1. Egy bináris fa csúcsai 0 és 9 közötti egész számokkal vannak megcímkézve. Az inorder bejárás során a címkék sorrendje: 9, 3, 1, 0, 4, 2, 7, 6, 8, 5, a posztorder bejárásnál pedig 9, 1, 4, 0, 3, x, 7, 5, y, 2. Mi lehet az x és mi az y?

   Megoldás: Látszik, hogy a második listából a 6 és a 8 hiányzik, ezek egyike az x, a másik az y.

   Próbáljuk rekonstruálni a fát! A posztorder utolsó eleme a fa gyökere, tehát ez a 2. Az inorderben a 2 előttiek vannak a bal részfában, a 2 utániak a jobban. A 2 gyerekeit megint a posztorderből olvashatjuk ki, a 3 a bal oldalra került elemek közül az utolsó, tehát ez a 2 bal gyereke, a jobb pedig az y. Az y értékétől függetlenül a 7 az y bal részfájában lesz, hiszen az inorder sorrendben megelőzi. Nem lehet $y = 6$, mert akkor az inorder sorrend szerint a bal részfája csak a 7-ből állna, és nem állhatna a posztorder sorrendben előtte az x. Tehát csak $x = 6$ és $y = 8$ lehet. Ez pedig valóban egy jó megoldás, a 2 jobb gyereke az $y = 8$, ennek két fia a 7 és az 5, a 7-nek valamelyik oldali gyereke a 6.

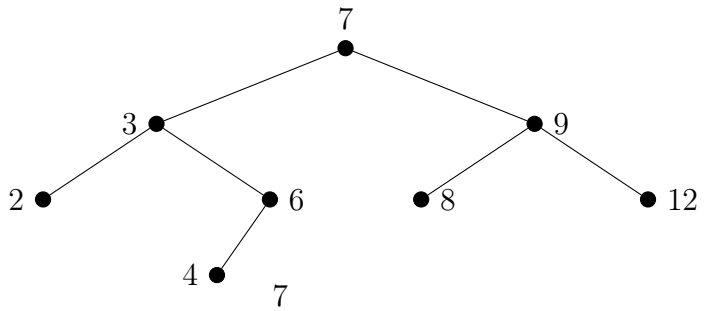
   Megjegyzés: ha látni akarjuk, hogy ez tényleg jó megoldás, azt is ellenőrizni kell, hogy a bal részfa sorrendjei sem mondanak ellent egymásnak.

2. Határozza meg azokat a bináris fákat, amelyekben a preorder bejárás szerinti sorrend éppen a posztorder bejárás által adott sorrend fordítottja!

   Megoldás: Ha egy csúcsnak két gyereke van, akkor ezek sorrendje ugyanaz a preorder és a posztorder bejárás szerint is. Tehát a fa egyetlen út lehet csak (nincs benne elágazás). Ebben az esetben pedig függetlenül attól, hogy az a gyerek melyik oldalon van, a preorder a fán lefelé haladva sorolja fel az elemeket, a posztorder meg felfelé.

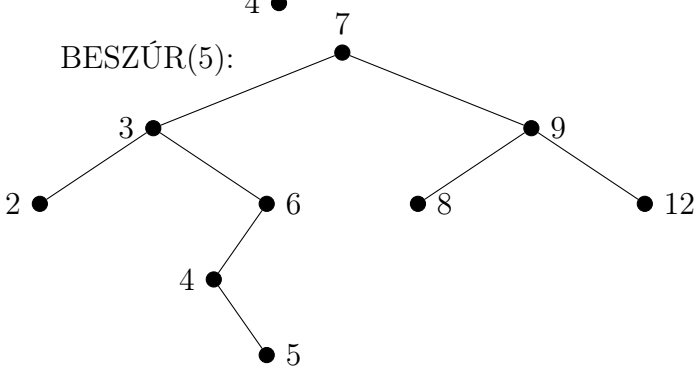
3. (a) Építsen beszúrásokkal bináris keresőfát az alábbi sorrendben érkező számokból: 7, 3, 2, 9, 8, 12, 6, 4.
   (b) Járja be pre-. in-, és posztorder bejárással a kapott fát!
   (c) Az (a) rész keresőfáján hajtsa végre az alábbi műveletsort: BESZÚR(5), TÖRÖL(2), TÖRÖL(7), TÖRÖL(6).

