**CSE 332 Project 1 Write up**

Ruijia Wang

1360765

1. **Who and what did you find helpful for this project?**

Google

1. **How did you test that your stack implementations were correct?**

The backwards of secret and bot are the same fore both array and linked list. Also, the backwards of secret is a meaningful sentence.

1. **The file secret.wav is a backwards recording of a word or short phrase. Use sox (or another converter) and your program to reverse it, and write that as the answer to this question.**

The scent of bitter almond always reminded him of the fate of unrequited love.

1. **Other than java.util.EmptyStackException, did you use any classes from the Java framework or other class library?**

No

1. **Your array stacks start with a small array and double in size if they become full. Assuming the computer had enough memory, how many times would this resizing occur (explain how you got the answer) for a .dat file with:**

**a) One million lines?**

**17**

**10\*2^17 = 1.3\*10^6**

**b) One billion lines?**

**27**

**10\*2^27 = 1.3\*10^9**

**c) One trillion lines?**

**37**

**10\*2^37 = 1.3\*10^12**

1. **How might you implement QueueStack (i.e. simulate a Stack using Queue as internal data structure) with one or more instances of a FIFO Queue? Assuming your Queue class provides following operations: enqueue(), dequeue(), isEmpty(), and size(), implement push() and pop() operations of QueueStack.**

public class QueueStack implements DStack {

private int size;

private Queue<Double> q;

public QueueStack() {

this.size = 0;

q = new LinkedList<Double>();

}

public void push(double d) {

q.enqueue(d);

size++;

}

public double pop() {

for (int i = 1; i < size; i++) {

q.enqueue(q.dequeue());

}

size--;

return q.dequeue();

}

// Assume other methods (peek & isEmpty) are implemented

}

1. **Why would a stack implementation using a queue, as you described in the previous problem, be worse than your array and linked-list stack implementations? Explain in terms of asymptotic bounds.**

|  |  |  |
| --- | --- | --- |
|  | push | pop |
| array | O(1) | O(1) |
| Linked-list | O(1) | O(1) |
| queue | O(1) | O(N) |

O(N) is slower than O(1).

1. **In the process of making your generic stack implementations from your non-generic ones, what sort of errors did you encounter and how did you resolve them?**

I encounter the error of creating a new generic array. I solved it by read the workaround #1.

1. **How much did you have to understand about the code in Reverse.java to make the changes to use your generic stacks?**

Generic stacks are similar to double stacks. I use the generic object gs instead of the double object s when I make changes.

1. **If you did “Above & Beyond”, describe each of your extra credit implementations in detail.**

1. **What did you enjoy about this assignment? What did you hate? What could you have done better?**

I did not know how to install sox on mac and there isn’t any detailed information about it in spec. That wastes a lot of my time.

1. **Anything else you would like to include?**