Shoot Script
There are a few variables at the start that collect a game object that I had to drag into the script in unity so that the script knows what game objects to use. There is also a float for the speed of the bullets set to public which allows me to change the speed in unity. There is an int for ammo, a vector2 for the direction of the bullets, a float for the angle of the bullets, and vector3 for determining if the mouse is positioned behind the player. Anything with 'public' in front of it is considered a global value which means that anything can access that if needed otherwise, if something does not have 'public' in front of it then it should only be able to be accessed within the script it is located in. In the update method, mouseDistance is set equal to the position of the mouse on the screen minus the position of the player. Ammo is set to the amount of ammo the player currently has. BulletDirection is set to the position of the mouse on the screen minus the position of the player in a vector3. BulletAngle is set to the calculation needed to determine what angle the bullet should be shot at. BulletSpawner.rotation is set to the bulletAngle allowing the bullet to be shot in the direction of the mouse. The first 'if' statement looks for if the ammo is greater than 0 and the mouse position is in front of the player to then move on to the next 'if' statement that looks for if the player clicks the left mouse button which spawns a bullet and a gunshot animation at a gameobject positioned at the edge of the player's gun in unity and the bullet will move in the direction of where the mouse was when clicked. The 'if' and 'else if' is for the gunshot animation to face the right way when the player fires a shot with the transform.localScale function that gets set to -1 in the x to face backwards and 1 in the x to face forwards. The bullet and gunshot animation are destroyed after their respective amount of time set and the bullet is only destroyed if it does not collide with anything otherwise it will be destroyed when it collides with something. Ammo is decreased by 1 and updates the ui in the game to show that the ammo is gone.

LevelThreeMovement Script
A float is declared for the move speed. In the update method, a vector3 is made that gets the players vertical position and is then used to calculate the new position of the player if they decide to move in the vertical direction at the speed of whatever moveSpeed is set to. The same thing happens for horizontal movement.

KeepGameUI Script
In the start method, an int is set to the length of the game objects that have a tag of player_ui and they will be destroyed if they are larger than 1 character. This prevents the ui from carrying over from the last game. In the Awake method, the ui will not be destroyed when loading new scenes.

GameUI Script
The start method declares ammo, health, and score which are used in KeepGameUI to start the game at what values I set them to for the ui in the game.

Buttons Script

The startIntro method is called by the play button on the main menu scene and starts the intro sequence animation. The intro sequence animation is in a new scene that plays a simple text animation on the screen that I made and at the end of the animation is an event that calls startLevel1 which loads the game where the player will play. The quitApp method is called by the quit button on the main menu and closes the game. The mainMenuButton method is called on the win and lose scenes and loads the main menu scene. In the start method, an if statement looks for if the win or lose scenes are loaded and sets the game's ui to nothing so that no ui shows up.

Melee Script

Floats are declared for attackSpeed, startAttackSpeed, and attackRange. A layerMask is made that looks for a layer in unity that I specify and a transform is made that looks for a game object in unity that I specify. An Animator and audioclip are made. In the update method, an if statement looks for if attackSpeed is less than or equal to zero to allow the player to attack when the melee is ready to be used again otherwise the attackSpeed is subtracted from Time.deltaTime which counts time in the game. Within the if statement, another if statement that looks for if right mouse button is pressed activates an animation and sounds that shows the sword swinging and activates a circle collider on a game object in front of the player with parameters to look for if a game object with the layer defined in 'enemies' is inside the range of the melee and then sets that to a variable which is then used in a foreach loop that calls the OnTriggerEnter2D method in the script with a parameter of the enemy object that was hit by the melee. The attackSpeed is also initiated and starts the cooldown of the melee. The OnTriggerEnter2D method takes a collider2D parameter and makes a string equal to the parameter game object with a tag. The next three if statements will call the correct health decreasing methods in the scripts needed to lower the health of either an enemy, an enemy spawner, or the boss. The OnDrawGizmosSelected method draws a green line in unity to show the range of effect of the melee's range when selecting the player.

Movement 2D Script

Floats are declared for speed and jumping force and a bool to check if the player is on the ground. A SpriteRenderer to show the sprite of the player. An animator for animations. Another bool to check which way the player is facing and a float for horizontal movement. In the start method, sprite is set to the spriteRenderer object on the player in unity and animator is set to the Animator object on the player in unity. In the update method, horizontalMove is set to the horizontal axis position of the player times the speed of the player. A vector3 is made that collects where the player is in the world. Transform.position takes the player's position and multiplies it with the speed and time to allow movement. The animator.SetFloat looks for the speed of the player in the horizontal direction and gets the absolute value of that to then play the walking animation properly. The if and else if statements swap the way the sprite is facing depending on which way the player wants to move. The landed and jump methods are also called. Landed checks if the player is on the ground and sets the IsJump animation trigger to false. Jump checks if the player presses spacebar and is on the ground and makes the player jump and plays the jump animation.

