Assignment 2
Linear Data Structures

Release Date	Due Date
January 30, 2018	**February 20, 2018**

Objectives

- Understanding the mechanisms of linked lists in depth
- Creating stacks and queues based on linked lists and arrays
- Using stacks and queues to solve practical problems

Problem Specification

In this assignment you will implement Stacks and Queues using both linked lists and arrays to be able to convert decimal numbers into binary and vice versa.
Note: In this assignment you are NOT allowed to use Java (or other languages) Collections API such as LinkedList, ArrayList, List, Stack, Queue, etc.

Write an application to solve the following problem:

1) Create a Class **MyNode** and with an integer attribute **myNum**.

2) Create a Class **MyLinkedList** that has at least two methods: **insert()** and **delete()**.

3) Create a Class **MyLLStack** that has at least two methods: **push()** and **pop()**. It must be implemented using **MyLinkedList**.

4) Create a Class **MyLLQueue** that has at least two methods: **enqueue()** and **dequeue()**. It must be implemented using **MyLinkedList**.

5) Also create two classes **MyAStack** and **MyAQueue** which are implemented using arrays.

6) In your Stack and Queue classes, implement a **store** function that takes an integer and stores its binary digits in the data structure.

    a. *Example: 13 in binary is 1101.*

    b. *myStack.store(13) should store the digits "1", "1", "0", "1" in your stack.. Each digit in a separate node of the stack, where myStack is either a LL-based or an Array-based stack*

7) In your Stack and Queue classes, implement a **toDecimal** function that reads the structure's stored binary digits and converts them into a decimal integer.

a. *Example: The queue contains the digits "1", "1", "0", "1".*

b. *myQ.toDecimal() returns 13, where myQ is either a LL-based or an Array-based queue.*

8) Test and measure time complexities of your two new functions and output the results. Generate *n* random positive integers and test the **store** and **toDecimal** functions of your stack and queue with each input Output the decimal number being tested and its binary representation stored in your stack or queue

9) Both your functions, **store** and **toDecimal**, should have input validation.

a. *Example: MyLLQueue.store(13) would store the digits {1, 1, 0, 1} in the queue.*

b. *Example: MyLLStack.store("hi_mom01") would report an error and gracefully exit or continue processing next input. Your message to mom would never be seen :(*

Sample output

```
n = 2

__LL-based Stack__

Decimal: 128
Binary: 10000000
Time taken: xx.xxxx ns

__Array-based Stack__

Decimal: 128
Binary: 10000000
Time taken: xx.xxxx ns

__LL-based Queue__

Decimal: 128
Binary: 10000000
Time taken: xx.xxxx ns

__Array-based Queue__

Decimal: 128
Binary: 10000000
Time taken: xx.xxxx ns

------

__LL-based Stack__

Decimal: 1951797
Binary: 000111011100100000110101
Time taken: xx.xxxx ns
```

```
__Array-based Stack__

Decimal: 1951797
Binary: 00011101110010000110101
Time taken: xx.xxxx ns

__LL-based Queue__

Decimal: 1951797
Binary: 00011101110010000110101
Time taken: xx.xxxx ns

__Array-based Queue__

Decimal: 1951797
Binary: 00011101110010000110101
Time taken: xx.xxxx ns


Average time for LL-based stack: xx.xxxx ns
Average time for Array-based stack: xx.xxxx ns
Average time for LL-based queue: xx.xxxx ns
Average time for Array-based queue: xx.xxxx ns
```

Design Requirements

Output requirements

You must output all your work to a text file called “hw2cs3310_yourName.txt” (obviously replace yourName with your name). Also, you **must** test your program with `n=1`, `n=50`, `n=500`, `n=1000`, `n=50000` and `n=100000` but give detailed output for `n=1`, `n=50` and `n=500` and only average timing results for `n=1000`, `n=50000` and `n=100000`. In your analysis report, you will compare your linked-list versus array-based implementations.

Code Documentation

For this assignment, you must include documentation for your code. This includes how to compile and run your program.

Coding Conventions and Programming Standards

You must adhere to all conventions in the CS 3310 coding standard. This includes the use of white spaces for readability and the use of comments to explain the meaning of various methods and attributes. Be sure to follow the conventions for naming files, classes, variables, method parameters and methods. Read the material linked from our class web-pages (in case you can’t recall programming styles and conventions from your CS1 and CS2 courses).

Testing

Make sure you test your application with several different values capturing different cases, to make sure it works.

Assignment Submission

- Generate a .zip file that contains all your files, including:
  - Signed Plagiarism Declaration
  - Source code files
  - Including any input or output files
  - Documentation of your code – e.g. using Javadoc if using Java
  - A brief report (in a pdf file) on your observations of comparing theoretical vs empirically observed time complexities. Note this report will include
    a) A brief description of problem statement(s).
    b) Algorithm descriptions (if these are standard, commonly known algorithms, then just mention their names along with customization to your specific solution(s), otherwise give the pseudo-code of your algorithms.
    c) Theoretically derived complexities of the algorithms used in your code
    d) Table(s) of the observed time complexities.
    e) Plots comparing theoretical vs. empirical along with your observations (e.g. do theoretical agree with your implementation, why? Why not?).
- Don't forget to follow the naming convention specified for submitting assignments.

PLAGIARISM DECLARATION

1. I know that plagiarism means taking and using the ideas, writings, programs, code, works, or inventions of another as if they were one's own. I know that plagiarism not only includes verbatim copying, but also the extensive use of another person's ideas without proper acknowledgement (which includes the proper use of quotation marks). I know that plagiarism covers this sort of use of material found in textual sources and from the Internet.

2. I acknowledge and understand that plagiarism is wrong.

3. This assignment is my own work (or my group's own unique group assignment for group-assignments). I acknowledge that copying someone else's assignment, or part of it, is wrong, and that submitting identical work to others constitutes a form of plagiarism.

4. I have not allowed, nor will I in the future allow, anyone to copy my work with the intention of passing it off as their own work.

Name…… Student #.....

Signed ……. Date …….