# Programování II pro matematiky

## Teoretické cvičení 02

**10. 3. 2021**

Martin Mareš a Tomáš Karella

# Plán cvičení

- Opakování pojmů
- Hledání kuličky (Příklad 4 ze cvičení 1)
- Časová a prostorová složitost
  - Příklady na určení složitosti
  - Počítání s asymptotickou časovou složitostí

# Opakování - Teorie

- Algoritmus (co my od něj chceme na *p2m*)
  - Skládá se z nějakých elementárních kroků
  - Skončí po konečném množství kroků
  - Vrátí správný výsledek

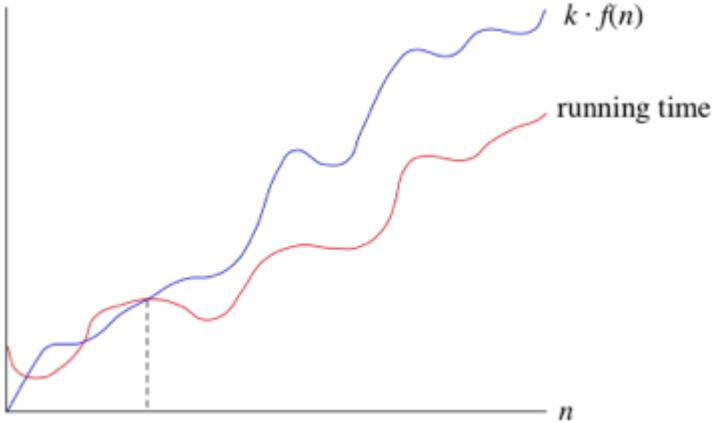
# Opakování - Teorie

- Algoritmus (co my od něj chceme na *p2m*)
  - Skládá se z nějakých elementárních kroků
  - Skončí po konečném množství kroků
  - Vrátí správný výsledek
- Porovnání algoritmů
  - Časová složitost
    - "... počet kroků v závislosti na velikosti vstupu"
    - např. Kolik vážení musíme provést, abychom našli nejmenší kuličku v závislosti na počtu kuliček.
  - Prostorová složitost
    - "... kolik paměti zaberu v závislosti na velikosti vstupu"
    - např. Kolik informací si musíme poznamenat, abychom našli nejmenší kuličku.

# Opakování - složitost

- O notace ("velké O"): Nechť $f, g : \mathrm{N} \to \mathrm{R}$ jsou dvě funkce. Řekneme, že funkce $f(n)$ je třídy $\mathrm{O}(g(n))$, jestliže existuje taková kladná reálná konstanta $c$, že existuje nějaké přirozené $n_0$ takové, že pro všechna $n \geq n_0$ platí $f(n) \leq c \cdot g(n)$.

$$f(n) \in \mathrm{O}(g(n)) \iff \exists c > 0 \; \exists n_o \; \forall n \geq n_0 : f(n) \leq c \cdot g(n)$$

