii

| vstupní text | Hash hodnota |
|---|---|
| Dobrý den! | D364965C90C53DBF140 |
| Dobrý den, vítám vás na přednášce ze základů forenzních databází. To jste netušilli, co všechno se tu dovíte,že? | B26BACAB73C46D844C1 |

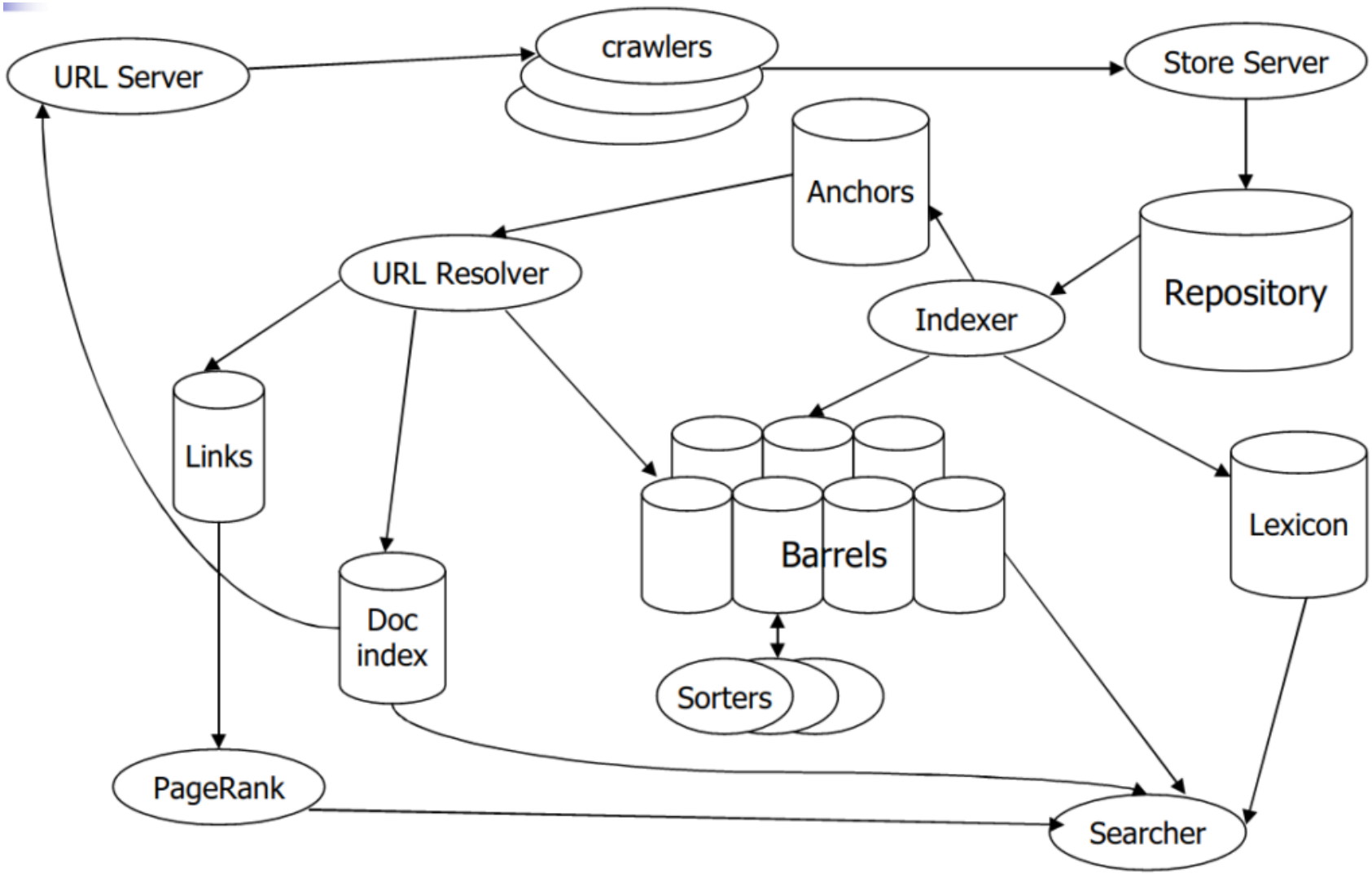
Šifra (Encryption) - je něco, co lze použí na transformaci textu do něčeho, co nelze přečíst použitím algoritmu a klíče. Ale!! zašifrovaný text lze zase nazpátek dešifrovat použitím stejného klíče (symetrická šifra) nebo jiným klíčem/algoritmem (asimetrická šifra).

