7. előadás

Hierarchikus adatszerkezetek

Fa, bináris fa, bejárások

Adatszerkezetek és algoritmusok előadás
2018. március 20.

Kósa Márk, Pánovics János és Szathmáry László
Debreceni Egyetem
Informatikai Kar

Hierarchikus adatszerkezetek

Hierarchikus adatszerkezet

A szekvenciális adatszerkezet általánosítása: minden adatelemének – egyet kivéve – pontosan egy megelőzője, és tetszőleges számú (akár 0) rákövetkezője lehet.

Hierarchikus adatszerkezetek:

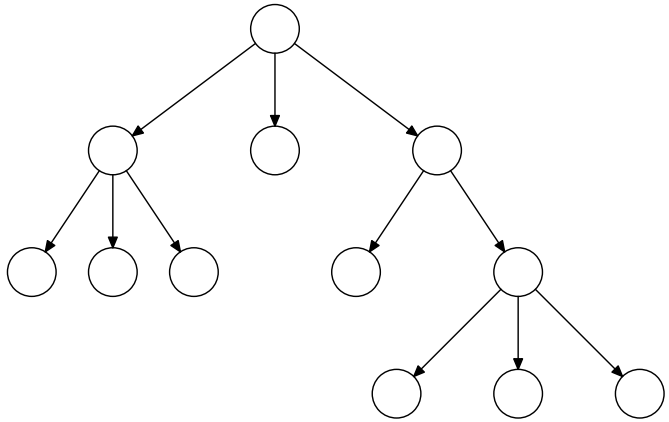
- fa
- hierarchikus lista

Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság

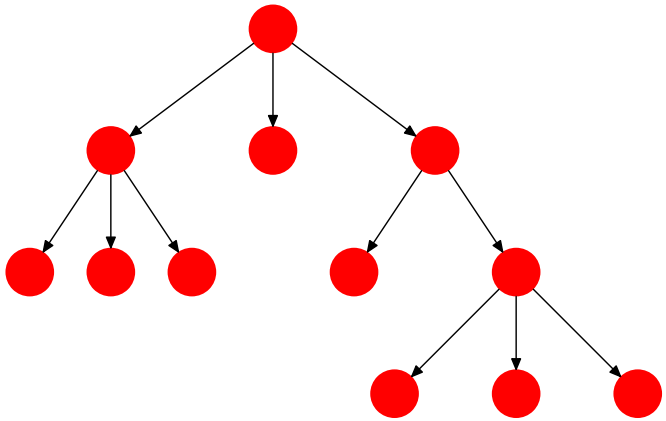


Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

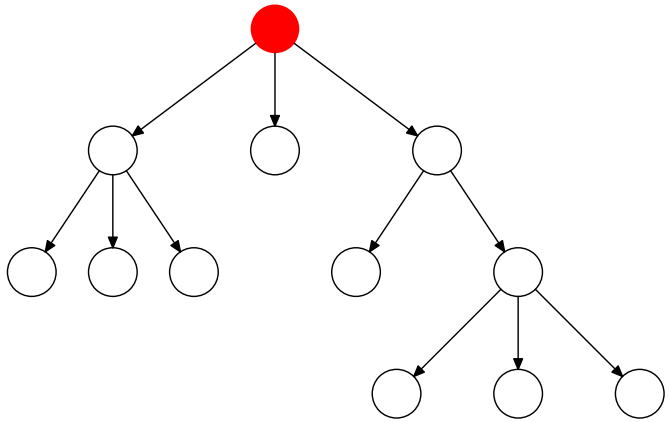
- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság



A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

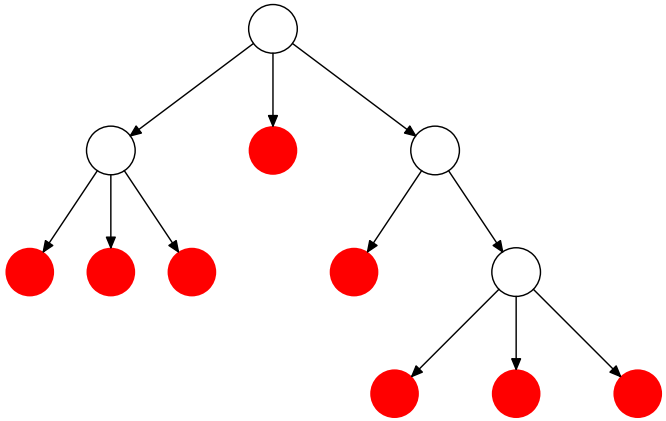
- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság



