Michelle Best, Eric Wilson, Lyuba Fridman

[PhoneBook](https://github.com/hunter-teacher-cert/csci70300-summer-2021/blob/main/code/ds_design/data-structure-design-challenge.md) should support the following operations:

1. add - you should be able to add new people in to the PhoneBook by lastName. This operation should be **optimized to be as fast as possible**.
2. find - you should be able to find a person by lastName. For example, pb.find("Smith") would return the person in the phone book with last name "Smith". Assume that last names are unique. Find should return null if the person is not in the phone book. **This operation should be optimized to be as fast as possible.**
3. remove - remove a person from the list based on lastName. This operation does not have to be optimized.
4. printList - this would print out an entire phone book. This operation does not have to be optimized.

Add a file or files into your repo describing this data structure and the algorithms for the operations

Individuals stored as:

public class person {

private String lastName;

private String firstName;

private String phoneNumber;

// other members not shown

}

Entry class:

public class PhoneBook{

// implementation not shown

}

### Our Design

#### Data Structures

Binary search tree will hold Nodes. Each Node has an instance variable called data of type Person and a next pointer of type Node.

TreeNode:

* data: Person
* left: TreeNode
* right: TreeNode

PhoneBook:

* root: TreeNode

#### Algorithms

Add - BST add

Find - BST find

Remove - BST remove

printList() - BST in order traversal