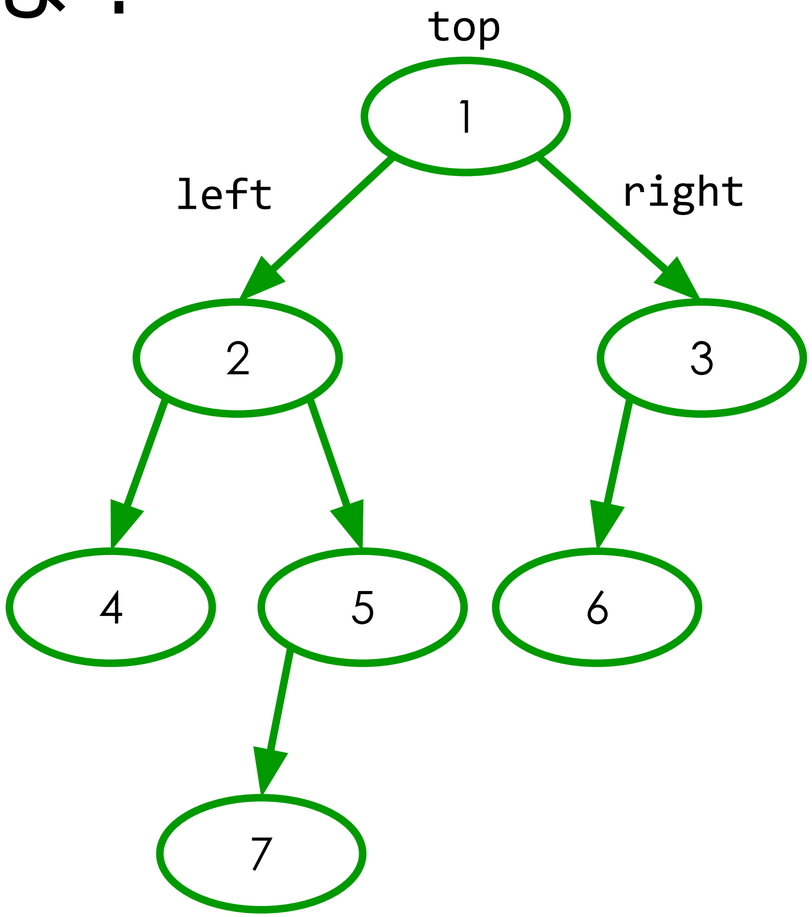
④

二分木の探索

二分木に含まれている全てのノードを探索する方法を学習します。

二分木のノードを全て表示するには？



- 問題：二分木の全てのノードを1度ずつ（リストのように）表示するは、どのようにすれば良いでしょうか？
  - いくつかの並べ方がありそうです
- プログラムで書くと、簡単そうで簡単ではありませんね？