   Megoldás: A végeredmény ez lesz:

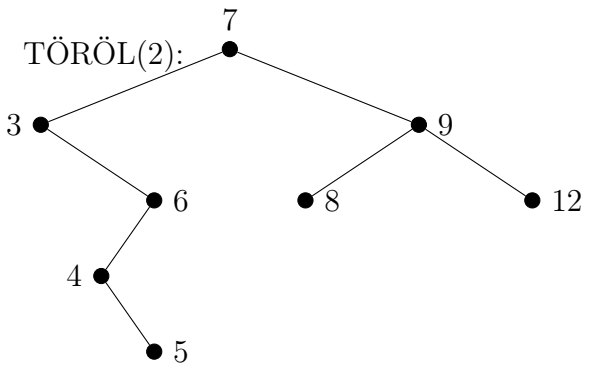


preorder: 7, 3, 2, 6, 4, 9, 8, 12
inorder: 2, 3, 4, 6, 7, 8, 9, 12
posztorder: 2, 4, 6, 3, 8, 12, 9, 7

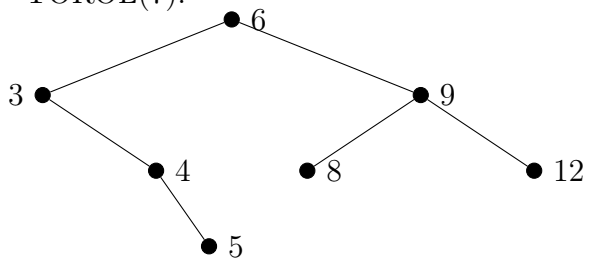
BESZÚR(5):



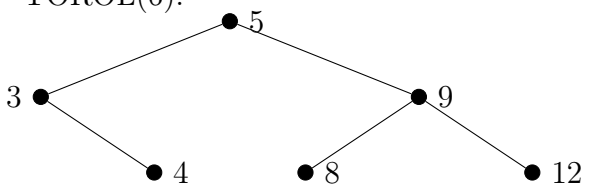
TÖRÖL(2):



TÖRÖL(7):



TÖRÖL(6):



4. Egy bináris keresőfában csupa különböző egész számot tárolunk. Mely x egész számokra lehetséges, hogy egy KERES(x) hívás során a keresési út mentén a 20, 18, 3, 15, 5, 8, 9 kulcsokat látjuk ebben a sorrendben?

   Megoldás: Nyilván a 20 van a fa gyökerében. Mivel utána a $18 < 20$ jön majd, nyilván balra léptünk, azaz $x < 20$. Hasonlóan a $3 < 18$ miatt megint balra léptünk, tehát $x < 18$. Ekkor egy 3-nál nagyobb szám következett, tehát $x > 3$, és így tovább, $x < 15$, $x > 5$, $x > 8$. Összegezve, biztos teljesül, hogy $8 < x < 15$. Azaz $x = 9, 10, 11, 12, 13, 14$ mindegyike lehet, az első esetben a keresés sikeres volt, a többiben nem.

5. Bizonyítsuk be, hogy ha az inorder bejárást egy bináris keresőfára futtatjuk, akkor az elemeket rendezett sorrendben kapjuk meg.

   Megoldás: Bizonyítás: Azt látjuk be, hogy ha x és y két elem a fában, melyekre teljesül, hogy $x < y$, akkor az inorder bejárás x-et írja ki előbb.

