ADATSZERKEZETEK ÉS ALGORITMUSOK

Bináris Keresőfa

Bejárások

- Preorder

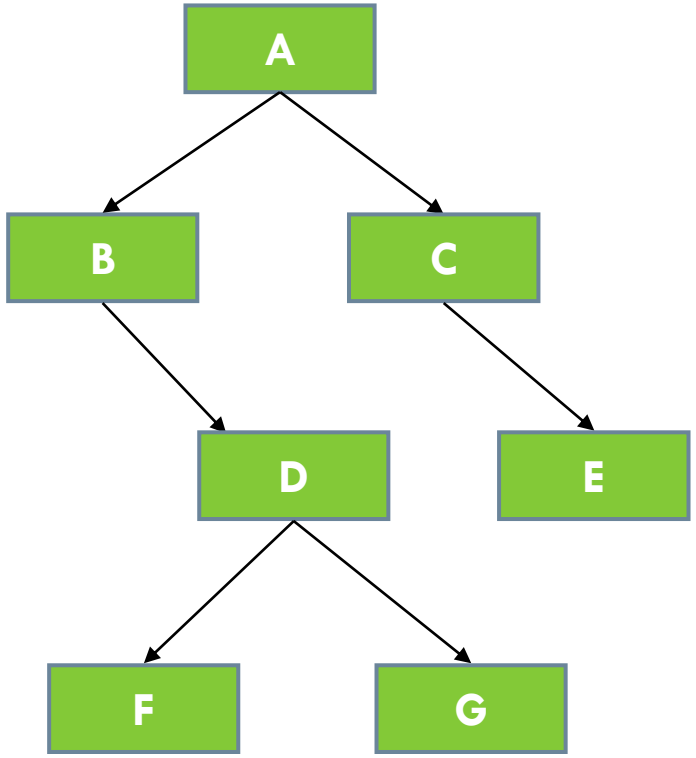
A, B, D, F, G, C, E

- Postorder

F, G, D, B, E, C, A

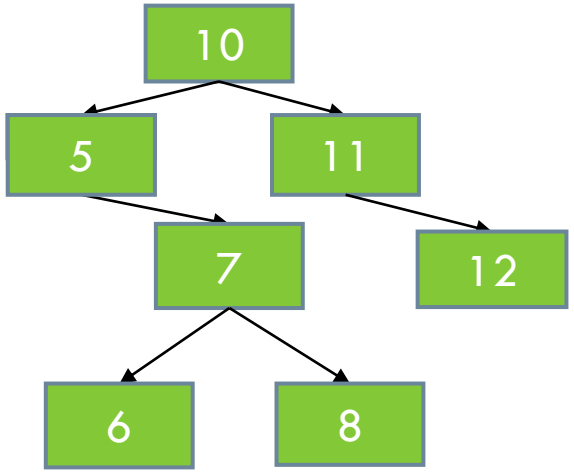
- Inorder

B, F, D, G, A, C, E



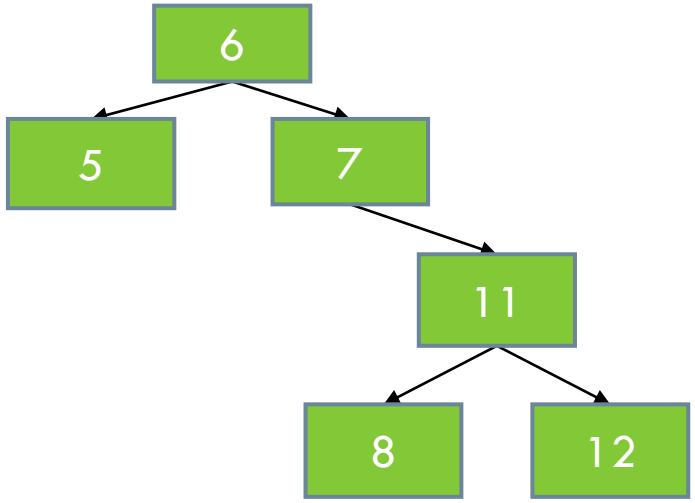
Bejárások – példa

10, 5, 7, 6, 8, 11, 12:



Preorder: 10, 5, 7, 6, 8, 11, 12
Postorder: 6, 8, 7, 5, 12, 11, 10
Inorder: 5, 6, 7, 8, 10, 11, 12

6, 5, 7, 11, 12, 8:



Preorder: 6, 5, 7, 11, 8, 12
Postorder: 5, 8, 12, 11, 7, 6
Inorder: 5, 6, 7, 8, 11, 12

A bejárások komplexitása N csúcsú fa esetében: O(N)

Vegyük észre, hogy bináris keresőfák esetében az inorder bejárással az adatelemek rendezett sorozatát kapjuk

Binkerfa C++-ban

- A műveleteknek van egy kívülről látható és egy belső változata. Ezekre azért van szükség, hogy azon műveletek, melyek megvalósítását egy fától elvárjuk, függetlenek legyenek a reprezentációhoz felhasznált (kisegítő) osztálytól.