1，2，3，4，5，6，7

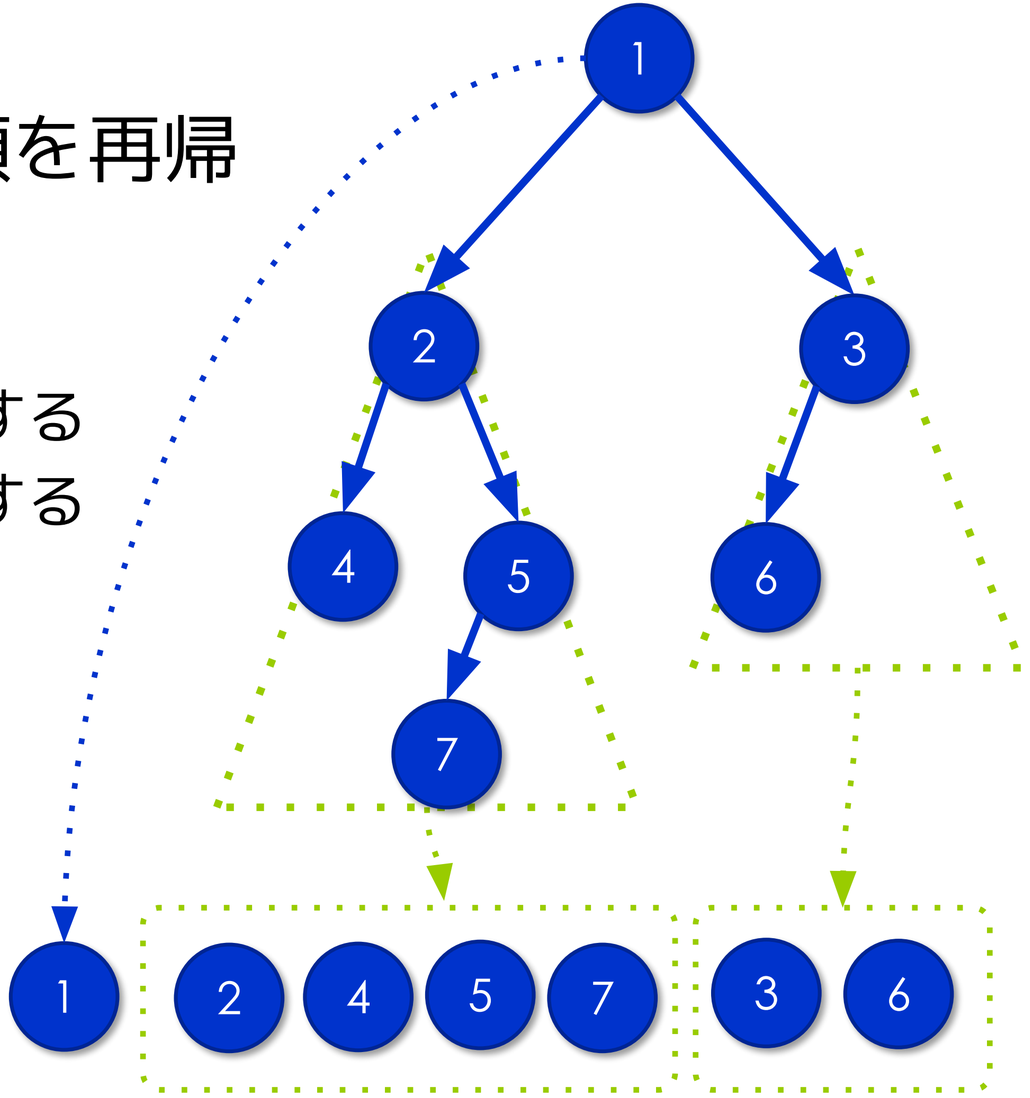
1，2，4，5，7，3，6

木の探索

- 木の全要素を順番に表示することを考えると、何らかのルールに従って、ノードを順に訪れる必要があります
- このような手続きを「木の探索」と呼び、いくつかの方法が考えられます
- ここではリンクを辿って子ノードを探索する「深さ優先探索」として、次の3種類の探索方法を覚えましょう
  - 行きがけ順の探索
  - 通りがけ順の探索
  - 帰りがけ順の探索
  - ※「深さ優先探索」以外に、後で学習する「幅優先探索」もあります
- 簡単のため二分木の場合で説明しますが、多分木の場合も同様です

行きがけ順の探索

- 各ノードについて、次のような手順を再帰的に適用します
  1. このノードを訪れる（＝表示する）
  2. 左の子ノードを根とする部分木を探索する
  3. 右の子ノードを根とする部分木を探索する



Pythonで表現すると…

```
def print_preorder(node):
    print(node, end=', ')

    if node.left != None:
        print_preorder(node.left)

    if node.right != None:
        print_preorder(node.right)
```

