Algoritmusok II. gyakorlat

6. gyakorlat, március 23.

6. gyakorlat

- Minimum (Maximum) kupactulajdonság

Egy fa minden p csúcsának minden q fiára igaz, hogy q = NULL VAGY p.kulcs < q.kulcs(p.kulcs > q.kulcs).

6. gyakorlat

Bináris kupac: majdnem teljes bináris fa egy tömbben ábrázolva. A fa gyökere a tömb első eleme, és ha *i* a fa egy adott csúcsának indexe a tömbben, akkor:

Szülő(i) = $i/2$,

Bal(i) = $2i$,

Jobb(i) = $2i+1$.

6. gyakorlat

```
MAXIMUM-KUPACOL(A, i)
l ← Bal(i)
r ← Jobb(i)
if l ≤ kupac-méret[A] és A[l] > A[i] then
   legnagyobb ← l
else
   legnagyobb ← i
if r ≤ kupac-méret[A] és A[r] > A[legnagyobb] then
   legnagyobb ← r
if legnagyobb <> i then
   A[i] ↔ A[legnagyobb] csere
   MAXIMUM-KUPACOL(A, legnagyobb)
```

6. gyakorlat

Elsőbbségi (prioritási) sor: a kupac egyik leggyakoribb alkalmazási területe. Kétféle elsőbbségi sorról beszélhetünk: maximum- és minimum elsőbbségi sor.

Az *S* maximum-elsőbbségi sort kezelő műveletek a következők:

SORBA(*S, x*) egy *x* elemet hozzáad az *S* sorhoz.

SORBOL(*S*) megadja és törli *S* legnagyobb kulcsú elemét.

KULCSOT-NÖVEL(*S, x, k*) megnöveli az x elem kulcsát, az új értéke k lesz, amiről feltesszük, hogy legalább akkora, mint az x elem kulcsának pillanatnyi értéke.

6. gyakorlat

```
SORBA(S, kulcs)
   kupac-méret[S] ← kupac-méret[S] + 1
   S[kupac-méret[S]] ← −∞
   KULCSOT-NÖVEL(S,kupac-méret[S], kulcs)

SORBÓL(S)
   if kupac-méret[S] < 1 then
      error ”kupacméret alulcsordulás”
   max ← S[1]
   S[1] ← S[kupac-méret[S]]
   kupac-méret[S] ← kupac-méret[S] – 1
   MAXIMUM-KUPACOL(A, 1)
   return max
```

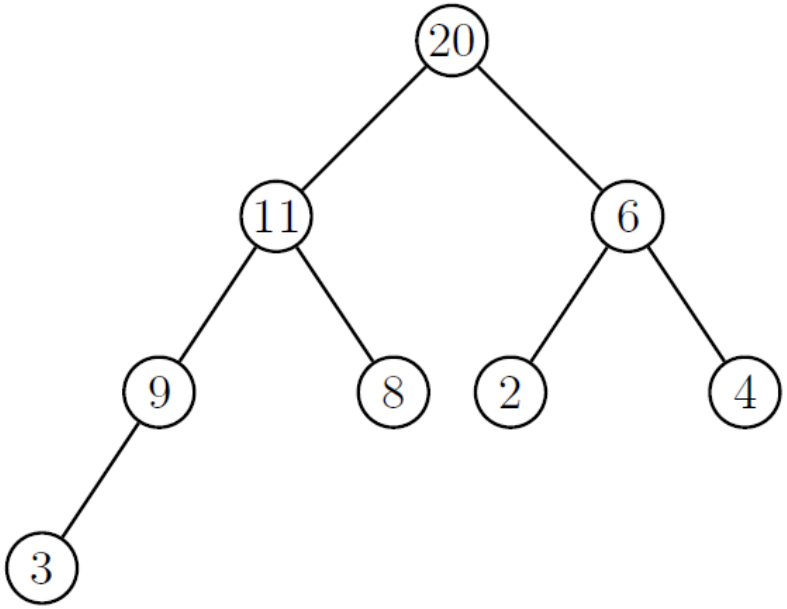
6. gyakorlat

```
KULCSOT-NÖVEL(S, i, kulcs)
   if kulcs < S[i] then
      error ”az új kulcs kisebb, mint az eredeti”
   S[i] ← kulcs
   while i > 1 és S[Szülő(i)] < S[i]do
      S[i] ↔ S[Szülő(i)] csere
      i ← Szülő(i)
```

