

Algoritmusok és adatstruktúrák 13. előadás

Pusztai Pál
pusztai@sze.hu

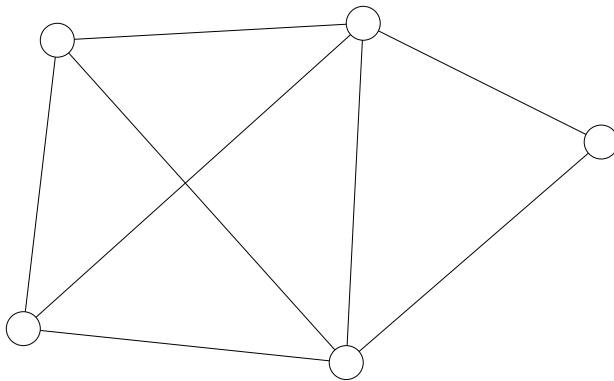
Tartalom

- Gráfok és fák
 - Alapfogalmak
 - Tárolás
 - Pontok, élek, fák, bináris fák, bináris kupacok
- Kidolgozott feladatok
 - Keresés bináris keresőfában
 - Keresőfa bővítése
 - Bináris keresőfa elemeinek listázása
 - Bináris kupac bővítése
 - Kupacrendezés

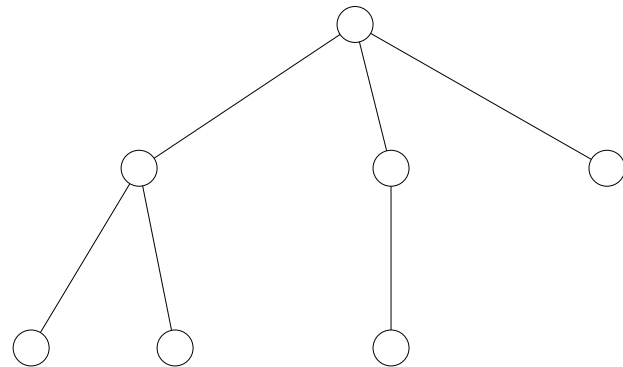


Gráfok

- **Gráfon** bizonyos elemek (**pontok**) és a köztük fennálló közvetlen kapcsolatok (**élek**) halmazát értjük.



Általános gráf



Fa

Gráfok

- **Irányított gráfról** akkor beszélünk, ha az élekhez irányítást rendelünk.
- **Egyszerű gráf** az olyan gráf, amelyben bármely két pont között egy irányban legfeljebb egy él van és nincs **hurokél** (egy pont önmagával vett kapcsolata).
- **Teljes gráf** az olyan gráf, amelyben bármely két pont között van él.
- **Út** a gráf egy olyan pontsorozata, amelyben a szomszédos pontok között van él.
- **Körmentes út** egy olyan út, amelyben minden pont különböző, egyébként (ha van ismétlődő pont) az út **körös út**.



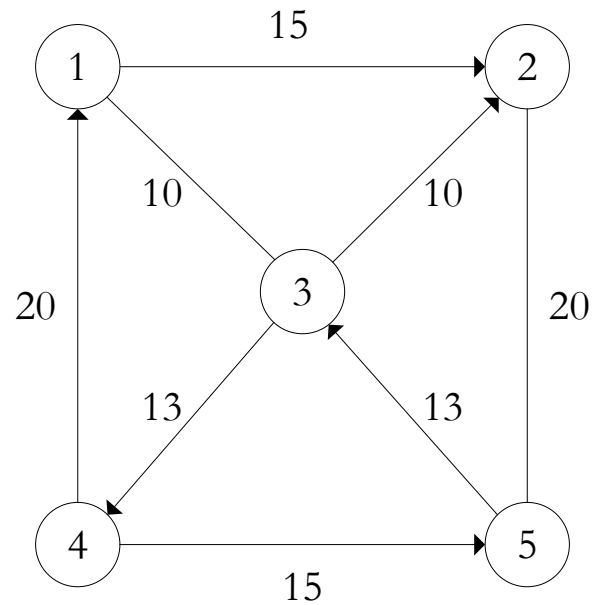
Gráfok

- Egy irányítatlan gráf **összefüggő**, ha bármely két pontja között van út.
- Fa struktúrájú gráf, vagy röviden **fa** az olyan összefüggő, irányítatlan gráf, amelyben nem hozható létre körös út.
- Egy gráf **részgráfján** értjük a pontok és a köztük lévő élek egy részhalmazát.
- Egy gráfot **szimmetrikusnak** nevezünk, ha irányítatlan, vagy minden irányított élnek van párja.



