1 Navrhovanie algoritmu

1. Napíš pseudokód pre procedúru `Merge(A, p, q, r)`.
2. Vypočítaj časovú zložitosť
$$T(n) = \begin{cases} 2 & n = 1 \\ 2T(\frac{n}{2}) + n & \text{inak} \end{cases}$$
3. Popíš algoritmus s časom výpočtu $\Theta(n \log n)$ ktorý pre daných n reálnych čísiel a číslo x zistí, či existuje dvojica čísliel z danej postupnosti, ktorej súčet je x.
4. Selection sort: napíš pseudokód, analýzu...
5. Zváž problém zistiť opakovanie v $< x_1, \ldots, x_n >$ v čase $\Theta(n \lg n)$.
6. Nahraď v algoritme Insert sort úplné prehľadávanie v utriedenej postupnosti binárnym vyhľadávaním. Môžeme potom dokázať výpočtu $\Theta(n \lg n)$?
7. Pre aké najmenšie n platí $100n^2 = 2^n$.

2 Rast funkcií

1. Nájdi $f(n)$ a $g(n)$ tak, aby neplatil žiaden zo vzťahov $\Theta, O, o, \Omega, \omega$ ($f(n)$ a $g(n)$ sú asymptoticky nezáporné funkcie)
2. Použitím Θ dokáž, že $max(f(n), g(n)) = \Theta(f(n) + g(n))$
3. Ukáž, že pre $a, b \in \mathbb{R}; b > 0$ platí $(n + a)^b = \Theta(n^b)$.
4. Platí
$$2^{n+1} = O(2^n)$$
$$2^{2n} = o(2^n)$$

3 Polynómy

1. Použi definíciu O-notácie na dôkaz:
$$T(n) = n^{O(1)} \Leftrightarrow T(n) = O(n^k) \text{ pre } k > 0$$
2. Dokáž $a^{\log_b n} = n^{\log_b a}$
3. Dokáž, že $\lg(n!) = \Theta(n \lg n)$ a $n! = o(n^n)$
4. Ktorá z funkcií je asymptoticky väčšia: $\lg(\lg^* n)$ alebo $\lg^*(\lg n)$

4 Rekurencie

1. Urči dobrú hornú asymptotickú hranicu pre rekurenciu $T(n) = 2T(\lfloor \frac{n}{2} \rfloor) + n$ iteráciou.
2. Nakresli rekurzívny strom pre $T(n) = 4T(\lfloor \frac{n}{2} \rfloor) + n$ a nájdi tesnú asymptotickú hranicu.
3. Použi rekurzívny strom na riešenie rekurencie $T(n) = T(\alpha n) + T((1-\alpha)n)$ kde α je konštanta $0 < \alpha < 1$.
4. Dokáž, že riešenie $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$ je $O(\lg n)$.
5. Dokáž, že riešenie $T(n) = 2T(\lfloor \frac{n}{2} \rfloor) + n$ je $\Omega(n \lg n)$.
6. Dokáž, že riešenie $T(n) = 2T(\lfloor \frac{n}{2} \rfloor + 17) + n$ je $O(n \lg n)$.
7. Použi Master Theorem na nasledovné rekurencie:
   (a) $T(n) = 4T(\frac{n}{2}) + n$
   (b) $T(n) = 4T(\frac{n}{2}) + n^2$
   (c) $T(n) = 4T(\frac{n}{2}) + n^3$
8. Čas výpočtu algoritmu A je popísaný rekurenciou
   $$T(n) = 7T(\frac{n}{2}) + n^2$$
   Konkurenčný algoritmus A' má čas výpočtu
   $$T'(n) = aT'(\frac{n}{4}) + n^2$$
   Aká je najväčšia celočíselná hodnota a taká, aby A' bol asymptoticky rýchlejší ako A?

5 Heapsort

1. Dokáž: výška je najdlhšia jednoduchá cesta od koreňa po nejaký list.
2. Ukáž, že nanajvýš $\lceil \frac{n}{2^{h+1}} \rceil$ uzlov výšky h je v ľubovoľnej n-prvkovej halde.
3. Aký je minimálny a maximálny počet prvkov haldy o výške h?
4. Ilustruj operáciu `Build-Heap` na poli $< 5, 3, 17, 10, 14, 19, 6, 11, 9 >$.
5. Ilustruj operáciu `Heap-Extract-Max` na halde $< 15, 13, 9, 5, 12, 8, 7, 4, 0, 6, 2, 1 >$.
6. Operácia `Heap-Delete(A, i)` odstráni prvok v uzle i z haldy A. Zostroj algoritmus `Heap-Delete`, ktorý bude mať čas výpočtu $O(n \lg n)$.

