# CS-273-1 Final Project (Fall, 2019)

**Group Members: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Goal 1: Correctly implement moderately complex software from specification.**

**Goal 2: Practice applying the concepts and techniques you have learned this semester.**

|  |  |  |
| --- | --- | --- |
| **Project Implementation** | **Points** |  |
| **FinalSpec.doc (see rubric)** |  | 35 |
| **FinalSummary.doc (see rubric)** |  | 20 |
| **Implementation:** |  |  |
| Use of class **inheritance** and **polymorphism** |  | 10 |
| Use of **set**, **map**, and/or **hash table** (1 or more) |  | 10 |
| Use of **queues** (a **priority queue** counts), or **some other structure** that you plausibly argue sounds useful. |  | 10 |
| **Project Demo (see rubric)** |  | 15 |
| **TOTAL** |  | 100 |

## Final Project Evaluation Rubric

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Emerging** | | **Developing** | | **Mastering** | |
| 0 | 4 | 5 | 8 | 9 | 10 |
| Project design uses class inheritance | More than 2 missing or poorly specified components: requirements specification, use cases, UML diagram, and pseudo-code. | | Adequate inheritance structure but does not use polymorphism. Alternatively, inadequate **use cases** or UML diagram in specification. | | Excellent **inheritance** structure, using **polymorphism**, with detailed **use cases** and **UML** diagrams. **Design should match implementation.** | |
| Project Summary Report | Poor answers to the questions. No summary of problems or lessons learned in the implementation. | | Good summary of problems encountered, final design, and lessons learned in the implementation. | | Excellent project summary, with good discussion of results, including the required graph. | |
| Short Project Demo (max of 5 minutes) | Poor presentation with many missing elements. | | Addresses some of the questions assigned for the project. Presenters speak clearly and audibly. Presenters look at the audience. Presenters explained clearly their design and the challenges encountered. | | Good presentation skills and all questions assigned for the project are addressed in the demo. Presenters explained clearly their design and the challenges encountered. | |

## Project Overview

* **You must work in a group of 2 or 3 persons.**
* **Pick one of the problems described in this document, and design/implement a solution for it.**
* **Final Presentation on day scheduled for the final.**

## Solution Strategy

* Read the problem description carefully. In **your own words** write out the **requirements specification** for this problem.
* **Understand the rubric, and know what your design and implementation will need to have.**
* At this point it will be helpful to re-read **section 6.5**. Also, I will make available some seed code that will help you see an event-driven simulator in action..
* Meet with your partner and work through the **use case** for this problem. Note that since this is a simulation, you will want to understand what your simulator needs to do at each **clock tick**. **Treat a clock tick like a user input and write down its use case.**
* Come up with the design by drawing a high-level UML diagram. You will not need to know all the properties or methods for your classes at the beginning. Just understand the high-level structure first, **including what will be your base and sub classes.**
* You can start implementing a **prototype** to flush out the details of your design. Fill in the details of your UML diagram.
* When you are confident about your design, **throw your prototype away**, and **begin implementing** your solution.

## What You Should Not Do

* **Do not just copy (or extend) the airport simulator seed solution.** You must pick a simulator from the shortlist I give you below and come up with your own interpretation of it.
* Do not spend less than 50% of the time working together as a group
* Do not forget to constantly communicate to your partner what you are thinking in terms of the software design. Communicate, communicate, communicate … Tell your partner what you are thinking!
* Do not forget to test everything you implement. Remember to unit-test as much as possible.

## Project Deliverables

* **Two or more pages design/specifications document called “FinalSpec.doc” (At Final):** It needs to include a requirements specification, use cases, UML diagrams, and any pseudo-code for your simulation. The more detailed the design, the higher your score will be. **One report per group. Keep a copy in all team members’ folders.**
  + Your design will inevitably change when you begin your implementation. Make sure you update your design documents constantly to reflect the current state of your design.
* **Event-based simulation in folder** **PR\_Final\_Simulation** **(At Final)**:
  + An implementation of the simulation of one of the problems described on the next page.
  + Simulation must include **inheritance** **and polymorphism**
  + Simulation must use **set**, **map**, and/or **hash table**. At least ONE of these data structures must be used.
  + Simulation must use one or more **queues (a priority queue counts as well)**. Alternatively, if you feel another data structure would be more useful given your problem domain, make an argument as to why and use it.
* **One page project summary report** **“FinalSummary.doc”** **(At Final)**: That includes answers to the questions given below in the project descriptions, and includes the required graph. Also indicate what was changed from your initial design in your final implementation. If implementation did not work, describe your design, implementation challenges, and lessons learned. **One report per group member.**