EnemyCollision Script
An int for enemy health and score are declared and an animator and audioClip are made. The start method sets the animator equal to the animator object on the enemy. The decreaseHP method lowers the enemy health by 1 whenever it is called and shows the enemy health in the console. A hurt sound effect is also played when hit and if the health goes to zero then the enemyDead method is called. The enemyDead method updates the player's score by whatever amount I choose and starts the death animation for the enemy and sets its shooting range and line of sight to zero so it can no longer move while the animation plays. The enemy is then destroyed after a set time and the updateScore method is called. The OnTriggerEnter2D method takes a parameter of a collider2D which is usually going to be the player bullet which is set to a string that looks for if an object with the tag of playerBullet collides with the enemy and if the enemy's health is more than zero then the bullet will be destroyed and DecreaseHP is called. The updateScore method sets the score ui value to an int of score and then adds the new score value from enemyDead to the ui.

EnemySpawner Script
Three game objects, three floats, a transform, two ints, and an audioClip are declared at the start. Start method sets a player variable equal to the game object with the player tag. Update method sets a float to the position of the player in the world and has an if statement looking for if the distance from the player is less than the line of sight of the enemy spawner and if the spawn time is off of its cooldown and if those conditions are met then an enemy will be spawned at a game object in front of the spawner and the spawn time cooldown is started. The DecreaseHP, OnTriggerEnter2D, updateScore, and OnDrawGizmosSelected methods do exactly the same thing as what I've previously mentioned for other scripts. The spawnerDead method only has a small difference from the enemyDead method from before and that is that instead of a death animation, there is an explosion animation that is spawned in place of the spawner when it is destroyed and the explosion is destroyed after it finishes playing.

Enemy Script
Five floats, two game objects, a transform, and an animator are declared at the start. The start method does the same stuff as what I've previously described in other scripts. The update method sets the distance of the player from the enemy equal to a float and then checks in an if statement if the player is in the enemy line of sight and outside of the shooting distance and if this is true then the enemy will move to the player and the MovingToPlayer method is called. The else if statement is for if the player is inside the shooting distance of the enemy in which the enemy will stop moving and shoot a bullet at the player when the nextFireTime is off of

cooldown. The MovingToPlayer method checks which side the player is on based on the enemy's position and makes the enemy look at the player depending on the side and starts the moving animation. OnDrawGizmosSelected shows the enemy line of sight and shooting range with green lines.

BossAttack Script
A float, a layerMask, a transform, and four game objects are declared. MeleeAttack method is called by the boss attack animation in an event and uses the range of effect of hitRange connected to the attackPoint game object to hit the player which is connected to the player layer under attackMask and if this happens then the player's DecreaseHealthFromBoss method is called which lowers the player's health. RangedAttack method is called by an animation event in the boss spell cast animation and spawns an attack animation and invisible hitbox above the player on a game object attached to the player and these will be destroyed after the attack animation is finished. The RangeAttackFinished method is called by the boss spell cast animation event at the end of the animation that spawns an invisible object that is destroyed after a specified amount of time to act as a cooldown for the boss range attack which in another script on the boss checks to see if that game object exists and if it does then the boss will not try to use the range attack. This was causing me trouble because the boss was designed to walk to the player and use the range attack when the player is in range and then keep walking to the player after the attack was used while it was on cooldown but the boss wouldn't keep walking and this was the way I discovered on my own to fix the issue. OnDrawGizmosSelected shows the boss hit range with a green line.

BossMovement Script
Eight floats, a transform, and an animator are declared. Start method does the same thing as described in previous scripts. Update method has the same moving to player lines as the regular enemy but with an additional range for the melee attack. The first if else statement is for what I described in the BossAttack script that looks for the invisible cooldown game object and sets the cast range to zero if that object exists otherwise the range will be its original range to use on the player. The next if statement is for the movement to the player. The first else if statement is for the ranged attack and plays the casting animation and stops the moving animation. The second else if statement does the same thing but plays the melee animation and has a smaller range of effect. MovingToPlayer method is an exact copy of the enemy script's movingToPlayer method. OnDrawGizmosSelected draws green lines for the line of sight, melee range, and casting range.

BossScript Script
Two ints, an animator, and an AudioClip are declared. Start method does the same thing as described in another script. DecreaseBossHP method is called when the player hits the boss with a bullet or melee and decreases the boss health by 1 and plays a hurt sound effect and if the boss health is at zero or less then bossDead is called. The bossDead method updates the player score

and plays the boss death animation and calls the updateScore method which is described in another script. The WinningScreenLoad method is called when the boss death animation finishes and gets to the animation event that calls it. OnTriggerenter2D does the same thing as the enemyCollisions script.