- Nyissuk meg a letöltött projektet és a az előadás diákban szereplő pszeudokódok alapján írjuk meg a hiányzó függvényeket.

- Az **`a==b`** kifejezés helyett használjuk a **`!(a < b) && !(b < a)`** kifejezést!
  (így egyetlen operátorral (<) rendezzük az egész fát)

Két típusparaméter

- Hogy kell két típusparaméterrel rendelkező osztályt készíteni?

```
template <class T, class U>
class A{ … };

template <class T, class U>
void A<T, U>::test()
{
    T var; U var2;
    // ...
}
```

Gyakorló feladat – GY05F01

- Az alábbi feladat megoldása bináris keresőfával.
- A könyvtárban a könyvekről a következő adatokat kell tárolnunk: (egyedi azonosító), `string` cím, `int` ertekeles, `bool` ki van-e épp kölcsönözve?
- Implementáld a fa iterátort (inorder járja be a fát) a hossz és cím mezőkre.
- Tölts fel 8 tetszőleges könyvet a könyvtáradba.
- A program futtatása során a következőkre legyen lehetőség:
  - új könyv hozzáadása
  - könyv kikölcsönzése/visszahozása
  - könyvek listázása cím szerint
  - könyv törlése

Gyakorló feladat – GY05F02 I.

- Hozz létre egy két template paraméteres bináris keresőfát. A két template paraméter itt azt jelenti, hogy az egyik paraméter a kulcs típusa (K), a másik, az ott tárolt érték típusa (V). Tehát ezek után a fa alkalmas kulcs-érték párok tárolására.
- Írj egy olyan alkalmazást, amelyben a felhasználó egy menün keresztül használni tud egy angol-magyar szótárt.
- Az angol-magyar szótárat a következőképpen hozd létre:
- Töltsd fel a megírt két template paraméteres fát úgy, hogy a kulcs az angol szó (`string`), az érték pedig az angol szó magyar megfelelő(i) (`string` / `list<string>`)
- Ezután pedig írj egy menüt, amin keresztül egy felhasználó használni tudja a szótárt. A menünek a következő funkciói legyenek:
  - Ki lehessen keresni kívánt angol szó magyar megfelelő(i)t. Ha ez eddig még nincs benne a szótárban, akkor legyen lehetőség ezen új szó bevitelére.
  - Ki lehessen listázni a szótárban tárolt angol szavakat és ezek magyar megfelelő(i)t, abc sorrendben.

Gyakorló feladat – GY05F02 II.

- Ha rákeresünk egy csúcsra a megadott angol szó segítségével, akkor hogyan nyerjük ki a csúcsban tárolt magyar megfelelőjét? Ehhez a következőt kell tenned: definiálj a `<K, V>` típus paraméteres bináris keresőfa műveletei között egy `V getValue(K k)` függvényt. (Tehát kulcs alapján keresek, és az értéket nyerem ki.)

Gyakorló feladat – GY05F03

- Adott két bináris keresőfa. Az egyikről annyit tudunk, hogy páratlan, a másikról pedig, hogy páros számokat tartalmaz.
- Készíts iterátort a bináris keresőfához!
- Bináris keresőfával és iterátor felhasználásával oldd meg a következőket:
  - a) A két bemeneti fából készíts egyetlen fát, majd inorder írd ki az eredményt.
  - b) Az iterátor és a bináris keresőfák felhasználásával készíts egy olyan programot, amely bementre egy számsort (listát) vár, a kimeneten pedig növekvő sorrendben adja azt vissza.

Gyakorló feladat – GY05F04

Egészítsd ki a Binkerfa implementációt szélességi bejárással (használj egy eddig tanult adatszerkezetet).

Készíts el egy algoritmust, ami minden elemről kiírja hogy levél-e.

Készítsd el a << operátort, ami a fa mind a négy bejárását kiírja.

Gyakorló feladat– GY05F05

- **Családfa**
- A bináris keresőfát alakítsuk át úgy, hogy egy ági családfa tárolására legyen alkalmas. (Csak egy szülő szerepel.)

1. Egy szülőnek több gyereke (tetszőleges számú) is lehet (tehát már nem lesz bináris)
2. A rendezésnél adjuk meg beszúrandó node-nál, hogy ki a szülője és ezt keresse meg a fában, továbbá az azonos szinteken ABC sorrendben legyenek tárolva a node-ok.
3. Ahhoz, hogy továbbra is keresőfa maradjon, találjatok ki egy egyedi kódolást az egyes személyekhez.
   Pl.: Nagyapa: „A” – Apa: „AA” – Apa testvér: „AB” – Apa gyerek1: „AAA” – Apa gyerek2: „AAB”
4. Törlést nem kell implementálni.

Gyakorló feladat folyt. – GY05F05