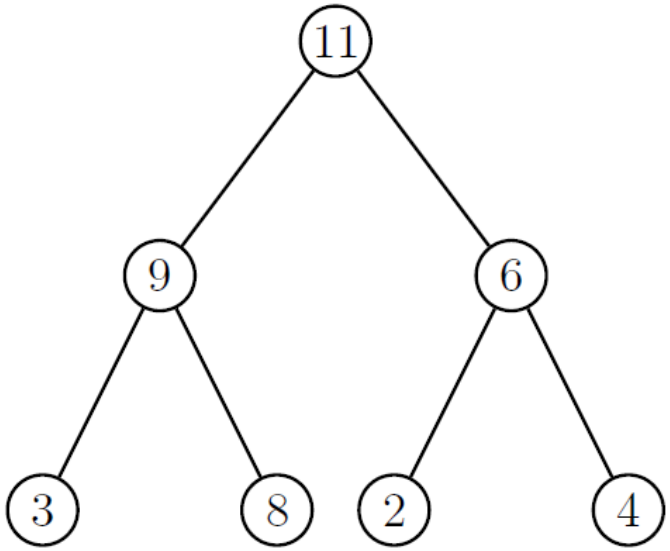
6. gyakorlat

- Hajtsuk végre a Sorba() műveletet egy üres maximum kupacon a következő elemekkel: 3, 8, 2, 11, 20, 4, 6, 9. Mi lesz a Sorbol() eredménye?

6. gyakorlat



(o) SORBA() műveletek után



(p) SORBOL() művelet után

6. gyakorlat

Binomiális fa:

A B_k binomiális fa egy rekurzív módon definiált rendezett fa

B_0 egy csúcsból áll

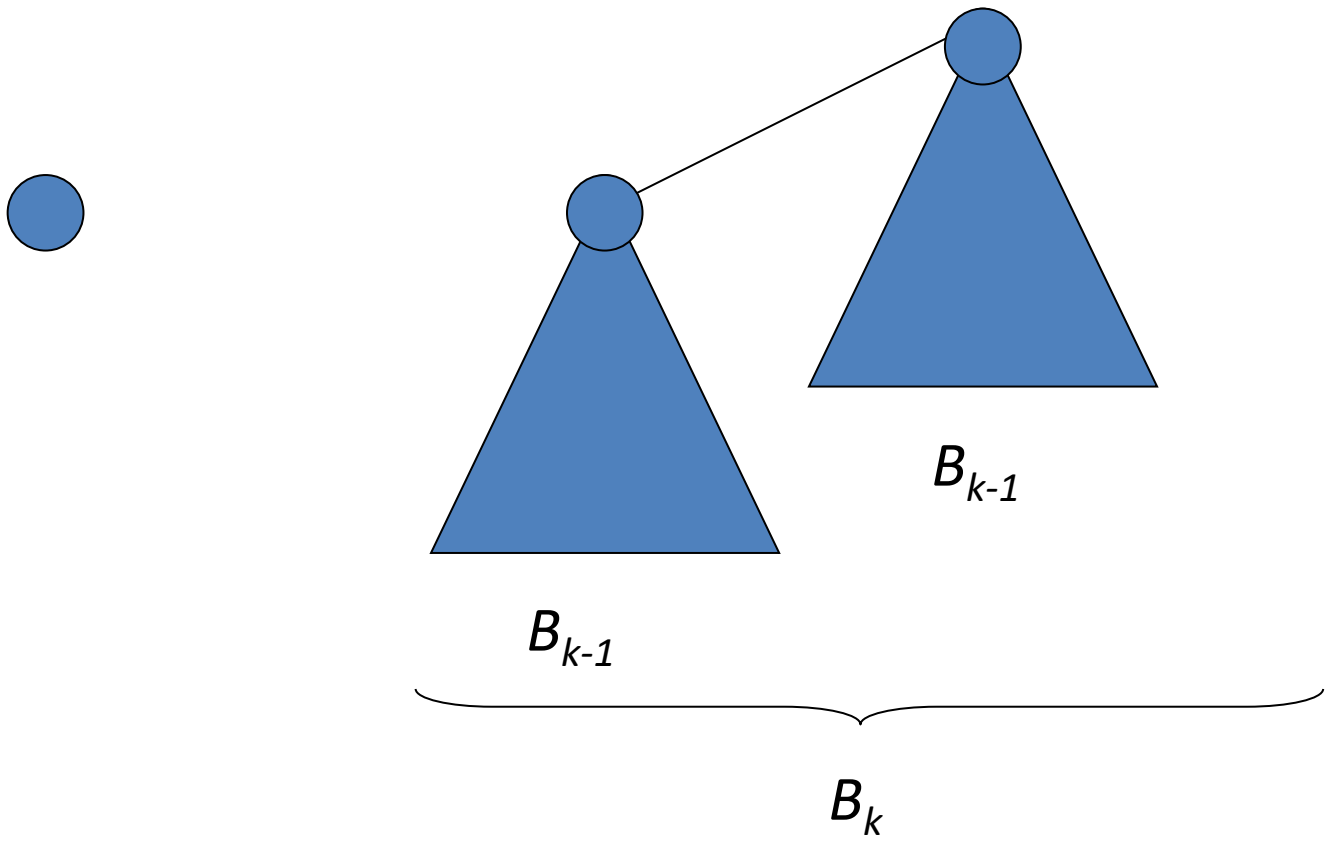
.

