

KSI 2014/2015

Úloha 1-5: Složité výpočty

Jan Horáček

Gymnázium, Brno, Vídeňská 47; jan.horacek@seznam.cz

2. listopadu 2014

1 *karlikuvP1(n)*

Algoritmus nápadně podobný naivní metodě výpočtu n -tého Fibonacciho čísla s sebou nese i časovou složitost tohoto algoritmu - $O(n^2)$.

Tato hodnota časové složitosti plyne z toho, že v každém volání funkce *karlikuvP1(n)* je tato funkce zavolána dvakrát. Funkce je tedy celkem zavolána 2^n krát.

2 *karlikuvP2(n)*

Stejným prvkem pro posouzení asymptotické časové složitosti je cyklus `for`. Tento cyklus proběhne celkem $n - 1$ krát, z čehož logicky vyplývá lineární časová složitost $O(n - 1)$.

Pokud uvážíme, že v každé iteraci cyklu nastanou 3 operace (což je značné zjednodušení), časová složitost tohoto algoritmu je $O(3(n - 1))$ — opět složitost lineární.

3 *karlikuvP3(n)*

Tato funkce se od předchozí liší horní mezí `for` cyklu, která je $n - n$. Jak už tiše tušíme, $n - n = 0$ a tudíž `for` cyklus nikdy neproběhne. Časová složitost této funkce je tedy konstantní: $O(1)$.

4 *karlikuvP4(n)*

Pakliže funkce *karlikuvP4(n)* vždy volá sama sebe s poloviční hodnotou argumentu, dokud argument nenabude hodnoty 1 (hodnoty 0 nemůže nabýt, protože se volání zastaví už při hodnotě 1), je asymptotická časová složitost $O(\log_2(n))$ vzhledem k počtu volání funkcí a hodnotě n .

Hezká ukázka toho, proč je časová složitost právě logaritmická, plyne například z počtu operací bitového posuvu z čísla n na číslo 1, když se posouváme vždy o jeden bit (vzhledem k číslu n).

5 *karlikuvP5(n)*

Ať už algoritmus dělá (resp. měl dělat) sebeobskurnější výpočet, je zřejmé, že se skládá ze dvou for cyklů. Oba dva cykly mají lineární časovou složitost, první cyklus $O(n-1)$, druhý $O(\frac{n-1}{2})$ když uvažíme, že nás zajímá pouze počet operací *vysledek.append(mezivysledek[i])*, nikoliv počet ověření podmínky *if i % 2 == 0*..

Celková asymptotická časová složitost je tedy součtem těchto dvou složitostí: $O\left(\frac{3(n-1)}{2}\right)$ — lineární.