Tárolás

- **Feladat:** Tároljuk egy gráf adatait!



Példahálózat

Tárolás

Sorszám	1	2	3	4	5
Azonosító	1	2	3	4	5
X koordináta	0	15	8	0	15
Y koordináta	20	20	12	0	0

Pontok tárolása

Tárolás

Élhossz	1	2	3	4	5
1	-	15	10	-	-
2	-	-	-	-	20
3	10	10	-	13	-
4	20	-	-	-	15
5	-	20	13	-	-

Élhosszak tárolása kétdimenziós tömbben

Tárolás

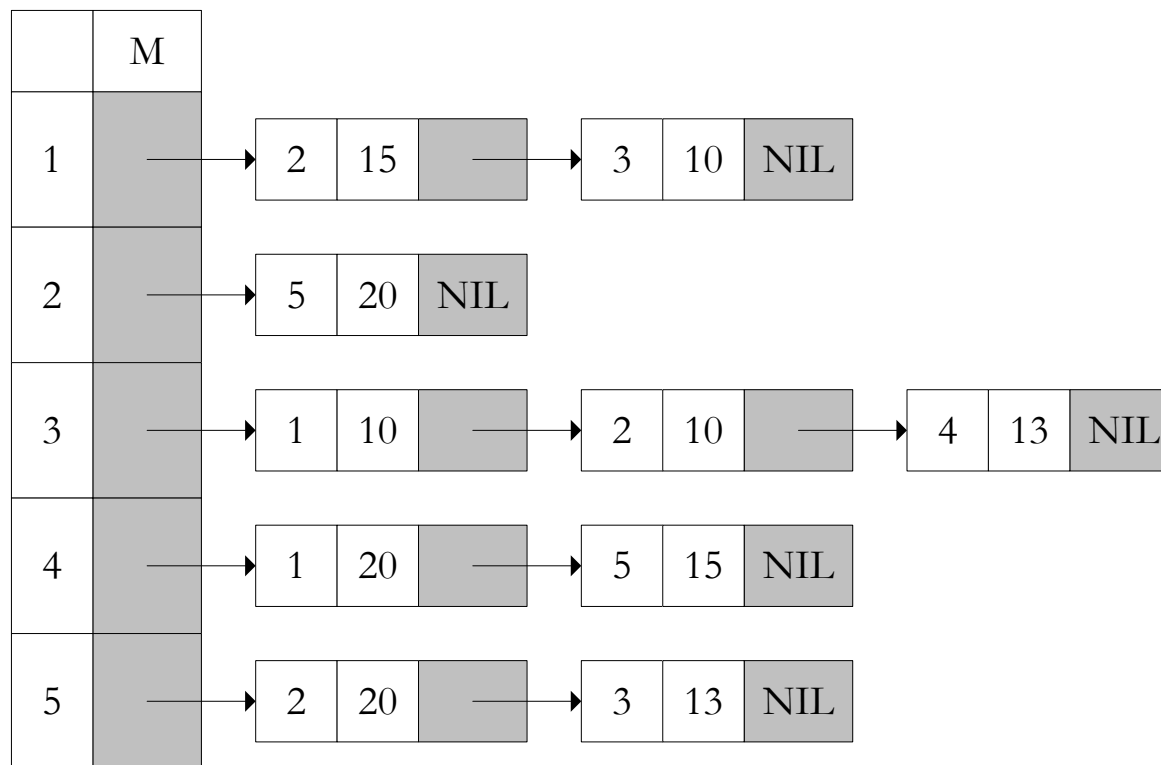
Sorszám	1	2	3	4	5	6
Élmutató	1	3	4	7	9	11

Pontok kiegészítése élmutatókkal

Sorszám	1	2	3	4	5	6	7	8	9	10
Végpont	2	3	5	1	2	4	1	5	2	3
Élhossz	15	10	20	10	10	13	20	15	20	13

