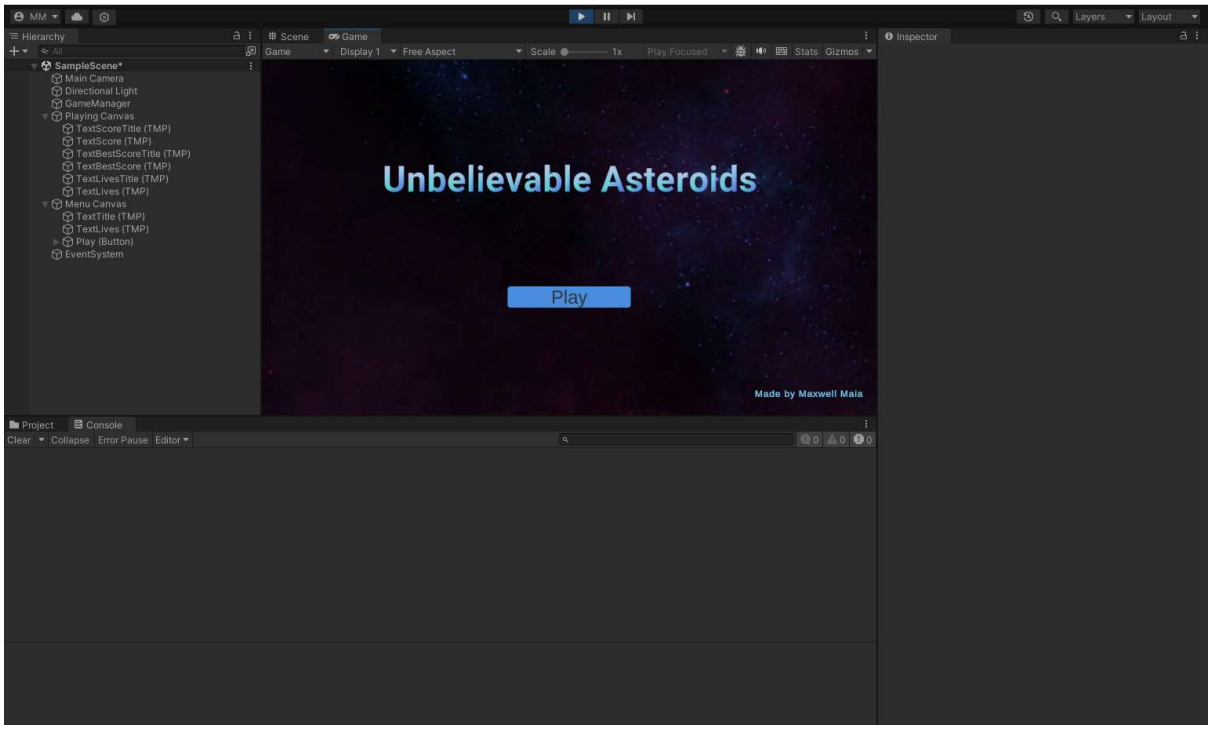
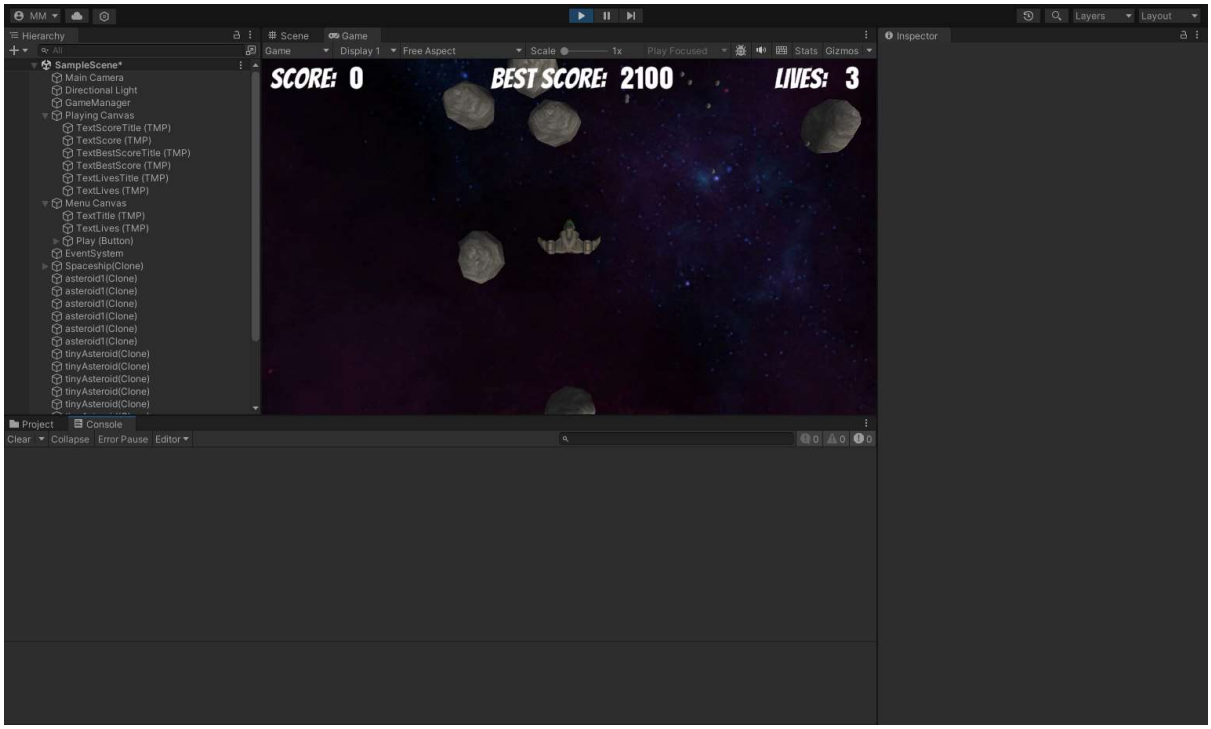
