Design 5 Template

Part 1: Recursive Fractals

Understanding the problem:

(1 pt) [Rephrase the problem statement]

To make sure that you understand the problem correctly, answer the following guiding questions:

For each line of the given examples on the assignment document:

1. (3 pts) How many leading white spaces are there? How many spaces are there between two stars?

2. (3 pts) What would \texttt{pattern (5, 7)} look like? \texttt{pattern (1, 0)}? \texttt{pattern (7, 1)}? Draw them out.

(1 pt) [List assumptions that you are planning to make for this program]

Program Design:

To help you break the problem down into smaller subtasks, answer the following guiding questions:

1. (1 pt) What is the general pattern of this recursive fractal?

2. (2 pts) For a general function call \texttt{pattern(n, col)}, how many recursive calls should it have? And what are they?

3. (1 pt) What is the base case of \texttt{pattern()}?
4. (1 pt) How many helper functions are you going to implement? What is the purpose of each of them?

(4 pts) [Create pseudocode (or flowchart) for `pattern()` and each of your helper function]

**Program Testing:**
To help you consider the possible test cases, answer the following guiding questions:
1. (1 pt) What is an example of good inputs?

2. (1 pt) Are there any inputs that would cause your program to crash?

3. (1 pt) What if the inputs are too large for your fractal output to fit into your screen?

(4 pts) [Create a testing table that has representative good, bad, and edge cases, and their expected outputs]

**Part 2: Linked List Implementation**

**Understanding the problem:**
(1 pt) [Rephrase the problem statement]

To make sure that you understand the problem correctly, answer the following guiding questions:
1. (2 pts) What would each `Node` object store? What does the `next` pointer point to?

2. (2 pts) In `LinkedList` class, what's the purpose of the `head` pointer? Why is it a pointer instead of a `Node` object?
3. (1 pt) Why is there no `set_length()` function?

(1 pt) [List assumptions that you made]

Program Design:
To help you break the problem down into smaller subtasks, answer the following guiding questions:

1. (2 pts) How would you iterate through the entire linked list? How could you tell if it reaches the end of the list?

2. (1 pt) How would you swap two nodes in a given linked list? Write down the steps in pseudocode.

3. (1 pt) Can any of the listed functions serve as a helper function to others?

4. (1 pt) How to apply the merge sort to `sort_ascending()`?

5. (1 pt) What algorithm are you going to use for `sort_descending()`? How would you apply it?

6. (1 pt) What other private member variables are you planning to add? What is the purpose of each?

7. (1 pt) How many helper functions are you going to implement? What is the purpose of each of them?

(4 pts) [Create pseudocode (or flowchart) for every function listed in the document, as well as your helper function(s)]
**Program Testing:**

To help you consider the possible test cases, answer the following guiding questions:

1. (1 pt) What is an example of good inputs?

2. (1 pt) Are there any inputs that would cause your program to crash?

3. (1 pt) What is your understanding of the providing test cases?

(4 pts) [Create a testing table that has representative good, bad, and edge cases, and their expected outputs. (You may use the provided test cases as reference, and you are encouraged to create your own test cases)]