Él adatok tárolása egydimenziós tömbben

Tárolás



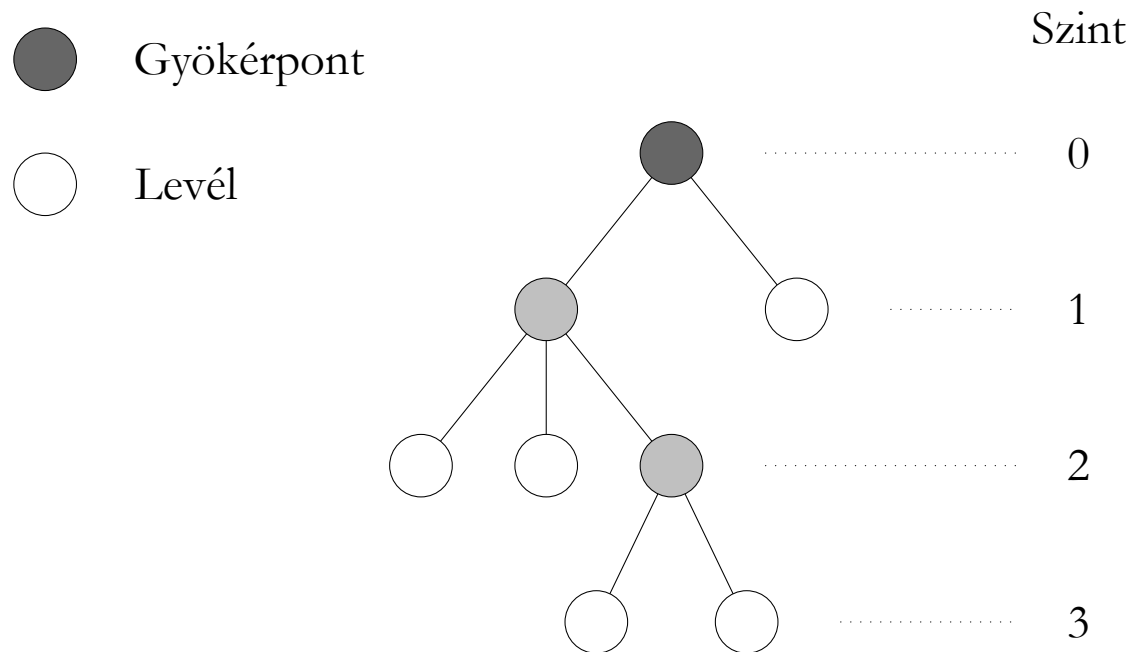
Élek tárolása dinamikusán

Fák

- Fa struktúrájú gráfon, vagy röviden **fán**, egy olyan összefüggő, irányítatlan gráfot értünk, amelyben nem hozható létre körös út.
- Néhány **ekvivalens** állítás a fákra vonatkozóan:
 - Bármely két pont között egyértelműen létezik egy őket összekötő út.
 - Éppen eggyel kevesebb él van, mint pont.
 - Akár csak egy élt hozzávéve az élekhez, a kapott gráf már tartalmaz kört, azaz nem lesz fa.
 - Akár csak egy élt is elhagyva az élek közül, a kapott gráf már nem lesz összefüggő, azaz már nem lesz fa.

Gyökeres fák

- Gyökeres fa** az olyan fa, amelyben egy pontot kitüntetünk. Ezt a fa **gyökérpontjának** nevezzük.

