O COMP 215 Algorithms Θ

Lab #1

For the sake of simplicity, all the elements in the classes that you will create below should be public.

1. Write a class 'cell' which contains an integer 'value' and a cell pointer 'next'.
2. Write a constructor for this class that takes an integer and creates a cell with this integer as its value, but without defining the pointer.
3. Write a second constructor for this class that takes an integer and a cell pointer, and that creates a cell with this integer as its value and this cell pointer as its 'next' pointer
4. Write a third constructor for this class that takes a cell and creates a *copy* of this cell.
5. Overwrite the operator '+' so that the sum of two cells is a new cell whose value is the sum of the values of the given cells (the 'next' pointer of this new cell is not assigned).
6. Write a class 'queue' that contains a cell pointer 'first' and a cell pointer 'last'.
7. Write a function 'pushLast' in the class 'queue' that takes an integer as input, creates a new cell with this integer value and places the cell at the end of the queue.
8. Write a function 'popFirst' in the class 'queue' that takes no input, removes the first cell of the list and returns the integer value of that cell.
9. Write a function 'maxValue' in the class 'queue' that returns the integer corresponding to the maximum value of contained in a cell of the queue. Be careful not to destroy the list in the process of finding this value.
10. Overwrite the operator '+' so that the sum of two lists is a new list whose values in its cells are the sum of the two corresponding cells in the input lists, that is, the value of the first cell of the new list is the sum of the values of the first elements of the two input lists, the value of the second cell of the new list is the sum of the values of the second elements of the two input lists, etc. It must be possible to do the sum of two lists of unequal length: in this case, the "tail" of the new list has cells with values equal to the "tail" of the longer of the two lists. Be careful not to destroy the two input lists in the process of creating this new lists, and be also careful to ensure that, if you are adding lists of unequal length the "tail" of the new lists is a *copy* of the tail of the longer of the two lists, rather than being composed of exactly the same cells.
11. Write a main method that allows you to test all the functions above.

You can leave when you have completed writing all this code, or when you have convinced yourself that you could if you wanted to (you really should try to do it though). There is nothing to hand in, but you should really make sure that you understand everything here, the course will proceed assuming that all of you could do the above.