BIL 212
Summer 2021

Lab 1

Download DoublyLinkedList.java and DLLDriver.java from LinkedLists section under Resources on the course Piazza site. Run the DLLDriver and make sure you understand the existing methods.

Part 1.

Add the following methods to the DoublyLinkedList class and **test** them by adding appropriate statements into DLLDriver. Make sure to test all cases.

- private void addBefore(Node<E> node, E newData) : adds newData immediately before the given node.
- private void removeBefore(Node<E> node) : removes the node that comes immediately before the given node.
- private Node<E> getNode(int i) : returns the address of the i-th node. The node immediately after the header sentinel is the 0-th node. You must start iterating from head if i is closer to the start, and tail node if i is closer to the end. You can use size variable to determine.
- public E get(int i): returns the data at the i-th position (in the i-th node). Write this method by making a call to the getNode method you have written above.
- public void add(int i, E newData): adds newData at the i-th position. Write this method by making calls to the getNode and addBefore methods you have written above.
- public E remove(int i): removes the i-th node, and returns the data that is removed. Write this method by making calls to the getNode and removeBefore methods you have written above.
- public int find(E data): finds and returns the index of node that contains the given data, returns -1 if no node contains it. In order to compare nodes' data, you must ensure that E implements comparable. Then you can use compareTo() on data. You can ensure it exists by changing the definitions of Node and DoublyLinkedList classes.

Part 2.

- private void swap(Node<E> node1, Node<E> node2): Write a member method that swaps two given nodes in a doubly linked list. Note that you are not allowed to change the data inside the nodes, you must swap them by changing only the next and prev references. Also note that you cannot create a new node in the process.

- public void sort(): Write a member method that sorts the doubly linked list in increasing order using the bubble sort algorithm and the swap method you just wrote. You can make changes to the definition of DoublyLinkedList and Node in order to make sure E is a Comparable class, and use compareTo() on data. You can assume that your code will be tested using only comparable classes (like Integer or String). Again, you cannot create new nodes and cannot change the data inside of nodes, you must sort using the swap method.

  Construct an unsorted list in the DLLDriver.java, sort it using the sort() method you wrote, and print it to the console.

Make sure to check your section before attending. Please upload your DLLDriver.java and DoublyLinkedList.java files without compression to uzak.etu.edu.tr before 27.05.2021 Thursday 11:00.