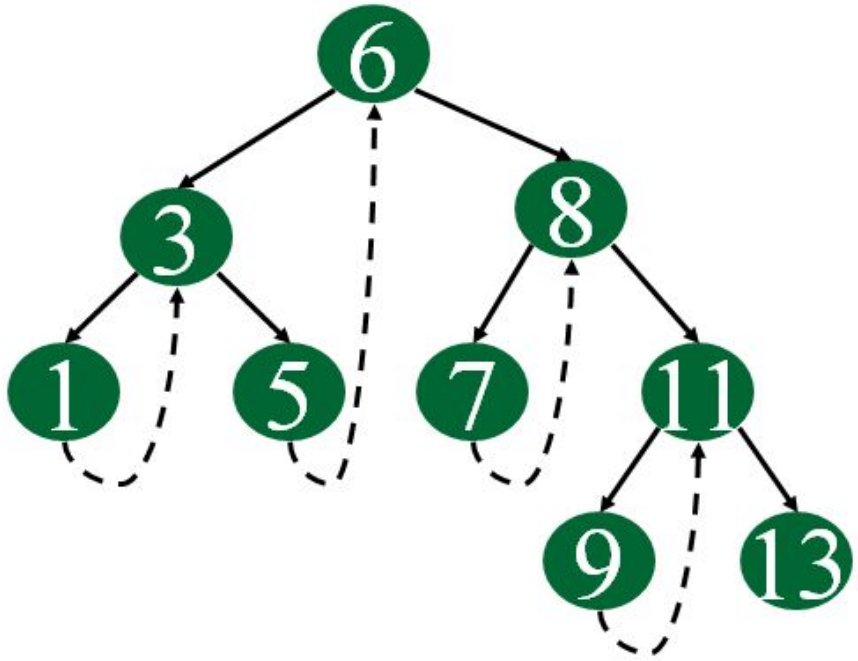
Threaded Binary Tree

DS openlab

오지민, 박태준



What is Threaded Binary Tree?

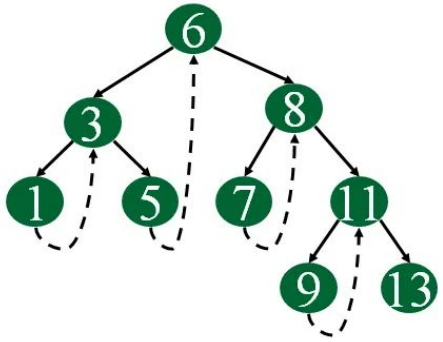
A binary tree is made threaded by making all right child pointers that would normally be NULL point to the inorder successor of the node (if it exists).

Single Threaded Binary Tree

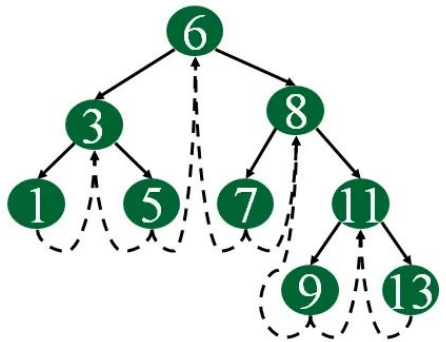
* right pointers is made to point to the **inorder successor**(if exists)
* can only construct Inorder successor

Double Threaded Binary Tree

Both left and right pointers are made to point to **inorder predecessor and inorder successor respectively.**



