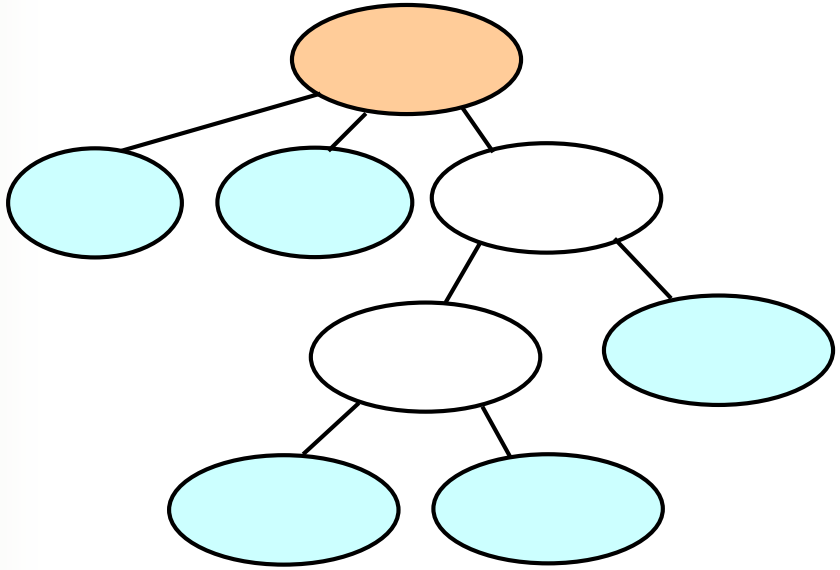
アルゴリズムとデータ構造a
7 – 木構造

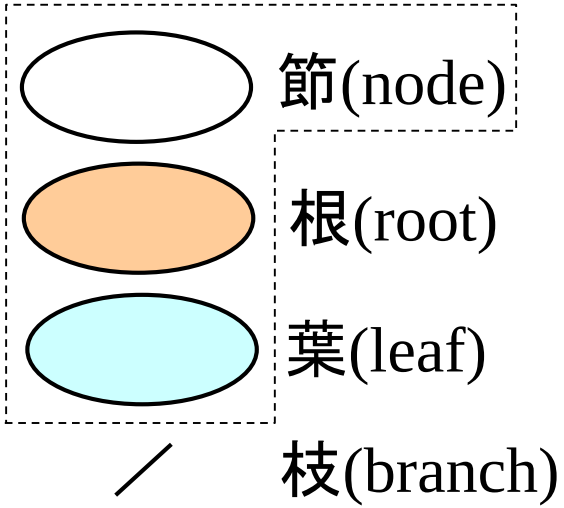
大見　嘉弘

今日の授業

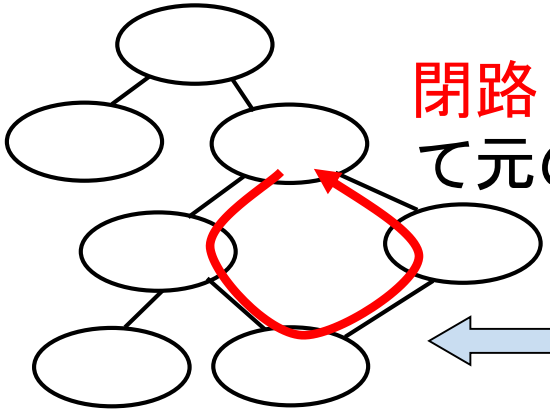
- 木構造
  - 構造
  - 二分木、探索木
  - B木
  - 二分探索木の操作

木構造(ツリー構造)