## Problem 1: An Emergency Room Simulator

C:\Users\walker\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\E52NQBY4\MC900198489[1].wmf

* You need to simulate a hospital emergency room located in the town of *Simville*, population 2000.
* Matt will post a list of names of all residents of said town on the GoogleDrive.
* Every person in 273ville is equally healthy, but they do occasionally need to go to the emergency room.
  + I.e. there is equal probability that anyone in town will be admitted into the emergency room as a patient.
* Your emergency room is small, but it has a big heart. It tries to run its operation as efficiently as possible.
* When a patient arrives at the emergency room, he/she is triaged – that is, the patient is assigned a priority number from 1 to 20, depending on the severity of the illness. **Higher priority values indicate more serious illnesses and are always treated ahead of lower priority illnesses.**
  + Illnesses with priority 1 to 10 occur approximately 70% of the time with equal probability
  + Illnesses with priority 11 to 15 occur approximately 20% of the time with equal probability.
  + Illnesses with priority 16 to 20 occur approximately 10% of the time with equal probability.
* The emergency room has 2 categories of caregivers: **Doctors** and **Nurses**.
  + **Nurses** can treat patients with **priority** **1 to 10**, and **doctors** can treat patients with **priority** **1 to 20**.
  + Empirically, we also know that **nurses** take **1 to 10 minutes** to treat a patient, and **doctors** take **1 to 20 minutes** to treat a patient, on average.
* The hospital **keeps a record of all patients** that were treated in the emergency room. Each record stores:
  + the number of visits to the emergency room, and
  + the severity of illness on each visit
* Your simulation needs to examine a week in the life of the emergency room on a minute-by-minute basis (i.e. it needs to simulate at least 7x24x60 minutes).
* Allow the user to input the following data values:
  + The **average hourly patient arrival rate** (patients/ hour) at the emergency room –assume that there will not be more than 60 patients per hour.
  + The **number of doctors** working in the emergency room.
  + The **number of nurses** working in the emergency room.
* You simulation will need to calculate the **average visit time** (arrival to discharge time) for emergency room patients.
* At the end of the simulation, you will need to **display a menu** with options to list the names of all residents that were treated, and retrieve the record of a resident by “name”.
* For your **final report** and **presentation**:
  + **Compare and comment** on the average patient visit time (for some fixed patient arrival rate) when the emergency room has
    - 1 doctor and 1 nurse
    - 1 doctor and 2 nurse
    - 2 doctors and 1 nurse
  + **Display a plot** of the visit time for increasing patient arrival rates, for a combination of doctors and nurses of your choice

## Problem 2: Downtown Traffic Simulator



**Bank**

**School**

**Jackson St.**

**(3 miles)**

**Amber St.**

**(6 miles)**

**Tulip St.**

**(3 miles)**

**Birch St.**

**(5 miles)**

**James St.**

**(4 miles)**

**Travis St.**

**(3 miles)**

* You need to simulate the **inflow** traffic pattern for the downtown core of Simville, population 2000.
* Matt will post the names of all residents of Simville on the GoogleDrive.
* After studying the downtown visitation habits, we know that at any particular day, there is a 30% chance for a resident to visit the Bank, and a 70% chance for a resident to visit the School.
* When a resident visits the Bank, he/she will spend anyway from 10 to 20 minutes, and when a resident visits the School, he/she will spend anyway from 5 to 10 minutes.
* Residents visiting the downtown core can arrive and depart using only two streets: Jackson and Travis. *Residents are equally likely to arrive and depart from any of these streets.*
* The speed limit through downtown is strictly enforced, so citizen will typical drive within the range of 25 to 35 MPH (miles per hour). *You can randomly select a driving speed within this range for all visiting resident.*
  + *Note: The driving speed determines how long a resident takes to travel along each street. E.g. a car traveling at 25 MPH on Birch St. will take 12 minutes (5/25 \* 60) to travel down that street.*
* Additionally, all streets have a **road capacity**. E.g. if a street has a road capacity of 2, it will only allow 2 cars to be traveling on the street at any time. If more than 2 cars arrive at the street, the additional cars will have to wait until at least one car completes travelling down the road before a new car can begin travelling down the street.
* Your simulation needs to examine a week in the life of downtown 273ville on a minute-by-minute basis (i.e. it needs to simulate at least 7x24x60 minutes).
* Allow the user to input the following data values:
  + The **average hourly visitor arrival rate** (visitors/ hour) –assume that there will not be more than 60 visitors per hour.
  + The **road capacity** forall the streets (if you like, you can have your simulator read this from a file).
* Your simulation needs to keep a record of everyone who visited downtown. Each record stores:
  + the number of visits downtown
  + the destination on each visit
* Furthermore, you can assume the following decisions when a resident reaches a 3-way intersection:

