Harrisburg University of Science & Technology

CISC 610 Data Structures & Algorithms

Instructor: Caleb Druckemiller, M.S.

Assignment 2

Linked Lists
You have been provided the code for doubly linked list: `doubly_linked_list.py`. Using this code, perform the following tasks:

> An interesting use-case of a linked list is to track edits in a "Undo - Redo" interface. Thus far we have really only talked about connecting singular data points in the nodes. The following task will deviate from that:
> I want you to create an interface of an application that will alter the contents of a list. As you alter the contents of the list, you should be able to retrieve the previous state of the list as you change it, much like being able to undo and redo changes in a document. Your application interface should support three basic functions, `undo()`, `redo()`, and `do()` where "do()" is what actually alters the contents of the list. You can interpret "do()" as what happens when you type in a word processor. I want you to "do()" the following actions:
>
> - Create a list with the initial values of: [5,3,2,1,4,8]
> - Sort the list.
> - Reverse the list
> - Add a 9 to the end of the list.
> - Undo back to before you reversed the list.
> - Add a 9 to the end of the list again
> - Undo back to the original state of the list.
>
> After every operation, print the current state of the "application", remember that your application's only functionality is to change the state of the list.

Stacks
Using a linked structure (doubly, or singly linked list) create a stack structure. From this structure (the stack) and the included text file `palindrome.txt`, determine which of the words within the file are proper palindromes. Your solution must utilize the stack structure. However, You may use other tools/functions within the language of your choice to check your work.
Queues
Finally, this last section is not a coding challenge.
A remote printing system serving a large pool of individuals can be very complicated to support. In theory, a simple queue that takes in print requests and dequeue's them once they have been processed would serve all the required operations. However, there are significant problems that arise as far as user requests and other things. What are some of the issues that you can see with a simple queue that only supports First In First Out operations typical of a queue? (Enqueue, dequeue, peek, etc.) For your submission, you can either submit a second video of 1-2 minutes in length, or small report of approximately 1 page that discusses the various shortcomings of a queue to support a remote printer system.

Your submission should be accompanied by a 8 minute walk-through of your code. This analysis should include your decision making process, the logic behind you code, an your original thoughts that went into the decision making on why your code is written and performs in the manner in which you have written it. If you can not adequately explain how your code functions, it is difficult to believe that you created it yourself as it is inherently difficult to make that which you don't understand.
All video submissions must:

- Be narrated by your own voice - Silent submissions will not be considered
  - If you need accommodations regarding your voice recording, reach out to me
    BEFORE
    the due date of the assignment
- Capture your screen to include the source code and other assets required by the assignment if necessary for the comprehension of your explanation.

Your submission should include:

- A link to your YouTube/Loom Video upload submitted as a `.txt` file or as a submission comment
  OR
  a video file (`.mp4` preferred).
  AND
- Your code project (source code, resource files, etc.) unzipped.
- If necessary, provide a README file if an explanation is required to execute your code.