A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság

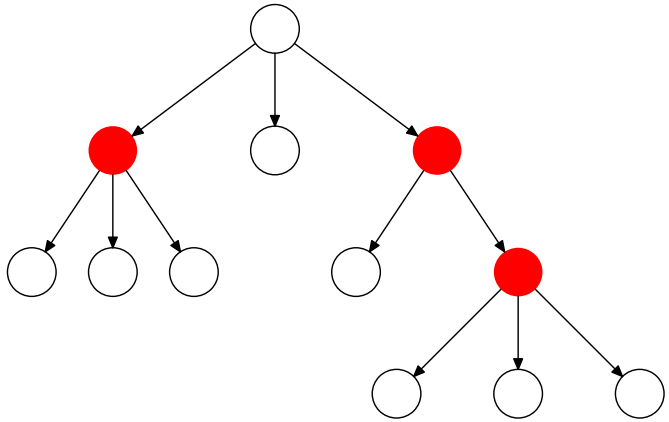


Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság

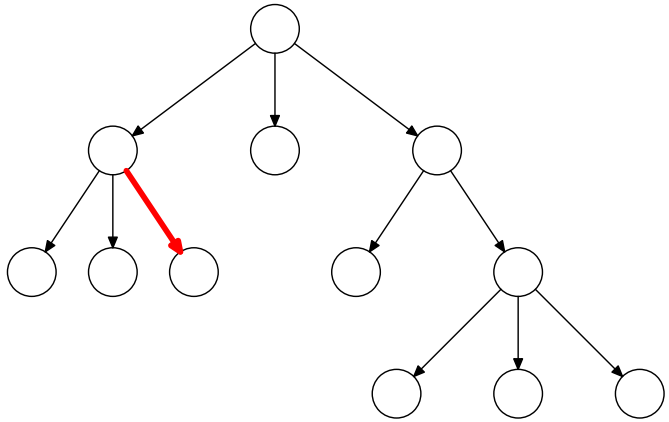


Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság

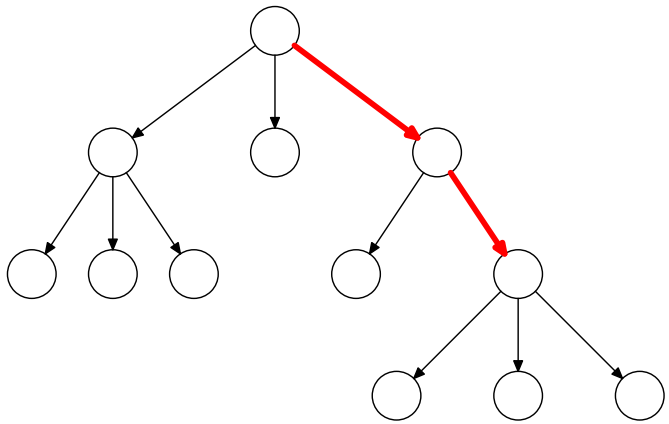


Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság

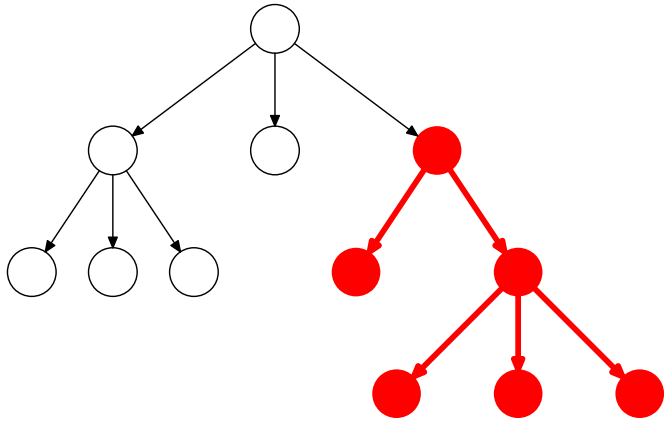


Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság

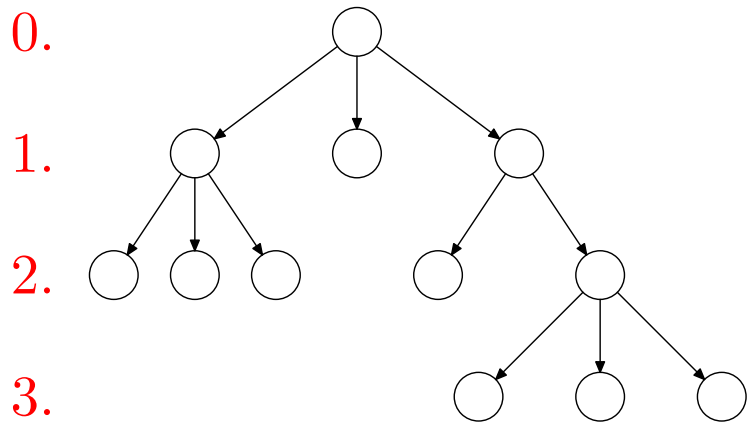


Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság

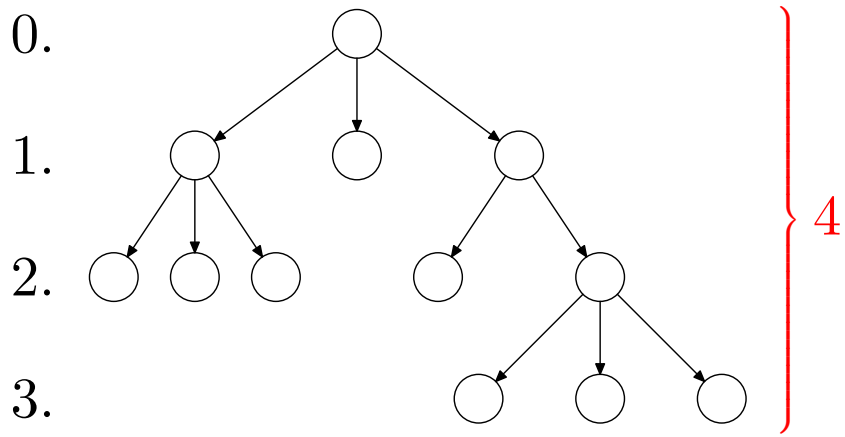


Kósa Márk
Pánovics János
Szathmáry László

A fa adatszerkezet

Homogén, dinamikus, hierarchikus adatszerkezet. Fa adatszerkezetekkel kapcsolatos fogalmak:

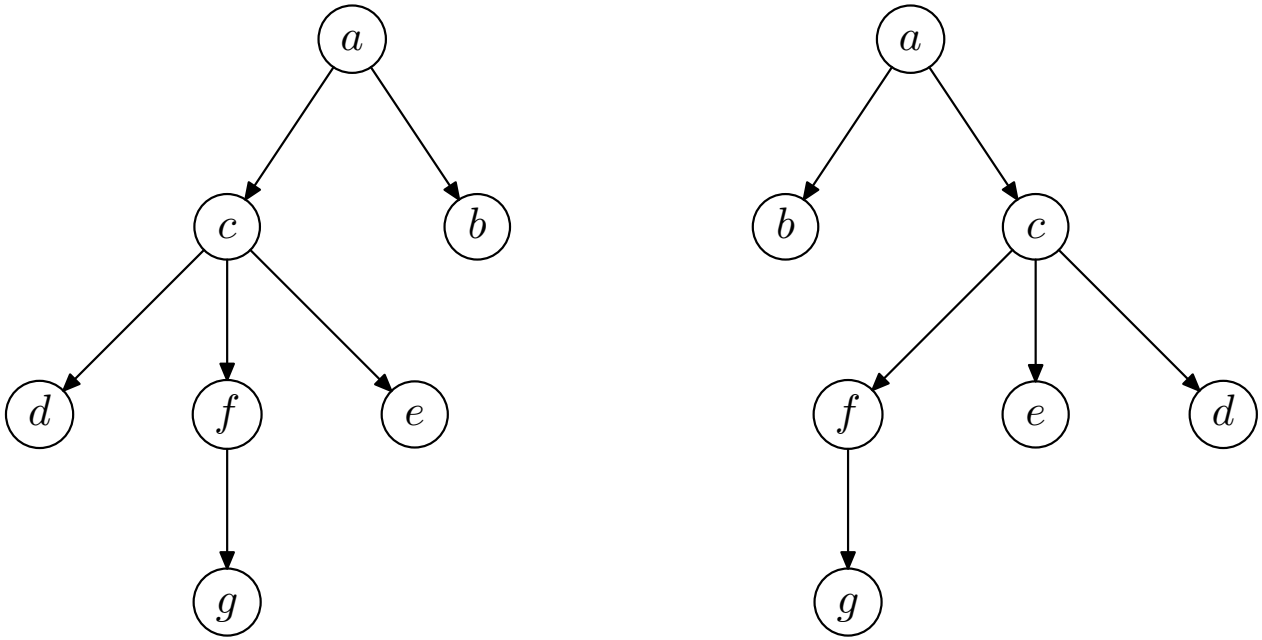
- csúcs, csomópont
- gyökérelem
- levélelem
- közbenső elem
- él
- út
- részfa
- szint
- magasság



Kósa Márk
Pánovics János
Szathmáry László

Rendezetlen és rendezett fák

Rendezetlen fáknál nem lényeges az ugyanazon csúcsból kiinduló élek sorrendje, rendezett fáknál viszont igen.



Az ábrán látható két fa ekvivalens egymással, ha eltekintünk az ugyanazon csúcsokból kiinduló élek sorrendjétől (azaz ha rendezetlen fáknak tekintjük őket).

Mivel az informatikában a rendezett adatszerkezetek játszanak fontos szerepet, a továbbiakban rendezett fákkal foglalkozunk.

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa

Bináris fa

Olyan fa, melyben minden adatelemnek legfeljebb két rákövetkezője van.

Szigorú értelemben vett bináris fa

Szigorú értelemben vett bináris fáról beszélünk, ha a bináris fában minden adatelemnek 0 vagy 2 rákövetkezője van.

Rendezett bináris fa

Rendezett bináris fa elemeire értelmezhetők a következő fogalmak:

- bal/jobb oldali rákövetkező
- bal/jobb oldali részfa

A továbbiakban bináris fa alatt – hacsak mást nem mondunk – rendezett bináris fát értünk.

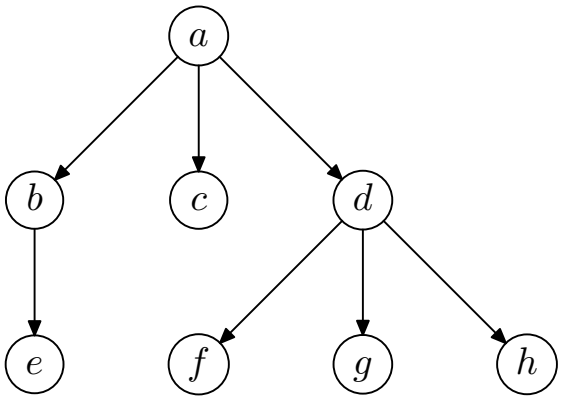
Kósa Márk
Pánovics János
Szathmáry László

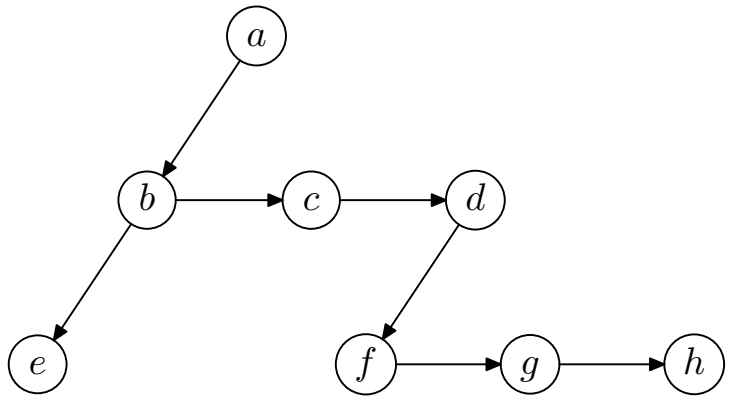
Nem bináris fa bináris fává alakítása

Minden nem bináris fa reprezentálható bináris fával.

Tetszőleges fa binarizálásának algoritmusa

1. Legyen a bináris fa gyökere a nem bináris fa gyökere.
2. A bináris fa egy tetszőleges elemének bal oldali rákövetkezője legyen a nem bináris fa megfelelő elemének bal oldali (első) rákövetkezője.
3. A bináris fa egy tetszőleges elemének jobb oldali rákövetkezője legyen a nem bináris fa megfelelő elemének következő (azonos szülőhöz tartozó) testvércsúcsa.