.

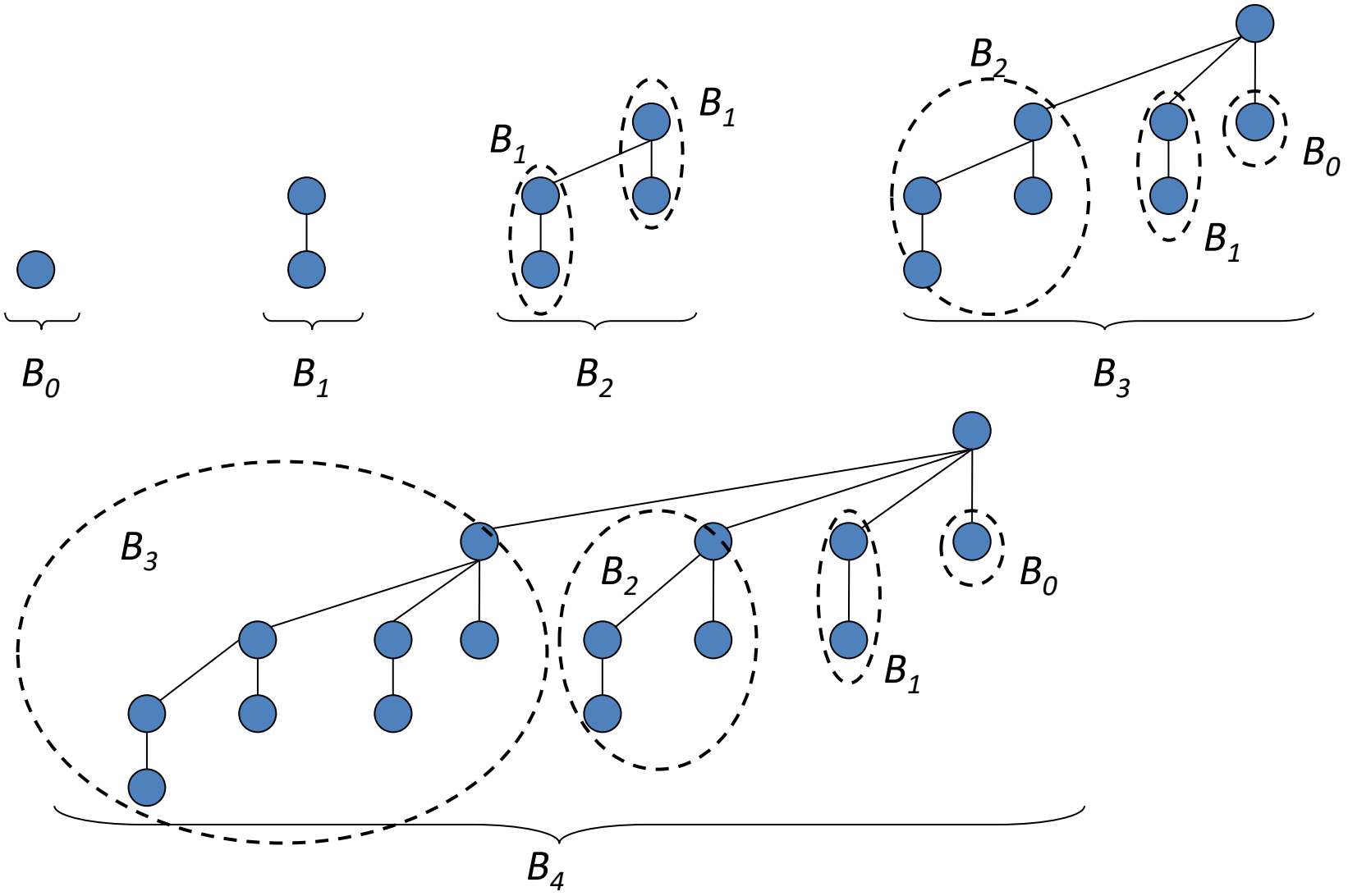
.

B_k két összekapcsolt B_{k-1} binomiális fából áll; az egyik fa gyökércsúcsa a másik fa gyökércsúcsának legbaloldalibb gyereke.

