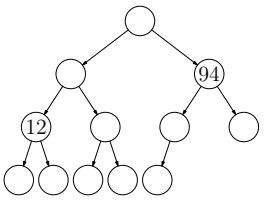
1 (True or False) and Justify [20 bodov]

1. Z in-order zápisu binárneho vyhľadávacieho stromu vieme tento strom jednoznačne zrekonštruovať.
2. Ak do vektoru, v ktorom už je n prvkov, postupne vložím n ďalších, bude zaručene celková časová zložitosť $O(n)$.
3. Dve polia obsahujú po n celých čísel. V čase $O(n \log n)$ sa dá zistiť, či sa niektoré číslo nachádza v oboch poliach.
4. Máme haldu s minimom v koreni, obsahujúcu n prvkov. Na nájdenie *maximálneho* prvku v nej treba čas $\Omega(n \log n)$.
5. Ak budeme v MergeSorte pole zakaždým deliť na štvrtinu a tri štvrtiny, bude časová zložitosť naďalej $O(n \log n)$.
6. Na paličke je n mravcov. Chcú sa stretnúť na jednom mieste. Tým miestom, pre ktoré bude súčet mravcami prejdených vzdialeností minimálny, je aritmetický priemer ich súradníc (t.j. ťažisko množiny mravcov).
7. Nech f je funkcia taká, že $f(n)$ je n-té Fibonacciho číslo. Potom f patrí do $O(2^n)$.
8. Ak pri HashSete zmeníme poradie, v ktorom doň vkladáme prvky, nezmeníme tým celkový počet kolízií.

2 Pohľad zvnútra: halda [4+6 bodov]



Na obrázku je binárna halda s maximom v koreni.
a) Vyfarbite všetky vrcholy, kde sa môže nachádzať prvok s hodnotou 47.
b) Vyplňte všetky prázdne vrcholy navzájom rôznymi prirodzenými číslami tak, aby vznikla korektná halda.

3 Pohľad zvnútra: prasa strikes back [15 bodov]

V pamäti máme pole $A[1..n]$. Toto pole bolo voľakedy usporiadané v ostro rastúcom poradí. Potom ale prišlo prasa. A tá sviňa nám toto pole zrotovala o niekoľko pozícií doprava. Príklad: pole $A = (6, 47, 1, 2, 4)$ bolo zrotované o 2 pozície. Napíšte čo najefektívnejšiu funkciu, ktorá pre dané x zistí, či sa v poli A nachádza. Dobré riešenie by to malo zvládnuť bez toho, aby sa pozrelo na väčšinu políčok poľa A.
Pomôcka: Asi to bude jednoduchšie, ak najskôr zistíte, o koľko tá sviňa zrotovala pole. Ale ide to aj bez toho.

4 Pohľad zvnútra: podmnožiny podmnožín [15 bodov a bonus]

Veliteľ má posádku tvorenú n vojakmi, očíslovanými 1 až n. Potrebuje sa rozhodnúť, ktorí vojaci budú v noci spať a ktorí hliadkovať. A o každom hliadkujúcom vojakovi sa potrebuje rozhodnúť, či bude hliadkovať pri bráne alebo okolo plota. Napíšte program, ktorý veliteľovi vypíše všetky možnosti, ktoré má na výber. (Ak máte program s optimálnou časovou zložitosťou, dostanete bonusové body, ak ju navyše správne odhadnete a dokážete, že je optimálna.)

5 Pohľad zvonka: realitná kancelária [7/13/20 bodov]

Realitná kancelária ponúka na svojej webstránke na predaj byty. Uvažujme niekoľko typov operácií:

1. `navstevnik(s)`: Keď príde návštevník na stránku, dostane otázku, ako drahý byt hľadá. Zadá sumu s a stránka mu ukáže k bytov, ktorých cena je najbližšia sume s. (Číslo k je rádovo menšie ako počet n všetkých bytov.)
2. `novy_byt(adresa,cena)`: Realitka má nový byt na predaj, treba ho pridať do ponuky.
3. `predane(adresa)`: Byt na danej adrese bol predaný, treba ho z ponuky odstrániť.

Vyberte si jednu z možností:
a) (max. 7 bodov) Navrhnite čo najlepšiu dátovú štruktúru umožňujúcu efektívne robiť operácie typu 1.
b) (max. 13 bodov) Navrhnite čo najlepšiu dátovú štruktúru umožňujúcu efektívne robiť operácie typu 1 a 2.
c) (max. 20 bodov) Navrhnite čo najlepšiu sadu dátových štruktúr umožňujúcu efektívne robiť operácie typu 1, 2 aj 3.
Bez ohľadu na vybranú možnosť: Slovne popíšte implementáciu navrhovaného riešenia pomocou dátových štruktúr z prednášky. Odhadnite časovú zložitosť jednotlivých operácií ako funkciu n a k.

6 Pohľad zvonka: problém vreca v krajine hojnosti [4 × 5 bodov]

V sklade je n typov vecí. Každá vec i-teho typu má kladnú celočíselnú hmotnosť m_i (v gramoch) a kladnú cenu c_i. Vecí každého typu je veľmi veľa. Do skladu prišiel zlodej s dostatočne veľkým vrecom. V ňom zvláda odniesť veci s celkovou hmotnosťou nanajvýš M. Nájdite najdrahšiu sadu vecí, ktoré zvláda odniesť. Príklad: pre $M = 65000$, $n = 3$ a typy vecí s $(m_i, c_i) = (30000, 1000), (16000, 120), (4047, 1)$ zlodej vezme 2 veci prvého typu a 1 vec tretieho typu.

a) Napíšte (ako pseudokód alebo kus programu) rekurzívny algoritmus skúšajúci všetky možnosti ako vybrať veci do vreca. (Začne napr. tým, že pre n-tú vec postupne rekurzívne vyskúša možnosti „ešte jednu takúto vezmem“ a „už žiadnu takúto nevezmem“.) Zdôvodnite, že váš program pre ľubovoľný možný vstup naozaj skončí.
b) Pridajte do predchádzajúceho algoritmu memoizáciu tak, aby vznikol algoritmus s časovou zložitosťou polynomiálnou od počtu vecí n. Odhadnite jeho časovú zložitosť.
c) Uveďte ekvivalentný algoritmus, ktorý túto úlohu rieši pomocou dynamického programovania.
d) Existujú situácie, kedy sa môžeme pozrieť na hmotnosti a ceny vecí a z nich priamo usúdiť, že niektoré veci určite v optimálnom riešení nepoužijeme. Pre nasledujúcu sadu typov vecí nájdite dva typy vecí, ktoré zlodej určite kradnúť nebude (bez ohľadu na M). Vyslovte kritérium, ktoré ste použili, všeobecne. Dokážte jeho správnosť.
Máme $n = 7$ typov vecí s $(m_i, c_i) = (3200, 1001), (447, 1), (3000, 1000), (1600, 120), (1310, 98), (2320, 84), (415, 1)$.