COMPUTER SCIENCE 21A (FALL, 2018)
DATA STRUCTURES AND ALGORITHMS

PROGRAMMING ASSIGNMENT 2

- **Due Saturday, November 17th, at 11:30PM**. This is a hard deadline, no exceptions. If you wait to submit until right before the deadline, you risk not turning it in.
- Late submissions will not receive credit.
- Submit as a zipped folder containing the .java files in the format LastnameFirstname_PA2.zip.
- Read these instructions in their entirety from start to finish before beginning and make sure you understand all of the specifications and requirements for submission.

Building a Digital Library Record

You have been tasked with building a Digital Library of hundreds of books so that scholars far and wide can find the resources they need for their research. Luckily for you, researchers will only want to search with two fields: Author Name, and ISBN (https://en.wikipedia.org/wiki/International_Standard_Book_Number). You will be given a text file containing book entries in them. Each line will have the book title, author, and ISBN separated by tabs.

The program you build will allow users to enter in an author name or ISBN, and your program should output the title, author, and ISBN to the user. This will help them when they check it out at their local or university library.

Of course, your boss wants the search to be efficient, so they tell you to implement the digital library using splay trees to manage your searches.

Your assignment will be graded using JUnit tests. When a structure is specified, you must follow it exactly so that we can test it with our JUnit tests. If we cannot test parts or all of your code, you will automatically receive a zero on those parts of the tests that are not passed. **This means we expect the explicit naming of the classes and methods specified. Again, if we cannot test your code with our JUnit suite, you will receive a zero on the assignment.**

Implementing the Splay Tree

You need to implement splay trees in Java. Unlike the previous assignment, you will not be making a class to hold the entire splay tree, and will pass the root node of your tree instead.

Specifications for SplayTreeNode

- The object name must be SplayTreeNode
- SplayTreeNode must be a Generic Class.
- An overloaded constructor that can accept either the data it stores, or nothing at all.
- A field data, of type Generic
- A field left, of type SplayTreeNode
- A field right, of type SplayTreeNode
- A field parent, of type SplayTreeNode
- toString(), which will return a string in the format:
    data.toString()
    LEFT | left.data.toString() or NULL if there is no left.data
    RIGHT | right.data.toString() or NULL if there is no right.data

Note that LEFT is followed by 4 spaces, a vertical bar, and 4 more spaces, and RIGHT is followed by 3 spaces, a vertical bar, and 4 more spaces.

- You may not have any more fields or methods than these specified ones.

Specifications for Book

- The object name must be Book
- A constructor that accepts a title, author, and ISBN in that order.
- A field title, of type String
- A field author, of type String
- A field ISBN, of type long
- toString(), which will return a string in the format:
    title, author, ISBN

Note that each comma is followed by exactly one space

Specifications for SplayTreeUtils

- The class name must be SplayTreeUtils
- SplayTreeUtils cannot store data. It is just a library of functions.
- zig(node) Performs zig rotation about SplayTreeNode *node*
- zag(node) Performs zag rotation about SplayTreeNode *node*
- insert(root, node, mode) Inserts a SplayTreeNode *node* into the splay tree with the root node *root* comparing by Author if mode is 0 and ISBN if mode is 1
- delete(root, node, mode) Removes a SplayTreeNode *node* from the splay tree with the root node *root* comparing by Author if mode is 0 and ISBN if mode is 1. Returns the root node.
- search(root, searchItem, mode) Searches for SplayTreeNode node with attribute *searchItem* and splay it (or the last searched item) to the root position, comparing by Author if mode is 0 and ISBN if mode is 1 Returns the root node.
- You **are allowed** more fields and methods than these specified ones.

Implementing the Digital Library

Now that you have built your data structures, you may proceed to build this digital catalog. When you run your program it should begin with the following output.

```
Welcome to the SPLTREE_DIGITAL_LIBRARY.
Loading library...
```

Your program should try to load data from three text files: spltree_digi_lib_auth.txt, spltree_digi_lib_isbn.txt, spltree_digi_lib_borrowed.txt. These files will hold the splay trees organized by author, by ISBN, and borrowed books respectively. You will be writing out to these files during the program, so it is up to you how they are formatted.

If these files don't exist, you will read in from a text file called spltree_digi_lib_baselib.txt, which will be in the format specified in on the first page. (The file contains book entries in them on each line. Each line will have the book title, author, and ISBN separated by tabs.)

Read the data from the appropriate source above and create two splay trees. One should be organized by author, and the other by ISBN. The borrowed books splay tree should be the contents of spltree_digi_lib_borrowed.txt, or empty if from the baselib. If you read in from spltree_digi_lib_baselib.txt, immediately write them out (in a format you decide) to the three files above.

Update your output to the user:

```
Welcome to the SPLTREE_DIGITAL_LIBRARY.
Loading library...  DONE.

Please enter 'author' to search by author name, 'ISBN' to search by
reference ISBN, 'popular' to see the top books, 'return' to return a book,
or 'exit' to leave the program:
```

Your program should loop until valid input is entered.

The following actions should be taken for each option:

author	Prompt the user with the string "You have selected Search by Author. Please enter the author name:" Search the author splay tree for the specified author name. If it is found, print the string "The following entry matched your search term:" followed by the toString method of the book. Prompt the user "Would you like to borrow this book? (y/n) " and loop for valid input. If yes, remove the book from both the author and ISBN splay trees and move it to the borrowed books splay tree. The borrowed books splay tree should be organized by author name. Write out the author, ISBN, and borrowed books splay trees to the text files. If no book was found, print "Sorry, no books were found with your search term."