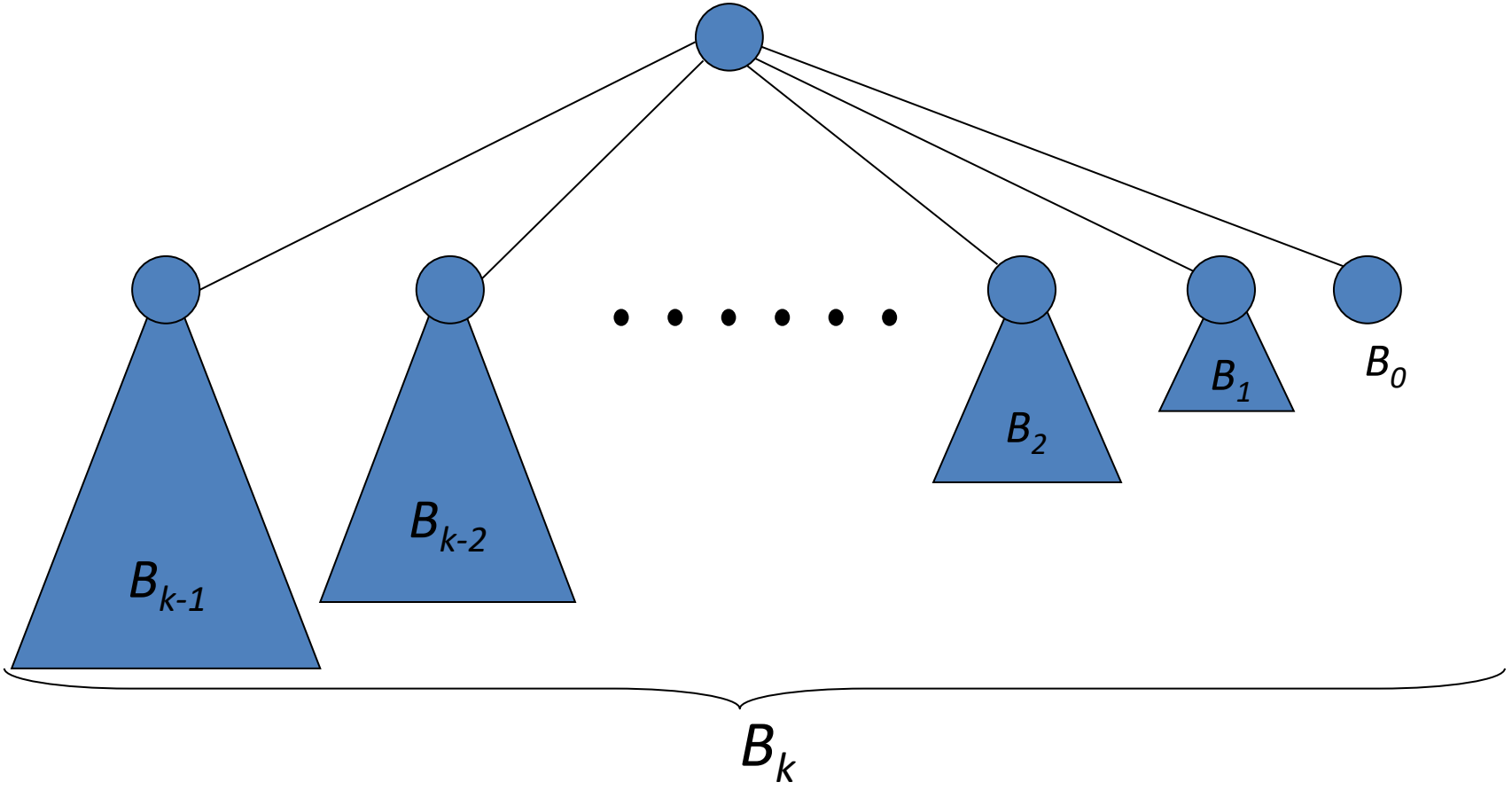
6. gyakorlat



6. gyakorlat



6. gyakorlat



6. gyakorlat

A B_k binomiális fa tulajdonságai

1. 2^k csúcsa van,
2. a magassága *k*,
3. az i-edik mélységben pontosan $\binom{k}{i}$ csúcs van, *i* = 0,1,..,*k,*
4. a gyökércsúcs fokszáma *k*, ami nagyobb, mint bármely másik csúcs fokszáma; továbbá,
5. ha a gyökércsúcs gyerekeit balról jobbra haladva megszámozzuk *k − 1, k − 2,. . . , 0-val*, akkor az i. gyerek a B_i részfa gyökércsúcsa.

6. gyakorlat

BİNOMİÁLİS KUPAC: binomiális fák olyan halmaza, amely kielégíti a következő binomiális-kupac tulajdonságokat.

1. H minden binomiális fája MİN(MAX)-KUPAC-RENDEZETT
   - egy csúcs kulcsa nagyobb(kisebb) vagy egyenlő, mint a szülőjének a kulcsa, azaz minden binomiális fa gyökércsúcsában van a fa legkisebb(legnagyobb) kulcsa.
2. H-ban legfeljebb egy olyan binomiális fa van, amelyikben a gyökércsúcsnak egy meghatározott fokszáma van.