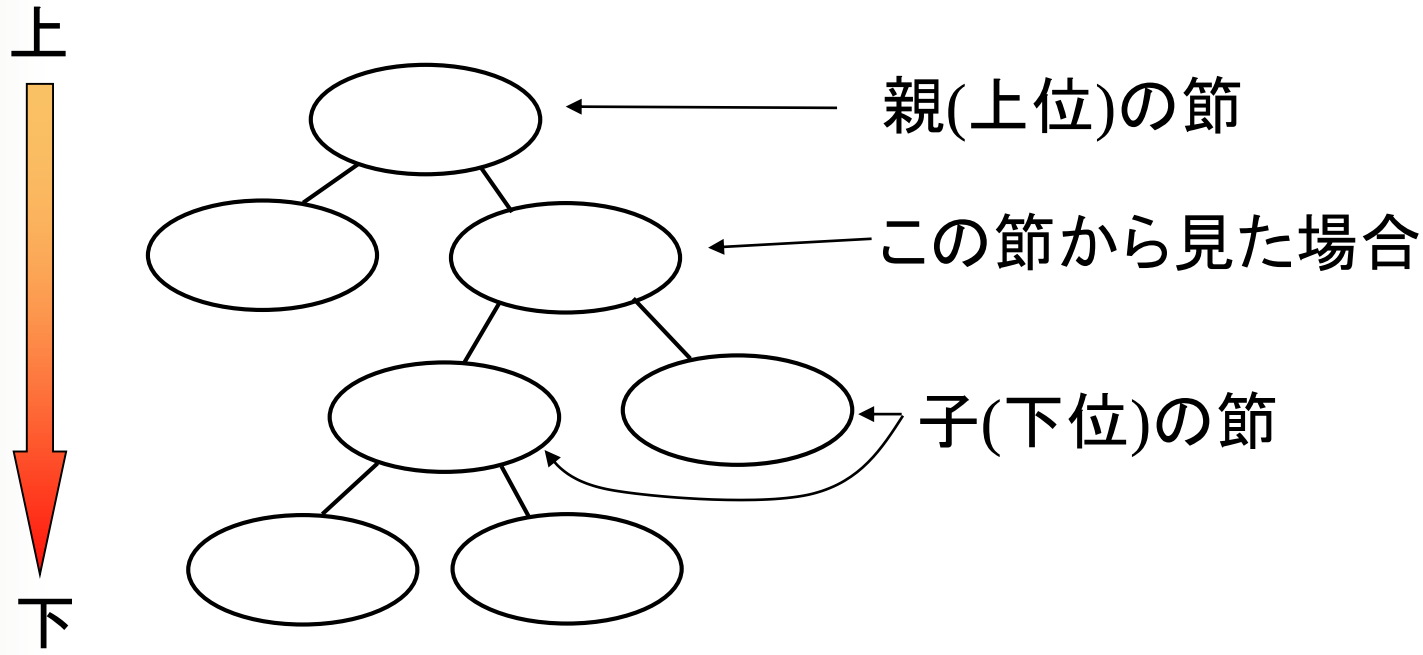
木構造は「閉路」を持たない



閉路：違う経路を通って元の場所に戻れる

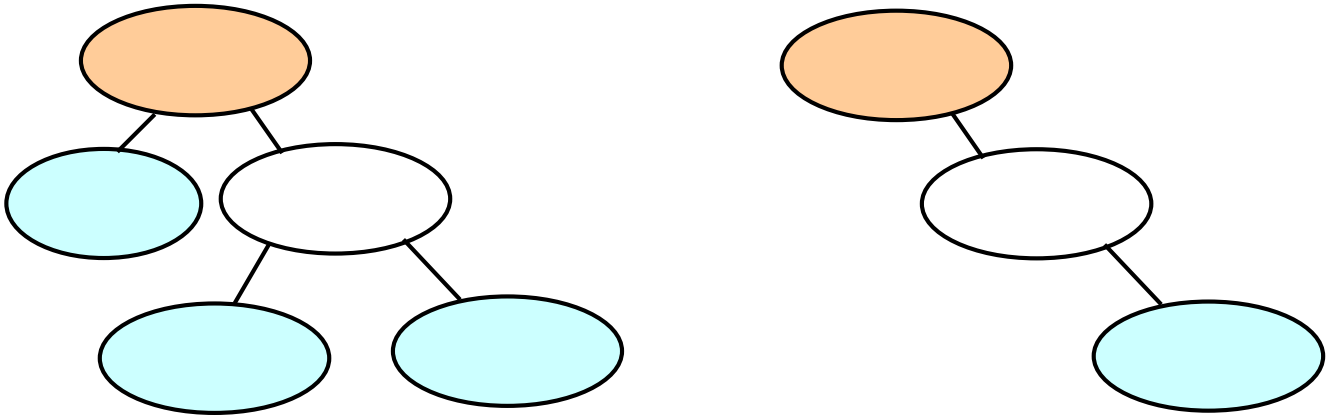