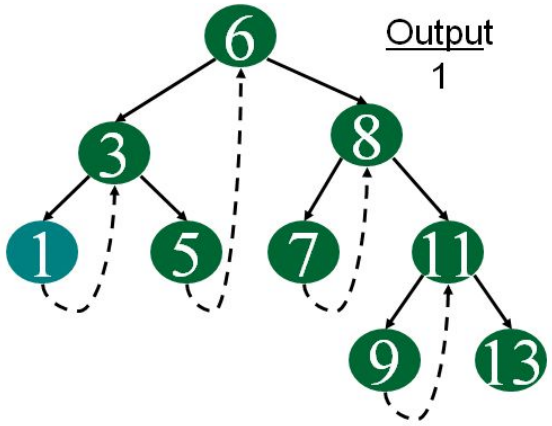
Single Threaded Binary Tree



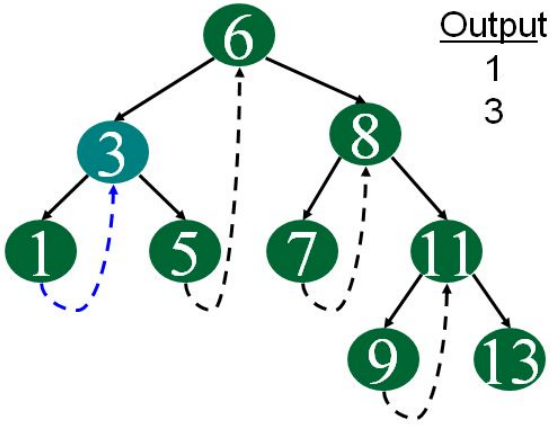
Double Threaded Binary Tree

Why Do We Use Threaded Binary Tree?

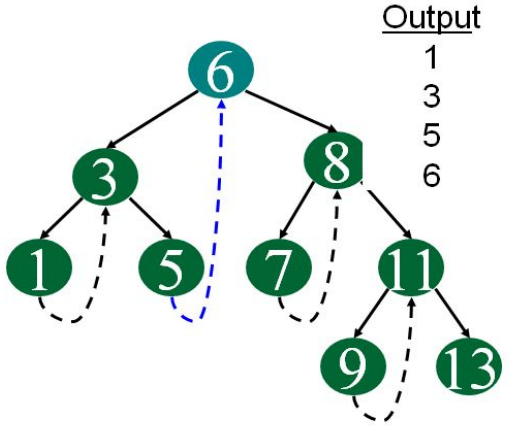
To make traversal faster and implement it **without recursion** and additional data structure



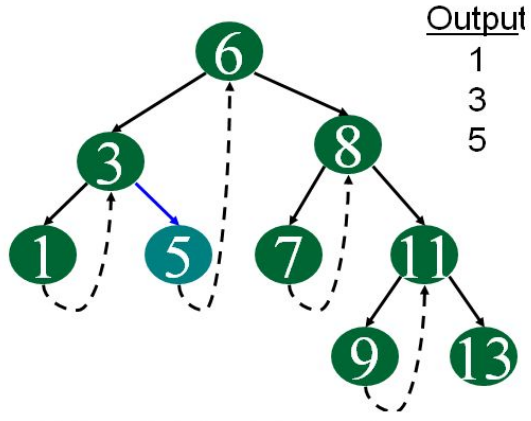
Start at leftmost node, print it



Follow thread to right, print node



Follow thread to right, print node



Follow link to right, go to leftmost node and print

Handling Threaded Binary Trees

- Node Insertion (On your own)
- Inorder Traversal (On your own)

Node Definition

```
typedef struct threadedTree {
    short int leftThread;
    threadedPointer leftChild;
    char data;
    threadedPointer rightChild;
    short int rightThread;
};
```