Kryptografická hašovací funkce - jakmile jednou zahašujete data, už je nazpátek nedostanete (jednosměrný proces).

# Hash v kryptografii



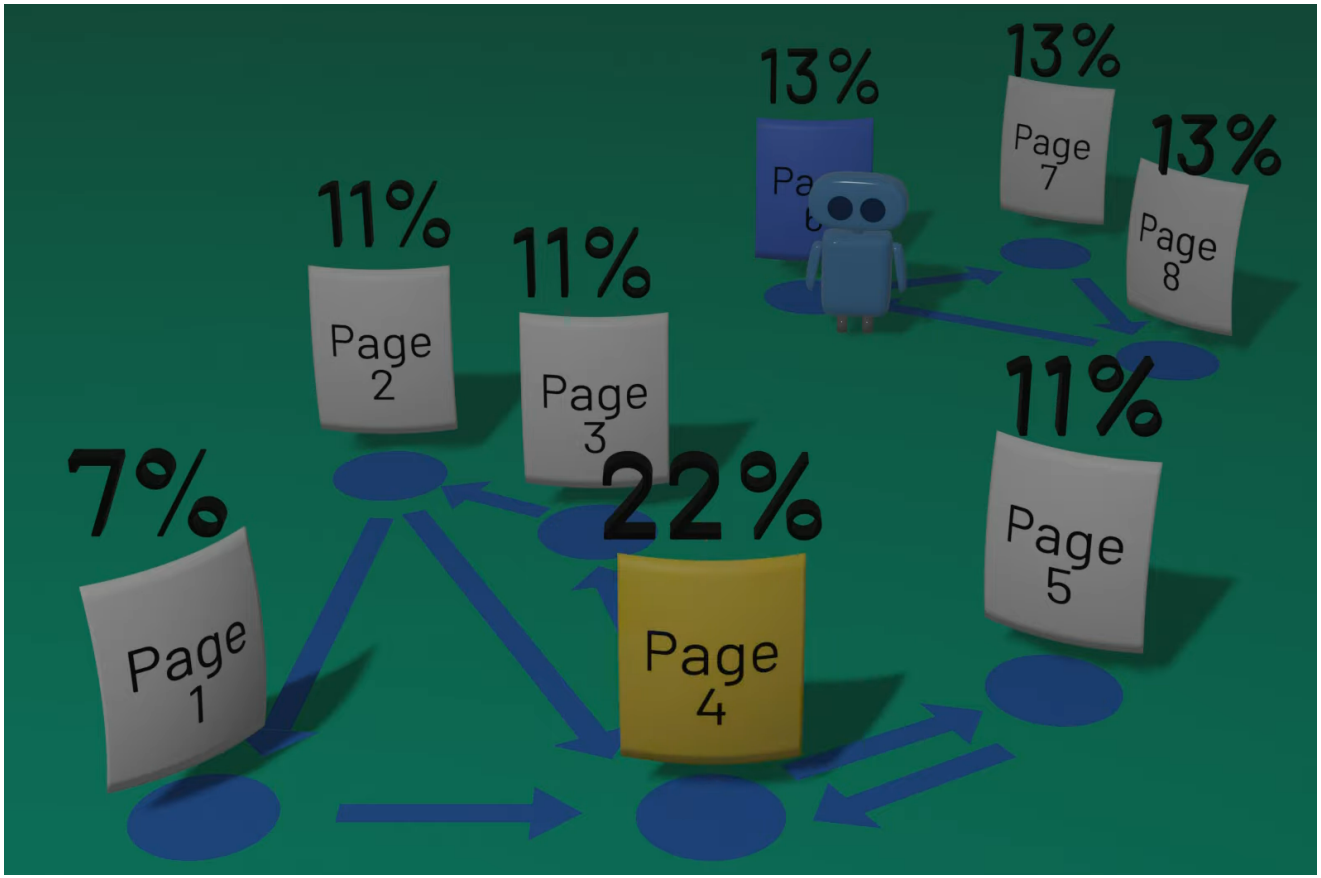
# Google


URL Server
crawlers
Store Server
Anchors
URL Resolver
Repository
Indexer
Links
Barrels
Lexicon
Doc index
Sorters
PageRank
Searcher

