# CS 4412 5512 Module 7 Project 7

# B+Tree

Module Objective: By the time students have completed this module, they will be able to do the following: Implement and analyze a B+Tree

Place your results in a zipped project folder named CS4412Pj7<yourLastName>.zip (or 5512) and submit through moodle. It should contain a .doc document containing the written question answers as well as the C++ code, that can all be contained in a .cpp file if you wish.

1. Implement a Person class in C++ for which each instance contains one tuple (row) in the spread sheet in this project. Assume the id number for each entry is the key.
2. Implement and test a B+Tree that provides access to instances of the above class. Keys are stored in the tree and pointers to instances of the class are contained in the leaf nodes. Use a constant for the number of keys in each B+tree node. Start with say 5.
   1. Implement Insert that inserts a “data record” into the B+Tree while maintaining proper B+Tree structure. Insert must correctly maintain the balanced nature of the B+tree.
   2. Implement Find that returns a data record given a key.
   3. Implement Delete that deletes a record given a key. Delete must maintain the balanced nature of the B+tree.
   4. Implement a Range-query that takes two arguments *low* and *high*, in the domain of the keys, and return a list of all data records with keys between *low* and *high*.
3. Implement 2 methods on your B+tree class that determine the min and max height of your B+tree.
4. Develop a written test plan and an acceptable user interface display to insure your B+tree class works correctly. It should allow reading in the data from the provided file, searching for items by key or other attributes, etc.
5. Modify your B+tree class to store unsigned ints. Create an array with 10^7 integers Insert these arrays into your B+tree. Delete the first ½ of them. Re-insert them. Delete the second ½. Re-initialize your array with a new set of integers. Insert the first ½, delete them. Insert the second ½. Etc. Do this a number of times.
   1. Using your min and max functions determine whether your tree has maintained its “balancedness.”
   2. Is this an ‘/; adequate and/or useful test of your data structure? What other tests might be needed to determine whether it works correctly?

Grading Rubric

1. 5
2. 10 1a 10 2b 10 2c 15 2d 15
3. 12
4. 5
5. 6 4a 6 4b 6
6. Style: up to -20 points. Style is critical. Each method (function), class, and program requires a block comment with your name and a really good sentence explaining the goal or purpose of the program, class, or method.