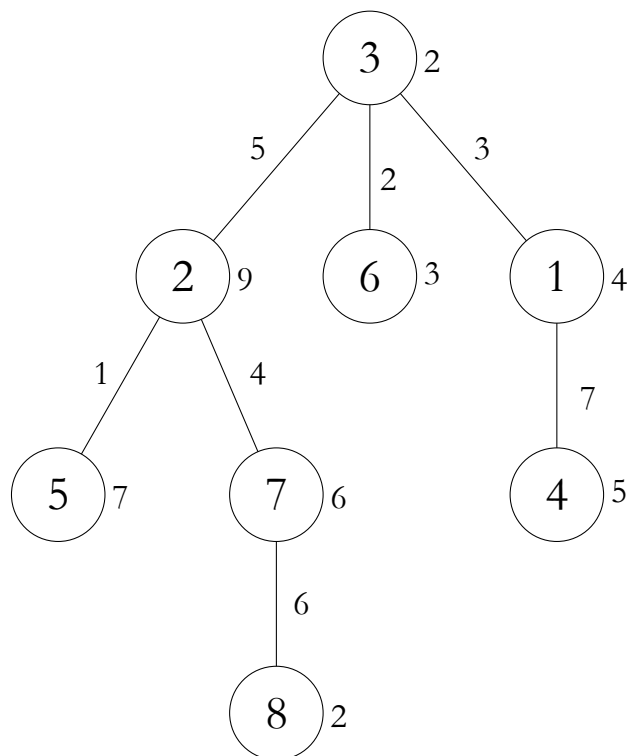


Gyökeres fa

Gyökeres fák

- A gyökérpont kivételével minden pontnak egyértelműen definiálható a **megelőzője** (szülő, ős), a gyökérpontból hozzávezető úton, az őt megelőző pont.
- Egy pont **követője** (gyermek, leszármazott) az a pont, amelynek ő a megelőzője.
- Egy fa bármely pontja, ha őt gyökérpontnak tekintjük, szintén meghatároz egy fát. Ezt a fát az eredeti fa **részfájának** nevezzük.
- **Rendezett fa** az olyan fa, amelyben a pontok leszármazottai között sorrendet értelmmezünk, azaz beszélünk első, második, stb. leszármazotról.

Gyökeres fák

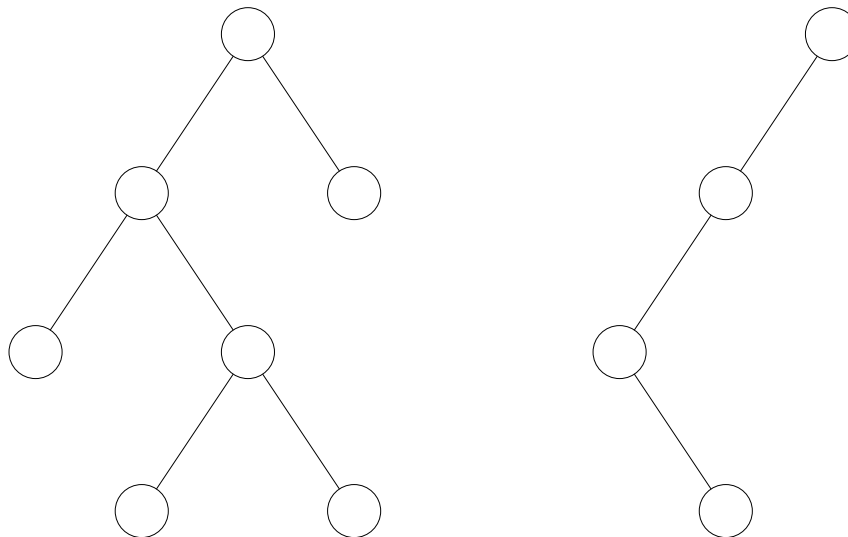


Pont	1	2	3	4	5	6	7	8
Címke	3	3	-	1	2	3	2	7
Pontjellemző	4	9	2	5	7	3	6	2
Éljellemző	3	5	-	7	1	2	4	6

Címketömb

Bináris fák

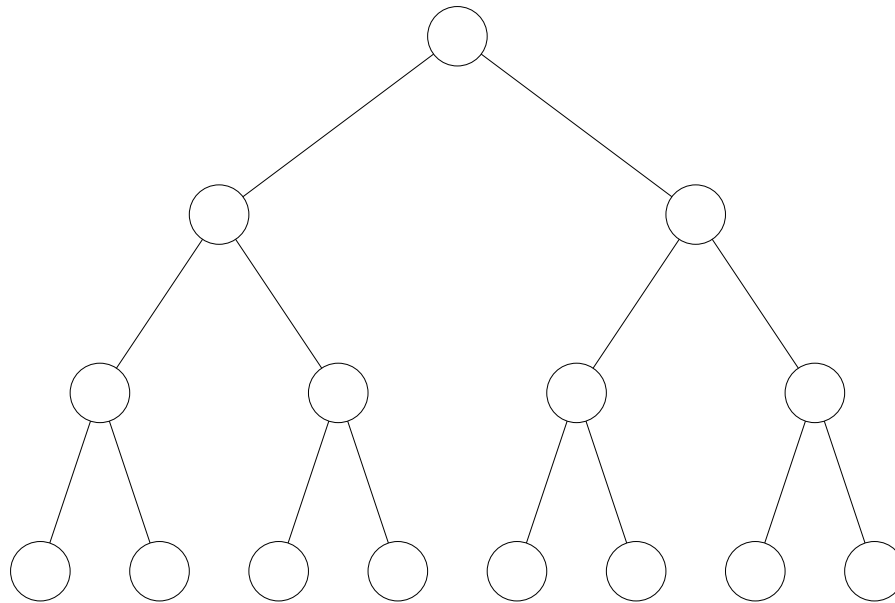
- **Bináris fa** az olyan rendezett fa, amelyben egy pontnak legfeljebb két leszármazottja van.
- A leszármazottakat **bal**, ill. **jobb leszármazottnak**, az általuk meghatározott részfat **bal**, ill. **jobb részfának** nevezzük.