Kósa Márk
Pánovics János
Szathmáry László

Bináris fával végezhető műveletek

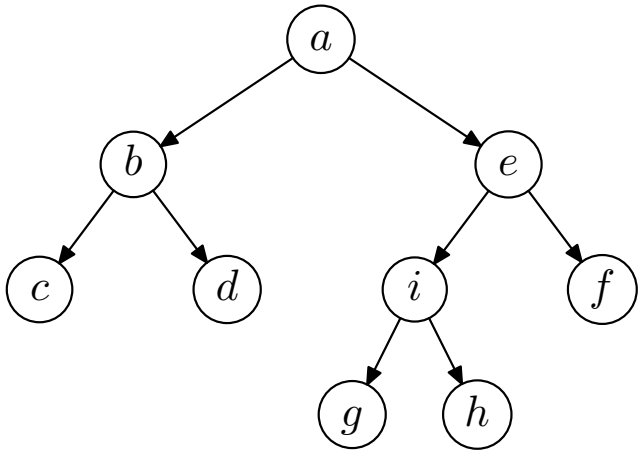
- Létrehozás: üres fa.
- Bővítés: egy elemmel vagy egy részfával, általában levélelemnél.
- Törlés: részfát vagy egy elemet, utóbbi esetben a fát a legtöbb esetben újra kell szervezni (hogy továbbra is fa maradjon).
- Csere: megengedett.
- Rendezés: nincs.
- Keresés, elérés és feldolgozás: a bejárás algoritmusa alapján.
- Bejárás: szokás szerint olyan algoritmus, amelynek segítségével a bináris fa elemeit leképezzük egy sorra (preorder, inorder vagy postorder módon).

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.

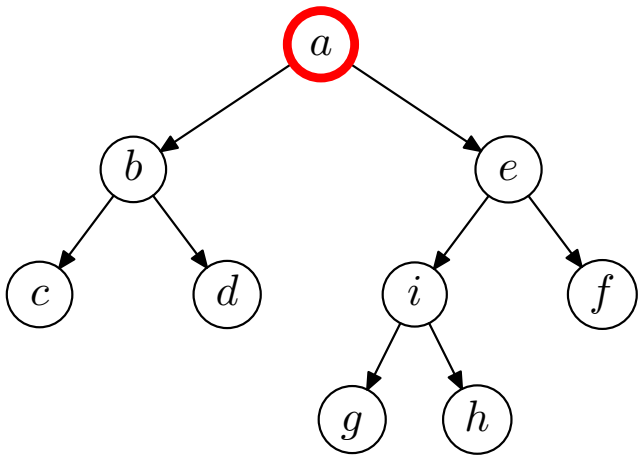


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.

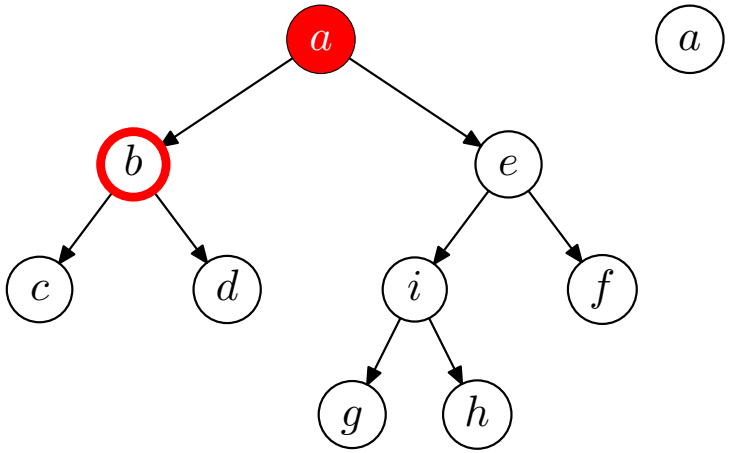


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.

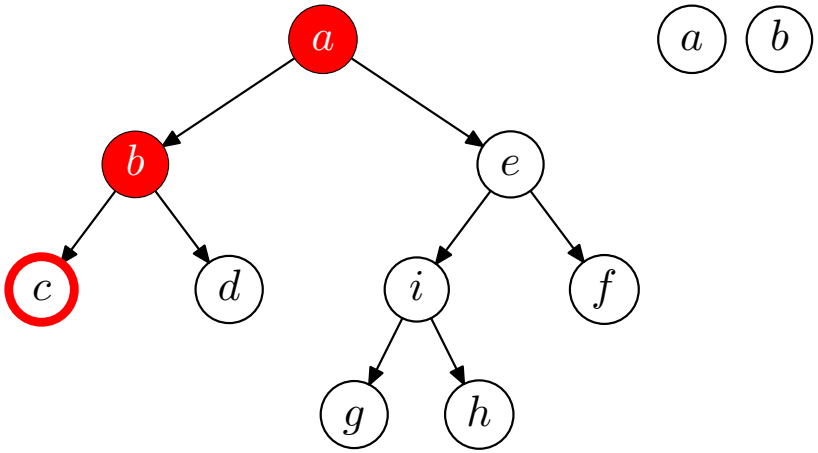


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.

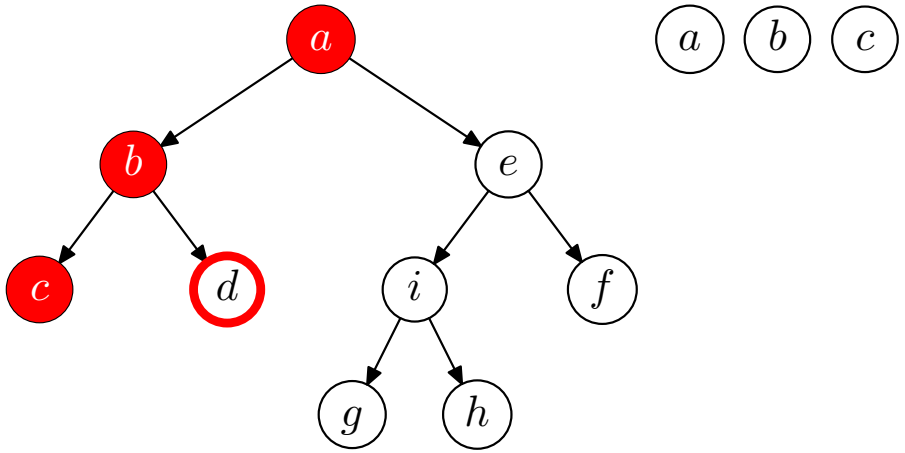


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.

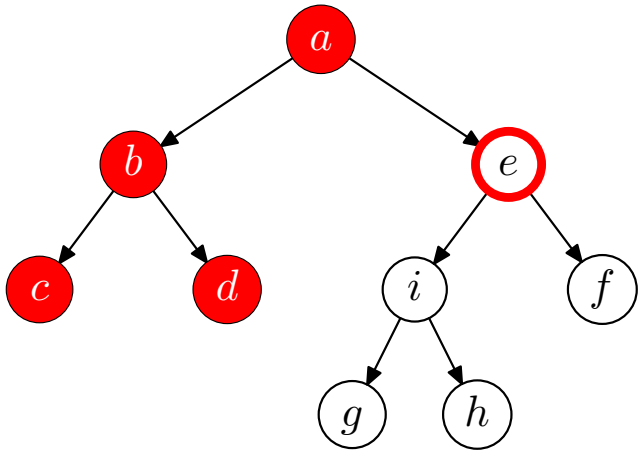


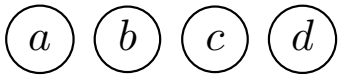
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.



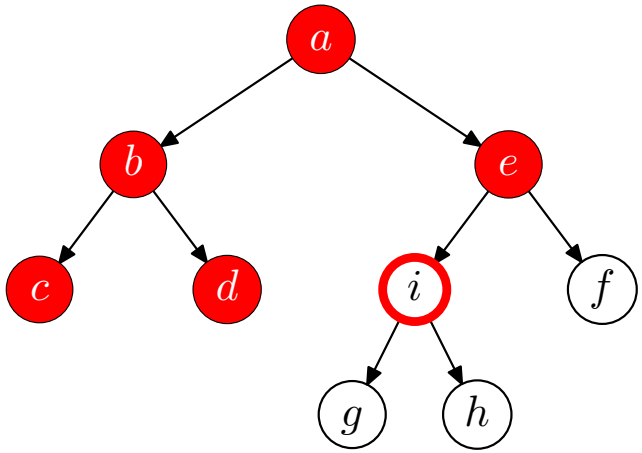


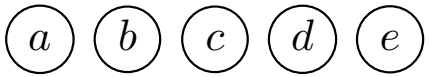
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.



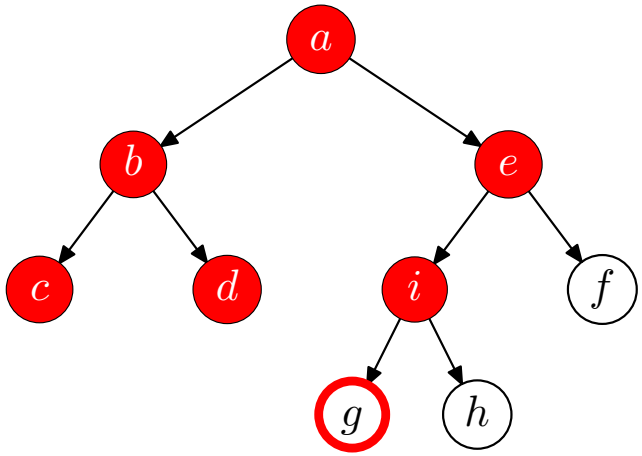


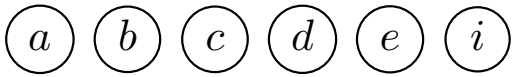
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.



