CS2233: Data Structures
Assignment 2B
8th September, 2018

Problem Statement

- Input: A set $S = \{a_1, a_2, \ldots, a_n\}$ of natural numbers given as a sequence. Additionally, a natural number t or a request to print the set.
- Output: Let B be the Binary Search Tree (BST) formed by inserting elements in S to an empty BST in the same order as the input. If the additional input is a number t:
  - If $t \in S$: output the location of t in B
  - Else, indicate that $t \notin S$.

  Else, on request to print the set: print the pre-order traversal of B.

Note: Full credit only if your program can handle numbers of arbitrary size. Partial credit (75%) otherwise.

Input Format

Each line of the input starts with one of three symbols:

- 'N' (stands for new set)
- 'S' (search within most recent set)
- 'P' (pre-order traversal)

Format in detail:

- A line that starts with '**N**' is folowed by a space and $a_1, a_2, \ldots, a_n \in \mathbb{N}$ with a space between consecutive numbers followed by a '\n' character.
- A line that starts with '**S**' is followed by a space and a $t \in \mathbb{N}$ followed by a '\n' character.
- A line that starts with '**P**' is followed immediately by a '\n' character.
- End of input is indicated by EOF.

Output Format

- If the input line started with 'N', no corresponding output.
- If input looked like 'S' followed by $t \in \mathbb{N}$:
  - Output "-1" if t is not found in the set formed by the numbers in the most recent line that started with "N".
  - Else, output a bit string that represents the path from root to the node containing t in the BST formed by the most recent sequence. Bits 0 and 1 represent left and right respectively. End the line with a '\n' character
- If input line was 'P', then output the pre-order traversal of the BST constructed most recently. Follow up with a '\n' character.

Implementation rules

- Use a BST to implement the set. Every node should maintain pointers to left child, right child and the parent.
- Handle the input line by line. i.e., your program should not wait till EOF to produce output.
- Input could have multiple occurences of the same number. Your BST should store each number exactly once.
- Remove any leading zeros from the input before inserting it into your BST. This ensures your output does not have any leading zeros.

Design decisions

- Decide if you want to allow arbitrary size integers. Mark this decision as a comment in your program. This affects the maximum score you can obtain.
- If you want to handle arbitrary size integers:
  - Think about whether you need to change any function signatures from your own programs in Assignment 1 and 2A. Also check if overloading some functions will be of use.
  - Do you want to handle NULL pointers or do you want to create a sentinel "NIL" node? Think about the benefits and drawbacks of both.

Other Remarks

- You are encouraged to use your own programs from Assignment 1 and 2A. You might have to write a procedure to check equality between two (large) integers.
- **Deadline:** Midnight of 16th September 2018.

Example

Input:

```
---------------------------------
N 12 15 9 34 8 14 10 78 3
S 16
S 14
S 34
S 12
S 35
P
S 10
N 321273 335442 8899188334 351000238891223 2 15
S 335442
S 321273
S 12
S 8
S 15
P
S 351000238891223

---------------------------------
```

Output:

```
---------------------------------
-1
10
11

-1
12 9 8 3 10 15 14 34 78
01
1

-1
-1
01
321273 2 15 335442 8899188334 351000238891223
111

---------------------------------
```