FAQ for CS417 Programming Assignment #7

1. Q: What is the point of this assignment?

   A: There are two goals:

   Notice that, with sentinel nodes, a lot of special cases drop out. In particular, in `insert` and `delete`, you don't have to check if the list is empty, or that you are inserting or deleting at the head or tail of the list.

   If you notice that you have code that handles those cases, you should probably get rid of it.

   The other goal is to prepare you for binary trees, which you will see in a week or two. Like linked lists, they consist of chained structures, where the data elements don't have names: they are accessed by walking down a chain of links.

2. Q: Can I have a hint?

   A: Here is `__setitem__`

```
    def __setitem__(self, index, value):
        if type(index) != int:
            # index must be an int
            raise TypeError
        elif index < 0 or index >= len(self):
            # index out of range
            raise IndexError
        else:
            # walk down the list, counting nodes.
            # start at the first DATA node (after the head sentinel)
            current = self._head._next
            for i in range(index):
                current = current._next
            current._value = value
```

3. Q: Why am I getting 55 pages of errors, and "maximum recursion depth exceeded". What's wrong?

   A: A couple of students had this issue. The problem was that `insert` calls `add_front`, and `add_front` calls insert. You have an infinite loop.

   The best way is to not check any special cases in insert. Look at the code in `__setitem__`: it walks down the list, arriving at the current node. Do something similar in `insert`: go to the current, node, where you will insert a new node.

Once you've arrived at the insertion point, create a new node, and carefully set its `_prev` and `_next` fields. Also, set those fields in its two neighboring nodes, so they point to the new node.

4. Q: The assignment handout is confusing. Which methods belong to class `Double_List`, and which ones to class `List_Node`?

   A: EVERY method listed in the assignment, except for the `__init__` method of the `List_Node` class, belongs to `Double_List`.

   To make things print out more nicely, you could also revamp the `__repr__` method of List_Node (which isn't mentioned in the handout). Every other method in the handout is part of `Double_List`.

   Indent your methods accordingly!!