   Három eset van:

   - Az x csúcs az y csúcs részfájában van: ekkor $x < y$ miatt csak a bal oldali fában lehet, de ekkor az inorder bejárás y bal fáját és így az x csúcsot is y előtt járja be.
   - Az y csúcs az x csúcs részfájában van: ekkor $x < y$ miatt csak a jobb oldali fában lehet, de ekkor az inorder bejárás x jobb fáját és így az y csúcsot is x után járja be.
   - Egyik csúcs sem leszármazottja a másiknak: ekkor van egy olyan közös ősük, aminek egyik részfájában van x, másik részfájában pedig y, legyen ez ős z. De ekkor $x < y$ miatt x a bal, y pedig a jobb részfájában van z-nek és így az inorder bejárás a teljes bal fát (beleértve x-et) z előtt, a teljes jobb fát (beleértve y-t) z után járja be, vagyis x előbb lesz, mint y.

6. Az A keresőfában n egész számot, a B-ben pedig m-et tárolunk. Rendezzük az $n+m$ elemet $O(n+m)$ lépésben!

   Megoldás: Tudjuk, hogy az inorder bejárás növekvő sorrendben adja meg a tárolt elemeket. Olvassuk így ki mindkét fából a tárolt számokat, és a két növekvő listát fésüljük össze. A kiolvasás (bejárás) lépésszáma arányos a fa méretével, az összefésülés $n + m - 1$ összehasonlítást és $n + m$ mozgatást igényel, összesen tehát a lépésszám $O(n + m)$.

7. Adott egy n csúcsú bináris keresőfa, melyben csupa különböző elemeket tárolunk. Ennek minden v csúcsára meg akarjuk határozni, hogy a v gyökerű részfában hány darab v-nél kisebb elem van tárolva. Adjon algoritmust, ami ezt a feladatot $O(n)$ lépésben megoldja!

   Megoldás: Egy keresőfa v gyökerű részfájában v-nél kisebb elemek csak a v bal részfájában lehetnek, a feladatunk minden csúcsra a bal részfa méretének meghatározása. Helyette dinamikus programozással határozzuk meg minden v csúcsra a bal részfa $B[v]$ és a jobb részfa $J[v]$ méretét is! Posztorder sorrendben menjünk végig a csúcsokon. Ha v levél, akkor $B[v] = J[v] = 0$. Ha v nem levél és x a bal, y a jobb gyereke, akkor $B[v] = B[x] + J[x] + 1$ és $J[v] = B[y] + J[y] + 1$. Amennyiben x vagy y nem létezik, akkor $B[v]$ vagy $J[v]$ értéke 0.

   Végül a $B[v]$ értékekre van szükségünk.

   A lépésszám: a bejárás lineáris, minden csúcsnál konstans sok további műveletet végzünk, ezért összesen $O(n)$.

8. Igazolja, hogy minden olyan algoritmus, ami csak összehasonlításokkal fel tud építeni egy bináris keresőfát n elem esetén $\Omega(n \log n)$ összehasonlítást használ!

   Megoldás: Mivel egy bináris keresőfából az elemek növekvő sorrendjének előállításához már nem kell további összehasonlítás (csak az inorder bejárás), ezért az összes szükséges összehasonlítás a fa elkészítésekor történik. A rendezéshez kell $\Omega(n \log n)$ összehasonlítás, tehát a fa építéséhez is szükség volt ennyire.

9. Nyitott címzéssel hash-elünk egy kezdetben üres $M = 11$ méretű táblába a $h(x) = x \pmod M$ hash-függvénnyel. Mi lesz a tábla állapota, ha a 4, 5, 14, 15, 16, 26, 3 kulcsokat a megadott sorrendben beszúrjuk és az ütközések feloldására

   (a) lineáris próbát használunk?

   (b) kvadratikus maradék próbát használunk?

   (c) kettős hash-elést használunk, amikor $h'(x) = 7x \pmod{(M-1)}$ a második hash-függvény? Hány ütközés történt az egyes esetekben?

   Megoldás: (a) Az ütközések száma: $0+0+0+2+4+4+4 = 14$, a végén a tábla elemei sorban 22, 16, 15, 14, 4, 5, –, –, –, –, 3

   (b) Az ütközések száma (lefelé indulok): $0+0+0+3+2+4+1 = 10$, a végén a tábla elemei sorban 15, –, 3, 14, 4, 5, 16, –, 26, –, –

   (c) Az ütközések száma (most is lefelé indulok): $0+0+0+1+2+1+3 = 7$, a végén a tábla elemei sorban 3, 16, 26, 14, 4, 5, –, –, –, –, 15

10. Előfordulhat-e nyitott címzéses hash-elés esetén, hogy az $n > 3$ méretű táblában csak 3 elem van, de a keresés lépésszáma n ?

