```
                                                                           /*
----------------------------------------------------------------------
SpectreAI
Written by Jeff Fisher
GSP494-497 Senior capstone
June 28, 2014
----------------------------------------------------------------------
                                                                           */
using UnityEngine;
using System;
using System.Collections;


public class SpectreAI : MonoBehaviour {

   private Vector3 currVelocity;

   public Transform playerTransform;
   public Transform horrorTransform;
   public Vector3 playerPos;
   private Vector3 horrorPos;
   private Vector3 horrorPosCheck;

   public float lookAtRange = 15.0f;
   public float attackRange = 10.0f;
   public float rotationSpeed = 1.0f;
   private int loopCount = 0;
   private Vector3 monAccel = new Vector3( 0.0f, 0.0f, 0.0f );
   private float squaredLookAtRange = 0.0f;
   private float squaredAttackRange = 0.0f;

   // Initializes Spectre
   void Start () {
      // Setting up the transform to control the Spectre
      horrorTransform = transform;
      transform.rigidbody.isKinematic = false;
      transform.rigidbody.detectCollisions = true;
      gameObject.tag = "Horror";

      // Setting up the transform to contain the information on the player
      playerPos = playerTransform.position;

      squaredLookAtRange = Mathf.Pow( lookAtRange, 2.0f );
      squaredAttackRange = Mathf.Pow( attackRange, 2.0f );
   }

   // Additional initialization
   void Awake(){
      // Detecting the player information
      playerTransform = GameObject.FindWithTag("Player").transform;
   }
   // Update is called once per frame
   void FixedUpdate () {
      // Set up temp variables
      float tempVectorX = 0.0f;
      float tempVectorZ = 0.0f;
      float totalVectorX = 0.0f;
      float totalVectorZ = 0.0f;

      // Find X and Y vectors to player and calculate square distance
      tempVectorX = playerTransform.position.x - transform.position.x;
      tempVectorZ = playerTransform.position.z - transform.position.z;
      float sqDistToPlayer = Mathf.Pow( tempVectorX, 2.0f ) + Mathf.Pow( tempVectorZ, 2.0f );

      // Track real time to control damage rate
      loopCount++;
      loopCount %= 50; // .02 second update, so 1x per second
      if( sqDistToPlayer < attackRange ){

         // Turn ghost in direction of player
         transform.rotation = Quaternion.Slerp(transform.rotation, Quaternion.LookRotation(
               (new Vector3(playerTransform.position.x, 0.0f, playerTransform.position.z)) -
               (new Vector3(transform.position.x, 0.0f, transform.position.z)) ), rotationSpeed*Time.deltaTime);

         // Calculate approximate steering vector
         float truDistToPlayer = Mathf.Sqrt( sqDistToPlayer );
         totalVectorX += ((tempVectorX / truDistToPlayer) * .025f);
         totalVectorZ += ((tempVectorZ / truDistToPlayer) * .025f);

         // Calculate acceleration (used to use rigidbody.AddForce but that didn't work well for this)
         monAccel = new Vector3( (totalVectorX *100), 0.0f, (totalVectorZ*100) );

         // Applies damage once per second
         if (loopCount == 0){
            if( sqDistToPlayer < 9 ){
               PlayerHealth.currentHealth -= 1;
            }
         }
      }
   }

   void Update(){
      // Move and animate character
      CharacterController spectreMoveController = GetComponent<CharacterController>();
      spectreMoveController.SimpleMove(monAccel);
      animation.Play();
   }
}
```