ISBN	Prompt the user with the string “You have selected Search by ISBN. Please enter the ISBN:” Search the ISBN splay tree for the specified ISBN. If it is found, print the string “The following entry matched your search term:” followed by the toString method of the book. Prompt the user “Would you like to borrow this book? (y/n) ” and loop for valid input. If yes, remove the book from both the author and ISBN splay trees and move it to the borrowed books splay tree. The borrowed books splay tree should be organized by author name. Write out the author, ISBN, and borrowed books splay trees to the text files. If no book was found, print “Sorry, no books were found with your search term.”
popular	Print the toString methods for both the root of the author and ISBN trees on separate lines.
return	Prompt the user with the string “Please enter the author for the book you are returning:” If it is in the borrowed books splay tree, print “Thank you for returning this book.” Remove the book from the borrowed books splay tree and insert it into the author and ISBN splay trees. Write out the author, ISBN, and borrowed books splay trees to the text files. If no book was found, print “Sorry, no books were borrowed with that author.”
exit	Exit the program

These functions must belong to the following class:

Specifications for SplayTreeDigitalLibrary

- The class name must be SplayTreeDigitalLibrary
- String main(String[] args) Runs the entire program. If args is not empty, you need to accept each String as input as if it was entered into the console. This method must return a string with the exact output you’ve printed to the console.
- authorSearch(String authorName) Performs the author search described above
- isbnSearch(long isbn) Performs the ISBN search described above
- popular() Performs the popular command described above
- returnBook(String authorName) Performs the return command described above
- authorSplayTree, the root of your author splay tree (we need to access this for JUnit Testing)
- isbnSplayTree, the root of your author splay tree (we need to access this for JUnit Testing)
- borrowedSplayTree, the root of your author splay tree (we need to access this for JUnit Testing)
- You **are allowed** more fields and methods than these specified ones.

Furthermore, all of this must be in a package called pa2 with subfolders SPL_DIGITAL_LIB for your code, and testSuite for the JUnit testing suite. This structure is provided for you with the files uploaded to LATTE.

As always we expect that you will comment your code, and write in modular classes. We expect edge cases to be handled.

JUNIT Testing

We have given you a non-exhaustive file of JUnit tests for this assignment. Use these to learn JUnit, along with the slides from the first recitation. You must write and submit your own JUNIT tests. Code coverage is a consideration. We do not want 'happy path' programming, and do not make assumptions. Please handle any deviations from the happy path.

Grading

- You will be graded on the percentage of JUnit tests that your program passes. We have many more JUnit tests for the above specifications that will not be released. Follow the specs of this assignment, and you will pass them. If we cannot test parts or all of your code, you will automatically receive a zero on those parts of the tests that are not passed. **This includes the explicit naming of the classes and methods specified. Again, if we cannot test your code with our JUnit suite, you will receive a zero on the assignment.** This accounts for 65% of the project grade.

- You must write and submit your own JUNIT tests, and code coverage will be a consideration in the grade. This will account for 10% of the project grade.

- We will grade you on the code style and algorithm design. This project is object oriented, and we expect to see reasonable modularity in your design. This will account for 10% of the project grade.

- **The JavaDoc style documentation for each of your methods must contain the Big-O time complexity of the method.** Appropriate comments and documentation is required. We need to understand your thought process and your algorithm. **Comment your code**. This will account for 15% of the project grade.

Example Output

```
Welcome to the SPLTREE_DIGITAL_LIBRARY.
Loading library...  DONE.

Please enter ‘author’ to search by author name, ‘ISBN’ to search by
reference ISBN, ‘popular’ to see the top books, ‘return’ to return a book,
or ‘exit’ to leave the program: author
You have selected Search by Author. Please enter the author name: asdf
Sorry, no books were found with your search term.

Please enter ‘author’ to search by author name, ‘ISBN’ to search by
reference ISBN, ‘popular’ to see the top books, ‘return’ to return a book,
or ‘exit’ to leave the program: author
You have selected Search by Author. Please enter the author name: cormen
The following entry matched your search term:
Introduction to Algorithms    Thomas H Cormen   9780262033848

Would you like to borrow this book? (y/n) y

Please enter ‘author’ to search by author name, ‘ISBN’ to search by
reference ISBN, ‘popular’ to see the top books, ‘return’ to return a book,
or ‘exit’ to leave the program: author
You have selected Search by Author. Please enter the author name: cormen
Sorry, no books were found with your search term.

Please enter ‘author’ to search by author name, ‘ISBN’ to search by
reference ISBN, ‘popular’ to see the top books, ‘return’ to return a book,
or ‘exit’ to leave the program: return
Please enter the author for the book you are returning: cormen
Thank you for returning this book.

Please enter ‘author’ to search by author name, ‘ISBN’ to search by
reference ISBN, ‘popular’ to see the top books, ‘return’ to return a book,
or ‘exit’ to leave the program: foo

Please enter ‘author’ to search by author name, ‘ISBN’ to search by
reference ISBN, ‘popular’ to see the top books, ‘return’ to return a book,
or ‘exit’ to leave the program: exit
```