6. gyakorlat

BİNOMİÁLİS KUPAC MŰVELETEI:

1. Minimum(maximum) keresése: a binomiális fák gyökércsúcsaiban a minimális(maximális) kulcs megkeresése.
2. SORBÓL(H)
   - a H gyökérlistájában az minimális(maximális) kulcsú *x* gyökérelem megkeresése, és ennek az elemnek a H gyökérlistájából való eltávolítása
   - H’ binomiális kupac létrehozása
   - az *x* gyerekeit tartalmazó egyirányú listában a láncolás sorrendjét fordítsuk meg, és mutasson fej[H’] az így kapott lista fejelemére
   - EGYESÍT(H,H’)

6. gyakorlat

BİNOMİÁLİS KUPAC MŰVELETEI:

3. SORBA(H,x)
   - H’ egy csúcsból (x) álló binomiális kupac létrehozása
   - EGYESÍT(H,H’)
4. KULCSOT-MODOSIT(B, kulcsindex, érték)
   - módosítsuk a kulcsot az új értékre, majd a gyökérig haladva ellenőrizzük, hogy az aktuális kulcs nagyobb-e(kisebb-e), mint a szülő, ha igen, akkor cseréljük fel őket.

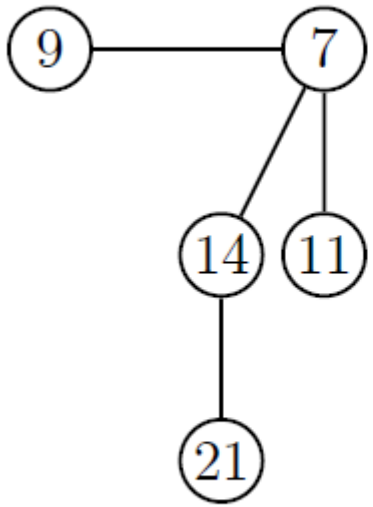
6. gyakorlat

BİNOMİÁLİS KUPAC MŰVELETEI:

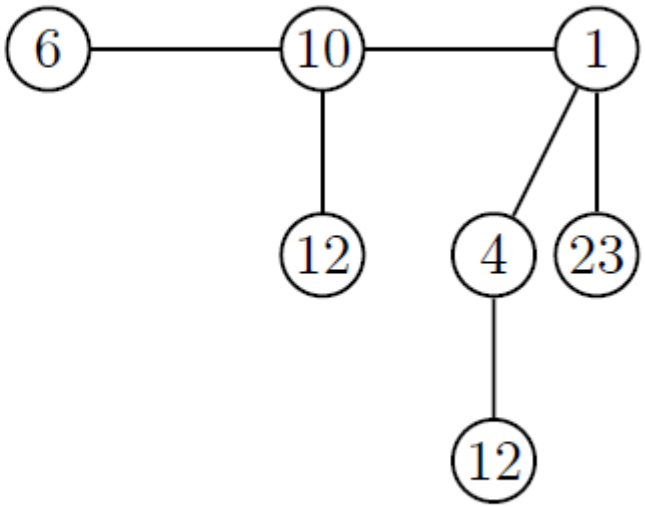
5. EGYESÍT(H1, H2):
   - Fésüljük össze H1 és H2 kupacot, legyen a kapott kupac H.
   - Szüntessük meg az azonos fokszámú fákat az alábbi módon:
     - x = H.fej
     - while(van következo fa)
       - a) ha x.fokszám != x.következo.fokszám, akkor x = x.következo
       - b) ha x.fokszám == x.következo.fokszám == x.következo.következo.fokszám, akkor x = x.következo
       - c) ha x.fokszám == x.következo.fokszám != x.következo.következo.fokszám, akkor kapcsoljuk össze az x és x.következo csúcsokat úgy, hogy a nagyobb (kisebb) kulcsú gyökércsúcs új fia legyen a kisebb(nagyobb) kulcsú gyökércsúcs, ezután x = újonnan kialakított fa

6. gyakorlat

Egyesítsük az alábbi két minimális binomiális kupacot!

