Part a: Code Analysis

```
class Mystery<T> {
    private ArrayList<T> data = new ArrayList<>();

    public void add(T elem) {
        data.add(0, elem);
    }

    public T remove() {
        return data.remove(0);
    }

    public T peek() {
        return data.get(0);
    }
}
```

For questions in this part, consider the following code:

Question 1 [**5 points**]

What are the tight, unqualified runtime bounds of `add`? If the Big-Θ bound does not exist, write **DNE**.

Big-O:

Big-Ω:

Big-Θ:

Question 2 [**5 points**]

What are the tight, unqualified runtime bounds of `remove`? If the Big-Θ bound does not exist, write **DNE**.

Big-O:

Big-Ω:

Big-Θ:

Question 3 [**5 points**]

What are the tight, unqualified runtime bounds of `peek`? If the Big-Θ bound does not exist, write **DNE**.

Big-O:

Big-Ω:

Big-Θ:

Question 4 [**5 points**]

Does `Mystery` exhibit the behavior of a `Stack`, `Queue`, or neither? In at most 2 sentences, explain your answer.

Part b: Asymptotic Analysis

For each question in this section, give the unqualified big-O, big-Ω, and big-Θ bounds for the specified function. If the big-Θ bound does not exist, write **DNE**. For this section you are not required to show any work or give a proof. To get full credit your bounds should be as simplified as possible.

Question 1 [**5 points**]

$$f_1(n) = 47n^3 + n^2 + n^2 \log(2^{n^2})$$

Big-O:

Big-Ω:

Big-Θ:

Question 2 [**5 points**]

$$f_2(n) = \sum_{i=1}^{15} \sum_{j=1}^{10} 2^i$$

Big-O:

Big-Ω:

Big-Θ:

Question 3 [**5 points**]

$$f_4(n) = \begin{cases} 5n^3 & \text{if n is prime} \\ 3n & \text{if n is greater than 2 and even} \\ \log(n) + 100n^2 & \text{otherwise} \end{cases}$$

Big-O:

Big-Ω:

Big-Θ:

Question 4 [**5 points**]

Is it possible for a function to be in both $\Theta(n^2)$ and $O(n^3)$? In at most two sentences, explain your answer.

Part c: Bounds Proofs

For each question in this part, you must prove the bound in question by coming up with constants c and n_0 that satisfy the inequalities as defined in class. You must show all work. **Answers given without showing sufficient work will receive no credit.**

Question 1 [10 points]

Let $g_1(n) = 6n\log(2^n) + \log(n) + 7n$. Prove $g_1(n) \in O(n^2)$.

Question 2 [10 points]

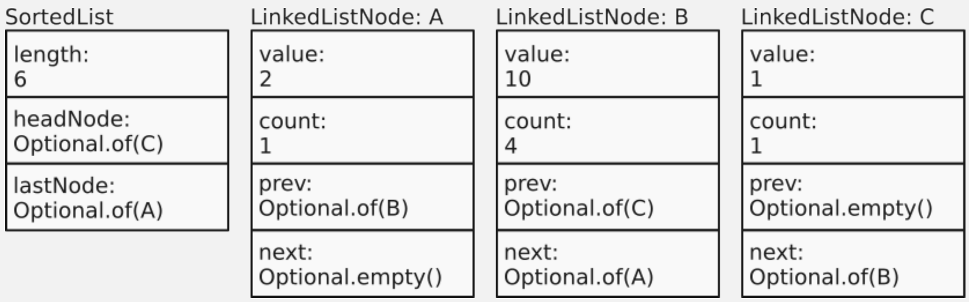
Let $g_2(n) = n^3 + 7n^2$. Prove $g_2(n) \in \Omega(n^3)$.

Part d: PA1 Review

The following two questions pertain to the `SortedList` data structure you implemented in PA1.

Question 1 [**10 points**]

The diagram below shows the nodes of a nearly valid `SortedList` data structure. There is exactly one error in the structure. Identify the error.



Question 2 [**10 points**]

Assume that the variable `list` is a `SortedList` containing N integers in the range from 0 to MAX. Suppose the following code has already been run:

```
// Generates a random integer i between 0 and MAX
Random r = new Random()
Integer i = Random.nextInt(MAX)

// Retrieve the node for value i, and save it as a hint.
LinkedListNode<Integer> hint = list.findRefBefore(i)
```

Assuming that `list.length()` is N, give a tight asymptotic upper (Big-O) bound on the runtime of the following block of code:

```
list.insert(i+2, hint)
```

Justify your answer by explaining which `LinkedListNode`s the `insert` operation would need to access in the worst case.

Part e: Data Structure Design

For each of the following scenarios, noting in particular the bolded text, state the data structure (Array, LinkedList, or ArrayList) you would use. In *at most 2 short sentences*, justify your answer in terms of how the properties of the data structure relate to the (bolded) requirements.

Question 1 [**10 points**]

Smart Watch Faces: You are implementing a 'watch face' manager for a smartwatch, and need a way to store a pointer to the region of memory used to store each watch face's state. Specifically, **you need to store one 8 byte pointer for each watch face**. You need to be able to jump to arbitrary watch faces quickly, so **you need to be able to access the ith pointer in constant time**. Memory on the watch is very limited, so **there will never be more than 19 watch faces open at a time**.

Question 2 [**10 points**]

Intrusion Detection System: You are implementing an intrusion detection system that works in two phases: First, **a large number of event objects are created and need to be stored**. Throughput is important, so it is critical that the **total cost of inserting all of the event objects is linear in the number of objects**. Then, in the second phase, the events are analyzed, requiring **constant-time access to elements by their index.**

Part f: Bonus

Question 1 [**5 points**]

Suppose you know that the function `foo()` has an **expected runtime** of $O(n)$.
What **guarantees** can you make about the unqualified runtime of the following code:

```
for (int i = 0; i < n; i++) {
  foo();
}
```

Scrap Page

Scrap Page