

# CSE 206 파일 처리론 과제 # 2

## 과제: Binary Search Tree 구현

### 1. 구현 내용: Binary Search Tree의 검색, 삽입, 삭제를 구현한다.

삭제에 있어서 삭제되는 키가 좌, 우 Subtree를 모두 가질 경우, 오른쪽 Subtree에서 가장 작은 key로 대체한다.

### 2. 입출력 포맷

2.1 입력: 파일이름은 **bst\_input.txt** 이다. 내용은 아래와 같다.

첫째 줄은 Test Case의 개수  $t$  를 나타낸다.  $1 \leq t \leq 10$ )

다음 줄부터 각 Test Case에 대한 데이터가 반복된다.

각 Test Case마다 Empty BST Tree에서 시작한다. 각 Test Case에 대해

- 첫째 줄은 삽입할 키의 개수  $i$  가 표기된다. ( $2 \leq i \leq 50$ )
- 둘째 줄은 삽입할  $i$ 개의 키가 삽입 순서대로 스페이스로 구분되어 표기된다. 입력되는 키 (값은 1이상의 정수)에 중복은 없다. (프로그램은 삽입할 키 정보를 사용하여 BST를 구성한다.)
- 셋째 줄은 검색할 키의 개수  $s_1$  가 표기된다. ( $1 \leq s_1 \leq i$ )
- 넷째 줄은 검색할  $s_1$ 개의 키가 검색 순서대로 스페이스로 구분되어 표기된다. (존재하지 않는 키를 검색하지는 않는다.)
- 다섯째 줄은 삭제할 키의 개수  $d$ 가 표기된다. ( $1 \leq d < i$ )
- 여섯째 줄은 삭제할  $d$ 개의 키가 삭제 순서대로 스페이스로 구분되어 표기된다. 삭제되는 키에 중복은 없고 입력되지 않은 키는 삭제하지 않는다. (프로그램은 삭제할 키 정보를 사용하여 키 삽입에서 구성한 BST를 변경한다.)
- 일곱째 줄은 검색할 키의 개수  $s_2$  가 표기된다. ( $1 \leq s_2 \leq i - d$ )
- 여덟째 줄은 검색할  $s_2$ 개의 키가 검색 순서대로 스페이스로 구분되어 표기된다.

(존재하지 않는 키를 검색하지는 않는다.)

**2.2 출력:** 파일이름은 **bst\_output.txt** 이다. 내용은 이하와 같다.

입력 파일에서 정의된 테스트 케이스에서 검색한 각 키에 이르기까지의 검색 경로를 띄우지 않고 표기한다.

하나의 검색에 대한 결과(경로)를 한 줄에 표기한다. 또한 입력파일에서 먼저 검색한 검색의 결과가 이후에 검색한 검색 결과보다 앞줄에 위치한다.

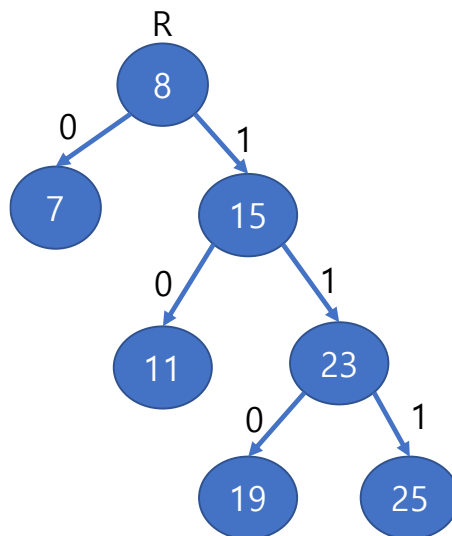
모든 경로는 루트 노드를 표현하는 문자 "R" 로 시작하고, 목적하는 키를 찾기 위해 각 노드에서 택해야 할 경로를 경로에 따라 "0" 혹은 "1" 로 표기한다. 구체적으로는 왼쪽 Subtree를 택했을 경우 "0", 오른쪽 Subtree를 택했을 경우 "1"로 표기한다.