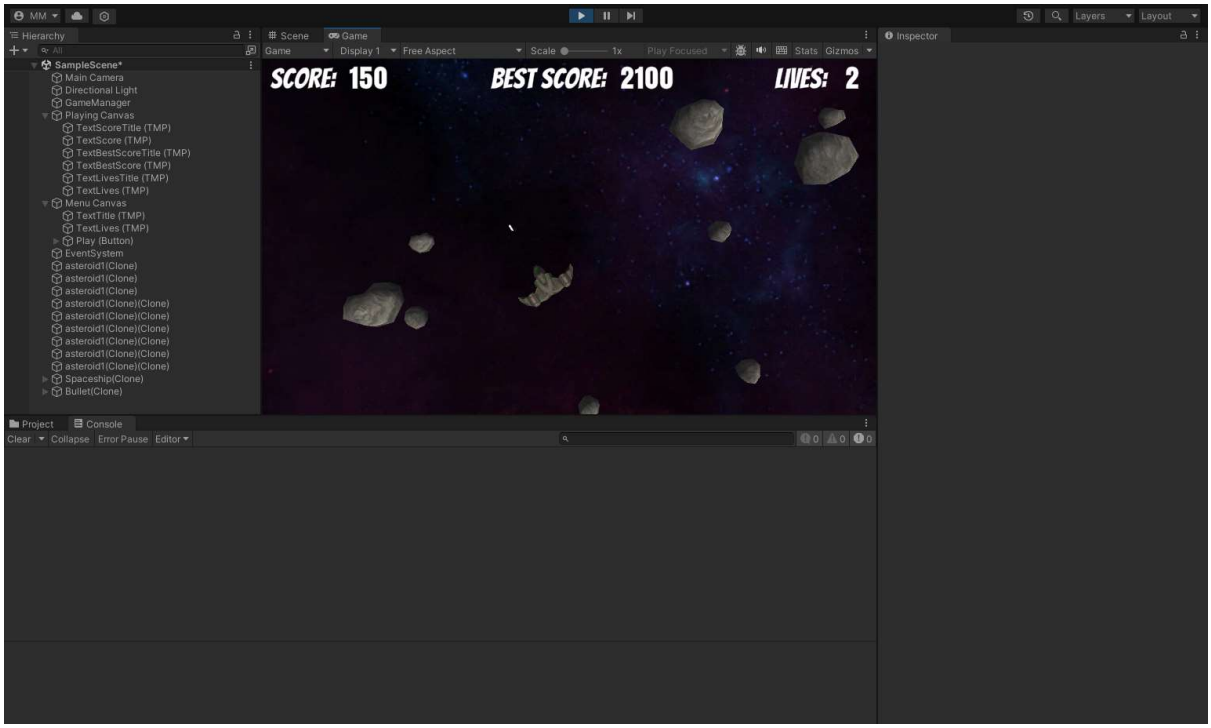
```
//Maxwell Maia
//21236277
```