Bináris fák

Bináris fák

- **Teljes bináris fa** az olyan bináris fa, amelyben az utolsó szinten lévő pontokat kivéve minden pontnak megvan mind a két leszármazottja, és az utolsó szinten csak levelek vannak.



Teljes bináris fa

A decorative graphic consisting of a grid of colored squares. The squares are arranged in a pattern that includes shades of blue, green, yellow, and red, with some squares being solid and others having a gradient.

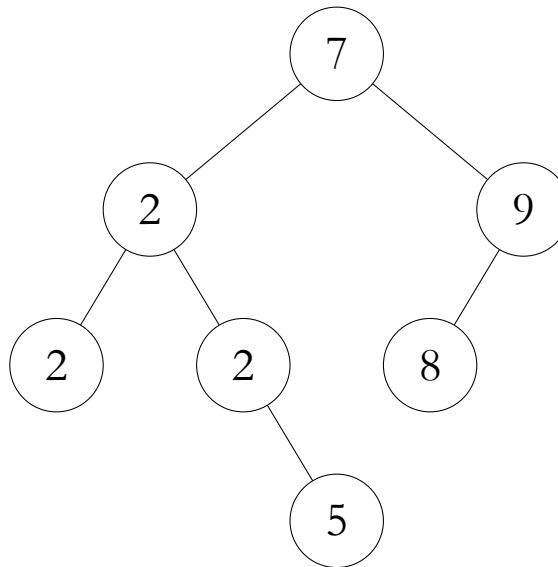
-
- A decorative graphic consisting of a grid of colored squares. The squares are arranged in a pattern that includes shades of blue, green, yellow, and red, with some squares being solid and others having a gradient.



A decorative graphic consisting of a grid of colored squares. The squares are arranged in a pattern that includes shades of blue, green, yellow, and red, with some squares being solid and others having a gradient.

Bináris fák

- **Bináris keresőfának** nevezünk egy olyan bináris fát, amelyben minden pontra igaz, hogy a ponthoz, mint gyökérponthoz tartozó **bal részfa** pontértékei **nem nagyobbak**, a **jobb részfa** pontértékei **nem kisebbek** a pont értékénél.