    Megoldás: Igen, ha korábban törlések is történtek, pontosabban, ha minden helyről, ahol most nincs elem volt törlés. Ekkor a keresés ezeken továbbmegy, és ha a 3 bent levő egyike sem a keresett elem, akkor mindenképpen bejárjuk az egész táblát.

11. Jó választás-e $M = 7$ méretű táblánál a $h(x) = x^2 \pmod 7$ hash-függvény?

    Megoldás: Nem, mert $f(1) = f(6)$, $f(2) = f(5)$ és $f(3) = f(4)$, tehát minden a 0, 1, 4, 2 helyek egyikénél kezd ahelyett, hogy a 7 lehetséges helyre szétoszlanának (több ütközés várható).

12. A $T[0 : M]$ táblában $2n$ elemet ($n < M/3$) helyeztünk el valamilyen hash-függvény segítségével, amikor azt tapasztaltuk, hogy az elemek mindegyike az első $3n$ hely egyikére került. Ha nem volt közben törlés és a végén a táblában minden $3i$ indexű hely üres maradt ($0 \le i < n$), akkor legfeljebb hány ütközés lehetett, ha

    (a) lineáris próbát használtunk?

    (b) kvadratikus maradék próbát használtunk?

    Megoldás: (a) Ezek szerint a táblázat elején mindig két foglalt helyet követ egy üres. Lineáris próbánál csak a két szomszédos ütközhetett a másodiknak érkező beszúrásakor, tehát legfeljebb n ütközés volt. Ennyi lehet is, pl. ha az elemek sorban 2, $M+2$, 5, $M+5$, stb.

    (b) Mivel mindenkinek legalább az egyik szomszédja üres, elemenként megint legfeljebb egy ütközés lehetett. Ezért az ütközések csak egy-egy szomszédos páron belül történhettek. Azaz most is legfeljebb n ütközés lehetett. Az előző példa mutatja, hogy ez lehetséges is.

13. Egy m méretű hash-táblában már van néhány elem. Adjon $O(m)$ lépésszámú algoritmust, amely meghatározza, hogy egy újabb elem lineáris próbával történő beszúrásakor maximum hány ütközés történhet!

    Megoldás: A feladat az, hogy a táblában levő legnagyobb foglalt blokk hosszát meghatározzuk. Ezt egyszerűen dinamikus programozással lineáris időben megtehetjük például így: Legyen M az eddigi

legnagyobb befejezett blokk hossza, B pedig az aktuális érték, kezdetben mindkettő 0. Az m méretű T táblában sorban az $i = 0,1,2\ldots$ indexekre (az indexet $\mod m$ értve):
Ha $T[i]$-ben nincs elem és $B > M$, akkor legyen $M = B$.
Ha $T[i]$-ben nincs elem, akkor $B = 0$.
Ha $T[i]$-ben van elem, akkor növeljük B-t eggyel.
Ha $i \geq m$ és $T[i]$-ben nincs elem, akkor megállunk. (Ez nyilván be fog következni valamely $i < 2m$ esetben.)
A végén $\max(M, B)$ a válasz.