Finishing Asteroids
Lab 7
CT3536







The code for this lab was implemented in **GameManager** and **Asteroid**.

All the other classes of this project have been included in this document too.

GameManager

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using TMPro;

public class GameManager : MonoBehaviour
{
    public int currentGameLevel;
    public GameObject asteroidPrefab;
    public GameObject spaceshipPrefab;
    public static GameManager gamemanager;
    private bool isPlaying;
    public GameObject menuCanvas;
    public GameObject playingCanvas;
    public Button playButton;
    public TMP_Text txtScore;
    public TMP_Text txtBestScore;
    public TMP_Text txtLives;
    public List<GameObject> activeAsteroids;
    private int score;
    private int bestScore;
```

```
    // Start is called before the first frame update
    void Start()
    {
        //Position camera
        Camera.main.transform.position = new Vector3(0, 30, 0);
        Camera.main.transform.LookAt(new Vector3(0, 0, 0), new Vector3(0, 0, 1));

        gamemanager = this;

        isPlaying = false;
        goToMenu();

        playButton.onClick.AddListener(PlayOnClick);
    }

    void PlayOnClick()
    {
        isPlaying = true;
        goToPlaying();
        StartNewGame();
    }

    void StartNewGame()
    {
        currentGameLevel = 0;

        //Set score to 0
        score = 0;

        //Load the saved best score into the best score variable
        LoadBestScore();

        //Set the UI to default values
        txtScore.text = "0";
        txtLives.text = "3";
        txtBestScore.text = bestScore.ToString();

        CreatePlayerSpaceship();

        StartNextLevel();
    }

    // Update is called once per frame
    void Update()
    {
        if(isPlaying == true)
        {
            //The active asteroids list is said to be empty when all of the indexes
are null
            bool isEmpty = true;

            foreach(GameObject a in activeAsteroids)
            {
                if (a != null)
                {
                    isEmpty = false;
                }
            }

            //When there are no more active asteroids
            //(When the player has destroyed all the asteroids)
            if (isEmpty == true) //activeAsteroids.Count == 0
```

```
            {
                //Clear the asteroid list
                activeAsteroids.Clear();

                StartNextLevel();
            }
        }
    }

    //Score system
    public void IncreaseScore(bool isLargeAsteroid)
    {
        //Get current score
        score = int.Parse(txtScore.text);

        //Increase score by 50 for large asteroids and 100 for small asteroids
        if (isLargeAsteroid)
        {
            score += 50;
        }
        else
        {
            score += 100;
        }

        //Update the score text
        txtScore.text = score.ToString();

        //It's nice to see the best score keep increasing as you are playing if you
have beat it
        CheckAndUpdateBestScore();
    }

    //BestScore System
    public void CheckAndUpdateBestScore()
    {
        LoadBestScore();

        //If you beat the score update the best score
        if(score > bestScore)
        {
            SaveBestScore(score);

            //Update best score UI
            txtBestScore.text = score.ToString();
        }
    }

    //Helper method to save best score
    public void SaveBestScore(int newBestScore)
    {
        PlayerPrefs.SetInt("localBestScore", newBestScore);
    }

    //Helper method to load best score
    public void LoadBestScore()
    {
        bestScore = PlayerPrefs.GetInt("localBestScore");
    }

    //Lives system
    public bool DecreaseLives()
```

```
{
    int lives;
    lives = int.Parse(txtLives.text);

    lives--;

    if (lives <= 0)
    {
        //Gameover
        EndGame();
        return false;
    }
    else
    {
        //Update lives
        txtLives.text = lives.ToString();
        return true;
    }
}

public void EndGame()
{
    //Check if you beat the best score and update it if you have
    CheckAndUpdateBestScore();