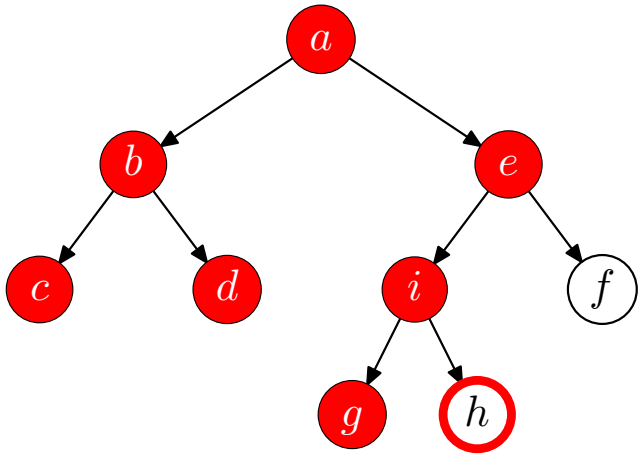


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.

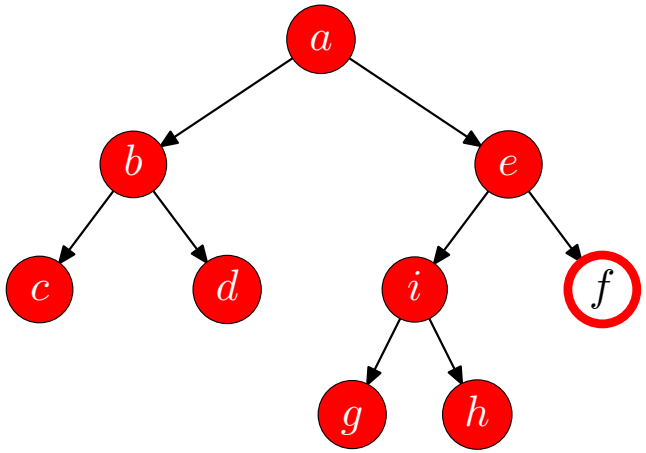


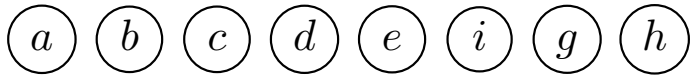
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.



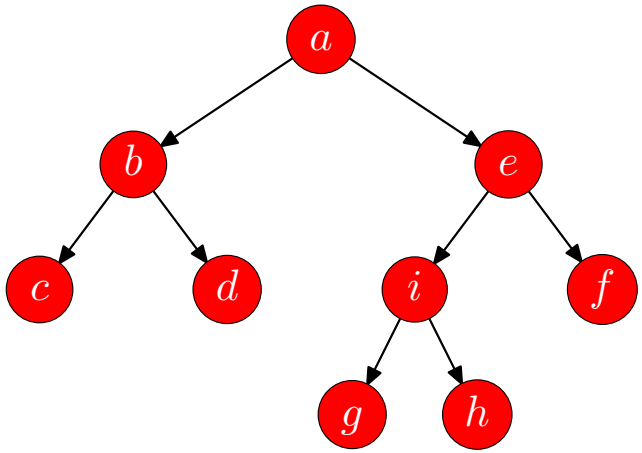


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Preorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
3. Járjuk be a gyökérelem bal oldali részfáját preorder módon.
4. Járjuk be a gyökérelem jobb oldali részfáját preorder módon.

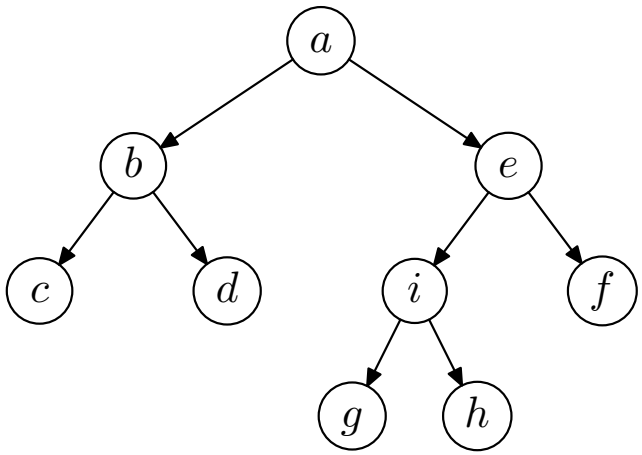


a b c d e i g h f

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.

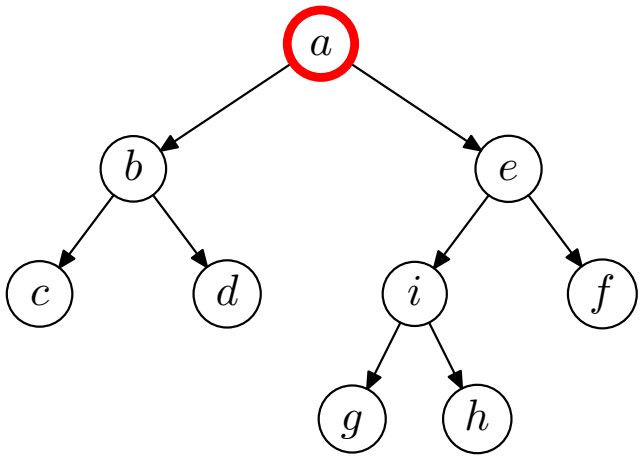


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.

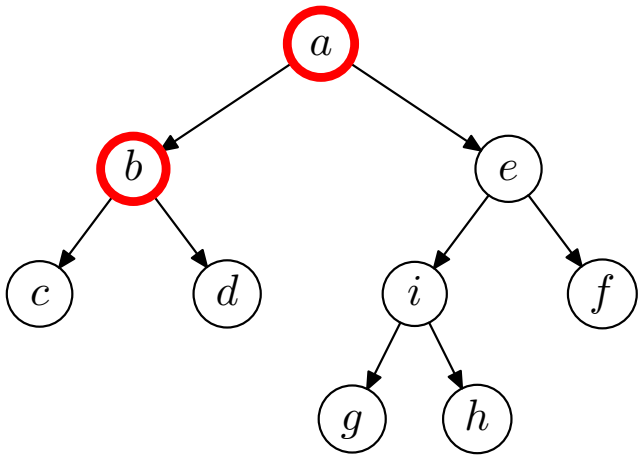


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.

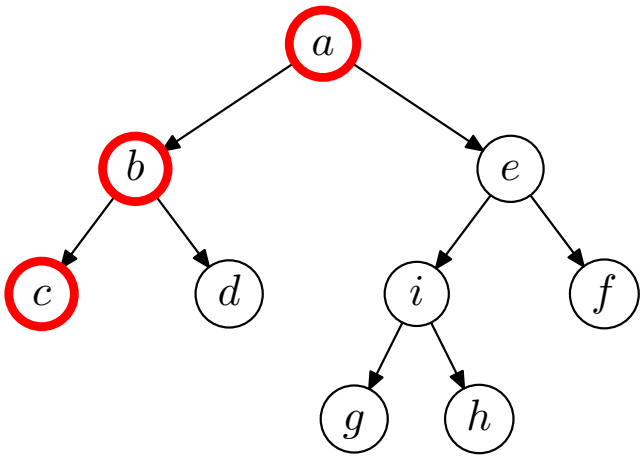


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.

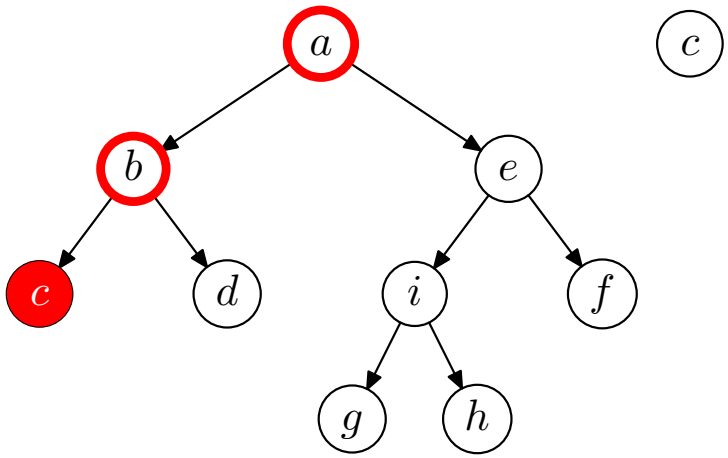


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.

