COMP 203 Data Structures and Algorithms, Fall 2022
Instructor: Zafer Aydın
Final Exam

Submit your codes and answers to Canvas. Submit your codes and template files as a single zip file.

1. Augment the ArrayQueue implementation with a new rotate() method having semantics identical to the combination, enqueue(dequeue()). But, your implementation should be more efficient than making two separate calls (for example, because there is no need to modify the size). Hint: Start by checking if the size of the queue is less than the capacity of the array. Then copy the first element to the end of the queue and set the first element to null. Finally increase the f variable by one. Use modular arithmetic (i.e. circular computations) when necessary to compute indices. Your method may have the following declaration and will go inside ArrayQueue class

```
public void rotate()
{

}
```

Implement a main method to test your rotate method.

2. (a) The bubblesort algorithm is given below for sorting an array A of numbers.

```
BUBBLESORT(A)
1  for i = 1 to A.length − 1
2      for j = A.length downto i + 1
3          if A[j] < A[j − 1]
4              exchange A[j] with A[j − 1]
```

Note that in this pseudo-code array indices start from 1. Also in this pseudo-code convention

for i = 1 to n

means i runs from 1 to n (including n).

for i = n downto 1

means i runs from n to 1 (including 1). Exchange means swap.

Implement this algorithm as a method that receives an ArrayList as input. Do not implement your method inside ArrayList class. Implement it in another file called ArrayListTest.java in another class. Use get and set methods of ArrayList class to access and modify the contents of the data array. Your method may have the following declaration

```
public static void bubblesort(ArrayList<Integer> list)
{

}
```

Implement a main method to test your method. Instantiate an ArrayList object with an array capacity of 5 (call constructor with parameter 5). Call the add method 5 times by adding an integer each time. Print the contents of the ArrayList. Call the bubblesort method. Print the contents of the ArrayList after sorting.

(b) What is the running time complexity of bubblesort in Θ-notation (i.e. BigTheta notation) assuming that the input ArrayList contains n numbers? You don't have to compute the running time line by line. Just stating the complexity in BigTheta notation is sufficient.

3. Re-implement the positional list methods addLast and addBefore realized by using only methods in the set {isEmpty, first, last, before, after, addAfter, addFirst}. Your methods may have the following declarations

```
public Position<E> addLast(E e) {
      //check if the list is empty and add the element using the
addFirst method
      //else add the element using the addAfter method
}

public Position<E> addBefore(Position<E> p, E e) {
      //check if p is the first position and add the element using the
addFirst method
      //else add the element using the addAfter method
}
```

You don't have to implement a main method in this question to test your methods.