# The Problem

The Leopold Medical Center in Cordinia, a small country made famous by Hallmark Christmas specials, has a problem in its emergency room. Patients are dying because they are not being seen on time. The hospital administration has suggested that they need extra beds. Since the country is fairly poor, the desire is to add as few beds as possible. They contracted with you to write a program to determine the minimum number of beds that they would need.

You traveled there and talked with their medical staff. As it turns out, they place patients on the emergency room tables on a first come, first serve basis (think FIFO). Cordinia is a very egalitarian society (except for royalty, of course) so they do not want to show any preferential treatment. From your data structures class, you (immediately?) think that perhaps there is a better approach. You start telling the staff about priority queues and how useful they are in scheduling algorithms. As their eyes start to glaze over, you switch tactics. You start talking about triage (they wake up) and how you could write a small program that might show the benefits of modifying how they schedule and perhaps not require extra beds. (At that time even the prince, who was sitting in on the meeting, wakes up; not hiring extra beds or doctors seemed like a great idea.)

They contract you to write a little simulation that could demonstrate how using the information produced by the triage nurse would be helpful in determining which patients to place on the beds as they become available.

# The program

The shell of the program has already been written. There are a number of classes:

**Patient** - represents a patient who needs to receive emergency treatment. This has five properties and a single method. The properties are

* **TimeToLive** - minutes until it is too late for the patient to survive if not on an ER table
* **TimeForProcedure** - minutes to perform the procedure for this patient
* **LastPossibleMoment** - time (relative to hospital time) when this patient MUST be on an ER table
* **IntakeTime** - time entering waiting room
* **TimeEnteringER** - time when patient is placed on an ER table
* ***setFinalOperationTime*** - adjust the LastPossibleMoment based on the current time

**ERTable**

Contains a **Patient** and an **ETC** (Estimated Time of Completion)

**TriageUnit**

Here, there is only a single method, ***getNewPatients***. The triage unit will return a random number of patients will random patient expiration times and operation times. These patients are stored in a list which is returned from ***getNewPatients.***

**EmergencyRoom**

This is the main class. This class holds the emergency room beds (an array). It also has a constant representing the number of tables and how long the simulation is to be run.

**Queue classes**

A **SimpleQueue** and a **PriorityQueue** which both implement an **IQueue** interface. This was simply so that we could easily swap out a standard FIFO queue with our priority queue.

**ProcessPatients** - this is the only method in the **EmergencyRoom** and where the bulk of the work happens in the program. The triage unit will return a random number of patients will random patient expiration times and operation times. Your job is to modify this method such that it will receive patients from the triage unit, place them into a waiting queue and then when hospital beds are empty, pull patients out of the queue and place them on a hospital bed. When the operation on that bed is complete, the bed becomes free for some other patient. It is your job to see that that bed is emptied.

Your program should do the following until the simulation ends:

* empty any beds that are currently occupied
* get a list of new patients from the triage unit
* for each new patient
  + set intake time
  + set final operation time
  + add patient to the waiting queue based on their last possible moment
* while there are empty beds
  + remove the next patient from the waiting queue
  + if the patient didn’t make it
    - increment the ‘expiry’ count
  + otherwise
    - set the time entering the ER
    - add the patient to the table

The program runs for a certain duration (a constant). At the end of the run, the program should show, at a minimum, the following data:

* Number of patients served.
* Number of Patients who didn’t make it.
* Maximum patients in the waiting room at the end of a cycle.
* Average wait time per patient.
* Average stay at the ER (wait time plus time on the ER table).

Make certain you begin with the provided code. You can accomplish this assignment by editing only the **ProcessPatients** method, though you may decide there are other methods that you find useful to implement. Run your code with the usePriority flag set to false and then to true. This toggles between a priority queue and just a standard queue. Note with the current values, the hospital will process roughly 400 patients during a shift, so don’t be surprised. These are the values I received for the current setting and no priority. I counted patients as they were placed into the waiting room.

* Total Elapsed Time: 720
* Total Patients: 410
* Total Expired: 20
* Longest Wait: 50
* Average Wait: 15
* Average Stay: 29

Questions:

What differences did you notice when you switched from a simple queue to a priority queue? It is probably obvious that more would survive. Did the differences in the other times surprise you? What causes these differences?

A priority queue is not the only way this might have been accomplished. What other data structure(s) might you have used to operate on the most urgent patients first? Are there any disadvantages to that choice? Advantages?