# Jak hledá google

1. Prolézací modul (crawling) je program, který prochází obsahem URL adres na internetu, zkoumá obash a prvotně jej indexuje.
2. Indexování (indexing) - Obsah obsažený v adresách URL je označen atributy a metadaty, které vyhledávači pomáhají obsah kategorizovat.
3. Barrels - předpřipravená, slinkovaná odpověď na dotaz
4. PageRank - "hodnota stránky"
5. Třídění informací na základě: Meaning, Relevance, Quality, Usability, Context...cca 200 kritérií

# PageRank

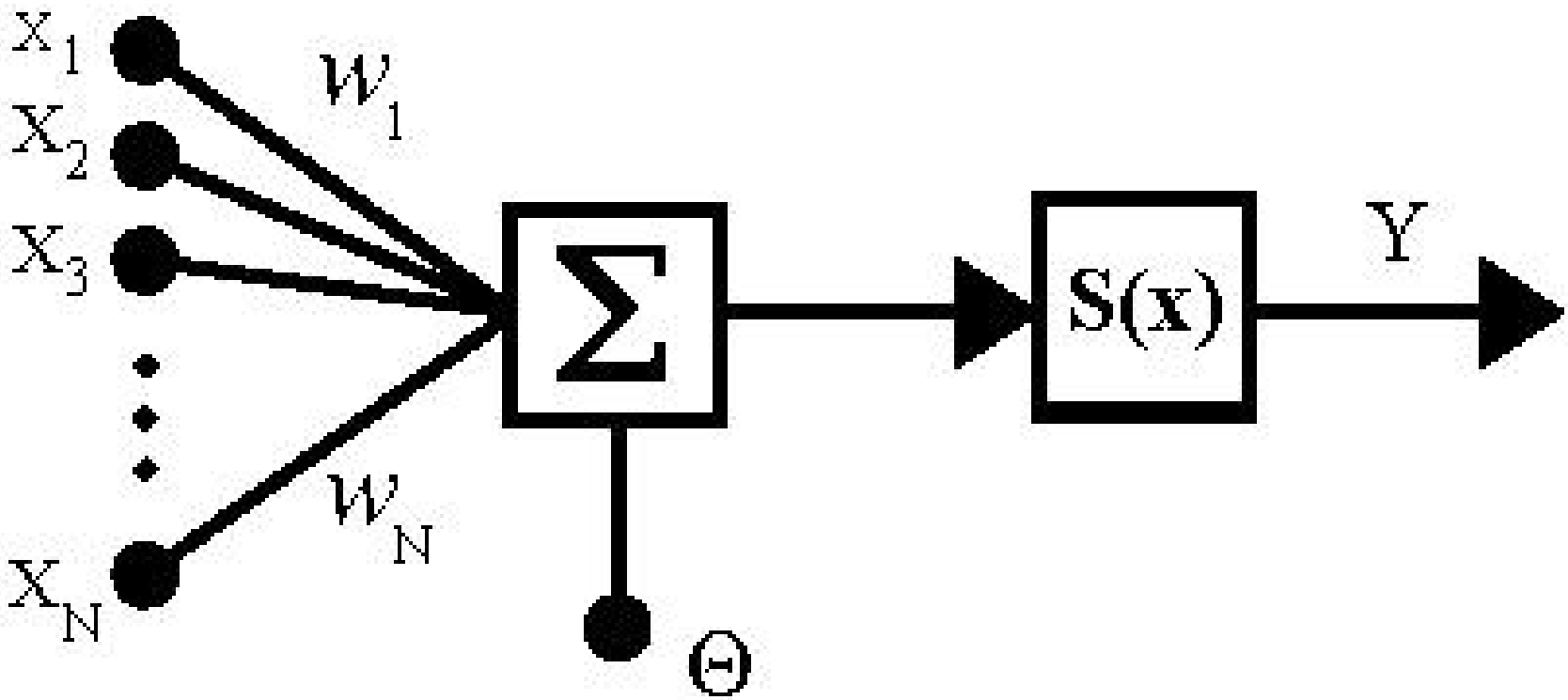



matematický vzorec, který posuzuje „hodnotu stránky“ pohledem na množství a kvalitu dalších stránek, které na ni odkazují. Jeho účelem je určit relativní důležitost dané webové stránky v internetové síti.