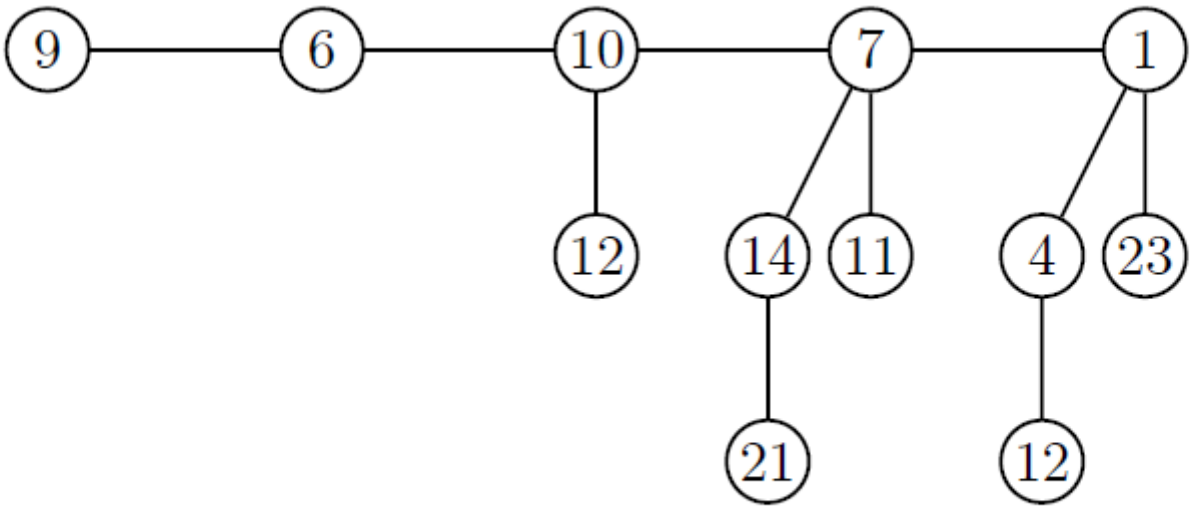


(q)



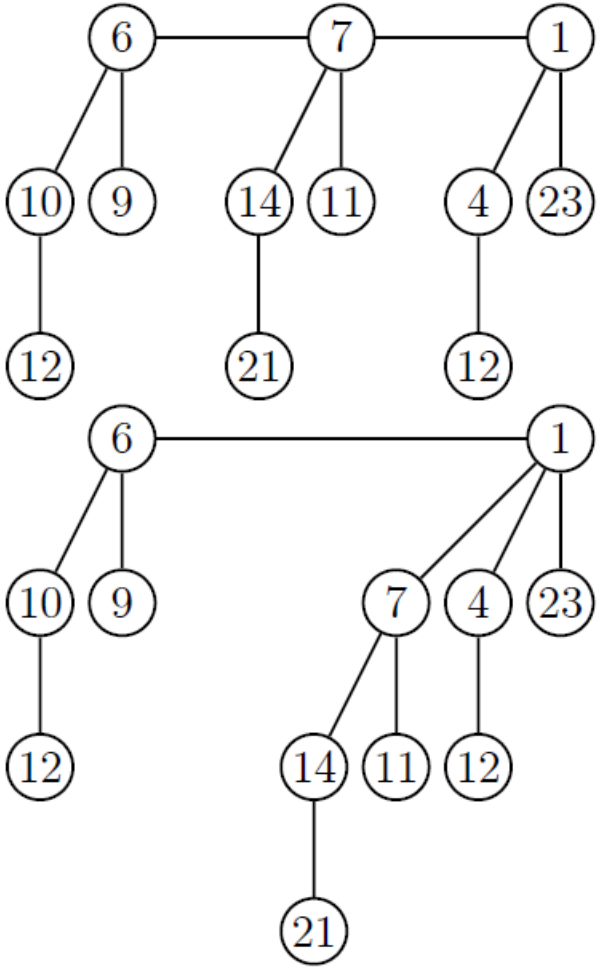
(r)