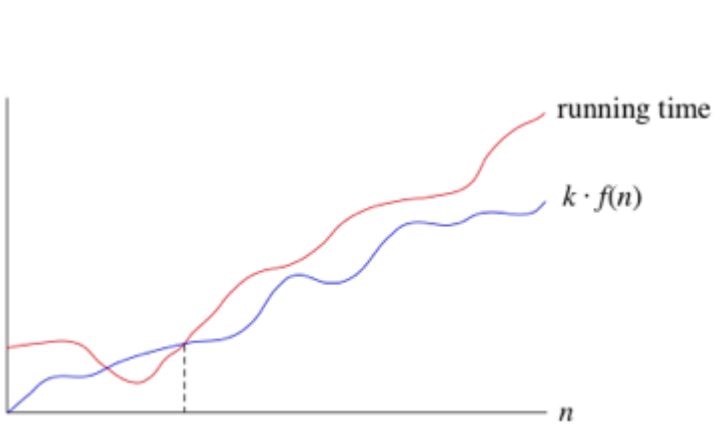


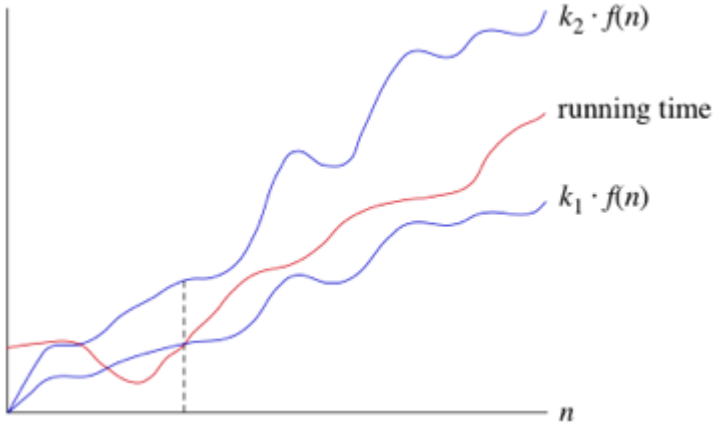
# Opakování - složitost

- $\Omega$ notace ("Omega"): Nechť $f, g : \mathrm{N} \to \mathrm{R}$ jsou dvě funkce. Řekneme, že funkce $f(n)$ je třídy $\Omega(g(n))$, jestliže existuje taková kladná reálná konstanta $c$, že existuje nějaké přirozené $n_0$ takové, že pro všechna $n \geq n_0$ platí $f(n) \geq c \cdot g(n)$.

$$f(n) \in \Omega(g(n)) \iff \exists c > 0 \; \exists n_o \; \forall n \geq n_0 : f(n) \geq c \cdot g(n)$$

- $\Theta$ notace ("Theta"): Řekneme, že funkce $f(n)$ je třídy $\Theta(g(n))$, jestliže $f(n)$ je jak třídy $\mathrm{O}(g(n))$, tak třídy $\Omega(g(n))$.



# Příklad 4 ze cvičení 1

## Hledání kuličky

Nepřítel nám dal 9 opticky stejných kovových kuliček a rovnoramenné váhy a prozradil nám, že jedna z kuliček je nepatrně těžší, než ostatní.

- Kolikrát musíme použít váhy, abychom tuto kuličku identifikovali?
- Jaká je odpověď, je-li kuliček 8? Jaká je odpověď, je-li kuliček 10?
- Jaká je odpověď, může-li být odlišná kulička i lehčí? (příklad s *)

# Rozbor hvězdiček

Najděte funkci $f(n)$, která určuje počet hvězdiček v závislosti na $n$, najděte funkci $\Theta$, do které tato funkce patří.

*Na vstupu máme číslo n*

**Algoritmus 1**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , n opakujeme:
3.       Vytiskneme *.
```

**Algoritmus 2**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , i opakujeme:
3.       Vytiskneme *.
```

**Algoritmus 1**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , n opakujeme:
3.       Vytiskneme *.
```

**Algoritmus 2**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , i opakujeme:
3.       Vytiskneme *.
```

**Algoritmus 3**

```
1. Dokud n ≥ 1, opakujeme:
2.    Vytiskneme *.
3.    n ← Dolní část z n/2.
```

**Algoritmus 4** (příklad s *)

```
1. Dokud je n > 0, opakujeme:
2.    Je-li n liché:
3.       Pro i = 1, . . . , n opakujeme:
4.          Vytiskneme *.
5.    n ← Dolní část z n/2.
```

**Algoritmus 1**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , n opakujeme:
3.       Vytiskneme *.
```

**Algoritmus 1**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , n opakujeme:
3.       Vytiskneme *.
```

- vypíšeme $n^2$ hvězdiček
- cca $4n^2$ kroků (změna `i` a `j` , podmínky a vypsání)

**Algoritmus 1**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , n opakujeme:
3.       Vytiskneme *.
```

- vypíšeme $n^2$ hvězdiček
- cca $4n^2$ kroků (změna `i` a `j` , podmínky a vypsání)
- $4n^2 \in \Theta(n^2)$

**Algoritmus 1**

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , n opakujeme:
3.       Vytiskneme *.
```

- vypíšeme $n^2$ hvězdiček
- cca $4n^2$ kroků (změna `i` a `j` , podmínky a vypsání)
- $4n^2 \in \Theta(n^2)$
- Časová složitost $\Theta(n^2)$
- Prostorová složitost $\Theta(1)$

**Algoritmus 2** kroků

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , i opakujeme:
3.       Vytiskneme *.
```

- vypíšeme $\frac{n(n+1)}{2}$ hvězdiček
  - $\frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n$
- cca $2n^2 + 2n$ kroků

**Algoritmus 2** kroků

