Greed Specification

Notes from zoom meeting

* Copy
  + main file over.
  + Keyboard services
  + Video\_services
* We will need to use the velocity in the new game.
  + Move\_next affects the velocity
* Create robot puts you in the middle of the screen
* Might want to inherit from actor to character for out game.

Game specifications

* Gems (\*) and rocks (o) **randomly appear and fall** from the **top** of the screen.
  + Will need activate the velocity. And make sure the y is 0 and x randomly changes for the starting point of the objects.
    - Lines 65 through 68 in \_\_main\_\_.py affect this.
  + Will need to change line 61 in main file since we are only dealing with \* and o’s as the ascii characters.
* The player (#) can **move left or right** along the **bottom** of the screen.
  + Will need to modify keyboard\_services so only left or right apply.
    - Actually, I think it will be in main file on line 45 we will just set to the max y.
* If the player touches a gem they earn a point.
  + Will need to create a score somehow
  + Display it on the top of the screen where the messages appear.
* If the player touches a rock they lose a point.
  + Logic for this. Probably need to use inheritance to say if object touches rock where rock is pulled from another class then lose point.
  + Would be method
    - Use setter and getter
    - \_Score = 0 (attribute)
    - Set\_score(self)
      * Return self.\_score
    - Get\_score(self, …)
      * Do math
* Gems and rocks are removed when the player touches them.
  + Whatever method we write it would be in the artifact class or actor class
* The game continues until the player closes the window.
  + Sounds like the video\_services

Requirements:

* The program must have a README file.
* The program must have at least eight classes.
* Each module, class and method must have a corresponding comment.
* The game must remain generally true to the order of play described earlier.

Classes

* Services
  + **KEYBOARD\_SERVICES**
  + Video\_services
* Shared
  + Color
  + Point
* Directing
  + Director
* Casting
  + **ACTOR**
  + **ARTIFACT**
  + **CAST**
* Main file

Dallan will look into how to get the score to show on the screen.

Agenda

* Create blueprint
* Divide and conquer