Za domácí úkol máte příklady 2 (jen druhá otázka) a 3. Důkladně si přečtěte zadání!

Definice 1. (a, b)-strom pro parametry $a \geq 2$, $b \geq 2a - 1$ je obecný vyhledávací strom, pro který navíc platí:

- Kořen má 2 až b synů, ostatní vnitřní vrcholy a až b synů.
- Všechny vnější vrcholy jsou ve stejné hloubce.

Definice 2. LLRB (červeno-černé, left-leaning red-black) strom je binární vyhledávací strom s vnějšími vrcholy, jehož hrany jsou obarveny červeně a černě. Přitom platí následující axiomy:

1. Neexistují dvě červené hrany bezprostředně nad sebou.
2. Jestliže z vrcholu vede dolů jediná červená hrana, pak vede doleva.
3. Hrany do listů jsou vždy obarveny černě. (To se hodí, jelikož listy jsou pouze virtuální, takže do nich neumíme barvu hrany uložit.)
4. Na všech cestách z kořene do listu leží stejný počet černých hran.

Uvažme rekurzivní algoritmus, který vstup rozloží na a podproblémů velikosti n/b a z jejich výsledků složí celkovou odpověď v čase $\Theta(n^c)$, přičemž případná zaokrouhlení zanedbejme. Příslušná rekurentní formule je $T(n) = aT(n/b) + \Theta(n^c)$ a předpokládejme, že $T(1) = 1$. Řešení této rekurentní formule závisí na poměru $\frac{a}{b^c}$:

- Jestliže $a < b^c$, pak $T(n) = \Theta(n^c)$.
- Jestliže $a = b^c$, pak $T(n) = \Theta(n^c \log n)$.
- Jestliže $a > b^c$, pak $T(n) = \Theta(n^{\log_b(a)})$.

Příklad 1. Složitost následujících rekurzivních algoritmů popište rekurentní formulí a tu vyřešte přímo i pomocí kuchařky (master theorem). U stromů předpokládejme, že jsou dokonale vyvážené.

1. Hledání v poli pomocí binárního půlení, hledání v binárním stromě.
2. Konstrukce binárního stromu ze setříděného pole, výpis vyhledávacího binárního stromu v setříděném pořadí.
3. Hledání mediánu v lineárním čase (pomocí pětic).
4. Mergesort.
5. k-cestý mergesort: Pole nedělíme na dvě části, ale na k částí.
6. Násobení dlouhých čísel.
7. Strassenův algoritmus na násobení matic.
8. Počet sčítání při výpočtu n-tého Fibonacciho čísla (hloupě rekurzí i sekvenčně).

Příklad 2. Mějme posloupnost matic $A_1, \ldots, A_n$, kterou si chceme uložit tak, abychom rychle uměli odpovídat následující dotazy: Pro dané indexy i a j urči součin matic $A_i \cdots A_j$. Součin matic pomalá operace, takže chceme ukládat pomocné informace takové, že k vyhodnocení dotazu potřebujeme co nejmenší počet součinů matic. Mohli bychom si pamatovat součin matic $A_i \cdots A_j$ pro všechny indexy $i \leq j$, ale takových dvojic je $\binom{n+1}{2}$ a tolik paměti nemáme. Kolik paměti potřebujete, abyste dokázali dotaz vyhodnotit jen na jeden součin dvou matic? Kolik součinů potřebujete, pokud smíte použít jen $\mathcal{O}(n)$ paměti? Můžete předpokládat, že jednu matici lze uložit v $\mathcal{O}(1)$ paměti. Nezapomeňte, že násobení matic je sice asociativní, ale není komutativní ani nelze matice dělit.

Příklad 3. Máme hodně dlouhý kabel, z jehož obou konců vede n drátů. Jak zjistit, které dvojice konců drátů k sobě odpovídají, chceme-li se co nejméně naběhat?

Příklad 4. Na stole leží n šroubků a n odpovídajících matiček různých velikostí. Umíme pro dvojici (šroubeček, matička) rozlišit stavy "pasují", "šroub moc velký", "šroub moc malý". Vymyslete, jak šrouby s matičkami co nejrychleji spárovat.