**AG 131  
Lab 6 – Behaviors**

**NOTES:**

* A Unity package called “Arcs” has been provided to you. You can use these models to help determine if the code you have written is working correctly.
* The Packages made for this lab assignment must be imported in 2018.3 or later.

**Part 1: Let’s take a look at that…**

* Create a new Script called “**VisionControl**”
  + It should have the following public members

float VisionRange  
 Maximum distance allowed to be considered in range

float VisionAngle  
 Angle of view

bool UseActorsOnly  
 True: filer out GameObjects that do not have an Actor Script (or child of) False: do not filter GameObjects

public bool HasLineOfSight (GameObject target)  
 Performs a ray cast from viewing object to target.   
 True: LOS (Line of sight) is clear  
 False: LOS is blocked by another object

public List<GameObject> PerformVision()  
 Performs vision check and   
 Returns a list of GameObjects that in LOS from the viewing object  
 Objects must meet conditions above to returned

* Set up a Scene
  + Have objects that are moving and not moving.
    - Moving on Use Paths
    - Using and not Using an Actor Script
  + Set up 4 Different objects Vision scripts attached.
    - Have a selection like above
  + Create a Canvas with four slots
    - Each Slot corresponds to one of the 4 Game Objects with a Vision Script
* Create a new Script called “TestVisionControl”
  + It should have the following public members   
     <UI Element> hud  
     VisionControl vision
  + This script will either get or add a gameObject a ControlVision script
  + When PerformVision() is called from its TestVision reference, it will pass the names of the game objects found to the UI Element assigned to it so it will be displayed on screen.
* Create a TestPrefab
  + It will have both VisionControl and TestVisionControl

**Part 2: A Heist (Simulation)**

The simulation will consist of the following 3 basic entities.  
Each Entity will have the following behavior states.

It’s okay to use cubes for this simulation

* Money [Entity]
  + Banked *[Behavior State]*
    - Available to be stolen
    - Doesn’t move.
  + Stolen *[Behavior State]*
    - Money is Stolen when Robber touches it
    - Is no longer active in the scene
* Guard: [Entity]
  + Patrol *[Behavior State]*
    - Walks a path around the money
  + Chase *[Behavior State]*
    - Moves towards the Robber
    - Moves at Walking Speed
  + Sprint *[Behavior State]*
* Robber [Entity]
  + Case *[Behavior State]*
    - Moves into position to steal the money
    - Moves at walking speed
  + Flee *[Behavior State]*
    - Oh Shit, it’s the cops
    - Flees to get away from a guard
  + Steal *[Behavior State]*
    - Short quick burst of movement to grab the money
  + Caught *[Behavior State]*
    - When touched by a guard, the robber is caught
    - Is no longer active in the scene

Simulation Ends when either

* All Robbers are caught
* All Money is stolen