3 A programmer is writing a class, `LinkedList`, to represent a linked list of unique integers. A linked list is a collection of data elements, whose order is not given by their physical placement in memory. Instead, each element points to the next.

For each of the sub-tasks, add a comment statement at the beginning of the code using the hash symbol '#' to indicate the sub-task the program code belongs to, for example:

```
In [1] :  # Task 3.1
          Program code

          Output:
```

Task 3.1

Write the `LinkedList` class in Python. Use of a simple Python list is not sufficient. Include the following methods:

- `insert(integer_value)` inserts the `integer_value` at the beginning (head) of the list
- `delete(integer_value)` attempts to delete `integer_value` from the list; if the item was not present, return `None`
- `search(integer_value)` returns a Boolean value: `True` if `integer_value` is in the list, `False` if not in the list
- `count()` should return the number of elements in the list, or zero if empty
- `to_String()` should return a string containing a suitably formatted list with the elements separated by a comma and a space, with square brackets at either end, e.g. in the form:

```
[11, 2, 7, 4]
```

[8]

Test `LinkedList` by using the data in the file `Task3data.txt`. Use the `to_String()` method to print the resulting contents of the list. [3]

Task 3.2

Write a Python subclass `SortedLinkedList` using `LinkedList` as its superclass.

The insert method in the `SortedLinkedList` subclass should ensure that the elements are stored in ascending order. [5]

Test `SortedLinkedList` by using the data in the file `Task3data.txt`. Use the `to_String()` method to print the resulting contents of the list.

Print the result of searching the `SortedLinkedList` for the value 94. [2]

Task 3.3

Write a Python subclass `Queue` using `LinkedList` as its superclass.

Additional `enqueue` and `dequeue` methods are to be defined on the `Queue` class:

- `enqueue(integer_value)` will insert `integer_value` to the end of the queue
- `dequeue()` will return the first element in the queue. If the queue is empty, return `None`. [6]

Test `Queue` by using the data in the file `Task3data.txt`. Print the first five elements to be dequeued from the list. [1]

Save your Jupyter notebook for Task 3.