Mehta Nimtrakul
I pledge my honor that I have abided by the Stevens Honor System.
CS 284 Midterm

Exercise 1:

1. True

Justification: Dynamic arrays allow for the use of a resizing method by doubling the capacity of the array to pop elements without having to reallocate memory every time. If the array was static, the array would have to be copied every time to a new array with size of N+1 and this would be inefficient.

Exercise 2:

1. B
2. B
3. C

Exercise 3:

a) O(n^2)
b) O(N log N)
c) O(N log N^2)
d) O(3N)
e) O(N^6)
f) O(log 4)^N

Exercise 4:

Original Array: [4,3,2,1]

1. Bubble Sort:

After first Iteration: [3,2,1,4]
After second Iteration: [2,1, 3, 4]
After third iteration: [1,2,3,4]
sorted.

2. Selection Sort:

After first Iteration: [1,3,2,4]
After second Iteration: [1,2,3,4]
After third iteration: [1,2,3,4]
// this checked that 3 was sorted
After fourth iteration: [1,2,3,4]
// this checked that 4 was sorted
Sorted.

3. Insertion sort

After first Iteration: [4,3,2,1]

- Mark first element as sorted

After second Iteration: [3,4,2,1]

- Insert 3 before 4 after comparing

After third iteration: [2,3,4,1]

- insert 2 before 3

After fourth iteration: [1,2,3,4]

- Insert 1 before 2

Exercise 5

1. High level description

   Make a new separate list with a head node and set the next node to the smaller value of both of the respective nodes of both linked lists while traversing through both.

2. Java code

```
Node temp = new Node(0);
Node current_node = temp

While (list1 != null && list2 != null){
        if( l1.data < l2.data){
                current_node.next = list1;
                list1 = list1.next;
        }
        Else{
                current_node.next = list2;
                list2=list2.next;
        }
        current_node = current_node.next
}
// cases to check if either did not traverse to the end
if(list1 != null) {
        current_node.next = list1;
        list1 = list1.next;
}
if(list2 != null {
        current_node.next = list2;
        list2 = list2.next;
}
Return temp.next;
}
```

3. Write time complexity

   Answer: O(n)

Exercise 6:

```
// built in function to sort array
```

```
Arrays.sort(array)
N = array.length

If n % 2 != 0:
        Return double(array[n/2])
Else:
        Return double(array[(n-1)/2] + array[n/2])
```