CSCI 230 Fall 2023
Instructor: Michael Levet

Homework Three

Oscar Jiang

9/13/2023

1 Homework Three: Honor Code

On my honor, my submission reflects the following:

- Everything I have submitted is my work and in my own words. Everything that I submit is something I understand.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

Honor Code Agreement Signature. **Oscar Jiang** □

2 Homework Three: Collaborators

The following people I have collaborated with are as listed:

- Matthew Jennings- We did not discuss much about how to go about doing our codes. However, we did exchange ideas on how to do the stringCheck method.

3 Homework Three: Problem One Reflection

One concept this assignment helped me cement was a stack. In class, I was quite confused with the concept of a stack- it seemed like a list that could only add to the end and remove to the end. As I worked on the assignment, I gained more of an understanding of it.. but the data structure that forces you to interact with the top element proved to be extremely limiting. The data structure class consisted only of three very simple methods: push(), which lets you add an element on top, peek(), which lets you see the element on top, and pop(), which lets you see the element on top, whilst removing it too.
In a similar manner, I did not know what to make of the checkString method with such limitations. To me, it would've been easier to check the last and first digits within the one loop... implementing that algorithm using a stack would have been difficult and unconventional. However, I understand that the limitations have a purpose and forces order on programs that would require such order– like a web browser– and efficiency for heavier programs- since it forces limiting access to traversing the stack.

4 Homework Three: Problem One B

I chose my own arrayList to support my stack. When I was between the choices of an arrayList or a linkedList, I thought of linked list to be easier since you have the tail– though, this was not accessible outside. However, an arrayList would have instant access to to whatever value at a select position. Specifically, it would retrieve the tail/last element relatively quickly for the for either the pop() or peek() method. In addition, using the pop() method would utilize the remove()- which when used on any other element than the last one would cause the entire list to be shifted- thus when removing the end, all it would have to do is decrement the length. As for the push() method, the arraylist would just had to add an element to the end, making it less process intensive as opposed to adding into the list. Overall, using arrayLists with smaller stacks would be more efficient as it would barely take up memory and it doesn't have to set the tail constantly. On the contrary, linkedLists would be better for larger stacks. This is because the arrayList used would not actually remove anything, but would decrement the list, effectively hiding the rest of the many elements popped from the user and it would create more space, potentially unnecessary, when the push() method finds itself occuppying the last 'true' space of the internal array.