6 Quicksort

1. Aká hodnota q bude výsledkom `Partition`, keď všetky prvky poľa A[p. .r] majú rovnakú hodnotu?
2. Dokáž, že čas výpočtu `Partition` je $\Theta(n)$.
3. Ako treba modifikovať Quicksort, aby triedil v nerastúcom poradí?
4. Ukáž, že čas výpočtu Quicksortu je $\Theta(n \lg n)$ keď sú prvky rovnaké.
5. Zostroj procedúru `Randomized`, ktorá pre vstupné pole A[p. .r] vygeneruje náhodnú permutáciu jeho prvkov v čase $\Theta(n)$.
6. Ukáž, že najlepší čas pre Quicksort je $\Omega(n \lg n)$.
7. Ukáž, že očakávaný čas výpočtu pre Randomized-Quicksort je $\Omega(n \lg n)$.

7 Bottom-Up Heapsort

1. Utrieď $< 4, 1, 3, 2, 16, 9, 10, 14, 8, 7 >$.
2. Čo sa stane, ak zameníme neostré nerovnosti za ostré?
3. Ako docielime opačné utriedenie?

8 Triedenie v lineárnom čase

1. Vypočítaj asymptoticky tesné ohraničenie pre $\lg n!$ bez Stirlingovej aproximácie ($\lg n! = \Sigma_{k=1}^{n} \lg k$).
2. Ukáž, že neexistuje triedenie porovnávaním, ktorého čas výpočtu je lineárny pre najmenej polovicu z $n!$ vstupov. Ako je to v prípadoch $\frac{1}{n}$ z $n!$; $\frac{1}{2^n}$ z $n!$?
3. Použi Countingsort na $< 7, 1, 3, 1, 2, 4, 8, 7, 4, 3 >$.
4. Dokáž, že triedenie Countingsort je stabilné (zachováva relatívne poradie rovnakých prvkov). Keby sme 9. riadok zmenili na

   $$\text{for j } \leftarrow 1 \text{ to length}\,[A]$$

   bol by Countingsort stabilný?
5. Použitím Radix sort utrieď pole objektov `COW, DOG, SEA, RUG, ROW, MOB, BOX, BAR, EAR, TAR, DIG, BUG, TEA FOX, NOW`
6. Ktorý z nasledujúcich algoritmov je stabilný: Insertsort, Mergesort, Heapsort, Quicksort? Koľko dodatočného času a priestoru (pamäte) je potrebné na stabilizáciu nestabilných algoritmov?
7. Ukáž, ako sa dá utriediť n celých čísiel z $\left[1, n^2\right]$ v čase $O(n)$.

9 Zoznamy, zásobníky, fronty

1. Napíš pseudokód zdvojeného radu.
2. Vysvetli ako implementovať 2 zásobníky v 1 poli A[1..n] tak, aby žiaden z nich nebol naplnený, pokiaľ nie je vyčerpané celé pole A[1..n].
3. Prepíš `Enqueue` a `Dequeue` na ošetrenie, či je Q prázdny alebo plný.
4. Popíš operáciu vkladani a vyberania do dvojitého radu.
5. Ukáž, ako implementovať rad pomocou 2 zásobníkov, analyzuj čas výpočtu operácii.
6. Implementuj zásobník použitím jednoduchého spájaného zoznamu (operácie `Push`, `Pop` v čase $O(1)$).
7. Implementuj slovníkové operácie `Insert`, `Delete`, `Search` použitím jednoduchého kruhového spájaného zoznamu.

10 Stromy

1. Napíš rekurzívnu procedúru s časom $O(n)$, ktorá vypíše uzly (kľúče) n-prvkového binárneho stromu.
2. Napíš nerekurzívnu procedúru, ktorá vypíše uzly (kľúče) n-prvkového binárneho stromu.