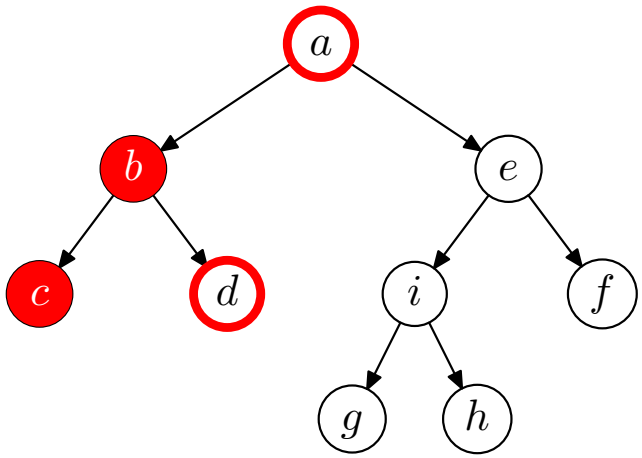


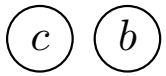
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



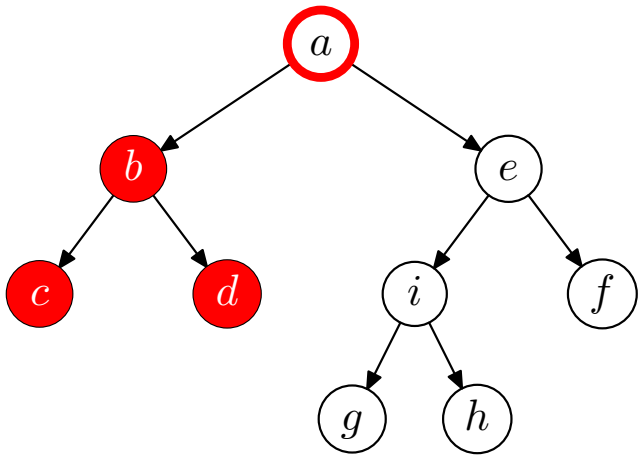


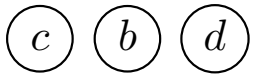
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



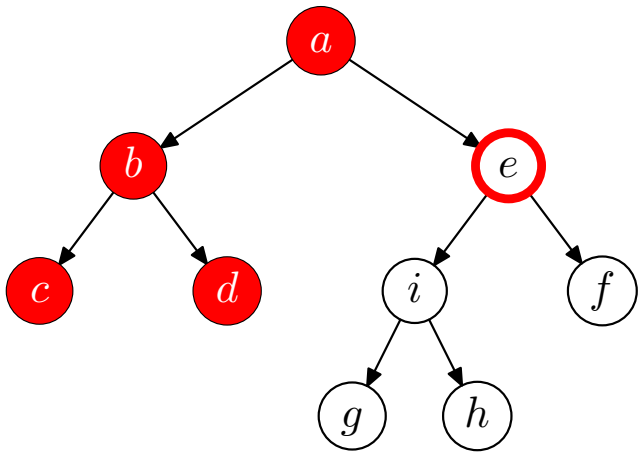


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



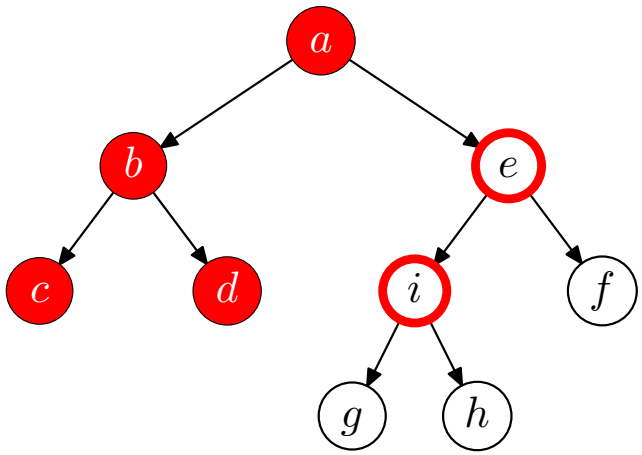
c b d a

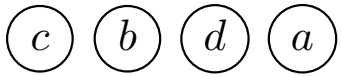
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



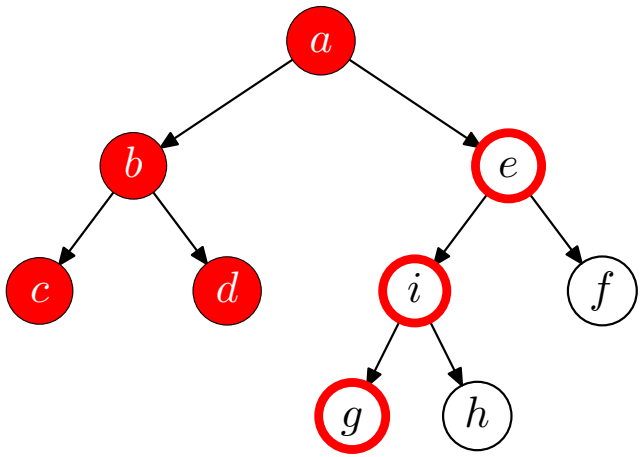


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



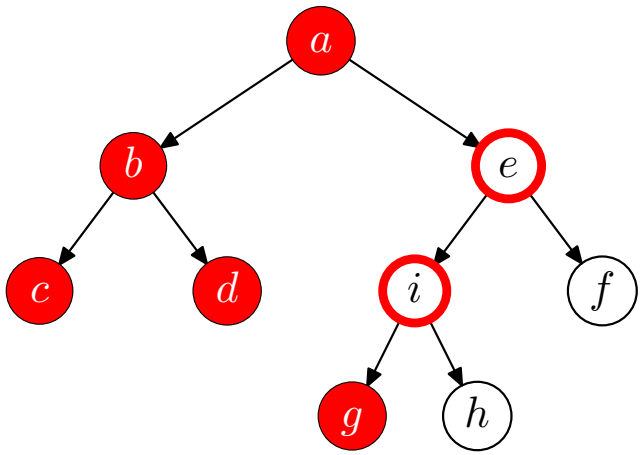
c b d a

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.

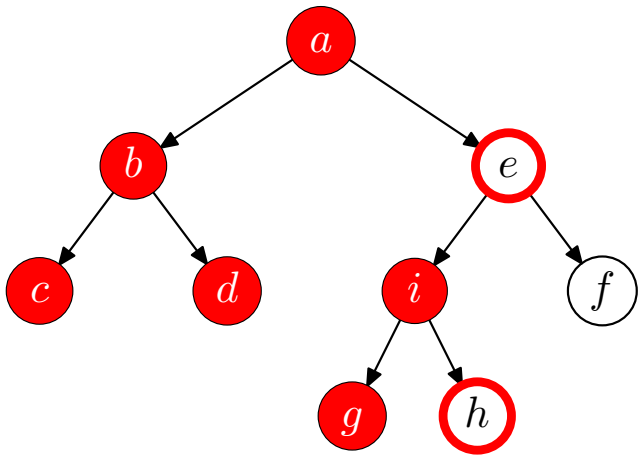


c b d a g

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



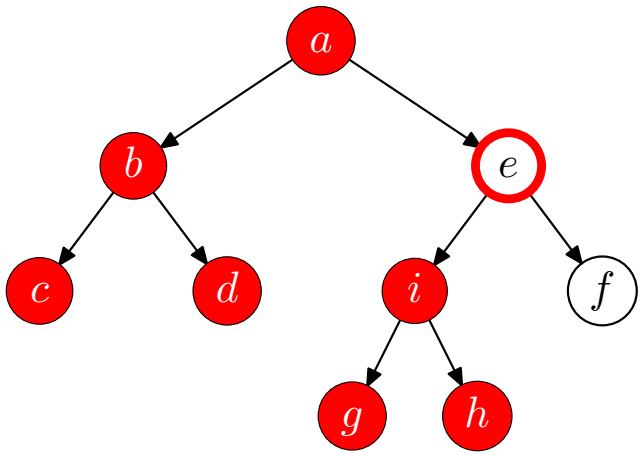
c b d a g i

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.