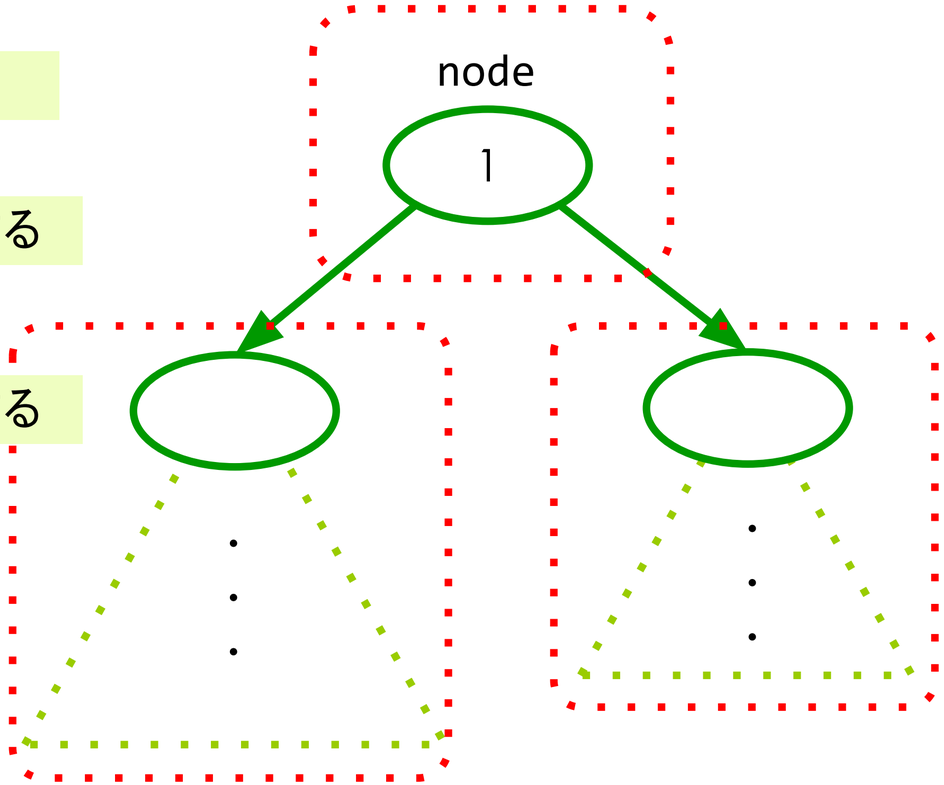
このノードを表示する

左の部分木を表示する

右の部分木を表示する

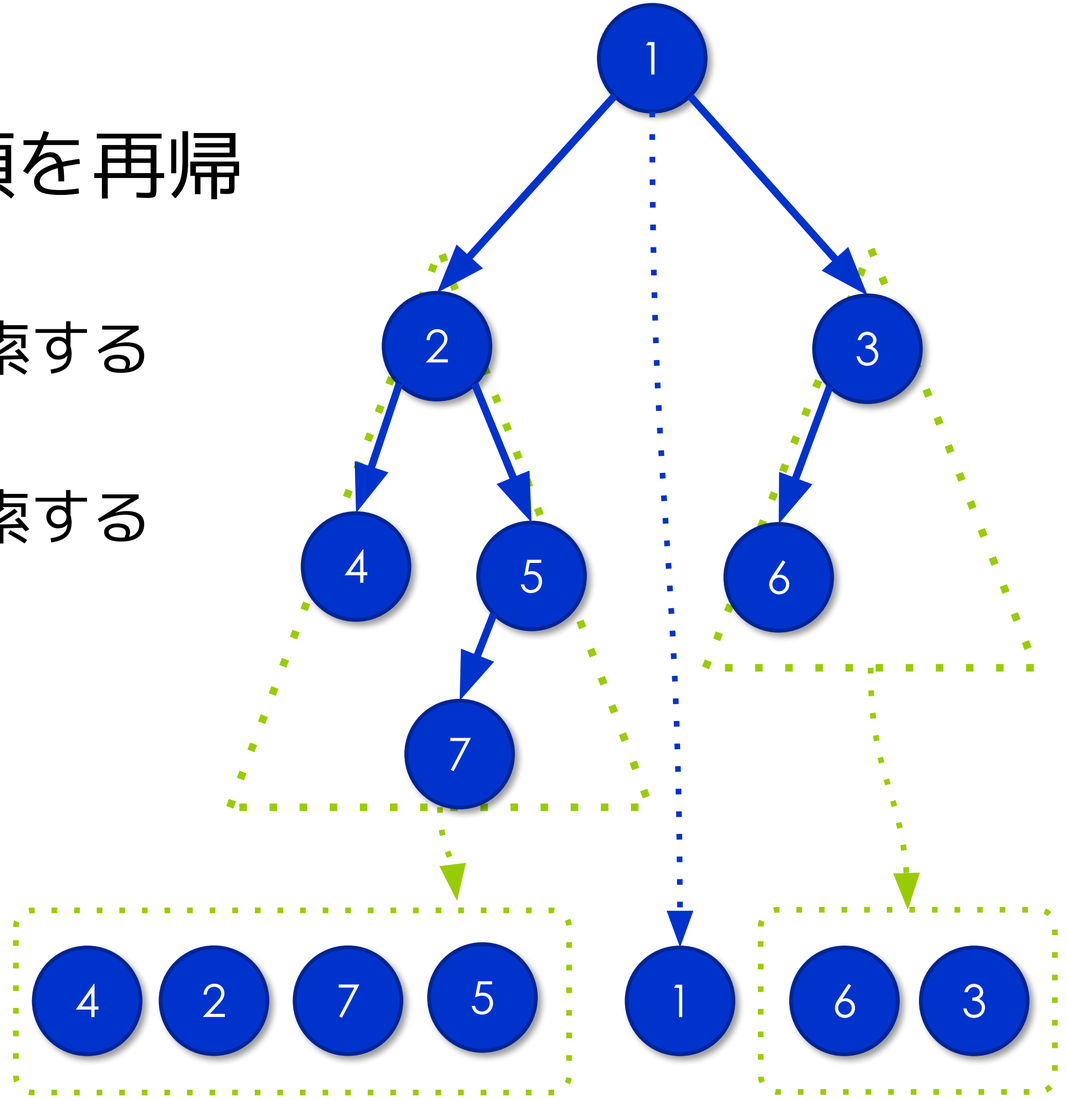
```
print_preorder(top)
```

```
1, 2, 4, 5, 7, 3, 6,
```



通りがけ順の探索

- 各ノードについて、次のような手順を再帰的に適用します
  1. 左の子ノードを頂点とする部分木を探索する
  2. このノードを訪れる（＝表示する）
