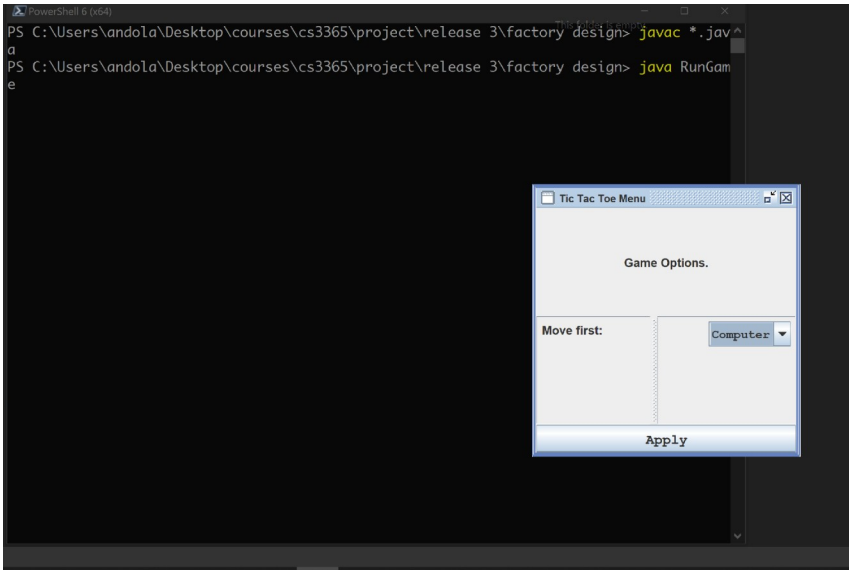
Description of design patterns

Andris Docaj

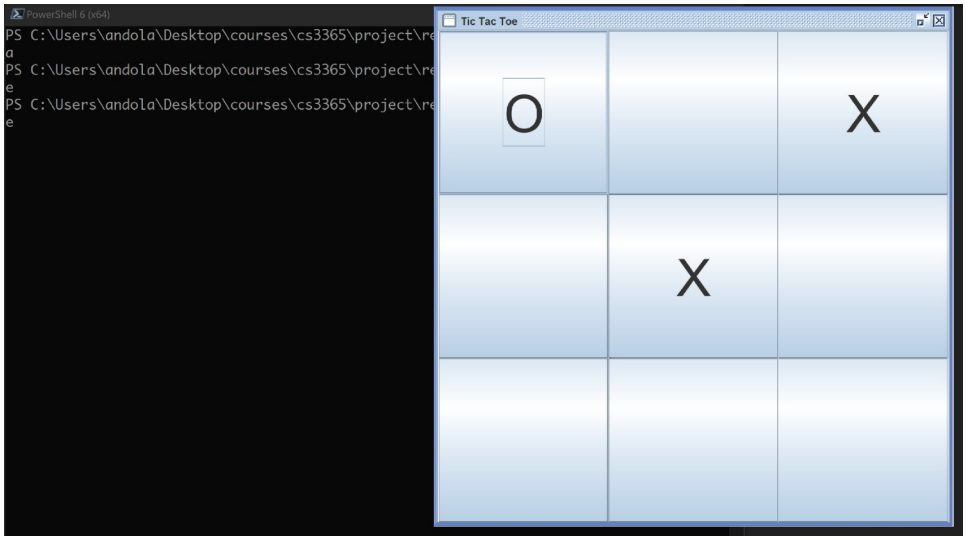
Factory Pattern

The first pattern implemented is the Factory pattern. We create an interface labeled Game, and in addition we create class labeled GameFactory. The GameFactory class is used to create all the objects that are needed to run the program. To run the program we use RunGame, as in



GameFactory is used in the constructor of the UserInteraction class to create a Board object. Moreover, GameFactory is used in the RunGame's main function to create a UserInteraction object. In addition, we use GameFactory to create a state object in the constructor of the MonteCarloSearch class. And finally, the Board class uses GameFactory to create UserInteraction, MonteCarloSearch and State objects in its constructor.

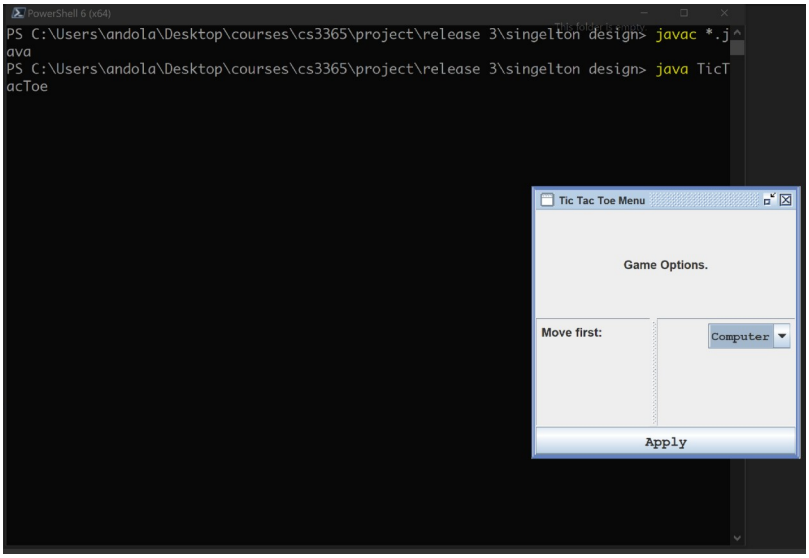
After selecting apply the user is presented with the following screen,



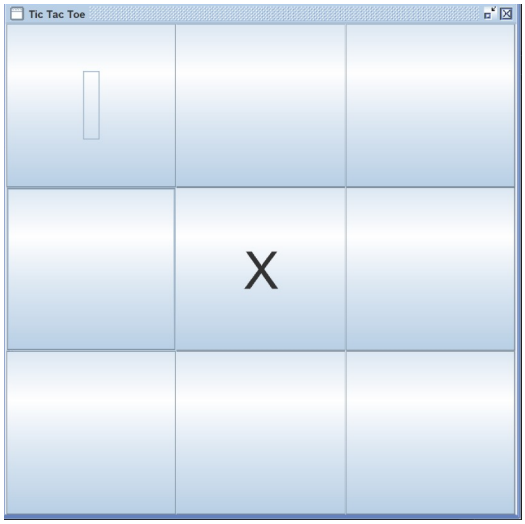
which they can use to play the game.

Singleton Pattern

The Singleton pattern is implemented in the TicTacToe class at its very beginning. We first initialize a private TicTacToe object to a null pointer. Afterwards we create a method called getTicTacToeInstance which returns a reference to a created TicTacToe object. The constructor is made private, so that the TicTacToe object can only be obtained through the public static method getTicTacToeInstance.
We run the program with the following command sequence,



After selecting Apply we are are presented with the following screen,



which can be used to play the game.
I was the sole contributor to this project.