Bináris keresőfa

Bináris fák

■ Típus

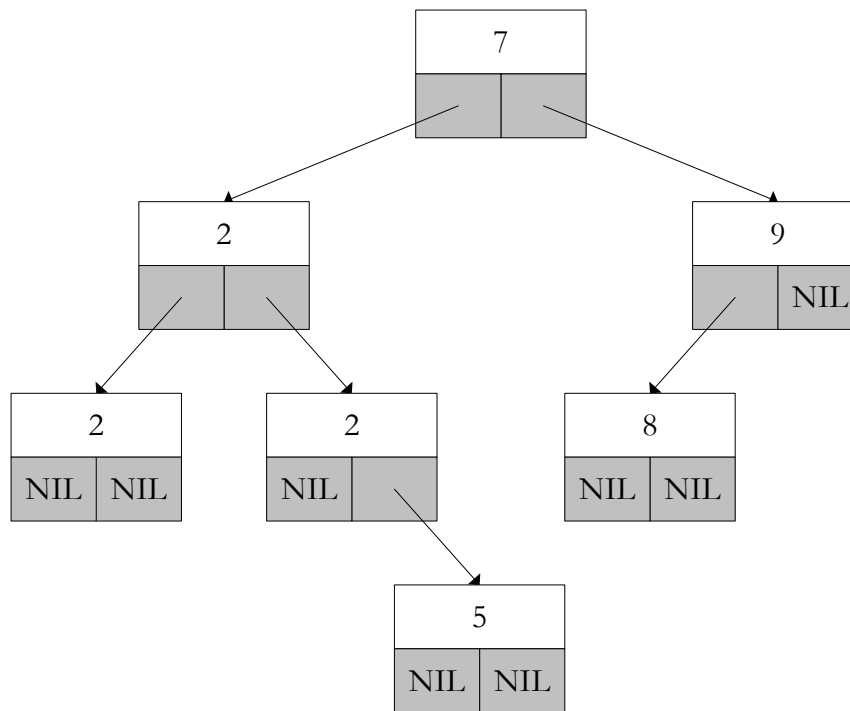
BINFAPONT Rekord

PONTJELL

BALAG, JOBBAG

Pontjellemző típus

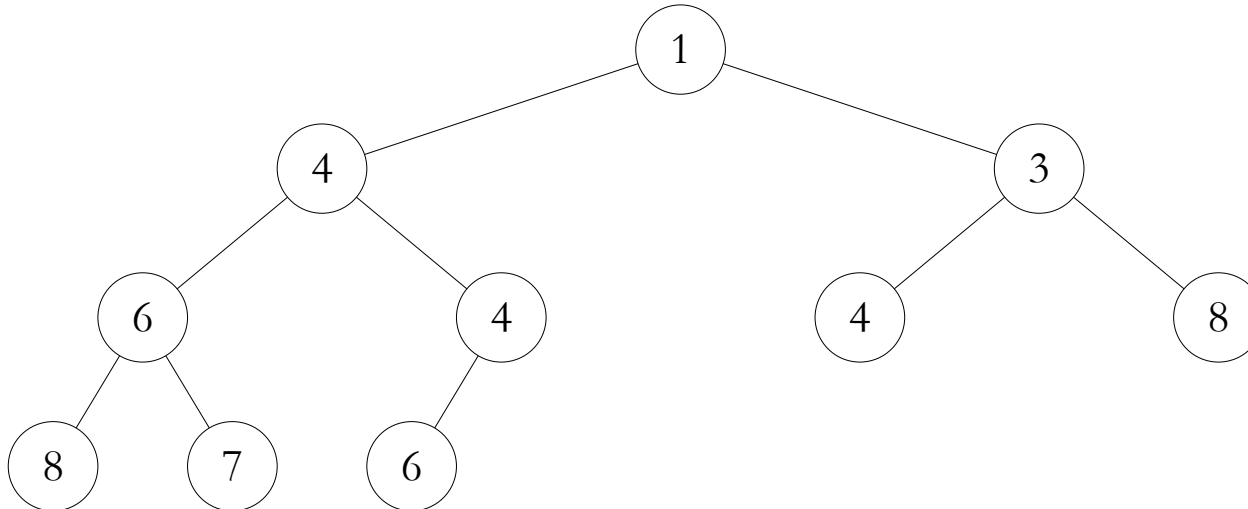
BINFAPONT rekordra mutató



Dinamikus adatszerkezettel megadott bináris fa

Bináris fák

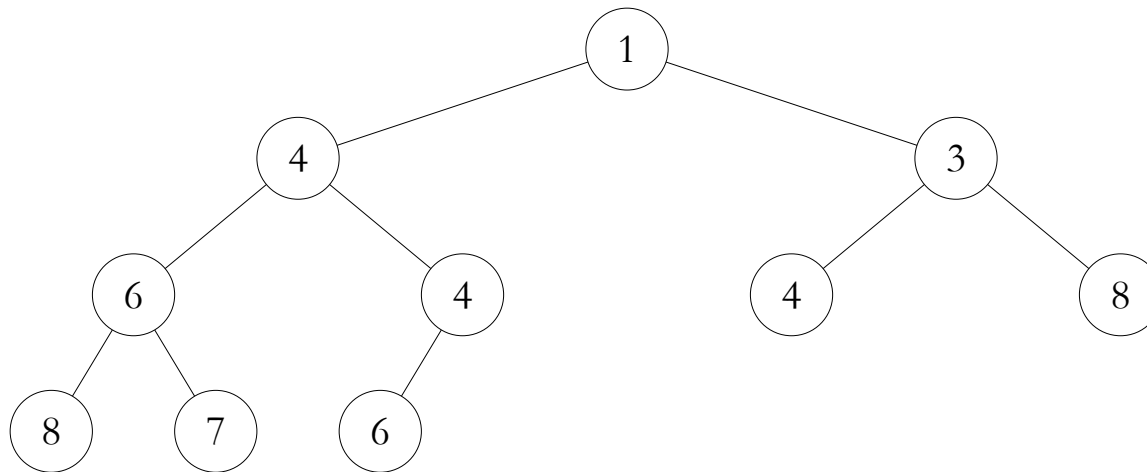
- **Bináris kupacnak** vagy röviden **kupacnak** nevezünk egy olyan, majdnem teljes bináris fát, amelyben minden pontra igaz, hogy a ponthoz, mint gyökerponthoz tartozó részfa pontértékei **nem kisebbek** a pont értékénél.