  3. 右の子ノードを頂点とする部分木を探索する



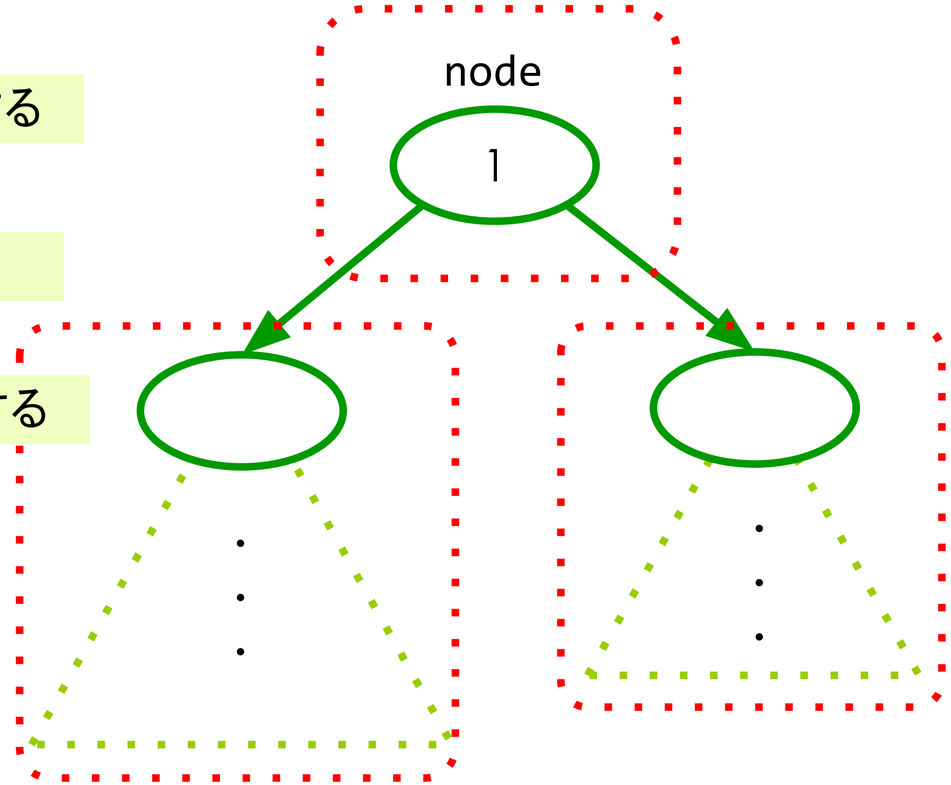
Pythonで表現すると…

```
def print_inorder(node):
    if node.left != None:
        print_inorder(node.left)

    print(node, end=', ')

    if node.right != None:
        print_inorder(node.right)
```