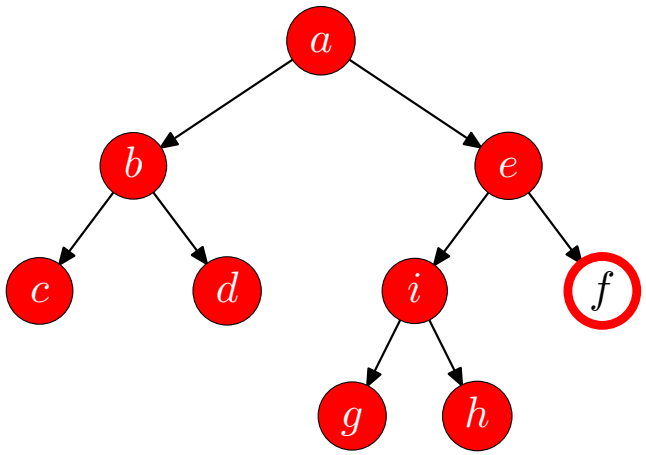


c b d a g i h

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



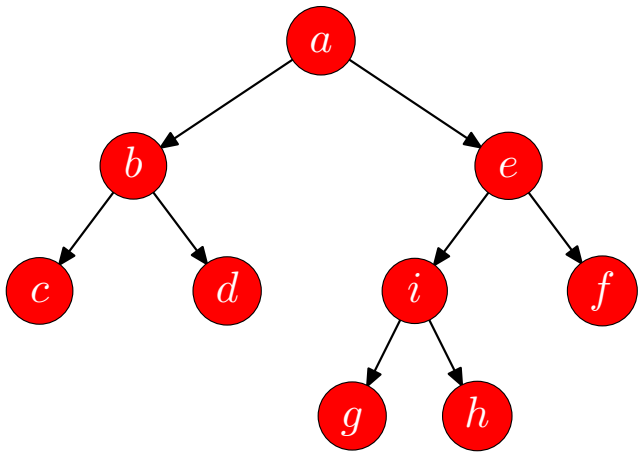
c b d a g i h e

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Inorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját inorder módon.
3. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).
4. Járjuk be a gyökérelem jobb oldali részfáját inorder módon.



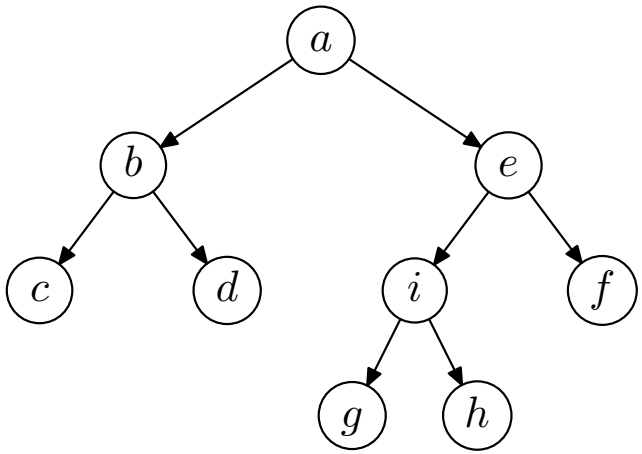
c b d a g i h e f

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

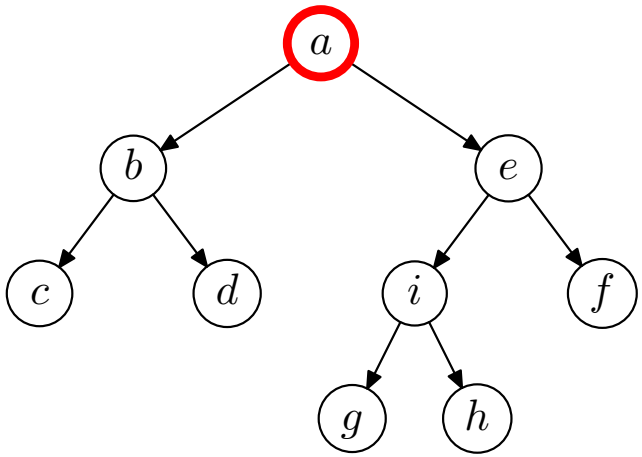


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

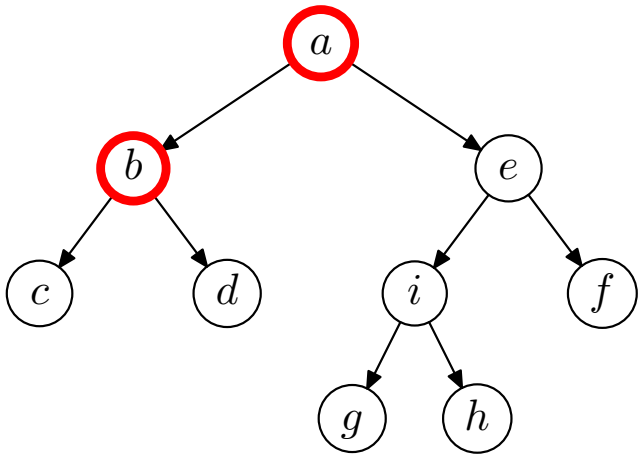


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

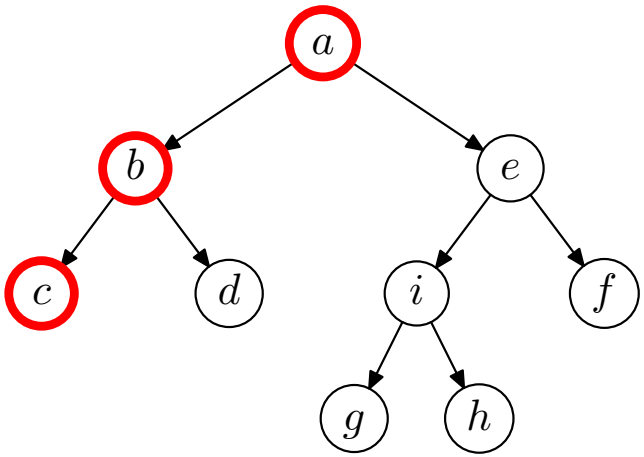


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

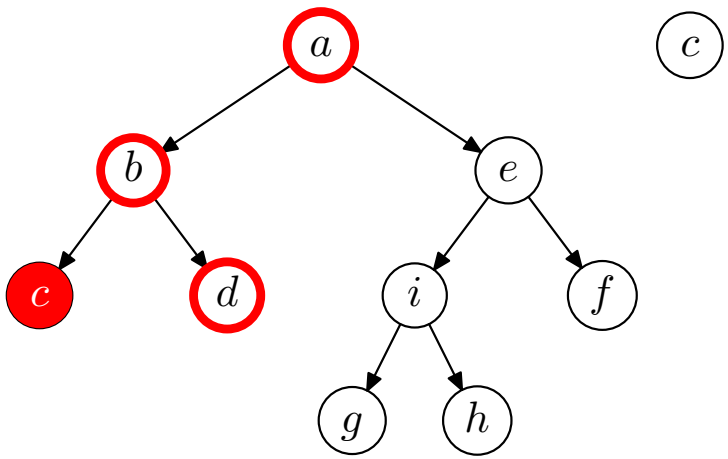


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

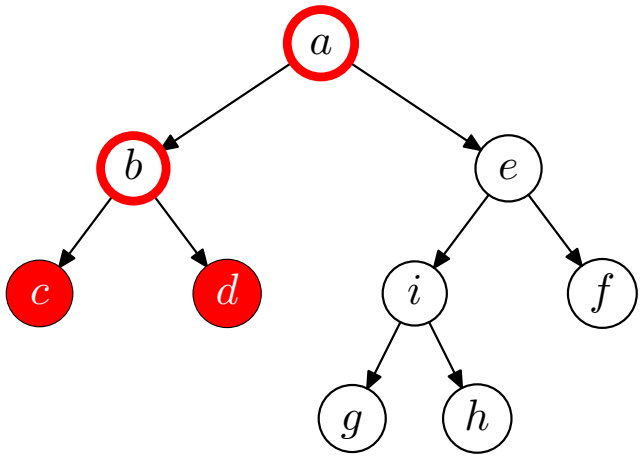


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).



