

4. Bináris fán alapuló keresés. Beszúrás, törlés és kiegyensúlyozás.

A **bináris fák** olyan fa adatstruktúrák, ahol minden csomópontnak legfeljebb két leszármazottja lehet. Rendszerint ilyen fákból alakítanak ki bináris keresőfákat és bináris kupacokat. A bináris fákon való kereséseknek két jellemző algoritmusuk van:

- **Mélységi keresés (depth-first):** az algoritmus minden lépésnél a legbaloldalibb, még nem bejárt csomópontot választja ki. Ha már nincs több lépés, akkor visszalép, és az utolsó elágazási pontban a következő, még nem bejárt alternatívát választja ki. A kiindulás történhet a gyökérpontból, vagy egy még nem bejárt csomópontból. Ha a kereső megtalálta a keresett csomópontot (adatot, kulcsot stb.), akkor leáll.
- **Széltében keresés (breadth-first):** az algoritmus mindig az adott kiindulási csúcsból, vagy az adott rétegből egy lépésben elérhető csúcsokat járja be. A gyökérpontot kiindulási pontnak tekintve a bináris fát olyan módon járja be, hogy mindig a gyökérhez legközelebb eső, még be nem járt csomópontot keresi fel.

Fa (Tree): csomópontok (nodes) halmaza, amelyeket élek (edges) kötnek össze, és teljesülnek az alábbi feltételek:

- létezik egy kitüntetett csomópont: a gyökér (root)
- a gyökértől különböző minden más csomópont egy éllel van összekötve a szülőjéhez
- a fa összefüggő: bármely nem-gyökér csomóponttól kiindulva a szülőkön keresztül a gyökérhez eljutunk

Bináris kereső fa (BST)

- **Rendezett fa,** emiatt pontosítjuk a struktúrát:
TBSTElem = Struktúra(kulcs, tartalom, balgyerek, jobbgyerek)
- **Kulcs alapján rendezett:** a fa minden csomópontjára igaz, hogy a baloldali részfája legfeljebb a kulcs nagyságú, a jobboldali részfája pedig legalább a kulcs nagyságú kulcsokat tartalmaz
- A rendezés természetesen lehet fordított is
- A kulcs lehetséges megvalósításai
 - különálló kulcs mező
 - a kulcs a tartalom része
 - a kulcs a tartalomból számítható érték

Bejárás

- **Bejárás:** az adatszerkezet valamennyi elemének egyszeri elérése (feldolgozása)
- A láncolt listához hasonlóan a bejárás algoritmusuk független a végrehajtandó tevékenységtől, ezért az algoritmuson belül csak utalást teszünk erre
- Mivel a láncolt listával ellentétben egy elemről több irányban is tovább lehet lépni, többféle bejárás is elképzelhető
- A csomópontokban található adatok (tartalom, bal, jobb) feldolgozásának sorrendje alapján három fajtát különböztetünk meg

különböztethet_ meg (ezen belül a bal és jobb megcserélhet_):

- PreOrder: tartalom, bal, jobb
- InOrder bejárás: bal, tartalom, jobb
 - PostOrder: bal, jobb, tartalom

Keresés BST-ben

- Alapelv: a fa gyökérelemének kulcsa vagy egyenl_ a keresett kulccsal, vagy egyértelm_en meghatározza, hogy melyik részében kell a keresést folytatni
- Ez ugyanúgy igaz a teljes bináris fa gyökérelemére, illetve bármelyik (a keresés során elért) részfájának gyökérére
- A keresés menete (keresend_ kulcs: x)
 - bázis
 - ha T üres, akkor x nincs benne
 - ha T gyökérének címkéje x, akkor talált
- indukció (legyen T gyökérének kulcsa y)
 - ha $x < y$, akkor x-et a bal részében,
 - ha $x > y$, akkor a jobb részében keressük

Beszúrás BST-be

- A beszúrás során az elem beláncolásán kívül ügyelnünk kell a keres_fa tulajdonság fenntartására is
- Ugyanazok az elemek többféleképpen is elhelyezkedhetnek egy bináris keres_fa-ban, beszúráskor ez alapján több stratégiánk is lehet
 - minél kisebb er_forrásigény_ legyen a beszúrás
 - minél kiegyensúlyozottabb legyen a fa a beszúrás(ok) után
- A beszúrás menete (beszúrandó kulcs: x)
 - bázis
 - ha T üres, akkor új csomópontot készítünk x kulccsal és visszatérünk a csomópont címével
 - ha T gyökérének kulcsa x, akkor visszatérünk a csomópont címével
- indukció (legyen T gyökérének kulcsa y)
 - ha $x < y$, akkor x-t a bal részébe,
 - ha $x > y$, akkor a jobb részébe szúrjuk be

Törlés BST-ből – 1. lépés

- A törlés során az elem kiláncolásán kívül ügyelnünk kell a keres_fa tulajdonság fenntartására is
- Törlés során az alábbi problémák merülhetnek fel
 - két gyerekkel rendelkező elem mindkét gyereket nem tudjuk az _szül_jének egy mutatójára rákapcsolni
 - gyökérelem törlése
- Törlés 1. lépése (beszúrandó kulcs: x)
 - bázis
 - ha T üres, akkor nincs ilyen elem
 - ha T gyökérének kulcsa x, akkor töröljük, majd tovább a 2. lépésre: BST tulajdonság visszaállítása
- indukció (legyen T gyökérének kulcsa y)
 - ha $x < y$, akkor x-t a bal részből,
 - ha $x > y$, akkor a jobb részből töröljük