1. *Rekurence: Vyřešte následující rekurence (vždy $T(1) = 1$):*

   - $T(n) = T(n/2) + \Theta(1)$ *(binární vyhledávání)*
   - $T(n) = 2T(n/2) + \Theta(n)$ *(merge sort)*
   - $T(n) = 3T(n/2) + \Theta(n)$ *(Karacubův algoritmus)*
   - $T(n) = 8T(n/2) + \Theta(n^2)$ *(triviální násobení matic)*
   - $T(n) = 7T(n/2) + \Theta(n^2)$ *(Strassenův algoritmus)*
   - $T(n) = 8T(n/2) + \Theta(n^3)$
   - $T(n) = 7T(n/2) + \Theta(n^3)$
   - $T(n) = 9T(n/2) + \Theta(n^3)$

   Aplikací kuchařkové věty nebo rozborem stromu rekurze dostaneme:

   - $T(n) = \Theta(\log n)$
   - $T(n) = \Theta(n \log n)$
   - $T(n) = \Theta(n^{\log_2 3})$
   - $T(n) = \Theta(n^{\log_2 8}) = \Theta(n^3)$
   - $T(n) = \Theta(n^{\log_2 7})$
   - $T(n) = \Theta(n^3 \log n)$
   - $T(n) = \Theta(n^3)$
   - $T(n) = \Theta(n^{\log_2 9})$

2. *Multimerge: Popište třídicí algoritmus, který bude vstup rozkládat na více než dvě části a ty pak rekurzivně třídit. Může být rychlejší než náš Mergesort?*

   Dělíme posloupnost na d částí velikosti n/d, které po návratu z rekurze slijeme do jediné posloupnosti. Pokud považujeme d za konstantu, dostáváme rekurenci $T(n) = d \cdot T(n/d) + \Theta(n)$, která vede na $T(n) = \Theta(n \log_2 n)$.

   Pokud ale d považujeme za parametr, bude záviset na provedení slévání. První možností je udržovat si d ukazatelů do setříděných posloupností a v čase $O(d)$ vždy vybrat nejmenší následující prvek. Potom dostáváme rekurenci $T_d(n) = dT_d(n/d) + \Theta(dn)$, která povede na $T_d(n) = \Theta(dn \log_d n)$. Pokud bychom ale při slévání použili vyvážený BVS na uchování těchto d ukazatelů, dostaneme $T_d(n) = dT_d(n/d) + \Theta(n \log_2 d)$ což povede na řešení $T_d(n) = \Theta(n \log_2 d \log_d n) = \Theta(n \log_2 n)$. Tedy pro libovolné d takto dostaneme asymptoticky stejně rychlé řešení.

3. *Karacuba pozorněji: Pozornému studentovi jistě neuniklo, že se v rozboru časové složitosti Karacubova algoritmu skrývá drobná chybička: čísla $A + B$ a $C + D$ mohou mít více než $n/2$ cifer, konkrétně $\lceil n/2 \rceil + 1$. Ukažte, že to časovou složitost algoritmu neovlivní.*

   Uvažme strom rekurze pro rekurenci $T(n) = 3T(n/2+1)+\Theta(n)$ namísto „ideální“ rekurence $T(n) = 3T(n/2) + \Theta(n)$. Změní se velikost podproblému na k-té vrstvě – namísto původní velikosti $n/2^k$ máme podproblém velikosti $n/2^k + 1/2^{k-1} + 1/2^{k-2} + \cdots + 1$ (přímočaře získáme dosazením do rekurence). Geometrická řada $1/2^{k-1}+1/2^{k-2}+\cdots+1$ se vždy sečte na hodnotu menší než 2. Tedy na $\log_2 n$-té vrstvě místo problémů velikosti 1 dostaneme problémy velikosti menší než 3. Zbytek argumentace zůstane shodný.

4. *Karacubův prostor: Dokažte, že Karacubův násobící algoritmus má lineární prostorovou složitost.* V libovolnou chvíli běhu algoritmu je v každé hladině prováděno nejvýše jedno volání – aktuální zásobník volání obsahuje nějakou cestu z kořene. Každé prováděné volání používá paměť lineární s velikostí vstupu a tedy dostáváme rekurenci $P(n) = P(n/2) + \Theta(n)$, jejímž řešením je $P(n) = \Theta(n)$.

Zbylé příklady si schováváme na příště.