11 Hašovanie

1. Demonštruj vkladanie kľúčov 5, 28, 19, 15, 20, 33, 17, 12, 10 do hašovacej tabuľky, kde kolízie sú riešené zreťazením, $h(k) = k \bmod 9$, tabuľka má 9 pozícii.
2. Zdôvodni, že očakávaný čas pre úspešné vyhľadávanie pri hašovaní zreťazením je rovnaký, či sa vkladajú prvky na začiatok zoznamu, alebo na koniec (hint: ukáž, že nezáleží, ako sú prvky usporiadané v zozname).
3. Ukáž, že ak obmedzíme každú zložku a_i z a v rovnici (14) na rôznu od nuly, potom množina $\mathcal{H} = \cup\{h_a\}$ definovaná vzťahom (15) nie je univerzálna (zváž hlavne $x = 0, x = 1$).
4. Uvažujme hašovaciu tabuľku veľkosti $m = 10000$, $h(k) = \lfloor m(k * A - \lfloor k * A \rfloor) \rfloor$ pre $A = \frac{\sqrt{5}-1}{2}$. Spočítaj pozcície, na ktoré sa hašujú kľúče 61, 62, 63, 64, 65.
5. Uvažujme verziu delenia $h(k) = k \bmod m$ kde $m = 2^p - 1$ a k je nejaké číslo (znakový reťazec interpertovaný v radix $2p$). Ukáž, že ak reťazec x môže byť vytvorený z reťazca y permutáciou jeho znakov, budú sa x a y hašovať do tej istej pozície. Ukáž prípad, kedy nám to bude vadiť.

6. Treba uložiť kľúče 10, 22, 31, 4, 15, 18, 17, 88, 59 do hašovacej tabuľky dĺžky $m = 11$ použitím otvorenej adresácie, pričom primárna haš. f. je $h'(k) = k \bmod m$. Ilustruj výsledok vkladania kľúčov použitím
   (a) lineárneho hašovania
   (b) kvadratického hašovania, $c_1 = 1, c_2 = 3$
   (c) dvojitého hašovania použitím $h_2(k) = 1 + (k \bmod (m - 1))$

7. Predpokladajme hašovaciu tabuľku s otvorenou adresáciou a ukladacím faktorom $\alpha = \frac{1}{2}$. Aký bude očakávaný počet pokusov pri úspešnom a neúspešnom vyhľadávaní? Zopakuj pre $\alpha = \frac{3}{4}$, $\alpha = \frac{7}{8}$, $\alpha = \frac{9}{10}$.

12 Binárne vyhľadávacie stromy

1. Pre kľúče 1, 4, 5, 10, 16, 17, 21 nakresli BVS výšky 2, 3, 4, 5, 6.

2. Aký je rozdiel medzi vlastnosťami BVS a vlastnosťami haldy? Môže sa vlastnosť haldy použiť na výpis n-uzlového stromu v usporiadanom poradí v čase $O(n)$?

3. Hľadáme v BVS kľúč s hodnotou 363. Sú tieto postupnosti kľúče v BVS?
   (a) 2, 252, 401, 398, 330, 344, 397, 363
   (b) 924, 220, 911, 244, 898, 258, 362, 363
   (c) 935, 278, 347, 621, 299, 392, 358, 363

4. Profesor Bujman si myslí, že objavil výnimočnú vlastnosť BVS: predpokladajme, že hľadanie kľúča k v BVS končí na liste, predpokladajme 3 množiny:
   A = kľúče vľavo od cesty preľhadávania
   B = kľúče na ceste prehľadávania
   C = kľúče vpravo od cesty prehľadávania
   Profesor Bujman tvrdí, že ľubovoľné 3 kľúče $a \in A, b \in B, c \in C$ spĺňajú nerovnosť $a \leq b \leq c$. Over, či má profesor Bujman pravdu, alebo ukáž kontrapríklad.

5. Inorder prechod BVS môže byť implementovaný nasledovne: nájdeme najmenší prvok procedúrou `Tree-Minimum`, potom $n-1$ krát voláme `Tree-Succesor`. Dokáž, že tento algoritmus bude mať zložitosť $O(n)$.

6. Zostroj rekurzívnu procedúru `Tree-Insert`.

7. Danú množinu n čísiel môžeme utriediť tak, že zostrojíme BVS použitím `Tree-Insert` n-krát a potom vypísaním prechodom Inorder. Aký je najlepší a najhorší čas výpočtu?