|  |  |  |
| --- | --- | --- |
| Intersection | Destination | Street to take |
| Jackson-Amber-Tulip | School | Tulip St. |
| Jackson-Amber-Tulip | Bank | Amber St. |
| Travis-Birch-James | School | Birch St. |
| Travis-Birch-James | Bank | James St. |

* You simulation will need to calculate the **average travel time** for the downtown visitors (i.e. time from arriving until departing the downtown core after visiting the School or Bank).
* At the end of the simulation, you will need to **display a menu** with options to list the names of all residents that visited downtown, and retrieve the record of a resident by “name”.
* For your **final report** and **presentation**:
  + **Compare and comment** on the average travel time (for some fixed visitor arrival rate) for a combination of road capacities (your choice).
  + **Display a plot** of the travel time for increasing visitor arrival rates, for a combination of road capacities

## Problem 3: 28 Days Later Zombie Apocalypse!

* Your simulation again takes place in *Simville*, soon-to-be miserable and damned initial population of 2000.
* Additionally, *Simville* is broken into districts:
  + Downtown
  + Uptown
  + Soho
  + The Docks
  + Medical Hill
  + The University
* Each district is adjacent to some other set of districts, but not necessarily all. (More interesting simulations are not fully interconnected.) You will need to decide at the outset how your districts are laid out.
* Denizens of *Simville* live in a district, but can travel to other districts for work (and then return home).You will need to decide the population of your districts and how to distribute denizens to them at initialization.
* Because denizens can move between districts, each time click you should decide which denizens travel to which other districts.
* Denizens can be in three states:
  + Ignorant (they don’t know about the zombie outbreak, so behave like they normally would)
  + Alarmed (they know about the zombie outbreak and are behaving accordingly)
  + Zombie (only their insatiable hunger for brains drives their behavior)
* You may assume that a denizen in any given district is able to communicate with any other denizen in that district and not with denizens of any other district. Ergo, a zombie in Soho can bite anyone in Soho, but no one in the Docks. Likewise, an alarmed denizen can alarm another denizen in the same district, etc.
* Also, time clicks are not all “equal”. They occur at different times of day. You should use the time of day to determine the likelihood of denizens moving, also taking into account the denizen’s state.
  + Ignorant denizens go to work either in their own or some other district in the morning and return home at night.
  + Alarmed denizens move frantically trying to avoid zombies, i.e. they no longer try to return home and they try to spread knowledge of the horrible doom facing the city, one alarm per alarmed denizen per time click.
  + Zombies aimlessly shamble about looking for non-zombies to eat, one bite per zombie per time click.
* Besides movement of denizens, the following events should also be modelled at time clicks:
  + A non-zombie (ignorant or alarmed) is successfully bitten and turned into a zombie.
  + A zombie attempts but fails to bite a denizen (ignorant or alarmed). If the denizen was ignorant they are so no longer.
  + An ignorant denizen becomes alarmed, either because a zombie tried and failed to bite them or they met an alarmed denizen.
* Once alarmed, a denizen never returns to being ignorant. Likewise, once a zombie forever a zombie!
* One major parameter is how many denizens start out as zombies. Another is their distribution, i.e. where they begin. (I suggest Medical Hill or the University for narrative realism.)
* Another major parameter is the relative likelihood of an alarmed vs. an ignorant denizen being successfully bitten.
* You should graph the progress of the outbreak and related descent into chaos and disorder, i.e. at every time click count how many ignorant, alarmed, and zombies there are.