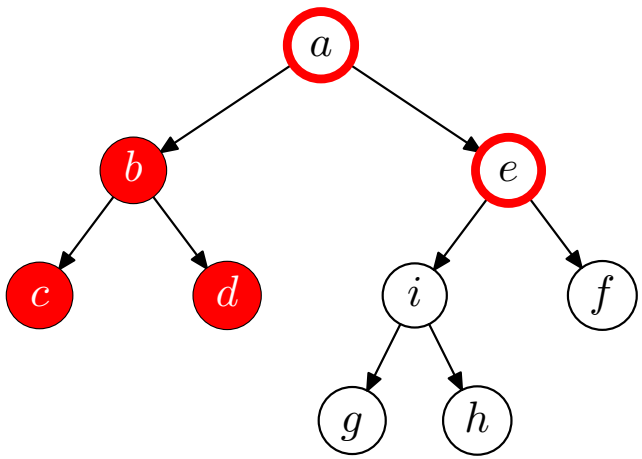
c d

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

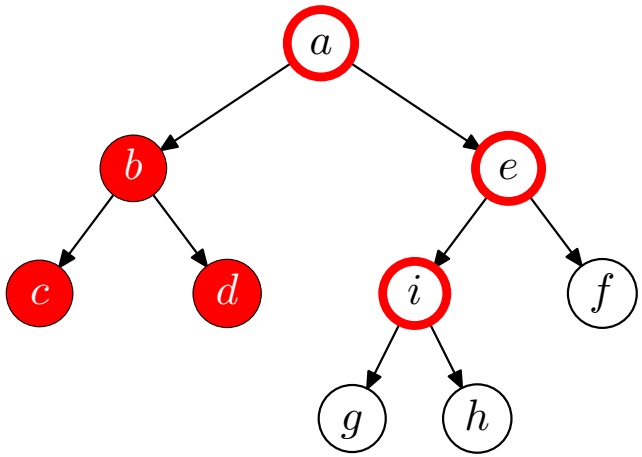


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

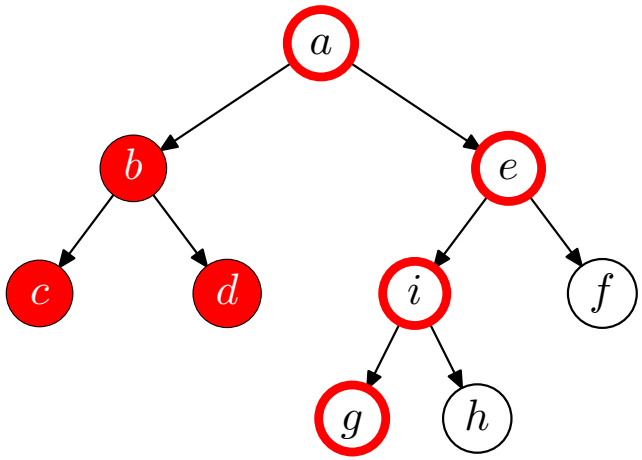


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

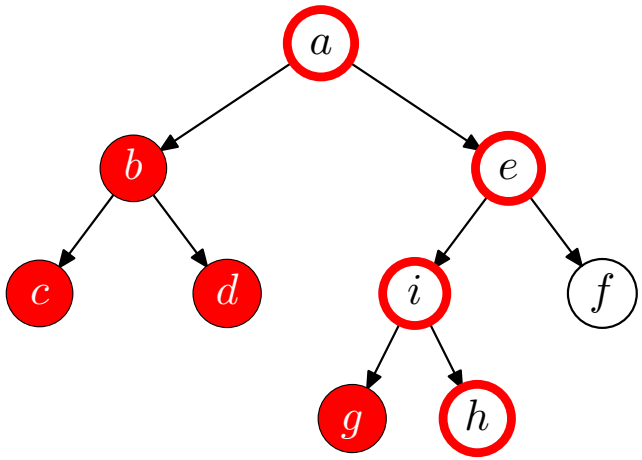


Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

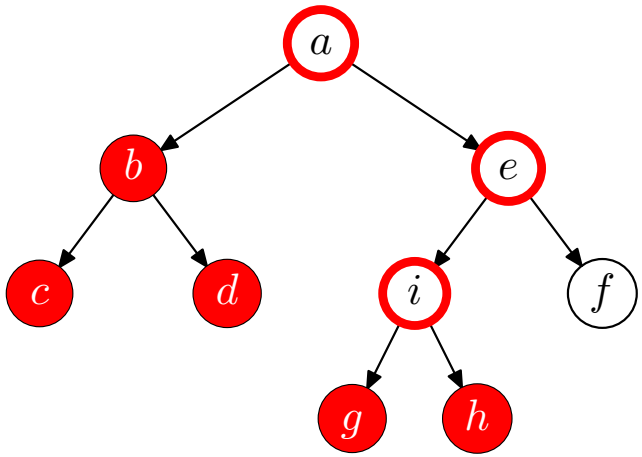


c d b g

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).



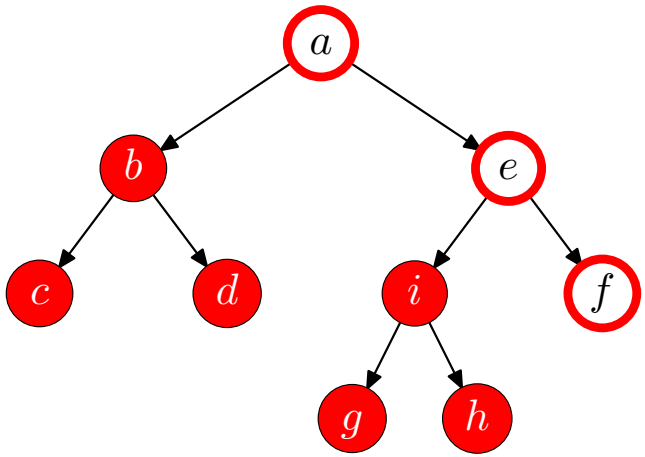
c d b g h

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).



c d b g h i

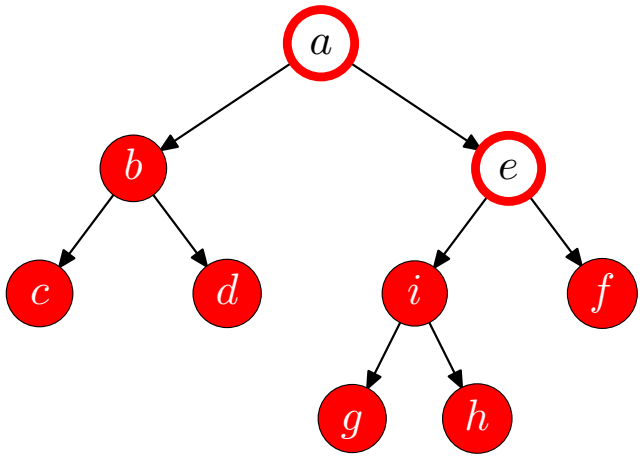
Hierarchikus adatszerkezetek

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).



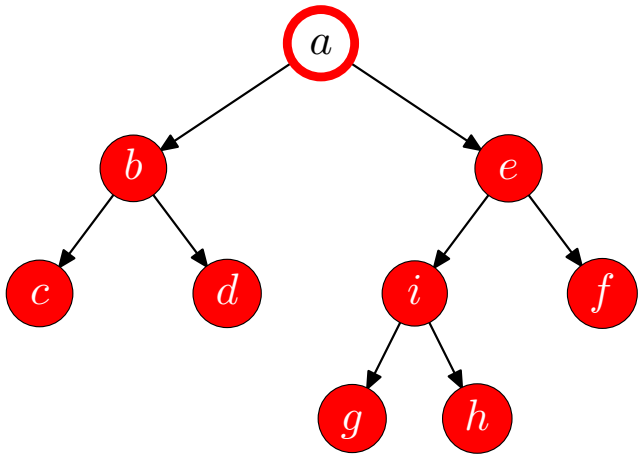
c d b g h i f

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).

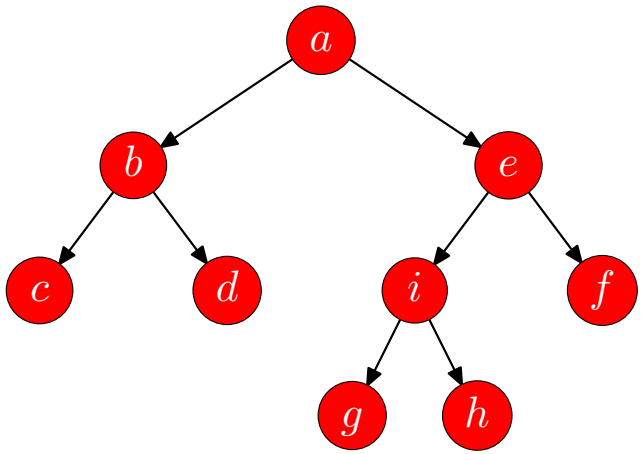


c d b g h i f e

Bináris fa bejárási algoritmusai

Postorder bejárás algoritmusa

1. Ha a bejárandó fa üres, az algoritmus véget ér.
2. Járjuk be a gyökérelem bal oldali részfáját postorder módon.
3. Járjuk be a gyökérelem jobb oldali részfáját postorder módon.
4. Dolgozzuk fel a gyökérelemet (más szavakkal: helyezzük a gyökérelemet a sor végére).



c d b g h i f e a

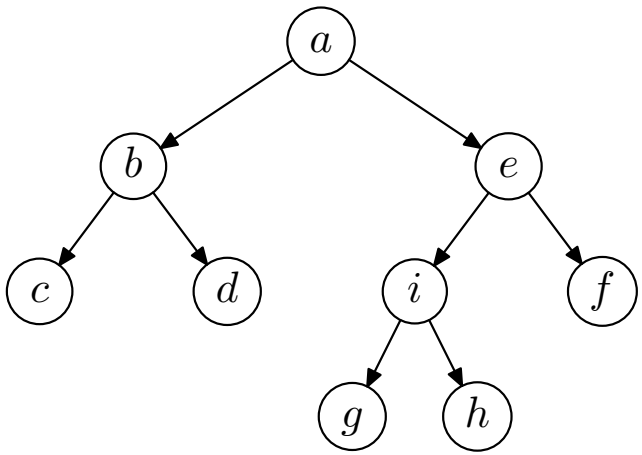
Kósa Márk
Pánovics János
Szathmáry László

Bináris fa folytonos reprezentációja

Három vektor segítségével, ahol a vektorok azonos indexű elemei a következő információkat tárolják:

- az ADAT vektorban az adatelem értékét,
- a BAL vektorban a bal oldali rákövetkező vektorbeli indexét,
- a JOBB vektorban a jobb oldali rákövetkező vektorbeli indexét.

