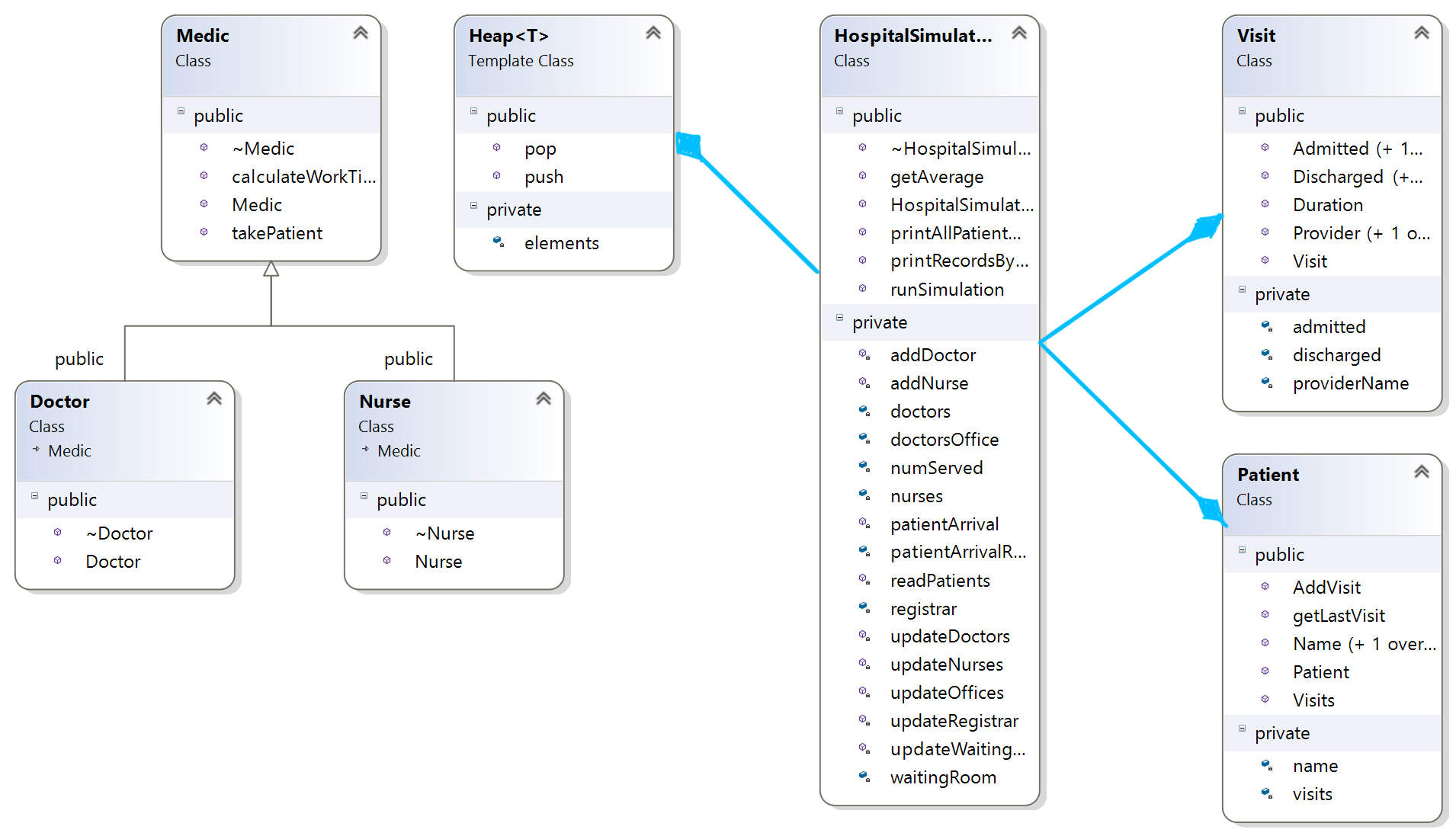
Design Specification

The Emergency Room Simulation software will allow the user to enter the average hourly patient arrival rate, the number of doctors working in the emergency room, and the number of nurses working in the operating room. As a patient arrives in the emergency room, they are assigned a priority value from one to twenty. Doctors can treat any priority value and require one to twenty minutes, while nurses can treat any priority less than eleven and require one to ten minutes. When a healthcare professional finishes with a patient, they take the highest priority eligible patient in the emergency room and begin treatment. The ER maintains records for each patient. Each patient record has a list of visits, documenting arrival time, departure time, and priority.

Use Cases

|  |  |  |  |
| --- | --- | --- | --- |
| Case Number | Use Case | Input | Response |
| 1. | User runs simulation |  | Prompt User for Patient Hourly Arrival Rate |
|  |  | User enters Patient Hourly Arrival Rate | If response is invalid, reprompt. Otherwise, prompt for number of doctors |
|  |  | User enters number of doctors | If response is invalid, reprompt. Otherwise, prompt for number of nurses. |
|  |  | User enters number of nurses | If response is invalid, reprompt. Otherwise, run simulation. Output summary of 24 hour simulation run. Display post-run menu. |
| 2. | User requests patient data. | User selects Single Patient Data from post-run menu. | Prompt User for patient name |
|  |  | User inputs patient name. | If name is found, output record of patient visits. Otherwise, ask whether the user would like to try another name. |
|  |  | User inputs yes or no. | If user inputs yes, reprompt for patient name. Otherwise display post-run menu. |
| 3. | User requests list of patient names. | User selects Get All Patient Names | Output the names of all patients discharged during this simulation |
|  |  |  | Output post-run menu |

UML



Pseudocode

Class Hospital Simulation – The class that will generate all the needed objects and will return all the needed values to the UI.

HospitalSimulation(int arrivalRate, int totalDoctors, int totalNurses)

Generates a HospitalSimulation object. It will instantiate the doctors vector and nurses vector.

Vector <String> readPatients()

Reads each line of names.txt into a vector and return it, if there is a problem it will cout a message and an empty vector.

Patient \* patientArrival(std::vector<std::string>& patients, int index)

Creates a new patient based on the name indicated by the index and then add the patient to the waiting room, will return the newly created patient object.

void HospitalSimulation::updateWaitingRoom(std::vector<std::string>& patients, int clock)

Creates a random number and use it to determine how many patients arrived in that instant and will add them to the heap.

void updateDoctors(int clock)

Iterates through all doctors, if the doctor is ready for a new patient then give him the most critical patient and calculate the visit, set the doctors timer to the correct number, if the doctor isn’t ready decrease his counter by 1.

void updateNurses(int clock)

Does the same as update doctors but for nurses.

void updateRegistrar(std::string patientName, Visit\* visit)

Adds the visit to the correct index in the registrar.