

문제

배열을 이용하여 **이진 탐색 트리 클래스**를 구현하십시오. 이 클래스는 다음과 같은 기능을 포함하고 있습니다.

1. 양수가 입력되면 이진 탐색 트리에 삽입 합니다.
2. 음수가 입력되면 그 수의 절대값을 탐색하여 해당 값을 출력하고 트리에서 삭제합니다. 이 때, 만약 해당 값이 트리에 없다면 0을 출력합니다. 단, 배열의 값을 모두 출력할 때 0은 생략합니다.
3. 0이 입력되면 배열의 전체 값을 출력하고 프로그램을 종료합니다.

입력 형식 :

1. 동시에 트리에 들어가는 숫자는 8개 이하입니다.
2. 중복은 존재하지 않기 때문에 별도의 처리는 불필요 합니다.
3. 배열의 0 번째 자리는 사용하지 않습니다. 또한, 배열의 값을 출력할 때에도 0 번째 자리는 생략합니다.

힌트:

1. search, insert, delete 함수가 필요합니다.
2. 배열 구현이므로 delete 함수 구현 시 재귀함수 호출이 필요할 수 있습니다. 따라서 인덱스를 받으면 해당 인덱스를 삭제해주는 deleter 함수를 별도로 구현하는 것이 용이할 수 있습니다.
3. 특정한 값의 인덱스를 찾는 getIndex 함수를 만들면 편할 수 있습니다.
4. 출력 후 삭제를 구현할 때에는 메인 함수에서 search와 delete 함수를 연달아 호출하는 것이 간단합니다.
5. 삭제를 구현할 때 다양한 방법이 있습니다. 어떤 방법이든 채점이 가능하도록 테스트 케이스가 구성되어 있으므로 자유롭게 구현하면 됩니다.

제한 사항:

1. 클래스로 구현합니다(C 언어의 경우 struct 사용)

입출력 예시 1:

```
8
1
7
2
6
3
5
4
-7
7
-7
0
-6
6
-5
5
-4
4
-3
3
-2
2
-1
1
0
8
```

예시 힌트 1

0을 입력하기 전에는 프로그램이 종료되지 않기 때문에 음수 값을 입력하면 절대값을 취한 후 해당 값을 찾아서 출력 및 삭제가 일어납니다. 마지막에 0이 입력되면 남아있는 값을 출력한 후 종료합니다.

입력 예시 2:

```
50
100
1
99
2
98
3
97
0
```

출력 예시 2:

```
50 1 100 2 99 3 98 97
```

예시 힌트 2

입력된 값을 토대로 BST를 그려보는 것을 추천합니다. 출력 예시는 root 노드부터 차례대로 depth를 기반으로 출력(배열로 구현했기 때문에) 된 모습입니다.

C / C++ 를 사용하시는 학생 분들은 아래의 품을 참고해서 작성해 주셔야 기본적인 컴파일 에러를 방지할 수 있습니다.

또한 C 언어의 경우 표준 컴파일러에서는 scanf_s 또는 printf_s 등과 같이 "_s"를 붙이는 경우 컴파일 에러가 발생하기 때문에 "_s"를 제거한 scanf / printf 등의 함수를 사용하시기 바랍니다.

C:

```
#include <stdio.h>

int main() {
    /* TODO */

    return 0;
}
```

C++:

```
#include <iostream>
using namespace std;

int main() {
    /* TODO */

    return 0;
}
```

모범답안

```
1 #include <iostream>
2 using namespace std;
3
4 class bst {
5 private:
6     int arr[501];
7     int current;
8 public:
9     bst() {
10         for (int i = 0; i <= 500; i++) arr[i] = 0;
11         current = 1;
12     }
13     int search(int input) {
14         current = 1;
15         for (int i = 1; i < 501; i++) {
16             if (arr[i] == input) return i;
17         }
18         return 0;
19     }
20     void insert(int input) {
21         current = 1;
22         if (input > 0) {
23             while (1) {
24                 if (arr[current] == 0) {
25                     arr[current] = input;
26                     break;
27                 }
28                 else if (arr[current] > input) current = current * 2;
29                 else if (arr[current] < input) current = current * 2 + 1;
30             }
31         }
32         else if (input < 0) {
33             if (search(-input) == 0) cout << 0 << endl;
34             else if (search(-input) != 0) {
35                 cout << -input << endl;
36                 deleter(-input, 1);
37             }
38         }
39     }
40     void deleter(int input, int index) {
41         int seindex = search(input);
42         if (arr[seindex * 2] == 0 && arr[seindex * 2 + 1] == 0) {
43             arr[seindex] = 0;
44         }
45         else if (arr[seindex * 2] != 0 && arr[seindex * 2 + 1] != 0) {
46             int curindex = seindex;
47             curindex = 2 * curindex + 1;
48             while (1) {
49                 if (arr[curindex * 2] == 0 && arr[curindex * 2 + 1] == 0) break;
50                 else {
51                     if (arr[curindex * 2] != 0) curindex = curindex * 2;
52                     else if (arr[curindex * 2 + 1] != 0) curindex = curindex * 2 + 1;
53                 }
54             }
55             arr[seindex] = arr[curindex];
56             arr[curindex] = 0;
57         }
58         else if (arr[index * 2] != 0 || arr[index * 2 + 1] != 0) {
59             int curindex = (arr[seindex * 2] == 0) ? seindex * 2 + 1 : seindex * 2;
60             arr[seindex] = arr[curindex];
61             arr[curindex] = 0;
62         }
63     }
64     void print() {
65         for (int i = 1; i <= 500; i++) {
66             if (arr[i] != 0) cout << arr[i] << " ";
67         }
68         cout << endl;
69     }
70 };
```

```
72 ▾ int main() {  
73     bst b;  
74     int value;  
75 ▾     for (;;) {  
76         cin >> value;  
77 ▾         if (value == 0) {  
78             b.print();  
79             break;  
80         }  
81         b.insert(value);  
82     }  
83     return 0;  
84 }
```