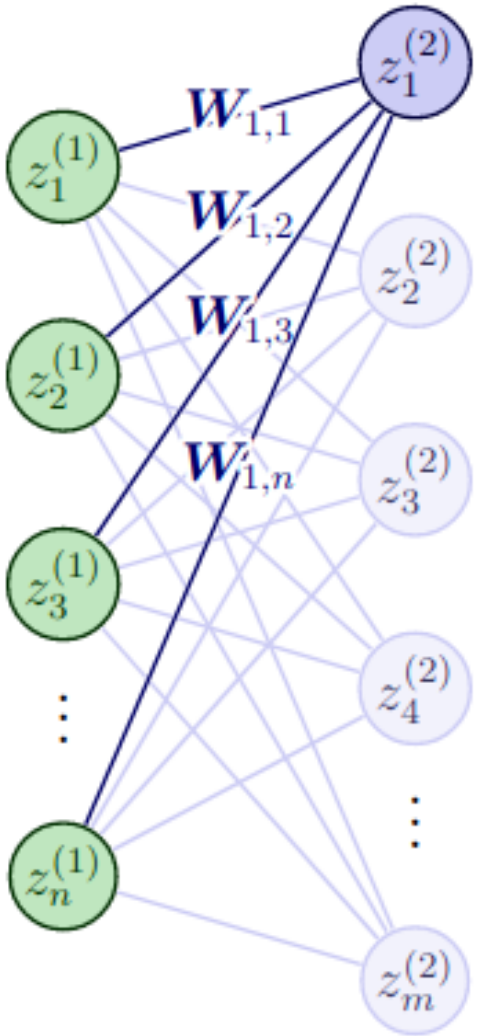
# Jeden neuron

# Jeden neuron

několik vstupů $x_i$ s vahami $w_i \rightarrow$ aktivační funkce $S(x) \rightarrow$ jeden výstup $Y$
jeden neuron - Perceptron