6. gyakorlat



(s) Az első összefésülő lépés után előálló kupac

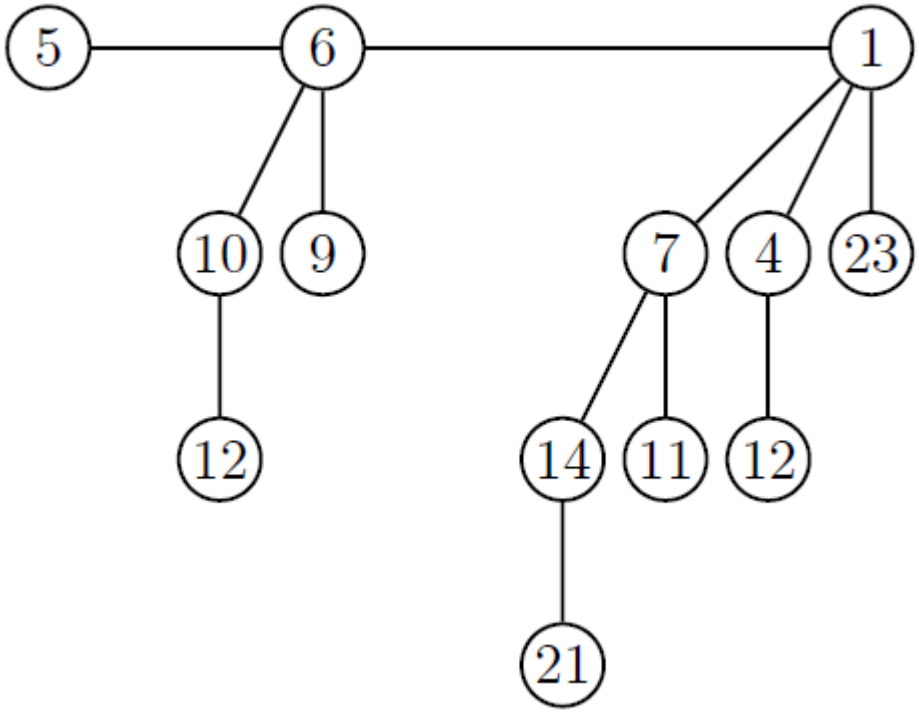
6. gyakorlat



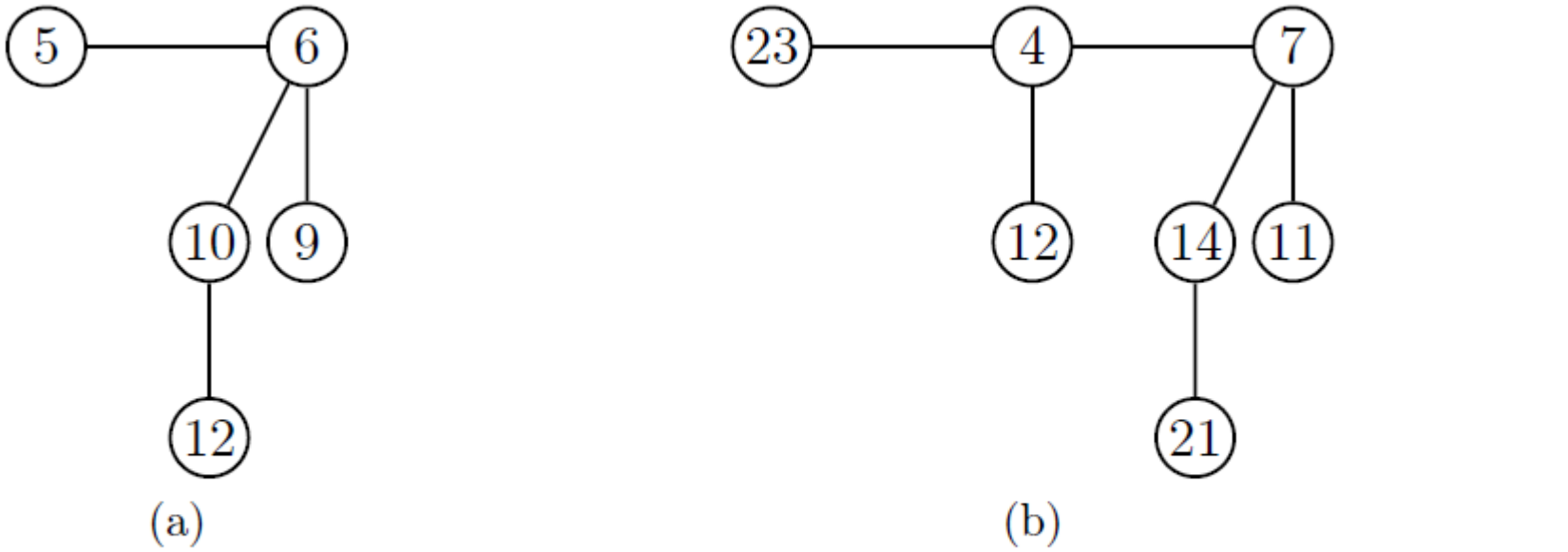
Egyesítés végeredményeként kapott kupac

6. gyakorlat

Az egyesített kupacra végezzük el a SORBA(5), SORBOL() műveleteket!