左の部分木を表示する

このノードを表示する

右の部分木を表示する

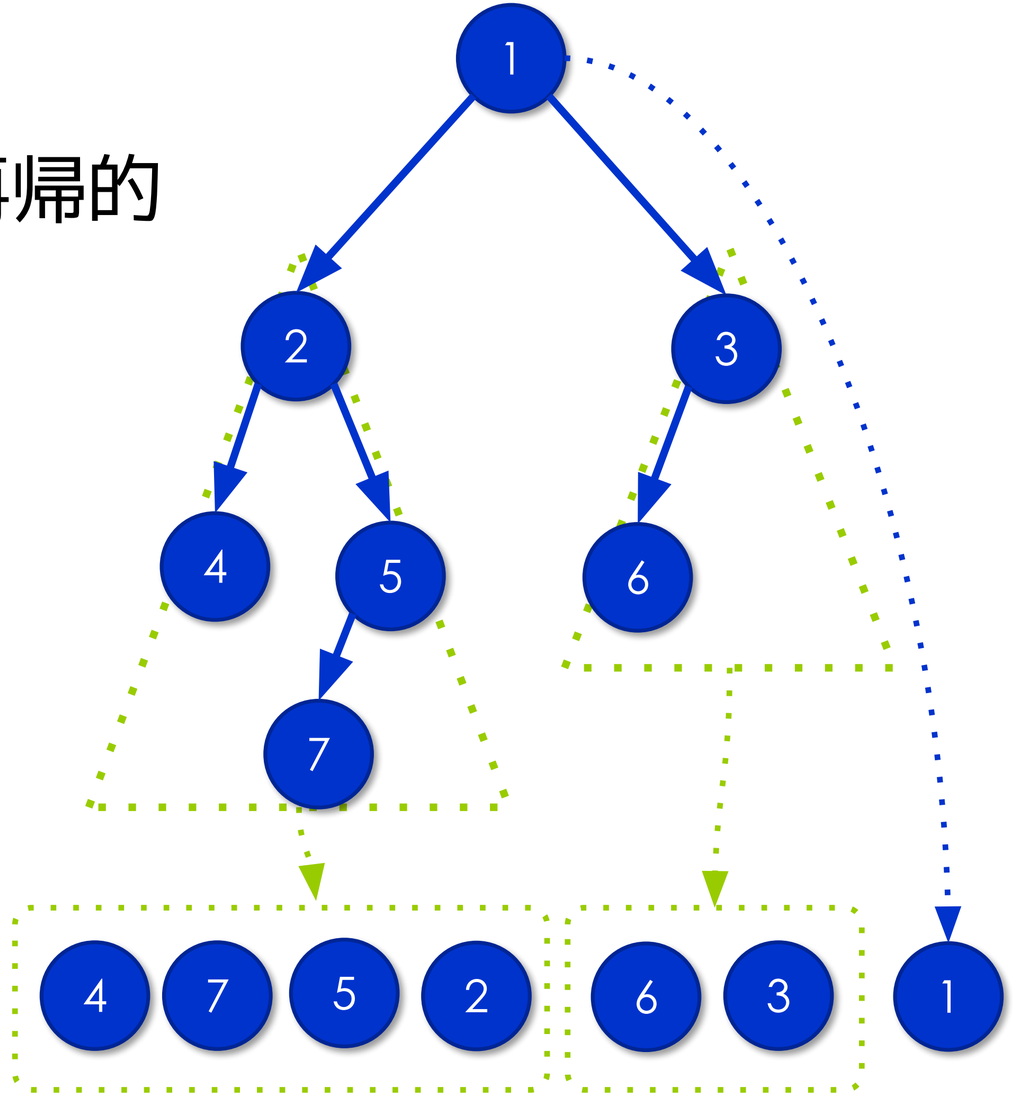



```
print_inorder(top)
```

```
4, 2, 7, 5, 1, 6, 3,
```