これは木構造ではない

木構造の階層



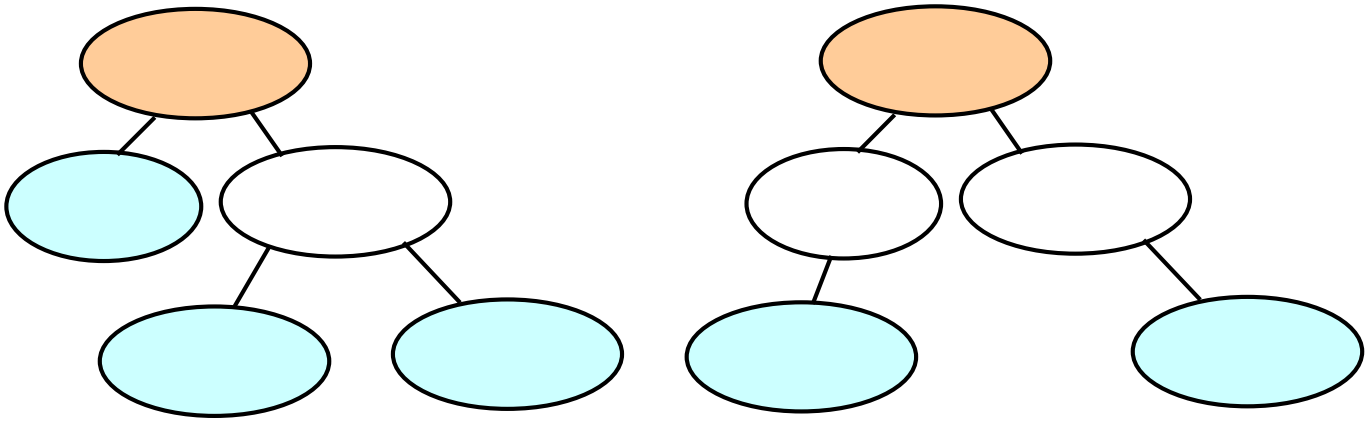
二分木

- 節から下に伸びている枝が最大2本の木(0～2本)