# Více neuronů



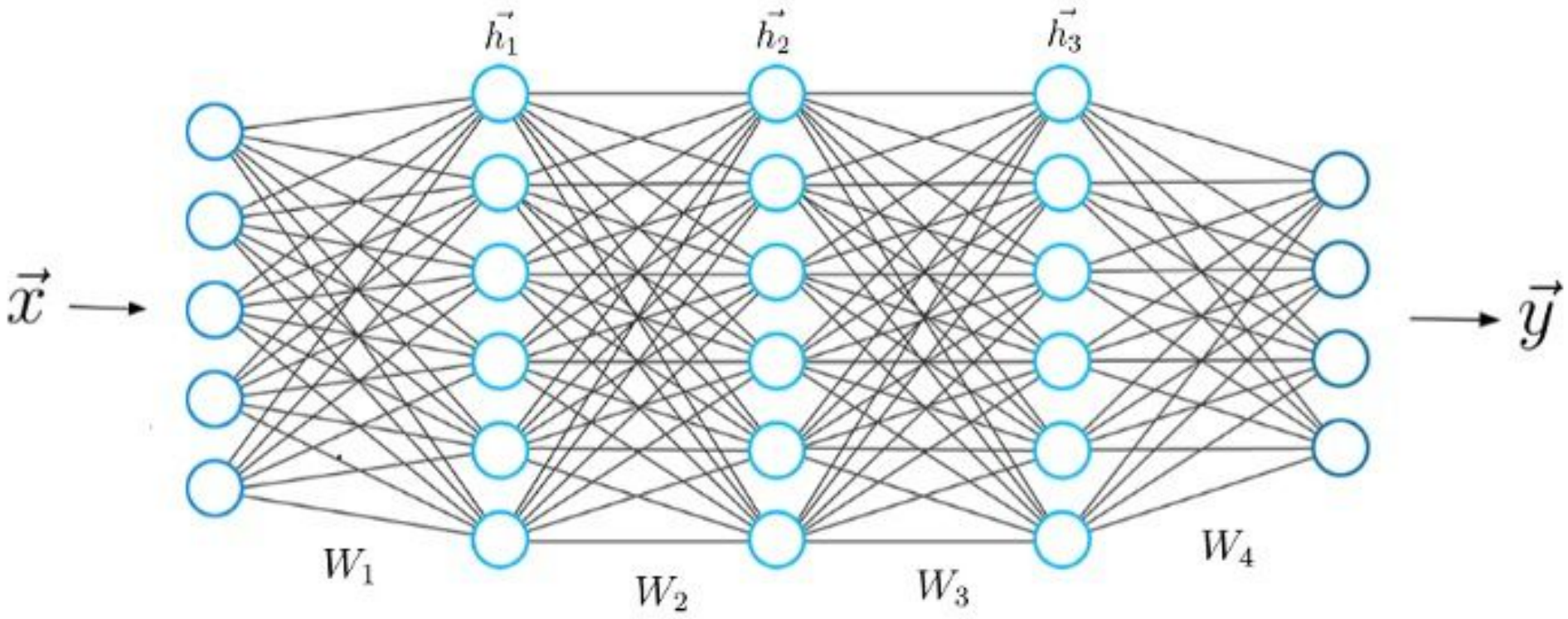

$$= f\left(\boldsymbol{W}_{1,1} z_1^{(1)} + \boldsymbol{W}_{1,2} z_2^{(1)} + \ldots + \boldsymbol{W}_{1,n} z_n^{(1)} + b_1^{(2)}\right)$$

$$= f\left(\sum_{i=1}^{n} \boldsymbol{W}_{1,i} z_i^{(1)} + b_1^{(2)}\right)$$

$$\begin{pmatrix} z_1^{(2)} \\ z_2^{(2)} \\ \vdots \\ z_m^{(2)} \end{pmatrix} = f\left[\begin{pmatrix} \boldsymbol{W}_{1,1} & \boldsymbol{W}_{1,2} & \ldots & \boldsymbol{W}_{1,n} \\ \boldsymbol{W}_{2,1} & \boldsymbol{W}_{2,2} & \ldots & \boldsymbol{W}_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{W}_{m,1} & \boldsymbol{W}_{m,2} & \ldots & \boldsymbol{W}_{m,n} \end{pmatrix} \begin{pmatrix} z_1^{(1)} \\ z_2^{(1)} \\ \vdots \\ z_n^{(1)} \end{pmatrix} + \begin{pmatrix} b_1^{(2)} \\ b_2^{(2)} \\ \vdots \\ b_m^{(2)} \end{pmatrix}\right]$$

$$\boldsymbol{z}^{(2)} = f\left(\boldsymbol{W}^{(2)} \boldsymbol{z}^{(1)} + \boldsymbol{b}^{(2)}\right)$$

