In this project, we ask you to implement a single display that flips between player 1

and player 2 as turns are switched. How would you change your code to instead have two displays,

one of which is player 1's view, and the other of which is player 2's?

**ANSWER:**

We will have the same implementation in our code, except the fact that in our **Subject** class, there will be two instances of observers, 1 for each player. Then, whenever, player1 has a change in the game, only the parts allowed to be viewed by player 2, will be viewed by player, and vice versa.

For the scenario, where we are not sure how many players will be playing (*i.e.* may be 3 or 4 players, we do not know), we will change the code in **Subject** class. We can simply have a vector of observers, where the necessary changes are updated accordingly.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

How can you design a general framework that makes adding abilities easy?

**ANSWER:**

Abilities can be classified into two categories:

1. Abilities which can affect the cell
2. Abilities which can affect the link of mine and/or opponent

If we want to add a new ability, we have to figure out, **who** is getting affected with our implementation.

If, the cells are getting affected, then we put the implementation in the **cell** class, and if it is the links who are getting affected, we implement it in the **link** class.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

One could conceivably extend the game of RAIInet to be a four player game by making

the board a plus shape (formed by the union of two 10x8 rectangles) and allowing links to escape

off the edge belonging to the opponent directly adjacent to them. Upon being eliminated by

downloading four viruses, each links and firewall controlled by that player would be removed, and

their server ports would become normal squares. What changes could you make in your code

to handle this change to the game? Would it be possible to have this mode in addition to the

two-player mode with minimal additional work?

**ANSWER:**

While creating the game, we would have to create a “plus” looking board. Therefore, first implementation would be to put a field in the cell, to identify whether the cell falls under the intersection of the two triangles, which looks like a plus.

Whenever a player is eliminated the destructor function would be called and that would remove all the links from the cell, as it would normally do for the 1v1 game. However, when it comes to firewall, we would have to consider in such a way that destructor also destroys the firewall in the cells.

To solve this: Whenever a player cast a firewall in a cell, the player keeps record of the firewall it created, as a field (as a vector of *firewalls*). Now, when the player gets eliminated, we take care of the firewalls and destroy the firewalls accordingly. Similarly, the player who gets eliminated, will have their end of the edge server pots turned into normal cell, since server ports are just another cell with different functionality.

For the scenario, where we are not sure how many players will be playing (*i.e.* may be 3 or 4 players, we do not know), we will change the code in **Subject** class. We can simply have a vector of observers, where the necessary changes are updated accordingly. Now, 1 player sees everything according to the rule of the game, and not anything extra which may tip the player to cheat.