Násobení polynomů

Uvažujte polynomy, reprezentované jako seznamy koeficientů bez vedoucích nul. Například polynom

$$f = x^5 + 5x^3 + 2x^2 + x + 1$$

by byl reprezentován jako seznam

$$\texttt{f} = [\texttt{1}, \texttt{1}, \texttt{2}, \texttt{5}, \texttt{0}, \texttt{1}].$$

Navrhněte algoritmus (zapište ho v rozumném pseudokódu nebo Pythonu), jehož vstupem jsou dva takto reprezentované polynomy f a g a jeho výstupem je jejich součin $f(x) \cdot g(x)$ ve stejné reprezentaci.

Odhadněte časovou složitost svého algoritmu pomocí notace O (za konstatní operaci můžete považovat sčítání, násobení, porovnávání a přístup do pole pomocí indexu).

Řešení. Využijeme klasického vzorce pro násobení polynomů, podle nějž platí

$$c_k = \sum_{i+j=k} a_i \cdot b_j = \sum_{i=0}^{k} a_i \cdot b_{k-i}$$

kde a_i, b_i jsou koeficienty původních polynomů a c_i jsou koeficienty součinu. Musíme si jen dát pozor na přetečení indexů.

Zde je jeden způsob, jak toto násobení v Pythonu implementovat (používá nějaké pythonovské triky pro práci se seznamy).

```
def nasob(f,g):
    df = stupen(f)
    dg = stupen(g)
    d = df + dg
    soucin = [
        sum([f[j]*g[i-j]
                 for j in range(max(0, i - dg),
                                min(i + 1,df + 1))])
        for i in range(d + 1)]
    return soucin
```

Složitost tohoto algoritmu je $O(m \cdot n)$, kde m a n jsou stupně vstupních polynomů. Určíme ji tak, že nahlédneme, že prvních několik řádků proběhne jedinkrát a jejich složitost je konstantní. V části s for-cykly si všimneme, že se každá dvojice koeficientů zpracuje právě jednou, všech takových dvojic je právě $m \cdot n$.