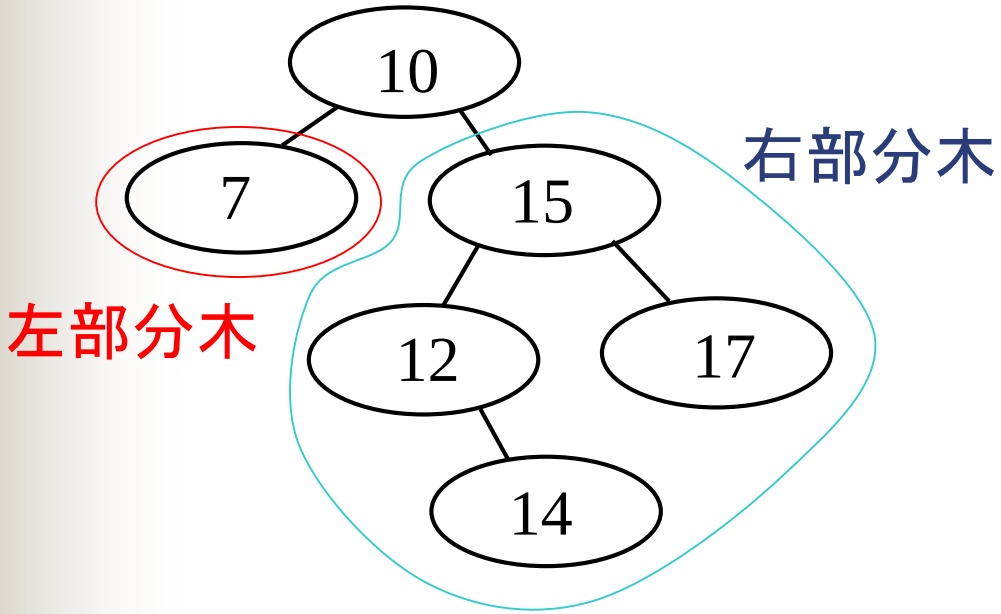


完全二分木

- 二説ある
  - 全ての葉が同じレベル(位)にある二分木
  - 根から葉までの枝数が同じか、1つしか違わない(1,2本)二分木



二分探索木



- 右部分木に含まれる値は根より大きい
- 左部分木に含まれる値は根より小さい

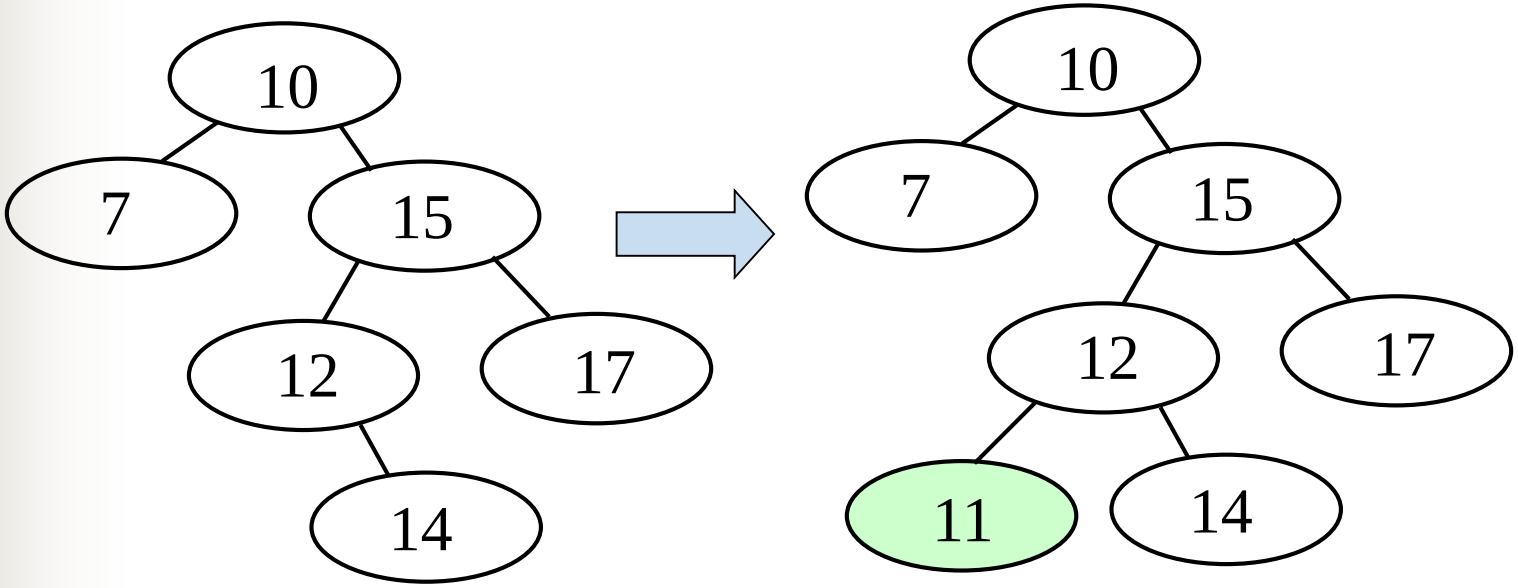
二分探索木の探索

1. 根の値と比較する
   - 一致したら「見つかった」。探索終了。
   - 根より大きければ、右部分木を見る。
   - 根より小さければ、左部分木を見る。
2. 以上を繰返して、見る先がなくなったら「見つからなかった」。探索終了。

