Homework Assignment 2

Linear Data Structures

This homework assignment consists of programming exercises derived from Chapter 3 in your textbook. We have provided a template (*hw2.py*) for you to extend along with a test program (*hw2_test.py*). We have also included an implementation of the UnorderedList class (in *UnorderedList.py*), which will be used for problems 2-4 and you'll need to modify for problem 5.

Start this assignment by downloading the template and the test program. Run the test program in the same directory as the assignment template before starting. It should tell you that none of the solutions have been implemented. You will complete the assignment by adding your code to the assignment template; each occurrence of "raise NotImplementedError" is a placeholder you need to replace with your solution. Running the test program along the way will give you feedback about your progress.

Upload the completed assignment template and associated files (*hw2.py* and *UnorderedList.py*, along with any other files you have used) to Gradescope before **Tue April 23 at 11:59pm**. You do not need to submit *hw2_test.py,* nor should you need to make any changes in that file. Gradescope will be grading your assignment based on a modified *hw2_test.py*, but will only publish the results of the "Sanity Tests" before the deadline. You may submit as many times as you like, but you should be testing on your own to ensure your code is working. The sanity test results are to let you know the upload was successful.

Problem 1 (20 points)

You are interested in a Queue with faster enqueue performance at the expense of dequeue. Modify the Python implementation of the Queue ADT (from Chapter 3.12 in the digital textbook) such that the rear of the queue is at the end of the list instead of the beginning. Name the new class QueueFE() for Fast Enqueue.

Problems 2-4 ask you to implement three abstract data types using the UnorderedList class described in chapter 3 in your book instead of the book's implementation using the Python list.

Problem 2 (20 points)

Implement the stack abstract data type using the UnorderedList class.

Problem 3 (20 points)

Implement a queue abstract data type using the UnorderedList class.

Problem 4 (20 points)

Implement a deque abstract data type using the UnorderedList class.

Problem 5 (20 points)

We will learn more about priority queues in the near future. For this problem we will extend the UnorderedList class so that it could be used to implement a priority queue. The key step is a method that removes and returns the minimum value in the list.

Extend the UnorderedList class so that it implements the method removeMin(self). This removes an item of minimum value. For a good starting point see the code for search(item) and remove(item) and read chapter 3.21.2. In this case you will have two steps. The first step is to find the minimum value. The second step is to remove it and return the value.