```
1. Pro i = 1, . . . , n opakujeme:
2.    Pro j = 1, . . . , i opakujeme:
3.       Vytiskneme *.
```

- vypíšeme $\frac{n(n+1)}{2}$ hvězdiček
  - $\frac{n(n+1)}{2} = \frac{1}{2}n^2 + \frac{1}{2}n$
- cca $2n^2 + 2n$ kroků
- $(\frac{1}{2}n^2 + \frac{1}{2}n) \in \Theta(n^2)$
- Časová složitost $\Theta(n^2)$ a prostorová složitost $\mathrm{O}(1)$

**Algoritmus 3**

```
1. Dokud n ≥ 1, opakujeme:
2.    Vytiskneme *.
3.    n ← Dolní část z n/2.
```

**Algoritmus 3**

```
1. Dokud n ≥ 1, opakujeme:
2.    Vytiskneme *.
3.    n ← Dolní část z n/2.
```

- v každém kroku se $n$ sníží na $\lfloor n/2 \rfloor$
  - po $k$ krocích je $n_k = \lfloor n/2^k \rfloor$
  - hledáme $l$ takové $1 = \lfloor n/2^l \rfloor$
  - $l = \log_2(n)$
- vypíšeme $\log_2(n)$ hvězdiček
- cca $4 \log_2(n)$ kroků

**Algoritmus 3**

```
1. Dokud n ≥ 1, opakujeme:
2.    Vytiskneme *.
3.    n ← Dolní část z n/2.
```

- v každém kroku se $n$ sníží na $\lfloor n/2 \rfloor$
  - po $k$ krocích je $n_k = \lfloor n/2^k \rfloor$
  - hledáme $l$ takové $1 = \lfloor n/2^l \rfloor$
  - $l = \log_2(n)$
- vypíšeme $\log_2(n)$ hvězdiček
- cca $4 \log_2(n)$ kroků
- $\log_2(n) \in \Theta(\log_2(n))$
- Časová složitost $\Theta(\log_2(n))$ a prostorová složitost $\mathrm{O}(1)$

**Algoritmus 4** (příklad s *)

```
1. Dokud je n > 0, opakujeme:
2.    Je-li n liché:
3.       Pro i = 1, . . . , n opakujeme:
4.          Vytiskneme *.
5.    n ← Dolní část z n/2.
```

- lze dokázat, že počet hvězdiček nebude vyšší než $2n$
- cca $8n$ kroků

**Algoritmus 4** (příklad s *)