B木

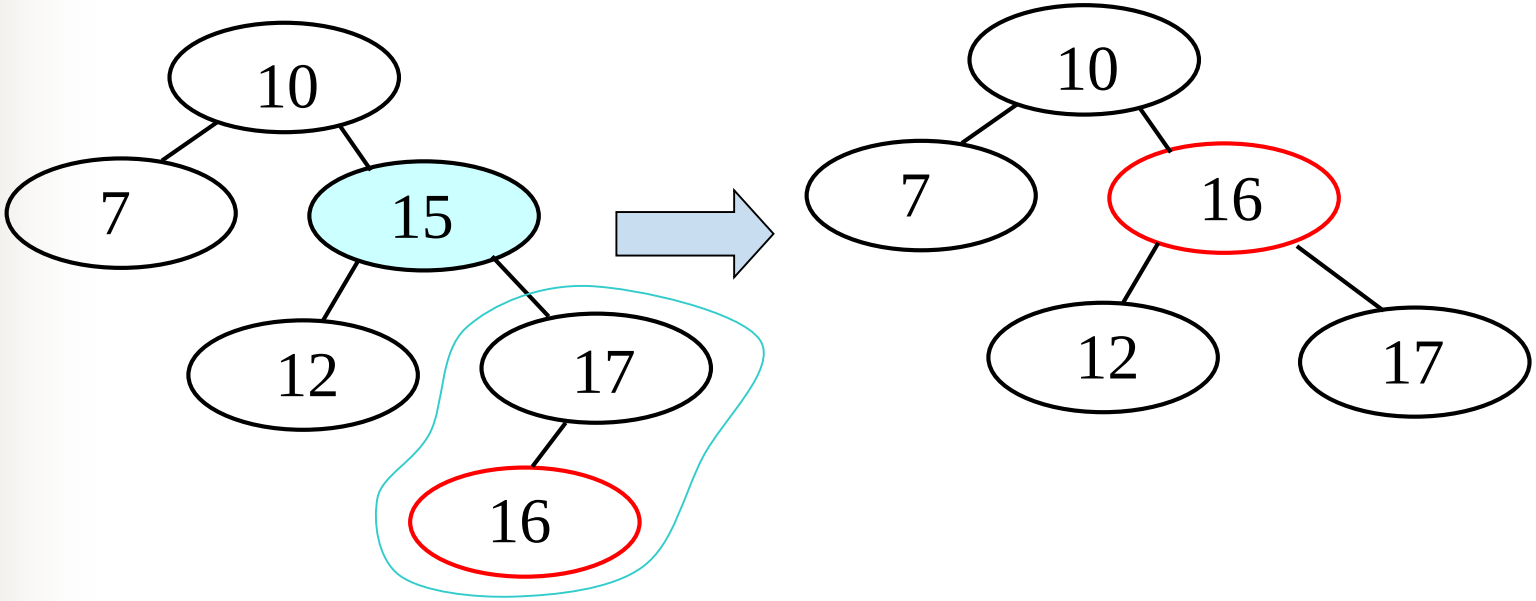
- データが多い木構造を扱うのに適する
- DBMS(データベース管理システム)等で使われている。
- B木の条件(m次(order m)のB木)
  1. 葉以外の各節がm個までの分岐点を持つ
  2. 各節は、(m-1)個までのデータ要素を持つ
  3. 根から全ての葉までの深さは同じ
  4. 各節内のデータは整列されている

二分探索木への挿入



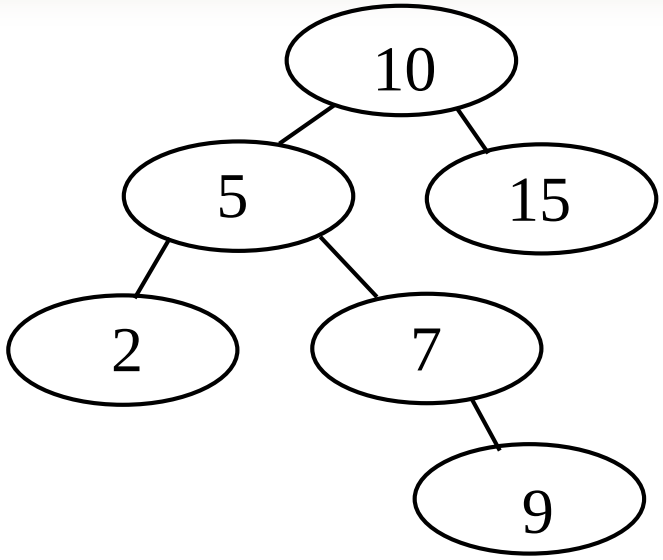
1. 11>10 だから右部分木に移動
2. 11<15 だから左部分木に移動
3. 11<12だから左部分木に移動⇒左部分木がない
   ⇒左部分木として11を追加

二分探索木の削除



1. 削除する節が葉なら、そのまま削除
2. 削除する節の子が1つなら、その子を削除した節の場所に移動
3. 子が2つなら、削除した節の右部分木のうち、最小の要素を削除した場所に移動
   (もしくは左部分木のうち、最大の要素を移動)

宿題



1. 上記の二分探索木に6を挿入する場合の手順を書き、最終的な木を描け。
2. 上記の二分探査木から5を削除する場合の手順を書き、最終的な木を描け。