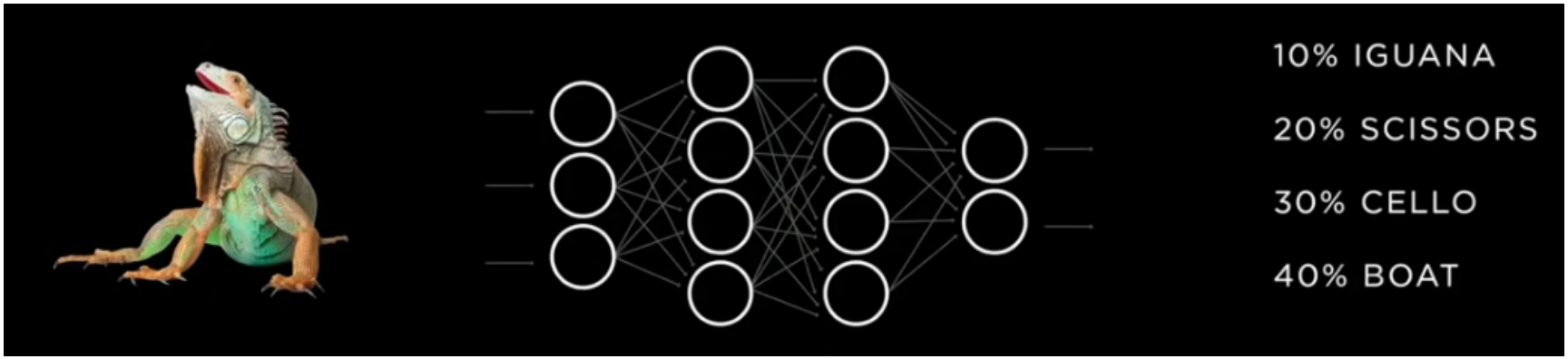
# Více neuronů



# Trénování neuronových sítí

# Trénování neuronových sítí

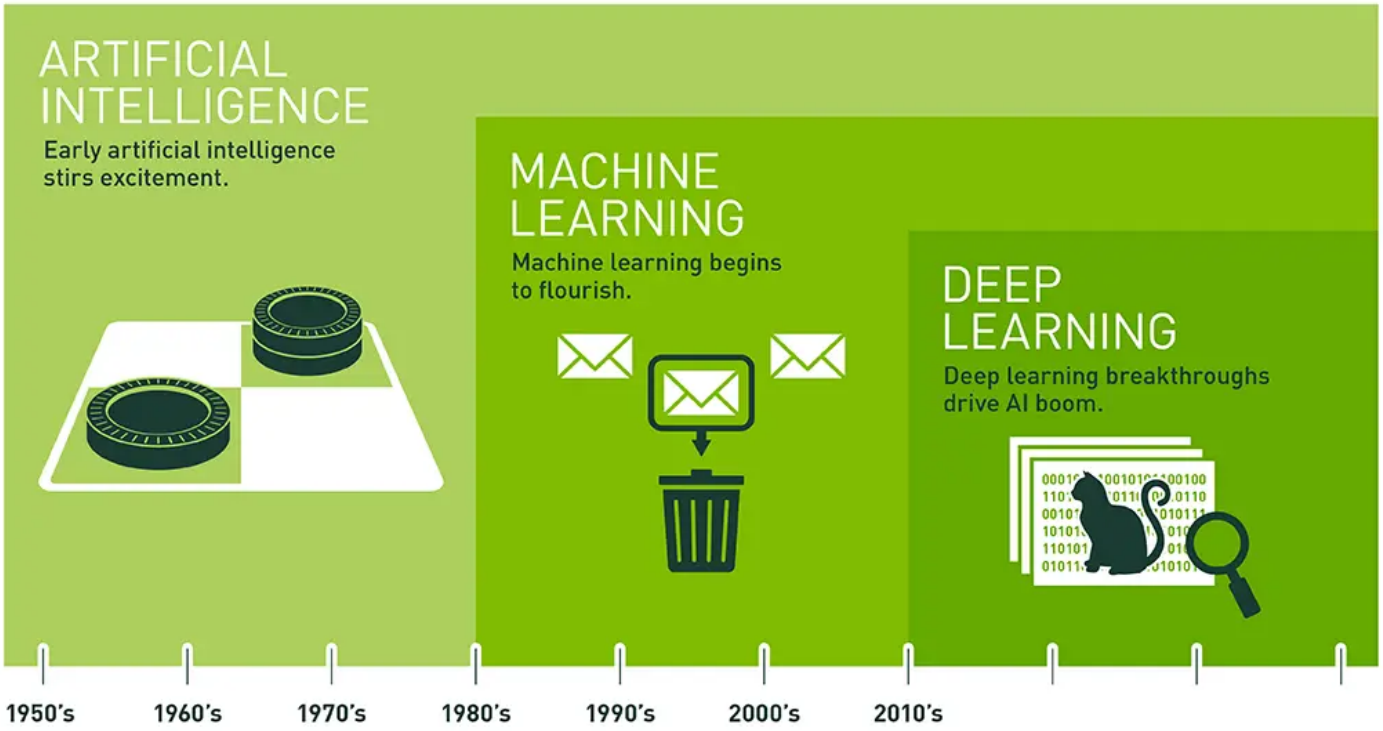
Natrénovaná sít zná všechny váhy w




Algoritmus zpětné propagace chyby
Učení s učitelem
Učení bez učitele

# Hloubkové učení, Strojové učení, Umělá inteligence

"What we want is a machine that can learn from experience." –Alan Turing (20th February 1947, London)




Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

# Hloubkové učení, Strojové učení, Umělá inteligence

1952 vznikla první aplikace schopná strojového učení od Arthura Samuela pro hraní dámy

1966 vznikl první chatbot ELIZA a o rok později Frank Rosenblatt vytvořil první neuronovou síť postavenou na perceptronech

1984 jsou obavy a nejsou peníze ... Terminátor

1997 superpočítač Deep Blue od IBM porazil Garryho Kasparova v šachu

2016 AlphaGo porazil světového šampiona Lee Se-dola ve hře Go

2023 boston dynamics
https://www.youtube.com/watch?v=-e1_QhJ1EhQ

# Co bude předmětem

| | | |
|---|---|---|
| 1 | 18.9. | Úvod do databází, jejich účel a historie |
| 2 | 25.9. | Architektura databází a informatika |
| 3 | 2.10. | Programování - algoritmus, kódy, výroková logika |
| 4 | 9.10. | Základy databází, klíče a kardinalita |
| 5 | 16.10. | ERA model, normalizace |
| 6 | 23.10. | Ukládání dat |
| 7 | 30.10. | Vyhledávání |
| 8 | 6.11. | Kvantové počítače pro práci s daty |
| 9 | 13.11. | SQL Standardizovaný strukturovaný dotazovací jazyk |
| 10 | 20.11. | MySQL, SQL Server, MS Access, pandas phyton |
| 11 | 27.11. | Návrh relační databáze I |
| 12 | 4.12. | Půlené cvičení |
| 13 | 11.12. | Půlené cvičení |
| 14 | 18.12. | Zkouška |

# Co bude předmětem

| | | |
|---|---|---|
| 1 | 18.9. | Úvod do databází, jejich účel a historie |
| 2 | 25.9. | Architektura databází a informatika |
| 3 | 2.10. | Programování - algoritmus, kódy, výroková logika |
| 4 | 9.10. | Základy databází, klíče a kardinalita |
| 5 | 16.10. | ERA model, normalizace |
| 6 | 23.10. | Ukládání dat |
| 7 | 30.10. | Vyhledávání |
| 8 | 6.11. | Kvantové počítače pro práci s daty |
| 9 | 13.11. | SQL + porovnání používaných (MySQL, SQL Server, MS Acce |
| 10 | 20.11. | Návrh relační databáze I + cvičení v BS2 od 8:00 |
| 11 | 27.11. | Návrh relační databáze I + cvičení v BS2 od 8:00 |
| 12 | 4.12. | Opakování |
| 13 | 11.12. | Zkouška |
| 14 | 18.12. | Zkouška |

# Skripta, stránky a materiály

obrazky:
https://www.fi.muni.cz/IB111/sbirka/08-datove_struktury.html
https://www.algoritmy.net/article/21/Binarni-vyhledavani
http://www.ryanhmckenna.com/2015/01/interpolation-search-explained.html
https://sectigostore.com/blog/hash-function-in-cryptography-how-does-it-work/
pagerank https://www.youtube.com/watch?v=meonLcN7LD4
Mechine learning college https://www.youtube.com/watch?v=0Hqz8u2TEcg
Deep learning https://www.youtube.com/watch?v=aircAruvnKk
Nierostek Jakub, Diplomová práce PřFUK 2023 Quantitative structure-activity relationship and machine learning
https://www.linkedin.com/pulse/securitys-greatest-innovation-cryptographic-hash-functions-bhanji
https://www.svethardware.cz/ai-strojove-a-hluboke-uceni-jak-se-vlastne-stroje-uci/59662

# Skripta, stránky a materiály

literatura:

https://www.geeksforgeeks.org/binary-search-tree-data-structure/
https://www.itnetwork.cz/algoritmy/vyhledavani/
ZÁKLADNÍ ALGORITMY. ARNOŠT VEČERKA. Univerzita Olomouc
ALGORITMY VYHLEDÁVÁNÍ V JAZYCE C BAKALÁŘSKÁ PRÁCE Ivan Nejezchleb:
https://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=115726
https://www.itnetwork.cz/navrh/algoritmy/algoritmy-vyhledavani