Node Insertion

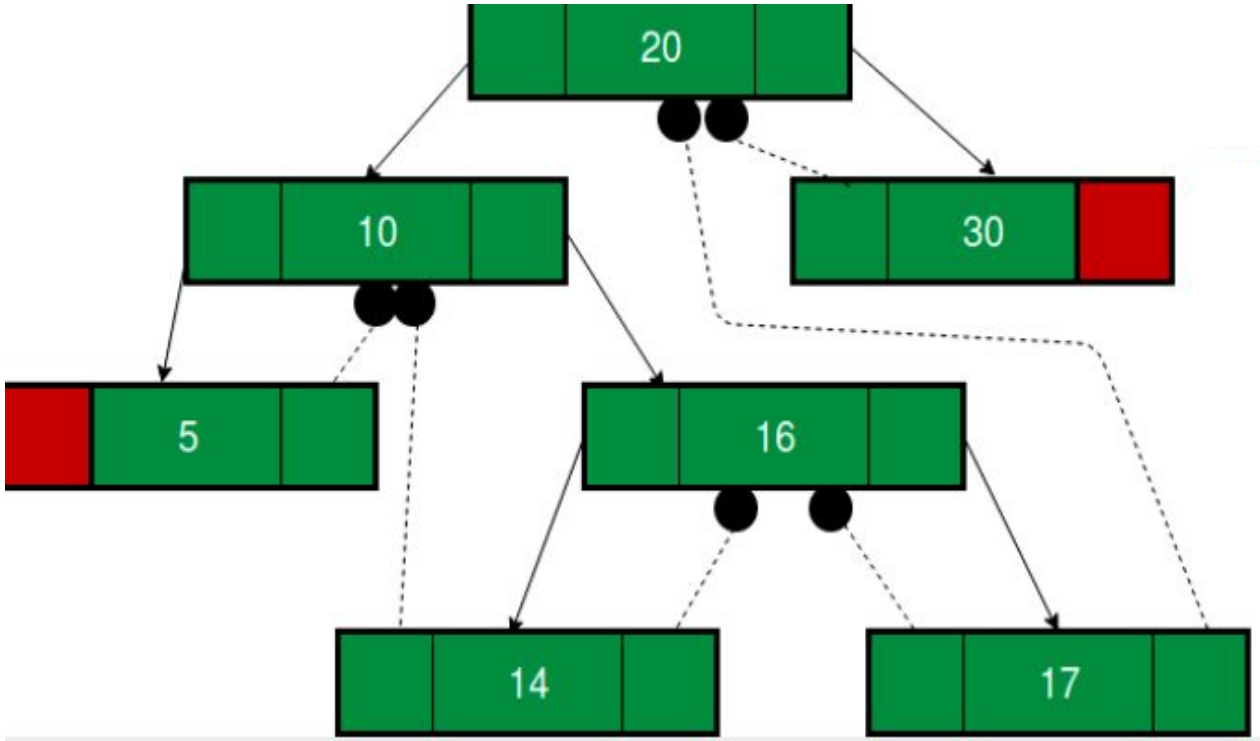
InsertRight :

Make created node stick be right child of designated node through parameter

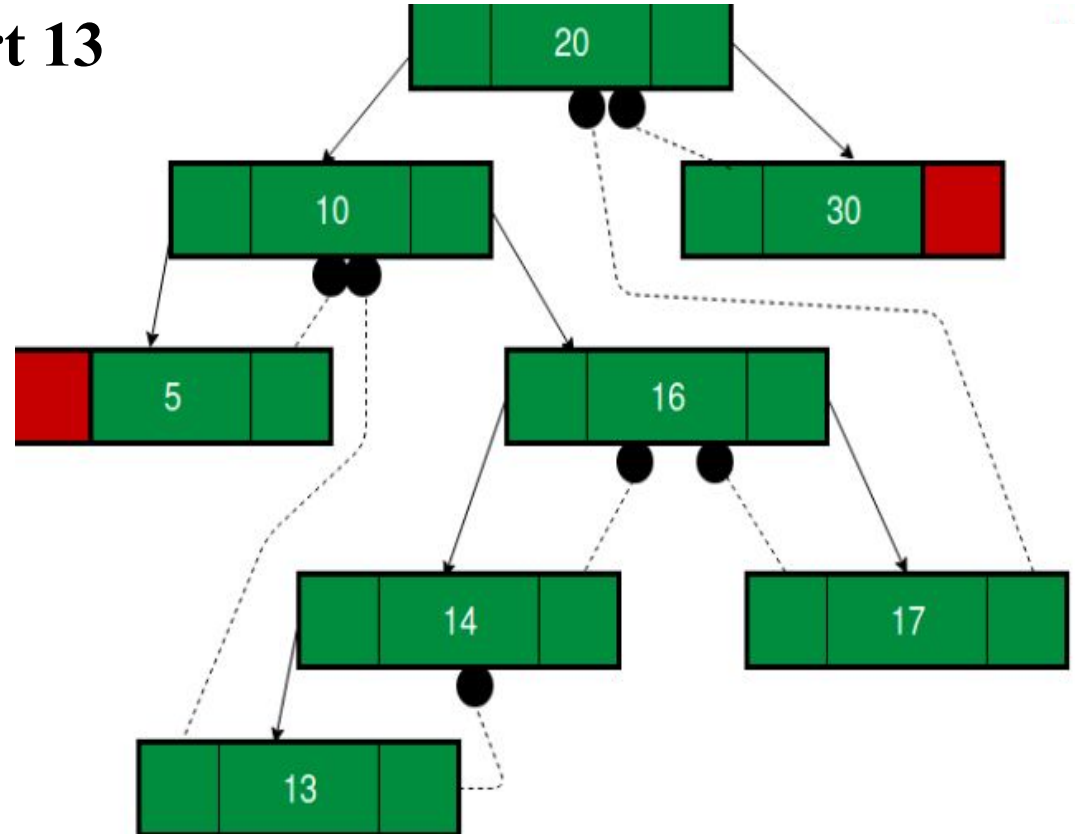
InsertLeft :

