Polynom

Polynomem $p(x)$ stupně n nazýváme výraz tvaru

$$p(x) = \sum_{i=0}^{n} a_i x^i = a_0 + a_1 \cdot x + \ldots + a_n \cdot x^n$$

a čísla $a_i \in \mathcal{R},\ i = 1, \ldots n$ se nazývají koeficienty.

Napište funkci, která vyhodnotí polynom zadaný n-ticí koeficientů v bodě x, tj.

```
# polynom(x, 3, 2, 1) vyhodnoti 3 + 2x + x^2
polynom(1, 3, 2, 1)
>> 6
```

Je vhodné pro zápis polynomu použít tzv. Hornerovo schéma (ušetří mnoho operací a snáze se implementuje). Pro polynom stupně 3 vypadá takto:

$$a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 = a_0 + x \cdot (a_1 + x \cdot (a_2 + a_3 \cdot x))$$

Faktoriál

V kombinatorice se často užívá symbolu $n!$, čteme n-faktoriál. Je definován takto:

$$n! = \prod_{i=1}^{n} i = n \cdot (n-1) \cdot \cdots \cdot 2 \cdot 1$$

Např.

$$\begin{aligned} 2! &= 2 \\ 3! &= 6 \\ 4! &= 24 \end{aligned}$$

Dodatečně se obvykle definuje $0! = 1$.

Naprogramujte: - funkci, která spočítá faktoriál čísla n - funkci, která spočítá faktoriál čísla n rekurzivně (tj. funkce volá sama sebe) - porovnejte časy, které jednotlivé implementace potřebují pro vyšší n, např $n = 20$. Pro měření času můžete využít následující funkci - porovnejte s implementacemi dostupnými v Pythonu: `math.factorial` a `numpy.factorial` - Změřte pro různé implentace závislot výpočetního času na n. Srovnejte v grafu (např. pomocí `matplotlib.pyplot.plot`)

Pro měření času můžete použít např.:

```python
def measure_time(n, fun):  # pouze Jupyter
    time = %timeit -q -n 100000 -o fun(n)
    return sum(time.timings) / len(time.timings)

def measure_time2(n, fun): # kdekoliv
    from time import time
    start = time()
    fun(n)
    end = time()
    return end-start
```