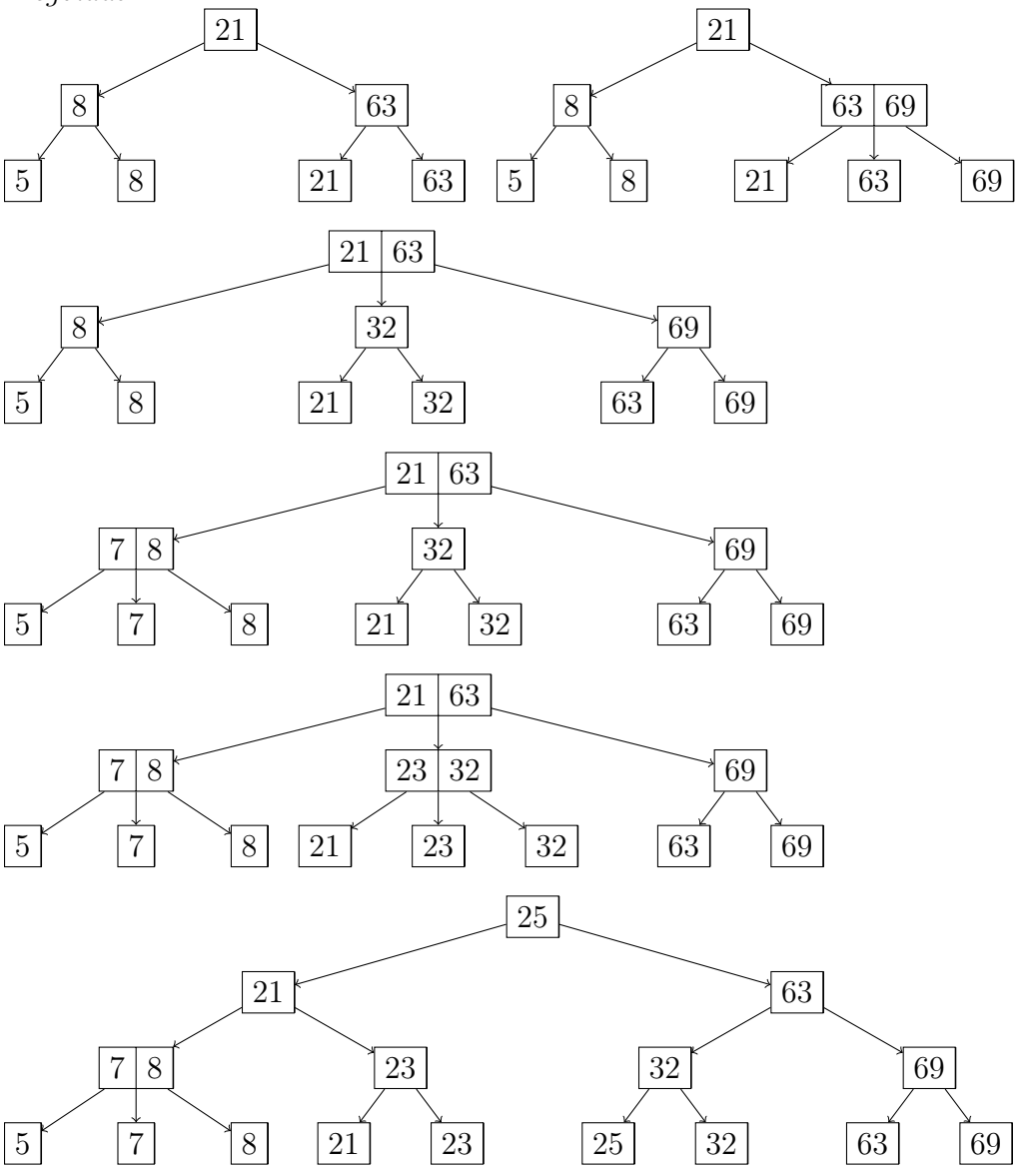
14. A $b_0 ... b_n$ alakú $n + 1$ hosszú bitsorozatokat akarjuk tárolni. Tudjuk, hogy a b_0 paritásbit (ami a sorozatban az egyesek számát párosra egészíti ki). Ha nyitott címzésű hash-elést használunk $h(x) \equiv x \pmod M$ hash-függvénnyel és lineáris próbával, akkor $M = 2^n$ vagy $M = 2^n + 1$ méretű hash-tábla esetén lesz kevesebb ütközés?

   Megoldás: Az ilyen számokon a mod 2^n levágja az első bitet, tehát $M = 2^n$ esetén különböző sorozatok nem ütköznek. Az $M = 2^n + 1$ esetben viszont lesz ütközés, a $00\cdots00$ ütközik az $10\cdots01$ sorozattal, hiszen mindkét esetben a h értéke 0.