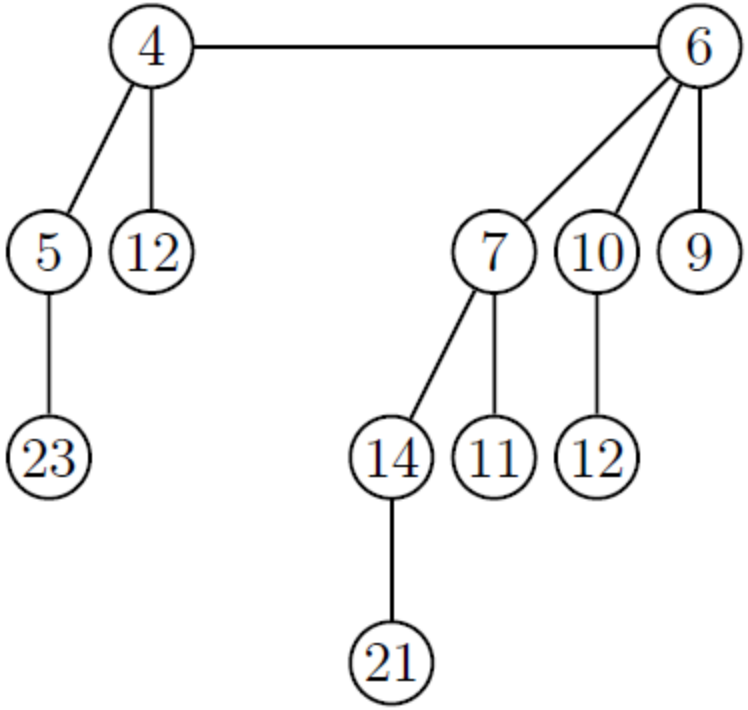


6. gyakorlat



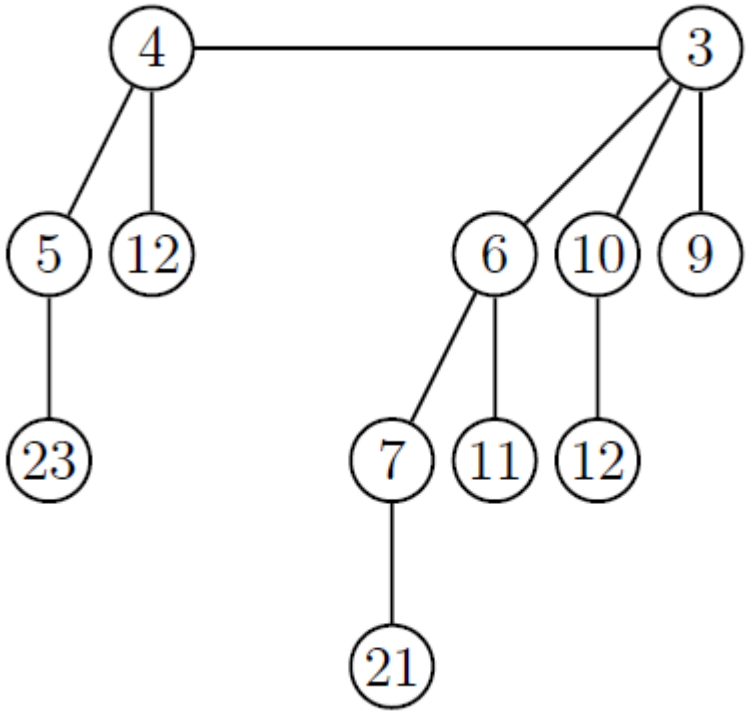
A minimum kulcs eltávolítása után egyesítendő binomiális kupacok.

6. gyakorlat

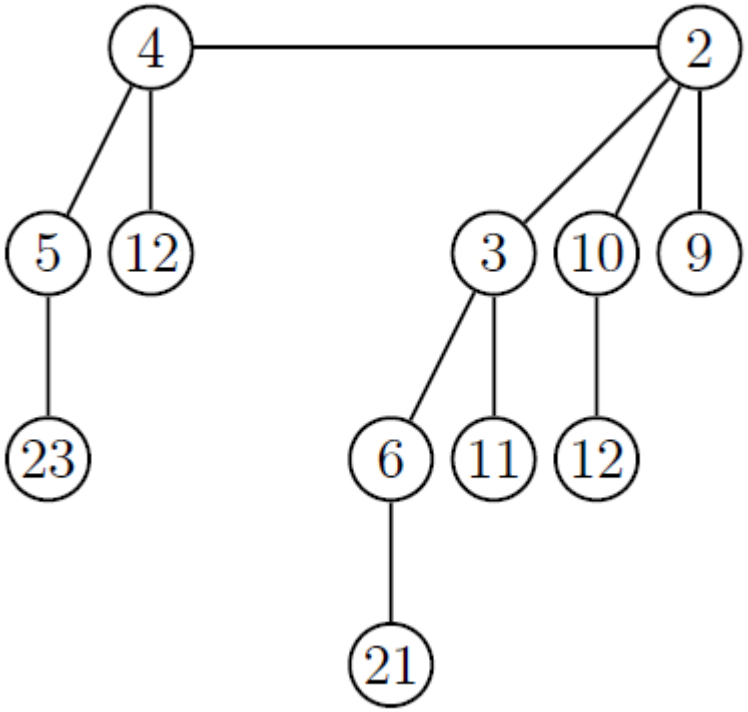


6. gyakorlat

Módosítsuk a 14-es kulcsot 3-ra, illetve a 7-es kulcsot 2-re.



6. gyakorlat



6. gyakorlat

Vizualizációk:

https://visualgo.net/en

https://www.cs.usfca.edu/~galles/visualization/Algorithms.html

http://people.ksp.sk/~kuko/bak/index.html

https://www.programiz.com/dsa