Bináris kupac

Bináris fák

- Kupac tárolása:
 - A gyökérpont indexe 1.
 - Ha egy pont indexe i , akkor a bal gyermek a $2*i$, a jobb gyermek a $2*i+1$ index alatt tárolódik.



Sorszám	1	2	3	4	5	6	7	8	9	10
Pontjellemző	1	4	3	6	4	4	8	8	7	6

Kupactömb

Bináris fák

- **Feladat:** Keressünk meg egy értéket egy (ezen érték szerinti) bináris keresőfában!



Bináris fák

- **Feladat:** Keressünk meg egy értéket egy (ezen érték szerinti) bináris keresőfában!

- **Típus**

BINFAPONT Rekord

PONTJELL

Pontjellemző típus

BALAG, JOBBAG

BINFAPONT rekordra mutató

Funkció	Azonosító	Típus	Jelleg
A fa gyökérpontja	GYOKER	BINFAPONT-ra mutató	I
A keresett érték	X	Pontjellemző típus	I
Az éppen vizsgált pont	AKT	BINFAPONT-ra mutató	M, O

Bináris fák

```
/* Keresés bináris keresőfában */  
KERES(GYOKER,X)  
AKT ← GYOKER  
while (AKT<>NIL) AND (X<>(*AKT).PONTJELL)  
    if X<(*AKT).PONTJELL  
        AKT ← (*AKT).BALAG  
    else  
        AKT ← (*AKT).JOBAG  
return AKT
```

Bináris fák

- **Feladat:** Bővítsünk egy keresőfát egy új ponttal!



Bináris fák

- **Feladat:** Bővítsünk egy keresőfát egy új ponttal!

Funkció	Azonosító	Típus	Jelleg
A fa gyökérpontja	GYOKER	BINFAPONT-ra mutató	I, O
Az új pont	X	BINFAPONT-ra mutató	I
Az éppen vizsgált pont	AKT	BINFAPONT-ra mutató	M
A szülő pont	SZULO	BINFAPONT-ra mutató	M

Bináris fák

```
/* Bináris keresőfa bővítése */  
BOVIT(GYOKER,X)  
if GYOKER=NIL  
    /* Üres fa */  
    GYOKER ← X  
else  
    /* Helykeresés */  
    AKT ← GYOKER  
    SZULO ← NIL  
    while AKT<>NIL  
        SZULO ← AKT  
        if (*X).PONTJELL<(*AKT).PONTJELL  
            AKT ← (*AKT).BALAG  
        else  
            AKT ← (*AKT).JOBAG  
    /* Beillesztés */  
    if (*X).PONTJELL<(*SZULO).PONTJELL  
        (*SZULO).BALAG ← X  
    else  
        (*SZULO).JOBAG ← X
```



Bináris fák

- **Feladat:** Írjuk ki egy bináris fa összes elemét a képernyőre!



Bináris fák

- **Feladat:** Írjuk ki egy bináris fa összes elemét a képernyőre!

Funkció	Azonosító	Típus	Jelleg
A fa gyökérpontja	GYOKER	BINFAPONT-ra mutató	I

```
/* Bináris fa elemeinek kiírása rekurzív eljárással */
```

```
KIIR(GYOKER)
```

```
if GYOKER<>NIL
```

```
    /* Rekurzív hívás a bal ágra */
```

```
    KIIR((*GYOKER).BALAG)
```

```
    /* A gyökérponthoz tartozó érték kiírása */
```