예를 들어 아래 BST에서

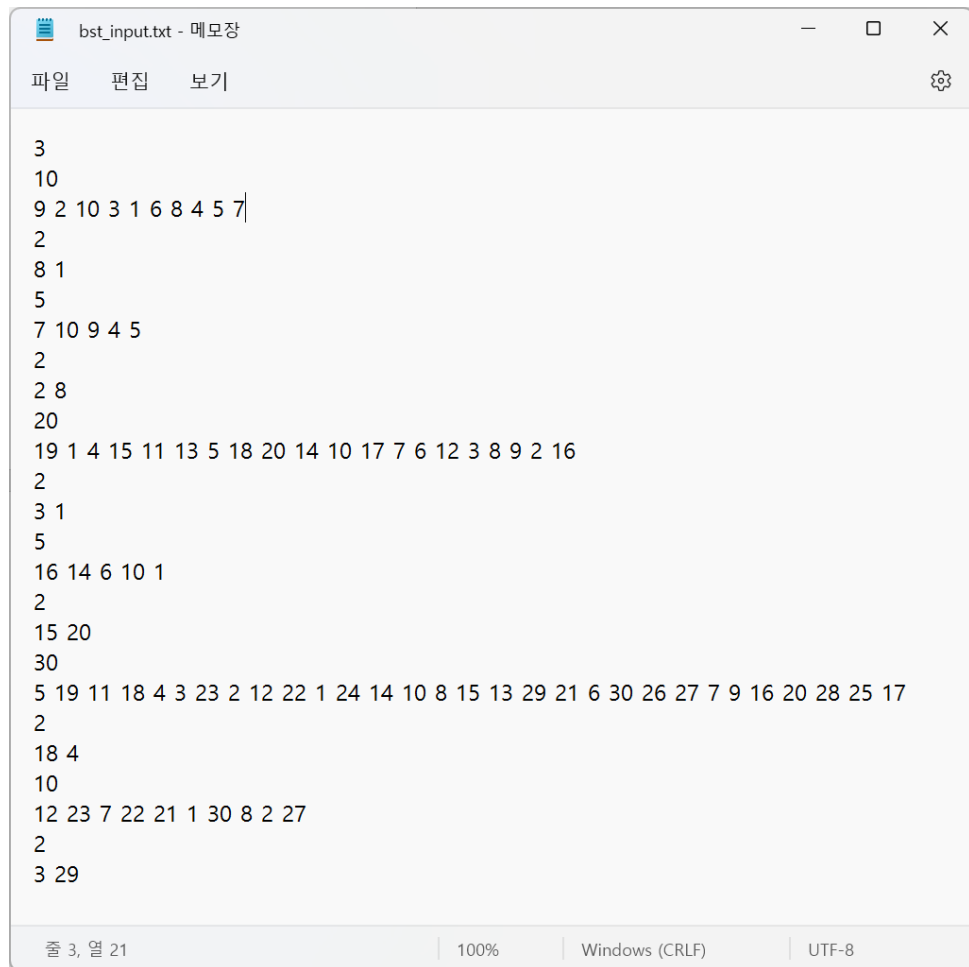
8을 검색할 경우의 경로는 "R",

11을 검색할 경우의 경로는 "R10"

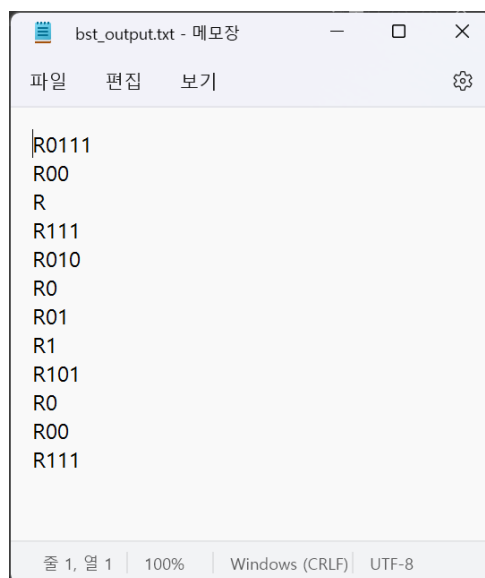
25를 검색할 경우의 경로는 "R111" 이다.



