오픈랩 과제#7 : Threaded Binary Trees

2018. 5. 24-25.

문제: Threaded binary tree가 주어졌을 때, 명시된 node의 오른쪽에 새로운 node를 삽입하는 함수를 구현하라. 노드의 삽입이 완료된 threaded binary tree는 inorder traversal를 사용하여 threaded binary tree가 잘 구성되어 있는지 확인한다.

자료구조와 알고리즘: 아래에 주어진 선언과 p.220의 program 5.10과 p.221의 program 5.11, 그리고 p.223 program 5.12를 참고.

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

void construct_tree(threadedPointer tree); // 주어진 threaded binary tree를 생성한다, 코드로 주어짐.
threadedPointer insucc(threadedPointer tree); // Program 5.10
void tinorder(threadedPointer tree); // Program 5.11
void insert(threadedPointer s, char data); // insertRight를 수행하기 전에 노드생성과 생성된 노드에 data를 저장한다, 직접작성
void insertRight(threadedPointer s, threadedPointer r); // Program 5.12

int main()
{
        /* initialize a head node */
        // 직접작성

        construct_tree(head);
        insert(head->leftChild->rightChild, 'E');
        insert(head->leftChild->leftChild->rightChild, 'F');
        insert(head->leftChild->leftChild, 'G');
        tinorder(head);
        return 0;
}
```

output

```
B   G   D   F   A   C   E
```