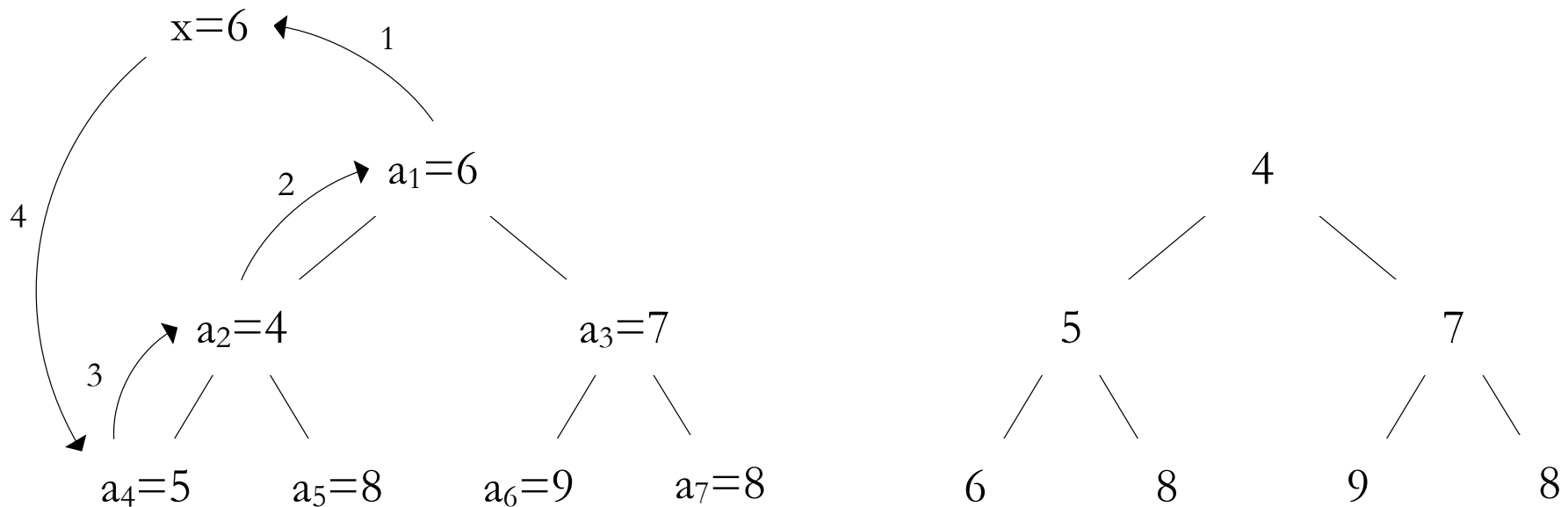
```
    Ki: (*GYOKER).PONTJELL
```

```
    /* Rekurzív hívás a jobb ágra */
```

```
    KIIR((*GYOKER).JOBAG)
```

Bináris kupacok

- Feladat:** Bővítsük az $a_{l+1}, a_{l+2}, \dots, a_r$ kupactulajdonsággal rendelkező elemeket egy új a_l elemmel úgy, hogy a kupactulajdonság megmaradjon!



Elem besüllyesztése

Bináris kupacok

■ Típus

ELEM Egész

/* Az elemek típusa */

TOMB Egydimenziós ELEM tömb

/* Az elemeket tároló tömb típusa */

Funkció	Azonosító	Típus	Jelleg
Az elemek	A	TOMB	I, M, O
A besüllyesztendő elem indexe	L	Egész	I
Az utolsó elem indexe	R	Egész	I
A besüllyesztendő elem tárolására	X	ELEM	M
A vizsgált elem indexe	I	Egész	M
A kisebbik gyerek indexe	J	Egész	M
Helyére süllyedt-e az elem	VEGE	Logikai	M

Bináris kupacok

/* A[L] besüllyesztése az A[L+1],...,A[R] elemek közé */

SULLYESZT(A,L,R)

X ← A[L]

I ← L

J ← 2*I

VEGE ← hamis

/* Helykészítés */

while (J ≤ R) AND NOT VEGE

/* A kisebb gyerekkel hasonlítsunk */

if (J < R) AND (A[J+1] < A[J])

J ← J+1

if X ≤ A[J]

/* Megvan a helye */

VEGE ← igaz

else

/* A gyerek feljebb léptetése */

A[I] ← A[J]

I ← J

J ← 2*I

/* Elemet a helyére */

A[I] ← X



Kupacrendezés

- **Feladat:** Bináris kupac adatstruktúra felhasználásával készítsünk rendező algoritmust!



Kupacrendezés

- **Feladat:** Bináris kupac adatstruktúra felhasználásával készítsünk rendező algoritmust!

Funkció	Azonosító	Típus	Jelleg
A rendezendő elemek	A	TOMB	I, M, O
Az elemek száma	N	Egész	I
A kupac első elemének indexe	L	Egész	M
A kupac utolsó elemének indexe	R	Egész	M
Két elem cseréjéhez	X	ELEM	M

Kupacrendezés

KUPACREND(A,N)

$L \leftarrow N \text{ DIV } 2 + 1$

$R \leftarrow N$

while $L > 1$

$L \leftarrow L - 1$

SULLYESZT(A,L,R)

while $R > 1$

/* Legkisebbet hátra */

$X \leftarrow A[1]$

$A[1] \leftarrow A[R]$

$A[R] \leftarrow X$

/* Kupacot kisebbre */

$R \leftarrow R - 1$

/* A hátul volt elem besüllyesztése */

SULLYESZT(A,L,R)