[**Closed book and notes.**] Show all of your work clearly in the space provided or on the additional page at the end of the exam. If the additional page is used, clearly identify to which exam question it is related. Be sure to **read each problem carefully**. Note that the exam is double sided.

1. (7 points) Explain what is wrong with the following method:

```
public static <E> E findMedian(Object[] data) {
  // ...
}
```

2. (7 points) Recall the LinkedList class that we developed in lecture. Give the big-oh time complexity for the implementation of the add(E element) we did in class. Justify your answer.

3. (6 points) Suppose that an additional attribute, tail, is added to the LinkedList class that points to the last node in the list. Give the big-oh time complexity for the most efficient implementation of add(E element) that you can think of. Justify your answer.

4. (20 points) Implement an inner class for the ArrayList we developed in lecture that implements the Iterator<E> interface. You do not need to provide an implementation for the remove method.

5. (15 points) Suppose that the LinkedList class that we developed in lecture was modified so that it only has one attribute: tail that points to the last node in the list. Further, suppose that the Node<E> class is modified so that the next attribute is replaced with a previous attribute that points to the previous node in the list instead of the next node in the list. Implement the contains method for the list.

6. (15 points) Recall that our implementation of the ArrayList<E> from lecture contains one attribute: data which is a reference to an array that stores all of the elements. Implement the add(int index, E element) method.

7. Consider the following method:

```
public static double getMedian(List<Double> nums)
{
  int N = nums.size();
  if(N<1)
  {
    throw new Exception("Grip a get");
  }
  double median = nums.get(0);
  int index = 0;
  for(int i=0; i<N/2; ++i)
  {
    int j;
    for(j=1; j<N-i; ++j)
    {
      if(nums.get(j)<median)
      {
        median = nums.get(j);
        index = j;
      }
    }
    swap(nums, j-1, index);
  }
  return median;
}
```

(a) (10 points) Suppose that the List passed to the method is an ArrayList and assume that swap swaps the element at $(j-1)$ and element at $index$ in nums and runs in $O(N^2)$ time. Using big-oh notation, describe the overall worst case time complexity for the getMedian method. Be sure to explain your reasoning and state any additional assumptions that you make.

(b) (10 points) Suppose that the List passed to the method is a java.util.LinkedList and assume that swap swaps the element at $(j-1)$ and element at $index$ in nums and runs in $O(N)$ time. Using big-oh notation, describe the overall worst case time complexity for the getMedian method. Be sure to explain your reasoning and state any additional assumptions that you make.

(c) (10 points) Suppose that the List passed to the method is the SinglyLinkedList developed in lecture and assume that swap swaps the element at $(j-1)$ and element at $index$ in nums and runs in $O(1)$ time. Using big-oh notation, describe the overall worst case time complexity for the getMedian method. Be sure to explain your reasoning and state any additional assumptions that you make.