**Asymptotické paměťové a časové složitosti**

**Asymptotická složitost**

**Co to vlastně je?**

Je to matematická složitost, která se používá k popisu efektivity konkrétních algoritmů nebo výpočetních operací

Způsob klasifikace

* Klasifikace podle 3 znaků
  + BIG O – Nejhorší možnost
  + THETA θ – Průměr
  + OMEGA Ω – nejlepší scénář

Zjednodušení analýzy algoritmů (podrobnější analýza by byla příliš složitá)

Nástroj pro hodnocení

* Jak rychle roste počet operací v závislosti na velikosti vstupních dat

Pří analýze se snažíme najít algoritmus s co nejnižší asymptotickou složitostí – nejrychlejší

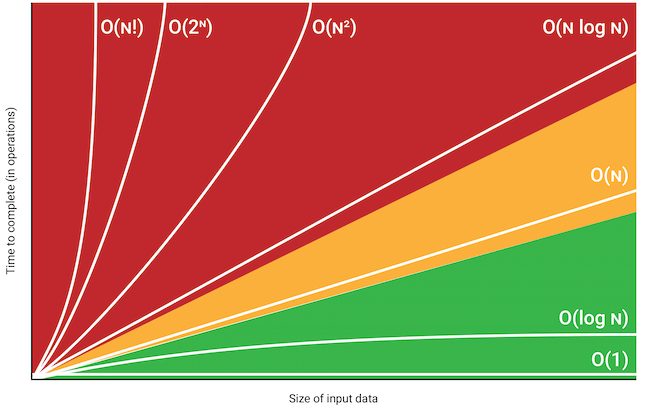
**HORNÍ ODHAD**

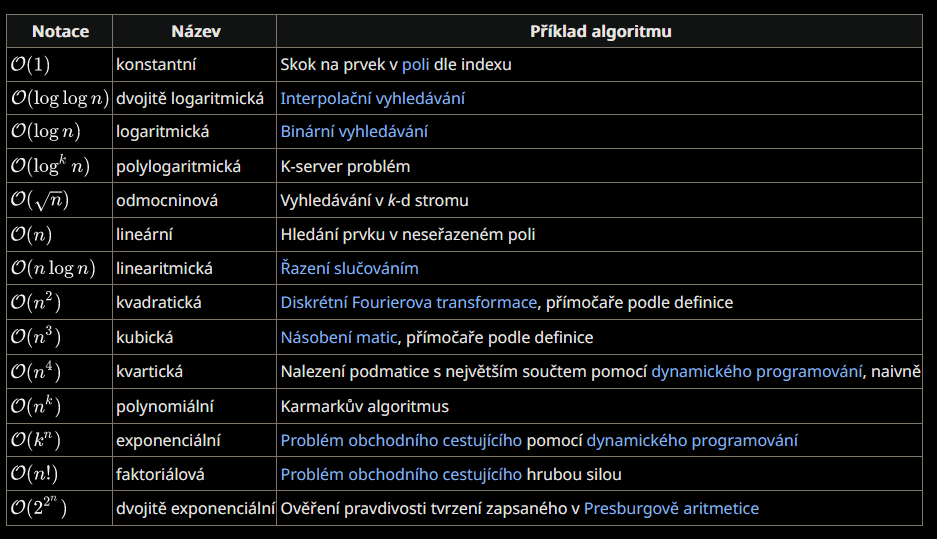
**Určuje nejhorší možnou rychlost růstu dané funkce, když se argument blíží k nekonečnu**

**Určuje maximální množství zdrojů potřebných pro běh algoritmu (paměť)**

Zápis se provádí pomocí “Velké Ó” -> O(f (n))

* f(n) – počet operací
* n – velikost vstupních dat
* O – nejhorší možný odhad





Můžeme analyzovat **ČASOVOU a PAMĚŤOVOU** složitost

**Jak zjistíme složitost?**

Můžeme kód ručně analyzovat

* Poměrně náročný
* Pomalý
* Nevhodný pro větší a složitější programy

Můžeme použít matematické vztahy

* Rychlejší než manuální analýza
  + Pokud ale ovládáme takové schopnosti

Můžeme použít SW nástroje

* Automaticky určí složitost
* Používají vlastní algoritmy a metody

**Měření časové složitosti v PYTHON**

* Tento kód měří námi udělanou funkci
* Funkce přidává do listu N prvků
* Tato funkce je prováděná několikrát, s různým počtem N prvků
  + Každé provedení funkce zaměříme a následně vypíšeme s počtem prvků
  + Tyto hodnoty můžeme přepsat do grafu ze kterého určíme o jakou složitost se jedná
  + V tomto případě je to **lineární složitost O(n) VYHUL SPATNE**

A screen shot of a computer program

Description automatically generated with low confidence

A screen shot of a computer

Description automatically generated with low confidence

**Měření paměťové složitosti v PYTHON**

* Tento kód měří námi udělanou funkci
* **1 Funkce** vytváří list o N prvků
  + Tato funkce je prováděná několikrát, s různým počtem N prvků
    - Každé provedení funkce zaměříme a následně vypíšeme s počtem prvků
    - Tyto hodnoty můžeme přepsat do grafu ze kterého určíme o jakou složitost se jedná
      * V tomto případě je to lineární složitost **O(n)**
* **2 Funkce** počítá logaritmus N o základě 2
  + Tato funkce je prováděná několikrát, s různým počtem N prvků
    - Každé provedení funkce zaměříme a následně vypíšeme s počtem prvků
    - Tyto hodnoty můžeme přepsat do grafu ze kterého určíme o jakou složitost se jedná
      * V tomto případě je to lineární složitost **O(log n)**

**A picture containing text, screenshot, font

Description automatically generated**

A picture containing text, screenshot, font

Description automatically generated

**Jak se měří časová složitost?**

Zaměříme čas průběhu programu v závislosti na počtu vstupních dat.

**A screen shot of a computer program

Description automatically generated with low confidence**

**Tyto složitosti by měli samozřejmě platit pokud předpokládáme, že APPEND je vykonáváno časovou složitostí O(1)**

**Jak se měří paměťová složitost?**

Určení množství paměti, kterou program využívá v závislosti na počtu vstupních dat.