帰りがけ順の探索

- 各ノードについて、次のような手順を再帰的に適用します
  1. 左の子ノードを頂点とする部分木を探索する
  2. 右の子ノードを頂点とする部分木を探索する
  3. このノードを訪れる（＝表示する）



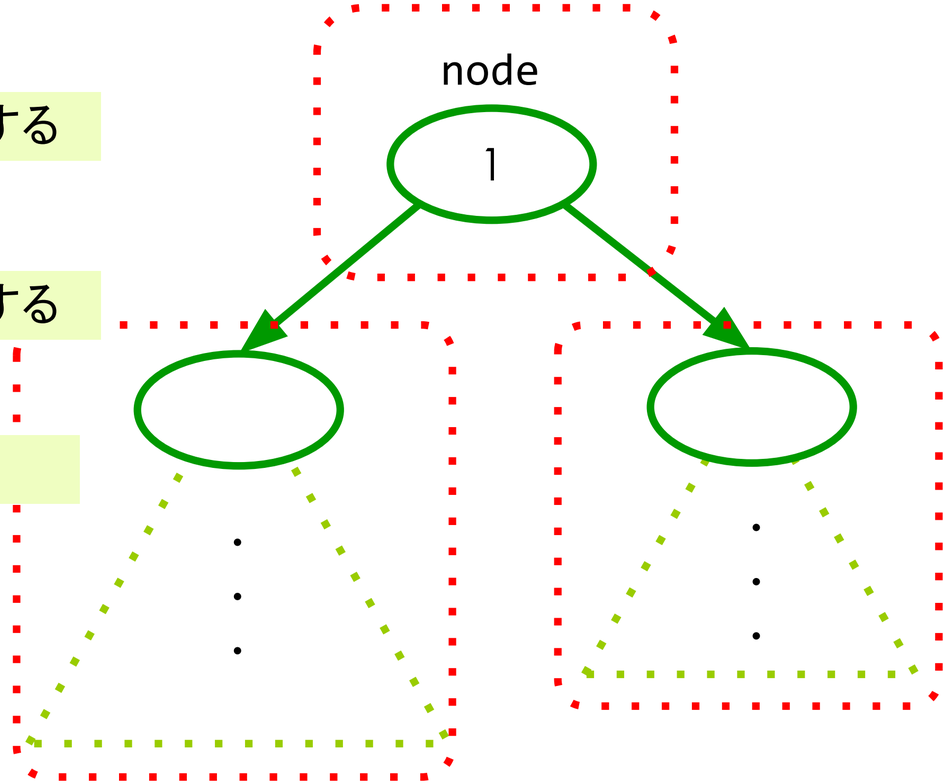
Pythonで表現すると…

```
def print_postorder(node):
    if node.left != None:
        print_postorder(node.left)

    if node.right != None:
        print_postorder(node.right)

    print(node, end=', ')
```

左の部分木を表示する

右の部分木を表示する

このノードを表示する




```
print_postorder(top)
```

```
4, 7, 5, 2, 6, 3, 1,
```