Course:	**INFO-3135 C++ Algorithms and Data Structures**
Project:	**Project #1 – Emergency Room Triage**
Due Date:	**See FoL Dropbox**
Submitting:	**Please see the Submission Guidelines below**
Professor:	**Daniel Maclam**
Student Name:	_____***Seolan Jin***_____ ***ID:*** ___***1037144***___

Project 1:

This is an individual project.

Objective: Write a console program that will simulate a patient emergency room triage system with a priority patient queue.

Where Do I Start?

- Download the Project #1 Starter Code from FoL
- All your code will be completed in this Solution

There are 3 projects in the solution:

- PriorityQueue
  - Update and modify the Patient, Ailment and PriorityQueue classes in this project.
- PriorityQueueTests
  - Contains tests for some of the functionality of the Patient, Ailment and PriorityQueue classes.
- Your_Main
  - This is where you will place your code for the complete application.

It is recommended to start with running the PriorityQueueTests project and enabling tests in order as you add functionality. To enable tests just open the test_config.h file and set the test you wish to run to true;

Once your priority queue passes the tests in the PriorityQueueTests project select Your_Main as the startup project and complete the implementation for the application.

Application Requirements:

- Create a menu that will allow a Triage Nurse to:
  - Add a patient to the queue in its appropriate position based on their triage status (priority) 중증정도
  - Service the next patient in the queue
  - Print the contents of the queue
  - Print the history of the queue (oldest first)
  - Load the queue from a csv file (see patients.csv for formatting.)
- **BONUS:** Save the queue to a csv file (in the same format as loading.)

```
**************************************************
* Welcome to Fanshawe College Medical Center F2020 *
* <Your Name Here>                               *
**************************************************


Please Make A Selection:

        1 - Add Patient
        2 - Process Next Patient In Queue
        3 - Display Queue
        4 - View Processed Patients History
        5 - Load Queue
        6 - Save Queue
        0 - Exit

>
```

- Add Patient
  - You will collect the following parameters for patients:
    - Name
    - Ailments
      - The patient can have more than one. Use an instance of the LinkedList class to hold all the ailments.
      - Contain Severity, time criticality, contagiousness for each condition.

```
Enter patient name: John Doe
        Enter ailment (leave blank when done): Heart Attack
        Enter severity: 8
        Enter time criticality: 10
        Enter contagiousness: 0

        Enter ailment (leave blank when done): Stroke
        Enter severity: 7
        Enter time criticality: 10
        Enter contagiousness: 0

        Enter ailment (leave blank when done):
```

- Process Next Patient
  - Removes the patient from the queue with the highest priority score
  - Displays the patient with the next highest priority score

```
John Doe moved to patient room!
Next in queue: Bob Smith with priority score 29
```

- Display Queue
  - Displays the patients in the queue including the following information:
    - Position in queue
    - Patient Name
    - Priority Score
    - Ailments

```
Patients In Queue:

0 : John Doe - 150 - Heart Attack, Stroke,
1 : Bob Smith - 29 - Flu,
```

- View Processed Patient History
  - Displays a list of patients that have been processed
    - Most recently treated patient first
    - This should list the following:
      - Patient Name
      - Priority Score
      - Ailments

```
History:
Bob Smith - 29 - Flu,
John Doe - 150 - Heart Attack, Stroke,
```

- Load Queue
  - Prompts the user to enter a path to a csv file containing a list of patients. (see patients.csv)
    - Loads the patients from the file into the queue.

```
Enter path to file: ./patients.csv
```

- Save Queue (**BONUS**)
  - Prompts the user to enter a path to a file to write the patient data from the queue to. (see patients.csv)

Technical Requirements:

- No STL (Standard Template Library)
  - Except where provided by me in PriorityQueueTests
- All inputs must be validated
- Pointer notation must be used throughout
- Patient Class Properties
  - Patient Name
  - Must maintain a list of Ailments using the provided LinkedList class
- Ailment Properties
  - Name
  - Severity
  - Time Criticality
  - Contagiousness
- PriorityQueue
  - Must use an instance of the LinkedList class to hold queued patients
  - Patients may only be in the queue once (no duplicates)
  - Patients position is determined by the sum of all ailments priority score. (> score is higher priority)
  - Score calculated by the following:
    - (Severity /Time Criticality) + Contagiousness
  - Score calculated when patient joins the queue

Submission Guidelines:

Electronic Submission (<u>mandatory</u>):

Submit your file to the "*Project 1 - Triage"* dropbox in *FanshaweOnline*.

Make sure to add your name and student number to all documents.

Submit your project on time!

Submissions must be made on time! Late projects will be subject to divisional policy on missed test and late projects.

10% per day for a maximum of 5 days after which the submission will receive a zero grade.

Submit your own work and *keep it to yourself*!

It is considered cheating to submit work done by another student or from another source. Helping another student cheat by sharing your work with them is also not tolerated. Students are encouraged to share ideas and to work together on practice

exercises, but any code or documentation prepared for a project must be done by the individual student. Penalties for cheating or helping another student cheat may include being assigned zero on the project with even more severe penalties if you are caught cheating more than once. Just submit your own work and benefit from having made the effort on your own.

All work will be subject to “TurnItIn” scrutiny.

Grading Scheme

Marks Available	Any amount of plagiarism will be awarded a zero Non compiling code will receive a zero	Marks Awarded
5	Code is commented	
5	Code formatting	
5	Naming conventions	
5	Following general coding guidelines	
5	Use of proper pointer notation	
10	Priority Queue Class	
10	Patient Class	
10	Ailment Class	
5	Proper priority score calculation (see formula above)	
10	CSV file import	
10	Correct operation of program, validated input, correct queueing etc.	
5	Correct output	
___ 85	TOTAL MARKS	
10 (BONUS*)	CSV file export *Bonus marks (but not to exceed the max of project grade)	
	Penalties	
-5	Penalty if the path of the csv is hard coded. It should just check your current working directory for the csv file.	
-100%	Plagiarism	
-100%	Code does not compile	
-100%	Starting code not used	
85	Total	