A 2-3 fák idén nem fértek bele az előadásba. Akit érdekel, a honlapon megnézheti a tudnivalókat, itt pedig pár gyakorló feladat következik.

15. Adjon egy 2-3-fát amely az 5,8,21,63 elemeket tartalmazza, majd sorban szúrja be a 69,32, 7,23,25 elemeket!

   Megoldás:



16. Egy 2-3-fa gyökerének három fia van, a benne szereplő két érték 40 és 50. Mennyi lehet a tárolt elemek minimális, illetve maximális száma, ha tudjuk, hogy csak pozitív egész számokat tárol a fa?

    Megoldás: A feltételekből következik, hogy a gyökérnek 3 fia van, a balban 1 és 39 között, középen 40 és 49 között, a jobb oldalon 50-től tárolunk elemeket.

    A minimális elemszámot nyilván úgy kapjuk, ha a gyökér gyerekei levelek, az x, 40, 50 elemeket tárolja a fa, ahol $1 \leq x \leq 39$ tetszőleges egész.

    A maximális elemszámhoz nézzük, mit tudunk mondani a fa magasságáról! Mivel jelen esetben a három részfa közül a középső tárolhatja a legkevesebb elemet, ez fogja a korlátot adni. Ennek a magassága legfeljebb 4, és a lehetséges maximális 10 elem tárolásához kell is ennyi. Tehát a másik két részfa magassága is 4 lesz. Ezekben akkor kapjuk a lehető legtöbb levelet (tárolt elemet), ha mindig 3 felé ágaznak, azaz 27 – 27 elemet tárolnak. Ennyi megengedett elem van is mindkét oldalon, tehát a maximális elemszám $10 + 2 \cdot 27 = 64$.

17. Az $[1, 178]$ intervallumba eső összes egész számot egy 2-3-fában tároljuk. Tudjuk, hogy a gyökérben két útjelző van, és az első ezekből a 17. Mi lehet a második?

    Megoldás: A második útjelző a jobb oldali részfában tárolt legkisebb szám, tehát ezt kell meghatároznunk.

    Az első útjelző miatt a bal részfa 1-től 16-ig tárolja a számokat. Ezért ennek a részfának a magassága legalább 4 és legfeljebb 5. Mivel két útjelzőnk van, a gyökérnek 3 gyereke van. Tudjuk, hogy mindhárom részfa magassága ugyanaz. Ha ez a közös magasság 4, akkor a középső és jobb részfában is legfeljebb $3^{4-1} = 3^3 = 27$ levél van, tehát legfeljebb ennyi értéket tárolhat, így nem fér a fába mind a 178 elem. Tehát a bal részfa, és így a másik kettő magassága is 5 kell legyen. Ekkor a másik két részfa mindegyike legfeljebb $3^{5-1} = 3^4 = 81$ értéket tárolhat. Mivel $16 + 2 \cdot 81 = 178$, ezért csak az lehet, hogy a középső és jobb részfában minden nem levél csúcsnak 3 gyereke van (a bal részfában meg mindenhol 2), a középső és jobb részfa is 81 elemet tárol.

    A gyökérben levő második útjelző a jobb részfa legkisebb eleme, ami ezek szerint $16 + 81 + 1 = 98$.

18. Egy 2-3-fában 4 elem van. Egyértelmű-e a 2-3-fa?

    Megoldás: A szintek száma $\log_3 4 + 1 = 2.2618\ldots$ és $\log_2 4 + 1 = 3$ között van, tehát pont 3. A három szintű fának pont 4 levele van, ha minden csúcsnak pont 2 gyereke van. Ha bármelyiknek 3 gyereke lenne, már legalább 5 levél lenne. Tehát a fa egy teljes bináris fa, ami egyértelmű. Mivel azt is tudjuk, hogy az elemek a levelekben balról jobbra növekednek, valamint a belső csúcsok címkéi is egyértelműen meghatározhatóak a levelekből, ezért a 2-3-fa egyértelmű.