**Game Name:**

Heroes of Miracles

**Overview of the game:**

A hero wanders the field of the game board to find and defeat the dragon to restore peace in the land. On the way to the dragon the hero will encounter all types of dangers and collecting spoils from enemies to fight the dragon.

**Functions of the game:**

1. Player move the heroes around the map with the “wasd” (up, left, down, right) keys.
2. Player can open the menu item to view the hero’s statistics and to view and equip new weapon power ups earned by killing enemies.
3. Player fight enemies with turn based between the characters, they can choose which move to take from the battle menu.
4. Player has to kill enemy to earn rewards from them. Bring their HP to 0. If enemy brings your characters to 0 HP, you lose the game.
5. Player’s character’s HP will be carried across different battles, they can choose to heal up outside a fight or during a fight.
6. Player can choose to exit the game when they open the item menu and select to quit the game.
7. Player wins the game when their hero defeats the dragon, bring their HP to 0 while yours is above 0.

**Design Patterns Used:**

Strategy:

* Used to return directly what the weapon power up is at run time when an enemy is defeated and the hero collects the power up.
* I chose this pattern because the power up should be randomly generated during run time of the game and not prefixed already before the game has started.

Decorator:

* Used when the player choose to equip the power up to a weapon that has been collected from a defeated enemy.
* I chose this pattern because the power ups is an extra extension to the weapon of the hero. It also gives options for later implementations to combine multiple power ups onto the same weapon, it makes it easier to calculate the total damage and describe which power ups had been combined.

State:

* Used when the player moves the hero character around the “game board”.

Template:

* Used as a default template for the type of different game characters that can spawn in the game with their statistics. Game characters can be various type of heroes controlled by the player or type of enemies controlled by the computer AI.
* I chose this pattern because all types of characters and creatures in the game is a game character. They all have almost the same statistics and method implementation for each of them. I want each subclass to override those methods as per needed.

Observer:

* Used when the dragon (boss) summons drakes during the battle. The boss gives its current data of statistics to the drakes.
* I chose this pattern because the drakes are a minion of the dragon, it is basically a one to many relationship between the 2 objects. I want the drakes to have the current statistics of the dragon during mid-battle of the final fight. By using the dragon as a subject, he can push his data to the drake observer when the time is right.

MVC:

* Used when the UI sends command objects (containing the states of the map, the character and the enemies) to the controller, who in turn updates the models, and have them calculate the results, and send back the the results (the new states) to the UI.