8. Dokáž, že ak uzol v BVS má 2 deti, potom jeho nasledovník nemá ľavé dieťa a predchodca nemá pravé dieťa.

13 RB-stromy

1. Napíš pseudokód pravej rotácie.
2. Zdôvodni, že rotácia zachováva poradie výpisu vrcholov pri inorder výpise.
3. Ukáž, že ľubovoľný n-uzlový binárny strom môže byť transformovaný na ľubovoľný iný strom pomocou $O(n)$ rotácii.
4. V riadku 2 je priradená farba Red do x. Keby sme vybrali Black, nebola by narušená vlastnosť RB-stromu. Prečo sme nevybrali túto možnosť?
5. V riadku 18 sme priradili Black do koreňa. Aká z toho plynie výhoda?
6. Ukáž proces zmeny RB-stromu postupným vkladaním kľúčov 41, 38, 31, 12, 19, 8 do počiatočne prázdneho stromu.
7. Po vytvorení RB-stromu z predchádzajúcej úlohy ukáž RB-stromy ako výsledky vymazávania s kľúčmi 8, 12, 19, 31, 38, 41 v tomto poradí.
8. V ktorých riadkoch `RB-Delete-Fixup` sa skúma alebo modifikuje sentinel `nil[T]`?
9. Zjednoduš kód `Left-Rotate` použitím sentinelu a zavedením sentinelu ukazujúceho na koreň `root[T]`.
10. Predpokladajme, že uzol x je vložený do RB-stromu, cez RB-insert a vzápätí vymazaný RB-delete. Je výsledný strom rovnaký ako na začiatku?

14 Dynamické programovanie

1. Nájdi optimálne uzátvorkovanie súčinu maticového reťazca, ktorého postupnosť rozmerov je 5, 10, 3, 12, 5, 50, 6.
2. Zostroj efektívny algoritmus `Print-Optimal` na vypísanie optimálneho uzátvorkovania maticového reťazca, ak máme danú tabuľku S vypočítanú algoritmom `Matrix-Chain-Order`.
3. Nech $R[i,j]$ je počet volaní $m[i,j]$ hodnoty tabuľky m z algoritmu `Matrix-Chain-Order`. Ukáž, že celkový počet volaní pre celú tabuľku je

   $$\sum_{i=1}^{n}\sum_{j=1}^{n} R[i,j] = \frac{n^3 - n}{3}$$

   Zváž užitočnosť rovnosti

   $$\sum_{i=1}^{n} n^2 = \frac{n(n+1)(2n+1)}{6}$$

4. Urči LCS postupností $[1,0,0,1,0,1,0,1]$ a $[0,1,0,1,1,0,1,1,0]$.
5. Zostroj alg. dyn. prog. pre problém výberu aktivít m_i pre $i = 1, 2, \ldots, n$. m_i najv. podm. komp. aktivít pre množinu $\{1, 2, \ldots, i\}$. Predpokladaj, že vstupy $[f_1, f_2, \ldots, f_n]$ sú utriedené. Spočítaj čas tohto algoritmu vs. greedy riešenie.

6. Dokáž, že zlomkový problém výberu batohu má vlastnosť greedy výberu.
7. Zostroj riešenie metódou dynamického programovania pre 0-1 problém batohu, ktorý má čas výpočtu $O(nW)$.
8. Popíš efektívny algoritmus, ktorý pre danú množinu bodov $\{x_1, x_2, \ldots, x_n\}$ na reálnej osi určí najmenšiu množinu jednotkových uzavretých intervalov, ktoré pokrývajú všetky dané body. Zdôvodni správnosť riešenia.

15 Huffmanov kód

1. Ukáž, že ak binárny strom nie je úplný, tak nemôže korešpondovať s optimálnym prefixovým kódom.
2. Aký je optimálny Huffmanov kód pre

   $$a : 1; b : 1; c : 2; d : 3; e : 5; f : 8; g : 13; h : 21$$

   Aký je zovšeobecňujúci vzťah pre generovanie Huffmanovho kódu pre frekvencie tvoriace Fibonacciho postupnosť?
3. Zovšeobecni Huffmanov algoritmus pre ternárne kódové slová (0, 1, 2) a dokáž, že vytvára optimálny kód.
4.

16 B-stromy

1. Prečo nepripustíme minimálny stupeň $t = 1$?
2. Ukáž všetky legálne B-stromy s $t = 2$, ktoré reprezentujú kľúče $1, 2, 3, 4, 5$.
3. Odvoď tesnú hornú hranicu pre počet kľúčov, ktoré môžu byť uložené v B-strome výšky h s minimálnym stupňom t.
4. Ukáž výsledok vkladania kľúčov `F, S, Q, K, C, L, H, T, V , W, M, R, N, P, A, B, X, Y, D, Z, E` v tomto poradí do prázdneho stromu. Nakresli náčrt, keď dochádza ku splitom a finálny strom.
5. Vysvetli ako nájsť minimálny kľúč uložený v B-strome a ako nájsť predchodcu k danému kľúču uloženému v B-strome.
6. Predpokladajme, že kľúče $\{1, 2, \ldots, n\}$ sú uložené do prázdneho B-stromu s minimálnym stupňom $t = 2$. Koľko uzlov má konečný B-strom?