Általában a fa gyökérelemét e vektorok első eleme írja le.

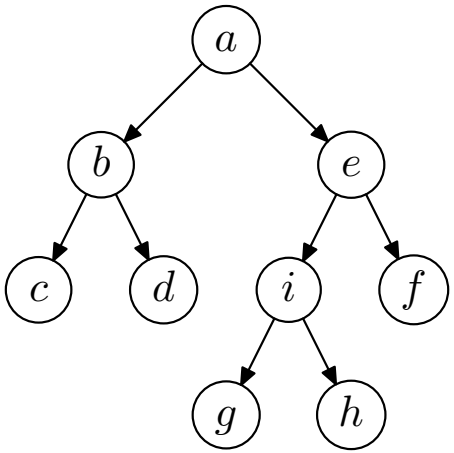


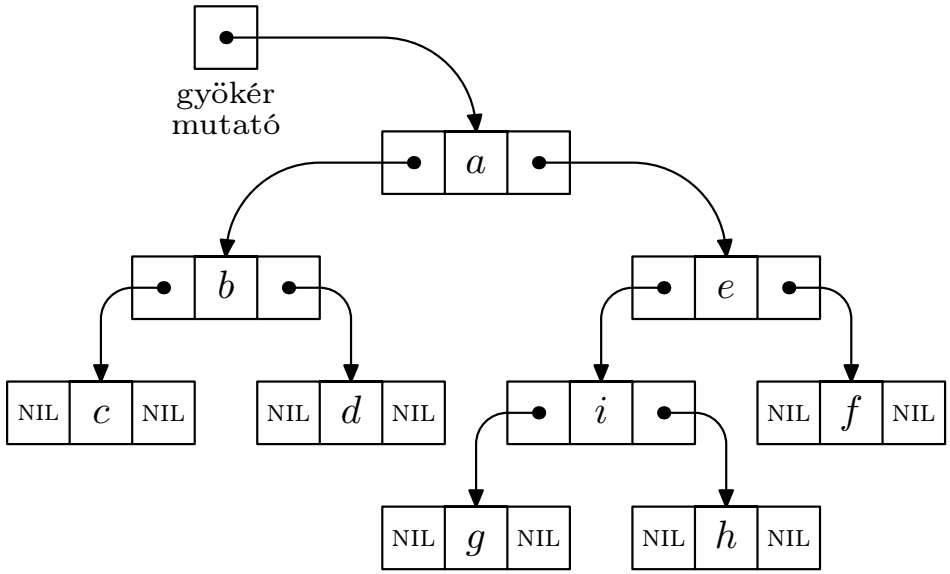
	ADAT	BAL	JOBB
1.	a	2	5
2.	b	3	4
3.	c	0	0
4.	d	0	0
5.	e	6	9
6.	i	7	8
7.	g	0	0
8.	h	0	0
9.	f	0	0

Kósa Márk
Pánovics János
Szathmáry László

Bináris fa szétszórt reprezentációja

A listaelemek adatrésze az adatelem értékét tartalmazza, a mutatórész pedig két mutatót: egyet, amely a bal oldali rákövetkezőt leíró listaelemet címzi, és egy másikat, amely a jobb oldali rákövetkezőt leíró listaelemet címzi. A gyökérelemhez (és rajta keresztül az adatszerkezet többi eleméhez) a „gyökér” mutató segítségével tudunk hozzáférni.



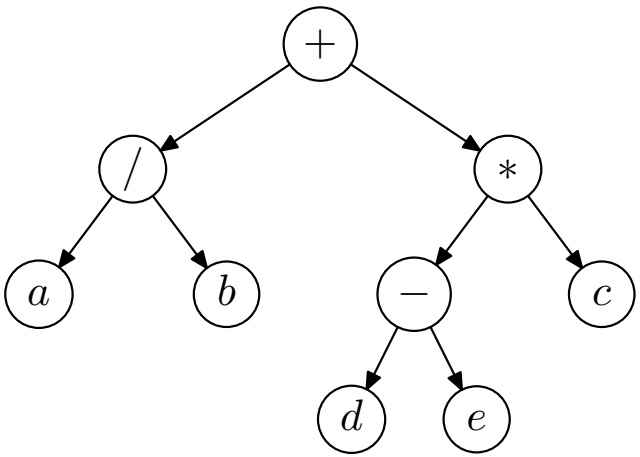


Kósa Márk
Pánovics János
Szathmáry László

Kifejezésfák és a bejárások kapcsolata

Kifejezésfa

A kifejezésfa olyan fa, melyben a levélelemek egy kifejezés operandusait, a nem levél elemek pedig ugyanazon kifejezés operátorait tartalmazzák.



prefix: $+ \; / \; a \; b \; * \; - \; d \; e \; c$

infix: $a \; / \; b \; + \; d \; - \; e \; * \; c$

postfix: $a \; b \; / \; d \; e \; - \; c \; * \; +$

Aszerint, hogy a kifejezésfát – a korábban említettek közül – melyik bejárási algoritmussal járjuk be, kapjuk a kifejezés prefix, infix és postfix alakját.

Kósa Márk
Pánovics János
Szathmáry László

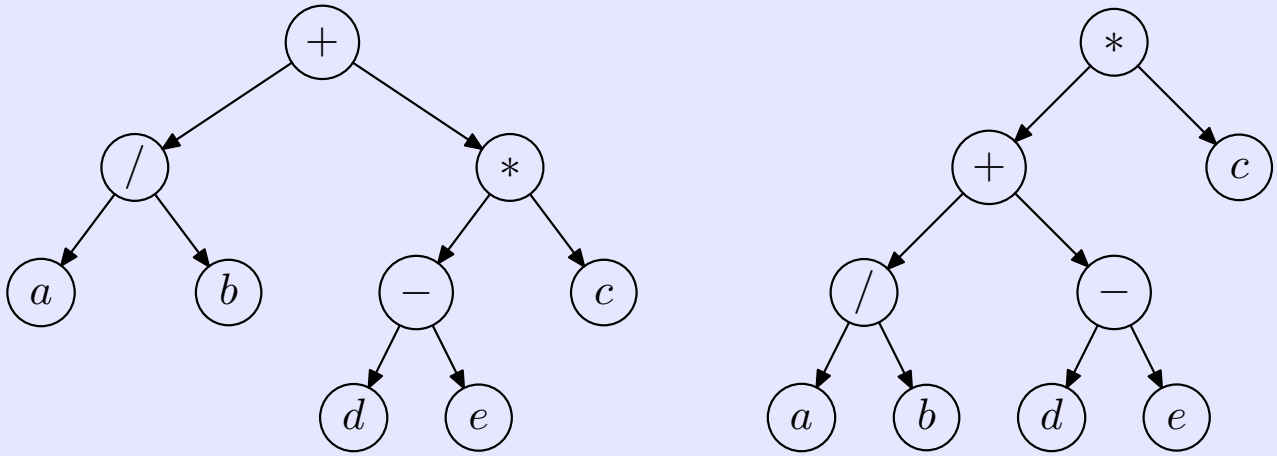
Kifejezésfák és a bejárások kapcsolata

A prefix és a postfix alak egyértelmű, az infix nem az (de zárójelek használatával azzá tehető).

Példa

Az alábbi két kifejezésfát inorder módon bejárva ugyanazt az infix kifejezést kapjuk:

$$a \,/\, b + d - e * c$$



Zárójelezést alkalmazva:

$$(a \,/\, b) + ((d - e) * c) \qquad ((a \,/\, b) + (d - e)) * c$$

Bináris fa implementálása

Üres fa létrehozása

```
typedef struct faelem {
    tipus adat;
    struct faelem *bal;
    struct faelem *jobb;
} FAELEM;

FAELEM *gyoker = NULL;
```

Bináris fa bejárása inorder stratégiával

```
void inorder_bejaras(FAELEM *csucs)
{
    if (csucs != NULL)
    {
        inorder_bejaras(csucs->bal);
        feldolgoz(csucs);
        inorder_bejaras(csucs->jobb);
    }
}
```


Kósa Márk
Pánovics János
Szathmáry László