CSL 201 Data Structures
Lab 5 Due on October 16, 11.55pm

Instructions

- You are to use Java as the programming language. Use Eclipse as the IDE.
- You have to work individually for this lab.
- You are not allowed to share code with your classmates nor allowed to use code from the internet. You are encouraged engage in high level discussions with your classmates; however ensure to include their names in the report. If you refer to any source on the Internet, include the corresponding citation in the report. If we find that you have copied code from your classmate or from the Internet, you will get a straight F grade in the course.
- The submission must be a zip file with the following naming convention - rollnumber.zip. The Java files should be contained in a folder named after the question number.
- Include appropriate comments to document the code. Include a `read me` file containing the instructions on for executing the code. The code should run on institute linux machines.
- Upload your submission to moodle by the due date and time. Do not email the submission to the instructor or the TA.

This lab will improve your understanding of search tree structures - AVL Trees. This involves a lot of programming, so please get started early.

1 Transaction Database (25 points)

In this assignment we will simulate the select, insert, update and delete operations on a transaction database. Every record of the database has the following attributes

- `tid` - a unique 8 digit integer identifying a transaction that has the natural number ordering defined on it.
- `item` - a string value representing the item in the transaction,
- `cost` - a double value representing the cost of the transaction

We will maintain two AVL trees, one for each of the attributes - `tid` and `cost`. The AVL tree built using `tid` will be the primary tree. That is, the node of the AVL tree built on the key `tid` will store the entire record. The node of the AVL trees built using `cost` will only store pointers to the actual records in the primary tree. We will also maintain a hash table on the attribute `item`, that uses chaining for collision resolution. The hash table records for an `item` will also only store the pointers to the actual records in the primary tree. The three storage data structures must be updated after an insert, update or delete query on the database. For the sake of simplicity we will assume that the attributes `tid` is unique.

Input Format

The input should read from a text file. Each line in the text file will be of the following format `op varargs`. The values that the character `op` can assume and the corresponding string of `varargs` are as follows:

- `i` - an insert operation. The value for `varags` will be `tidval itemval costval`. Inserts a new record into the database with the specified record values.

- `u` - an update operation based on the transaction identifier. We will be updating the database using only the transaction identifier. `varargs` is of the format `tidval itemval costval`. Updates a record with the transaction identifier `tidval` with the specified `itemval` and `costval` values.
- `d` - a delete operation on the database. `varargs` is a logical expression that will support equality queries such as `tid = tidval`, `item = itemval`, `item = itemval and cost = costval`, and range queries such as `tid < tidval`, `item = itemval and cost > costval` etc. All records that satisfy the logical expression must be deleted from the database. A delete operation might result in removal of more than one record from the database.
- `s` - a select operation. `varargs` is a logical expression that will support equality queries such as `tid = tidval`, `item = itemval`, `item = itemval and cost = costval`, and range queries such as `tid < tidval`, `item = itemval and cost > costval` etc. All the records that satisfy the logical expression must be printed in the format `tid item cost`, with one record per line.

Also, include a pdf document that describes your approaches for implementing delete and select queries efficiently. Discuss your approach to handle duplicate keys for the attribute `cost`.
The Lab folder also contains a sample data file for inserting 400 records into your database.

2 Documentation of Code (5 points)

Documentation of code will fetch 5 points.