4 Amortizovaná složitost, složitost algoritmů

4.1 Řešení příkladů z domácí přípravy.

1. Je dána funkce $T(n)$ je dána na přirozených číslech rekurentním vztahem. Vyřešte její asymptotické chování. Řešení zdůvodněte.
   (a) $T(n) = 2\,T(\frac{n}{3}) + 1$, $T(1) = 1$.
   (b) $T(n) = 4\,T(\frac{n}{3}) + n^{\frac{3}{2}} \log_2 n$, $T(1) = 1$.
2. Je dána funkce $T(n)$ je dána na přirozených číslech rekurentním vztahem. Vyřešte její asymptotické chování. Řešení zdůvodněte.
   (a) $T(n) = T(n-1) + c^n$, $T(1) = 1$, a kde $c > 1$ je konstanta z $\mathbb{R}^+$.

4.2 Spočítejte amortizovanou složitost funkce $pridej(x)$, která do pole přidá prvek x. Funkce se realizuje takto: Začínáme z jednoprvkového pole, v okamžiku, kdy už je pole zaplněné, pole zdvojnásobíme, původní pole do něj překopírujeme a pak teprve přidáme x.

4.3 Násobení dlouhých čísel: Mějme dvě binární čísla a, b a chceme spočítat jejich součin. Diskutujte časové složitosti a správnost následujících dvou algoritmů:

Algoritmus 1 (naivní): Předpokládejme, že součet dvou čísel trvá konstantní čas.

```
mezivysledek := 0
 for  (i=1; i< a+1;i++) do
    mezivysledek := mezivysledek + b
return mezivysledek
```

Algoritmus 2 (rekurzivní): Libovolné $2N$-ciferné číslo můžeme zapsat jako $2^N\,A + B$, kde A a B jsou N-ciferná. Součin dvou takových čísel pak bude

$$(2^N \cdot A + B) \cdot (2^N \cdot C + D) = (2^{2N} \cdot AC + 2^N\,(AD + BC) + BD).$$

Předpokládejme, že sčítání proběhne v konstantním čase, stejně jako násobení mocninou 2. N-ciferná čísla budeme násobit rekurzivním zavoláním téhož algoritmu.

4.4 Určete, co bude rozhodujícím parametrem pro určení časové a paměťové složitosti, je-li na vstupu:

1. posloupnost prvků $a_1, a_2, \ldots a_n$,
2. graf o n vrcholech a m hranách,
3. matice o rozměrech $n \times m$,
4. číslo x, jehož hodnota je podstatná pro celkovou délku výpočtu (například test prvočíselnosti).

4.5 Jsou dána dvě reálná čísla $a, b > 0$ a je dán pseudokód

```
while a > 0 do
    if a < b then
        (a, b) := (2a, b − a)
    else
        (a, b) = (a − b, 2b)
end while
return b
```

Jedná se o pseudokód algoritmu? Odpověď pečlivě zdůvodněte. Jestliže ano, spočítejte časové nároky algoritmu.

4.6 Je dán následující algoritmus

```
i = N;
while (i > 0) do {
    j = 0;
    while (j^2 < i) do {write(⋆); j + + }
    i = i − 2 }
```

Spočítejte časové nároky algoritmu, tj najděte nejjednodušší funkci $f(N)$ takovou, že časové nároky jsou v třídě $\Theta(f(N))$. (Samotná funkce nestačí, musíte zdůvodnit, proč je správně.)

Samostaná práce na příští cvičení

4.7 Je dán následující algoritmus

```
i = N
while (i > 0) do {
    j = 0;
    i := ⌊i/3⌋;
    while (j < i) do write(⋆); j := j + 1 }
```

Spočítejte časové nároky algoritmu, tj najděte nejjednodušší funkci $f(N)$ takovou, že časové nároky jsou v třídě $\Theta(f(N))$. (Samotná funkce nestačí, musíte zdůvodnit, proč je správně.)