Bullet Script
A game object, a float, and a rigidBody2D are declared. The Start method sets the bullet variable equal to the rigidBody2D element on the bullet object, and sets the target variable to the game object with the player tag. A vector2 is set to the player's position minus the enemy's position multiplied by the bullet speed and is then used to calculate where the bullet should be fired which will be at the player. The bullet is destroyed after a set time if it does not collide with anything.

CameraFollow2D Script
I'm not sure exactly what this script does since I couldn't find anywhere in the book that talks about it. A transform, five floats, and three vector3s are declared. The Start method sets the position of the camera to the be on the player. The update method sets a float to the player position minus the last position of the player in the x direction. A bool is set to the absolute value of that float being greater than the lookAheadMoveThreshold which looks ahead of the player. An if statement checks if that bool is true and tells the camera to look ahead of the player (it doesn't look ahead because it drags behind the player if you have the values set to something other than zero), else the camera will return to being on top of the player. This camera script kinda sucks because of how it drags behind the player making it so the player can't really see what's coming up in front of them so I ended up setting everything to zero and all the camera does is stay directly on top of the player which could probably be done with a much simpler script.

CheckGround Script
A gameobject is declared. In the start method, the player game object variable is set to the game object connected to the player in unity that is used for checking if they are on the ground. In OnCollisionEnter2D, a collision2D parameter is taken from the game object and checks if the object is colliding with objects tagged with ground or trap and sets the isGrounded bool in Movement2D to true to allow the player to jump. In OnCollisionExit2D, the bool is set to false when the player is not touching objects tagged with ground or trap so that the player can't jump because they will be in the air.

Collisions Script
Three ints, an animator, and two audioClips are declared. In the start method, the updateUI method is called. In OnCollisionEnter2D, a collision2D parameter is used and sets it to a string to look for tags on game objects in the game. If an object with the ammoPickup or healthPickup

tag is collided with, then the score is updated and the ammo or health is updated and the updateUI method is called and a collecting sound effect is played. If the player collides with objects tagged with enemy or trap then the DecreaseHealth method is called. The next four if statements check if the player collides with the invisible box colliders on the levels to allow the player to load the next scene to keep playing. The updateUI method updates everything on the player's ui with any new information when called. The DecreaseHealth method decreases the player's health by 1 and if the health reaches zero then the player dies and the lose scene is loaded. A hurt sound effect is also played when hit. DecreaseHealthFromBoss does exactly the same thing but the player loses three health instead of 1. OnTriggerEnter2D takes a collider2D parameter and sets it to a string that looks for tags on game objects. An if statement looks for if an enemy bullet collides with the player and destroys that bullet and calls DecreaseHealth and updateUI. The next four if statements are for the portals on the boss level and if the player collides with one of the portals on the bottom layer then they will be teleported to an invisible object on the top layer and if they collide with a portal on the top then they will be teleported to the bottom layer.

Asset Store Links Used
https://assetstore.unity.com/packages/2d/environments/warped-city-2-200208
https://assetstore.unity.com/packages/2d/gui/icons/wewe-free-game-icons-28604
https://assetstore.unity.com/packages/2d/environments/traffic-control-resource-pack-20085
https://assetstore.unity.com/packages/2d/free-2d-mega-pack-177430
https://assetstore.unity.com/packages/2d/characters/slime-character-157405
https://assetstore.unity.com/packages/2d/environments/pixel-starter-pack-by-gamertose-168809
https://assetstore.unity.com/packages/2d/textures-materials/futuristic-panel-textures-lite-80176
https://assetstore.unity.com/packages/2d/gui/icons/2d-hodgepodge-pack-by-gamertose-15710
https://assetstore.unity.com/packages/2d/characters/bringer-of-death-free-195719
https://assetstore.unity.com/packages/2d/textures-materials/2d-flat-explosion-66932
https://assetstore.unity.com/packages/audio/music/rock/eternity-djent-metal-scfi-horror-music-pack-194898
https://assetstore.unity.com/packages/audio/music/various-genres-music-collection-free-191462
https://assetstore.unity.com/packages/audio/action-music-taster-pack-100957
https://assetstore.unity.com/packages/audio/music/rock/7-track-hard-rock-game-music-pack-56544
https://assetstore.unity.com/packages/audio/music/rock/layered-rock-music-pack-action-129742
https://assetstore.unity.com/packages/audio/music/dynamic-music-35925
https://assetstore.unity.com/packages/audio/music/orchestral/free-game-music-collection-177094

Youtube links are inside the respective scripts inside the game