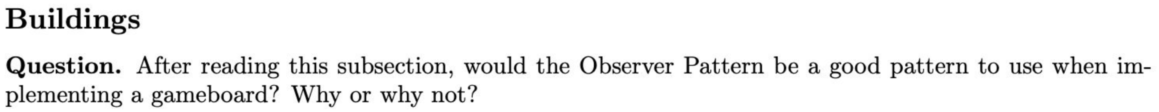
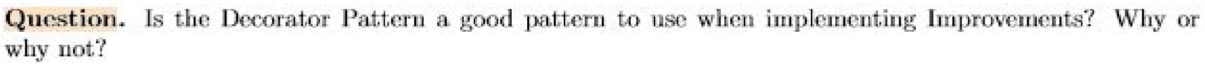
Questions to answer



Buildings

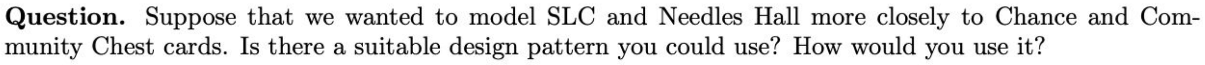
Question. After reading this subsection, would the Observer Pattern be a good pattern to use when implementing a gameboard? Why or why not?

If you decide to use MVC or otherwise decouple the interface from the board, then having the observer pattern would be a good choice between the interface acting as a subscriber and the board acting as the thing being observed. With 40 tiles this might not be entirely necessary, and in fact a visitor pattern would work as well (what you would do in that case is visit every tile). Overall the observer pattern would work well here.



Question. Is the Decorator Pattern a good pattern to use when implementing Improvements? Why or why not?

Decorator Pattern is not a good pattern to use when implementing Improvements, because it is intended to let you add functionality or features to an object at run-time rather than to the class as a whole. However, in this case, all the improvements costs increase differently, thus a single operation will not do it. Furthermore, there are only five improvements in total, so it is easier to just do it separately, whenever a player lands in their own tile, and they can and choose to improve it, we call an upgrade function within each tile and do it.



Question. Suppose that we wanted to model SLC and Needles Hall more closely to Chance and Community Chest cards. Is there a suitable design pattern you could use? How would you use it?

Template method will come in handy when trying to implement needles halls and SLC. A template method defines the steps of an algorithm in an operation, but lets the subclasses redefine certain steps though not the overall algorithm's structure. At the parent class of the Template, we will do the random odds drawing, and then we can have two sub classes, one for SLC and another for Needles Hall. We will have one to change the location, and take or give money based on the draw respectively.

A Breakdown of the Project

We plan to meet up on Thursday August the 6th, and do the work together. We have separated the works as above, Alex is doing these classes: Subject, Observer, Board. View, I/O. Karim is doing: Player, Properties, Res, Gym, and Academic. John is doing Tile, Non-properties, SLC, Needles Hall, Collect OSAP, Tuition, Goose Nesting, Coop Fee, Go to Tims, DC Tims Line. We aim to finish most of the coding by that day, if not we will do it on August the 7th. We have to submit the first copy by August the 7th in Github. Lastly, we will write the final report on August 14th. Please see below for the detailed Plan and work assignment.

Milestone 1: Having a functioning board where players can be added and move across the board. Tiles are not expected to produce any behavior yet but players should be able to roll the dice and move
Milestone 2: Implement the tuition mechanic where players can buy and own tiles, and receive payments when other players land on those tiles they own
Milestone 3: Implement the monopoly section for the academic tiles. The program should be able to know what tiles belong to what group, and if a monopoly exists. The improvements do not need to be yet implemented.
Milestone 4: By now the program should be able to recognize groupings of tiles, and should allow players to build improvements on those tiles.
Milestone 5: Implement the residences and gyms, with all their spec done accordingly
Milestone 6: Implement the non ownable properties regarding any cash moved to or away from a player.
Milestone 7: Implement the tims cup mechanic, including finally forcing players in the DC tims line when appropriate.
Milestone 8: Implement both SLC and Needles Hall tiles.
Milestone 9: Implement the mortgage mechanic
Milestone 10: Implement the bankruptcy mechanic
Milestone 11: Implement the trading mechanic between players
Milestone 12: Implement the auction mechanic
Milestone 13: implement the save and load features of the specification
Milestone 14: Add the testing mode to the program.
Milestone 15: Finish the final classes, and the MakeFile, and write the final report
Milestone 16: Debug program and add exception safety mechanisms to prevent program from quitting abruptly

Group one of classes: Alex August 6th
Subject
Observer
Board
View
I/O
Group two of classes: Karim August 6th
Player
Ownables:
Res
Gym
Academic
Group three of classes: John August 6th

Tile

Non-properties:

- SLC
- Needles Hall
- Collect OSAP
- Tuition
- Goose Nesting
- Coop Fee
- Go to Tims
- DC Tims Line

Final Report: John, Alex, Karim