### <입출력 파일 예>



```
bst_input.txt - 메모장
파일 편집 보기
3
10
9 2 10 3 1 6 8 4 5 7
2
8 1
5
7 10 9 4 5
2
2 8
20
19 1 4 15 11 13 5 18 20 14 10 17 7 6 12 3 8 9 2 16
2
3 1
5
16 14 6 10 1
2
15 20
30
5 19 11 18 4 3 23 2 12 22 1 24 14 10 8 15 13 29 21 6 30 26 27 7 9 16 20 28 25 17
2
18 4
10
12 23 7 22 21 1 30 8 2 27
2
3 29
줄 3, 열 21 | 100% | Windows (CRLF) | UTF-8
```



```
bst_output.txt - 메모장
파일 편집 보기
R0111
R00
R
R111
R010
R0
R01
R1
R101
R0
R00
R111
줄 1, 열 1 | 100% | Windows (CRLF) | UTF-8
```

위 예에서 입력파일의 case수  $t$ 가 3이고, 각 case마다 키 삽입 후 2번, 삭제 후 2번의 검색이 수행되었으므로 출력파일에는 12줄 ( $3 * (2 + 2)$ )의 경로가 출력된다.

### 3. 구현언어: Python으로 구현

프로그램은 bst.py로 한다.

Python: Python 3.9 이상 사용, Colab사용 가능

### 4. 제출물: 이하 제출물 중 하나라도 미제출일 경우 0 점, 과제 리포트에서 설명한 컴파일 방법에 따라 컴파일 불가능할 경우 0점.

1. 구현한 프로그램의 source code
2. 과제 리포트
  - 구현한 프로그램의 실행 결과 스크린 샷
  - 프로그램 구현에 있어서 어려웠던 점이 무엇이고 어떻게 해결하였는가?

## 5. 배점: 모든 제출물을 문제없이 제출했을 경우

- 프로그램 정확성: 70점, 과제 리포트: 25점으로 한다.
- 프로그램 정확성은 임의의 테스트 케이스 입력 파일에 대해
- 과제 풀이로 제출한 프로그램이 계산한 결과를 테스터에 입력하여 **테스터 (BST\_Tester.exe)의 테스트를 패스하는가로 판단**한다. 패스하지 않을 경우 프로그램 정확성 0 점.
- **테스터 사용법:**

bst\_input.txt (입력파일), bst\_output.txt (프로그램이 계산한 출력파일) 및 BST\_Tester.exe를 동일 폴더에 넣고 BST\_Tester.exe를 실행.

- 일부 기능만 구현하였을 경우, 부분점수를 받기 위해서는 구현하지 않은 기능에 대한 검색 결과는 더미 값을 넣어 둘 것.

예를 들어 삽입과 검색만 구현한 경우, 삭제 후 검색에 대한 답을 아무 값이나 넣어 놓아야 테스터가 해당 부분 오답 처리하고 삽입 후 검색에 대하여 바르게 채점함.

```
##### RESULTS #####
R0111
R00
R
R111
R010
R0
R01
R1
R101
R0
R00
R111
#####
[RIGHT] tester: R0111, yours: R0111
[RIGHT] tester: R00, yours: R00
[RIGHT] tester: R, yours: R
[RIGHT] tester: R111, yours: R111
[RIGHT] tester: R010, yours: R010
[RIGHT] tester: R0, yours: R0
[RIGHT] tester: R01, yours: R01
[RIGHT] tester: R1, yours: R1
[RIGHT] tester: R101, yours: R101
[RIGHT] tester: R0, yours: R0
[RIGHT] tester: R00, yours: R00
[RIGHT] tester: R111, yours: R111
Your Score is 100.00
press any key to terminate.
```

## 6. 기한 및 제출 방법:

2024-12-12(목) 23:59분까지 가상대학의 해당과제 제출을 통해 제출

지각제출 인정하지 않음.

## 7. 기타:

과제 수행 시 모르는 부분에 대해 수강생들끼리 논의 가능하나, 아이디어와 방법에 대해 논의할 것.

- 아이디어 및 방법을 논의하는 것은 가능하나, 코드를 직접 보여주거나 작성해 주는 행위 금지.
- 단톡방에서 논의할 경우 4명까지.
- 논의한 경우 과제 리포트에 누구와 어떤 논의를 하였는지 기입할 것. (기입없이 유사도 체크에 적발될 경우 유사한 모두 실격처리)