```
1. Dokud je n > 0, opakujeme:
2.    Je-li n liché:
3.       Pro i = 1, . . . , n opakujeme:
4.          Vytiskneme *.
5.    n ← Dolní část z n/2.
```

- lze dokázat, že počet hvězdiček nebude vyšší než $2n$
- cca $8n$ kroků
- $8n \in \Theta(n)$

Příklady vychází z Průvodce labyrintem algoritmů, str 43

# Počítání

Dokažte nebo vyvraťte následující tvrzení:

1. Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \mathrm{O}(f(n))$
2. Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = O(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) + f_2(n) = \mathrm{O}(g(n))$
3. Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = \mathrm{O}(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) * f_2(n) = \mathrm{O}(g(n))$
4. Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $2^{f(n)} = \mathrm{O}(2^{g(n)})$
5. Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \Omega(f(n))$
6. Pro každou funkci $f : \mathrm{N} \to \mathrm{R}^+$ platí: $f(n) = \mathrm{O}((f(n))^2)$

**1.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \mathrm{O}(f(n))$

Neplatí, $n \in \mathrm{O}(n^2)$, ale $n^2 \notin \mathrm{O}(n)$

**1.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \mathrm{O}(f(n))$

Neplatí, $n \in \mathrm{O}(n^2)$, ale $n^2 \notin \mathrm{O}(n)$

**2.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = O(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) + f_2(n) = \mathrm{O}(g(n))$

**1.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \mathrm{O}(f(n))$

Neplatí, $n \in \mathrm{O}(n^2)$, ale $n^2 \notin \mathrm{O}(n)$

**2.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = O(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) + f_2(n) = \mathrm{O}(g(n))$

Platí, dosadím do definice: $f_1(n) \geq c_1 \cdot g(n) \wedge f_2(n) \geq c_2 \cdot g(n) \implies f_1(n) + f_2(n) \geq (c_1 + c_2) \cdot g(n)$

**1.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \mathrm{O}(f(n))$

Neplatí, $n \in \mathrm{O}(n^2)$, ale $n^2 \notin \mathrm{O}(n)$

**2.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = O(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) + f_2(n) = \mathrm{O}(g(n))$

Platí, dosadím do definice: $f_1(n) \geq c_1 \cdot g(n) \wedge f_2(n) \geq c_2 \cdot g(n) \implies f_1(n) + f_2(n) \geq (c_1 + c_2) \cdot g(n)$

**3.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = \mathrm{O}(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) * f_2(n) = \mathrm{O}(g(n))$

Neplatí, $n^2 \in \mathrm{O}(n^2)$ a $n \in \mathrm{O}(n^2)$, ale $n^3 \notin \mathrm{O}(n^2)$

**1.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \mathrm{O}(f(n))$

Neplatí, $n \in \mathrm{O}(n^2)$, ale $n^2 \notin \mathrm{O}(n)$

**2.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = O(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) + f_2(n) = \mathrm{O}(g(n))$

Platí, dosadím do definice: $f_1(n) \geq c_1 \cdot g(n) \wedge f_2(n) \geq c_2 \cdot g(n) \implies f_1(n) + f_2(n) \geq (c_1 + c_2) \cdot g(n)$

**3.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = \mathrm{O}(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) * f_2(n) = \mathrm{O}(g(n))$

Neplatí, $n^2 \in \mathrm{O}(n^2)$ a $n \in \mathrm{O}(n^2)$, ale $n^3 \notin \mathrm{O}(n^2)$

**4.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $2^{f(n)} = \mathrm{O}(2^{g(n)})$

**1.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \mathrm{O}(f(n))$

Neplatí, $n \in \mathrm{O}(n^2)$, ale $n^2 \notin \mathrm{O}(n)$

**2.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = O(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) + f_2(n) = \mathrm{O}(g(n))$

Platí, dosadím do definice: $f_1(n) \geq c_1 \cdot g(n) \wedge f_2(n) \geq c_2 \cdot g(n) \implies f_1(n) + f_2(n) \geq (c_1 + c_2) \cdot g(n)$

**3.** Pro každou trojici funkcí $f_1, f_2, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f_1(n) = \mathrm{O}(g(n))$ a zároveň $f_2(n) = \mathrm{O}(g(n))$, pak $f_1(n) * f_2(n) = \mathrm{O}(g(n))$

Neplatí, $n^2 \in \mathrm{O}(n^2)$ a $n \in \mathrm{O}(n^2)$, ale $n^3 \notin \mathrm{O}(n^2)$

**4.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $2^{f(n)} = \mathrm{O}(2^{g(n)})$

Neplatí, uvažme $f(n) = 2 \log_2 n \wedge g(n) = \log_2 n$, potom platí, že $f(n) = \mathrm{O}(g(n))$, ale $2^{f(n)} = 2^{2 \log_2 n} = (2^{\log_2 n})^2 = n^2$ a $2^{g(n)} = 2^{\log_2 n} = n$ a ty nesplňují $2^{f(n)} = \mathrm{O}(2^{g(n)})$.
Pozn. $a^{xy} = (a^x)^y$

**5.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \Omega(f(n))$

**5.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \Omega(f(n))$

Platí, dosadím do definice: $f(n) \leq c \cdot g(n) \wedge c > 0 \implies \frac{1}{c} f(n) \leq g(n) \implies g(n) \geq \frac{1}{c} f(n)$

**5.** Pro každou dvojici funkcí $f, g : \mathrm{N} \to \mathrm{R}^+$ platí: pokud $f(n) = \mathrm{O}(g(n))$, pak $g(n) = \Omega(f(n))$

Platí, dosadím do definice: $f(n) \leq c \cdot g(n) \wedge c > 0 \implies \frac{1}{c} f(n) \leq g(n) \implies g(n) \geq \frac{1}{c} f(n)$

**6.** Pro každou funkci $f : \mathrm{N} \to \mathrm{R}^+$ platí: $f(n) = \mathrm{O}((f(n))^2)$

Neplatí, pro $f(n) = 1/n$ dostaneme $\frac{1}{n} \notin \mathrm{O}(\frac{1}{n^2})$ (protože $1 \leq c \cdot \frac{1}{n}$)

# Příklady na příště / na konec

1. Posloupnost N čísel – maximum a počet jeho výskytů
2. Posloupnost N čísel – maximální číslo bez opakování
3. Posloupnost N čísel – číslo s maximálním počtem výskytů

# Zdroje:

- Průvodce labyrintem algoritmů, Kapitola 2.4
- Khan Academy, Asymptotic notation