Make created node stick be left child of designated node through parameter

Node Insertion



insert 13



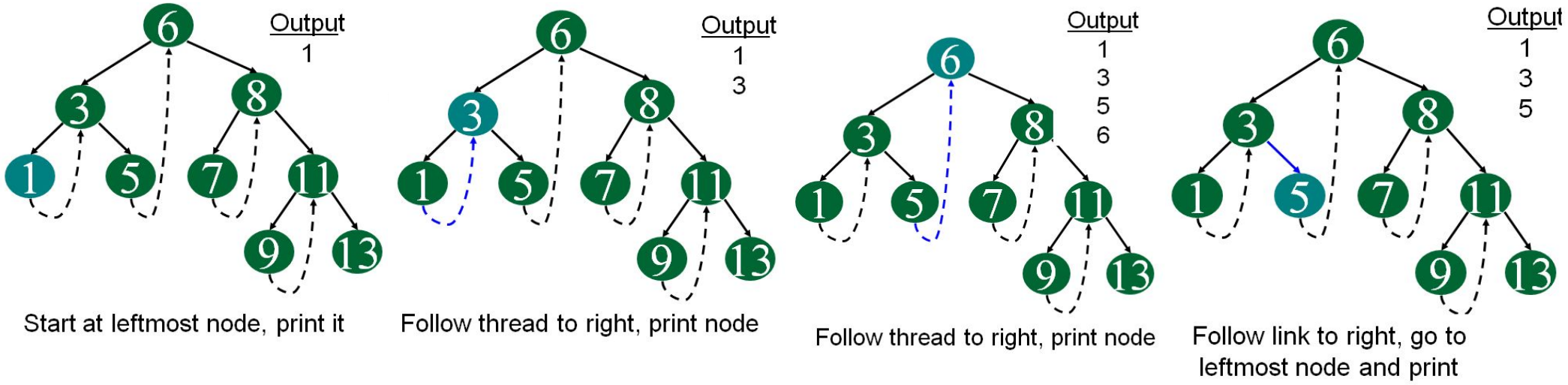
Inorder Traversal

Execute Inorder Traversal in iterative way using pointers.

You should NOT write recursive function.

tinorder :

Visit all nodes of tree without using recursive function



오픈랩 과제

Input
- Input.txt

```
S - 17
L 17 14
L 14 10
L 10 5
R 10 13
R 14 16
R 17 20
R 20 30
E
```

S : root node
L a b : 왼쪽 삽입
a : 부모노드 , b : 자식노드
R a b : 오른쪽 삽입
a : 부모노드, b : 자식노드
E : 종료

Output
- Command line

```
5
10
13
14
16
17
20
30
```

<Inorder traversal 결과 출력>

오픈랩 과제(Cont’)

오픈랩 요구사항

§ Tree.txt 에 저장된 트리 정보를 파일 입출력으로 읽어오기.

§ 다음 함수 구현하기:

- Insertion()
- InsertRight(), InsertLeft()

] 한 함수에 insert 구현해도 됨

- Traversal()
- insucc() 중위 순회 후속자 탐색

• 자율적으로 함수의 파라미터 설정 가능하고 자유롭게 함수 구현하세요.

• 전역변수사용불가.

• Single Threaded , Double Threaded Binary Tree의 제한은 없습니다.

오픈랩 과제(Cont')

- 제출메일

2019ds001@gmail.com

- 제출형식

  메일제목: [8주차]학번_이름

  파일이름:ds_open_8_학번.c (not cpp)

- Please double check the email, file format and input/ output format of your program before submission.
- Furthermore, please make sure that your file is ready to be submitted before actual submission.

  Re-submitted file will not be taken into account.

  Monday class's submission deadline date: 11/17.

  Friday class's submission deadline date: 11/21