CMSC 204: EXERCISE 1

For your first exercise, you will be integrating what you have learned about arrays and linked lists together with stacks and queues. Thus, the output required for Exercise 1 are two compiling C# codes bundled inside **LastnameExer1a.zip** and **LastnameExer1b.zip**.

IMPORTANT NOTES:

1. You are to follow the naming convention specified. You will be given a deduction if you fail to do so.
2. Make sure that you document your code (include your name, student number, the date and a brief description of the program in comments at the start of the code). Also put in the necessary comments that will help you (and me) understand your code.
3. To ensure the validity and integrity of your work, your programming exercise should come with a screencast of how you coded your work. You can use a screencasting freeware called Jing for Mac and Windows (http://www.techsmith.com/jing.html) or any alternatives. The screencast of your work should be at most 5 minutes (deductions if it exceeds the time), which will contain a coding demonstration of your solution. You don't have to capture the whole programming process. You can just highlight the important parts and explain how you did your assignment. The final section of the video should feature the finished product of your work and a short demo on how it works. You should be the one doing the voice-over for the explanation. The explanation can be in Filipino or English. Name the video as **LastnameExer1.** Note that the upload limit for the portal is 64MB. If your video is more than that, you can upload it to Youtube as "unlisted" and then you can include the link in your submission. Note that only one video is needed as an output for this exercise.

LastnameExer1a – STACKS

Using the **array implementation*** of stacks, create a **console program** that does the following:

1. Display the following menu:

[1] Add Notebook
[2] Check Notebook
[3] Peek at Notebook
[4] Check All
[5] Exit

2. When the user chooses [1], the user will be prompted to enter the name of the notebook's owner.

3. When the user chooses [2], a message "[name]'s notebook is being checked". The notebook is then popped from the stack.

4. Choosing [3] would display the name of the person who owns the notebook at the top of the stack.

5. Check all [4] would check all the notebooks and make the stack null.

6. Keep displaying the menu until the user chooses [5]

*IMPORTANT: Create the stack using your own implementation of arrays. **Do not make use of the built in Stack for this. Also do not make use of the built in Arrays class.**

FOR THE BRAVE: You can choose to implement the exercise using a Windows interface and accept input via forms or whatnot instead of a* ***console window*. As long as you can implement the functionalities specified, Windows based applications will be accepted. Bonus points will be given if you are able to deliver more than what is required.*

LastnameExer1b – QUEUES

Using the **linked list implementation*** of queues, create a console program that does the following:

1. Display the following menu:

[1] Sign up for doctor's consultation
[2] Enter room
[3] Begin consultation
[4] Closing time
[5] Exit

2. When the user chooses [1], the user will be asked to give the name of the person consulting. The user is also the nature of the concern. The person is then enqueued in the list.

3. When the user chooses [2], the name of the person in front of the queue is displayed on screen. The person, however, is not removed from the list.

4. When the user chooses [3], name of the person is displayed on the screen along with the nature of concern. After that, the person is removed from the list.

5. Choosing [4] would free up the whole queue.

6. Keep displaying the menu until the user chooses [5].

*IMPORTANT: Create your own linked list. **Do not make use of the built in lists and queues in C#.**

FOR THE BRAVE: You can choose to implement the exercise using a Windows interface and accept input via forms or whatnot instead of a* ***console window*. As long as you can implement the functionalities specified, Windows based applications will be accepted. Bonus points will be given if you are able to deliver more than what is required.*