    //Destory all asteroids
    foreach(GameObject ast in activeAsteroids)
    {
        Destroy(ast);
    }

    goToMenu();
}

//Hide the Playing GUI and show the Menu GUI
void goToMenu()
{
    //Set the alpha of each GUI
    playingCanvas.GetComponent<CanvasGroup>().alpha = 0;
    menuCanvas.GetComponent<CanvasGroup>().alpha = 1;
}

//Hide the Menu GUI and show the Playing GUI
void goToPlaying()
{
    //Set the alpha of each GUI
    menuCanvas.GetComponent<CanvasGroup>().alpha = 0;
    playingCanvas.GetComponent<CanvasGroup>().alpha = 1;
}

void StartNextLevel()
{
    currentGameLevel++;
    int numAsteroids = 6 * currentGameLevel;

    //Spawn asteroid depending on currentGameLevel
    for (int i = 0; i < numAsteroids; i++)
    {
        GameObject newAsteroid = Instantiate(asteroidPrefab);
        activeAsteroids.Add(newAsteroid);
    }
}
```

```
    public void CreatePlayerSpaceship()
    {
        GameObject spaceship = Instantiate(spaceshipPrefab);
        spaceship.transform.position = new Vector3(0, 0, 0);
    }
}
```

Asteroid

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Asteroid : MonoBehaviour
{
    public GameObject asteroidObject;

    private Vector3 spawn;

    public GameObject tinyAsteroid;
    private int numTinyAsteroidsToSpawn;

    private float timeOfLastSpawn = 0f;

    //Is the asteroid a large asteroid or debris fragment
    public bool isLarge = true;

    // Start is called before the first frame update
    void Start()
    {
        //Set the asteroid's position at a random position near the edges of the
screen
        //Select top, bottom, left or right to spawn the asteroid and pick a random
position along that edge
        if(Random.Range(0, 2) > 0.5f)
        {
            //Spawn along the top or bottom of screen
            if(Random.Range(0, 2) > 0.5f)
            {
                //Spawn along the bottom of screen
                spawn = Camera.main.ViewportToWorldPoint(new Vector3(Random.Range(0f,
1f), 0, 30));
            }
            else
            {
                //Spawn along the top of screen
                spawn = Camera.main.ViewportToWorldPoint(new Vector3(Random.Range(0f,
1f), 1, 30));
            }

        }
        else
        {
            //Spawn along the left or right of the screen
            if (Random.Range(0, 2) > 0.5f)
```

```
            {
                //Spawn along the left of screen
                spawn = Camera.main.ViewportToWorldPoint(new Vector3(0,
Random.Range(0f, 1f), 30));
            }
            else
            {
                //Spawn along the right of screen
                spawn = Camera.main.ViewportToWorldPoint(new Vector3(1,
Random.Range(0f, 1f), 30));
            }
        }

        //Set the asteroid's position
        asteroidObject.transform.position = spawn;

        //Set the asteroid moving in a random direction
        //Set the rotation of the asteroid randomly
        //asteroidObject.transform.Rotate(0.0f, 0.0f, Random.Range(0, 360));
        //Move the asteroid in a random direction
        asteroidObject.GetComponent<Rigidbody>().AddForce(new Vector3(Random.Range(-
500f, 500f), 0, Random.Range(-500f, 500f)));
    }

    void OnCollisionEnter(Collision collision)
    {
        //Create particle effect using smaller non-colliding game objects
        numTinyAsteroidsToSpawn = Random.Range(1, 3);
        for (int i = 0; i < numTinyAsteroidsToSpawn; i++)
        {
            //Spawn the tiny asteroid and set it's position and velocity in a random
direction
            GameObject newTinyAsteroid = Instantiate(tinyAsteroid);
            newTinyAsteroid.transform.position = collision.contacts[0].point;
            newTinyAsteroid.GetComponent<Rigidbody>().velocity = new
Vector3(Random.Range(-2f, 2f), 0, Random.Range(-2f, 2f));
        }

        //If the asteroid hit the player ship, destroy the player ship and re-create
it in the centre of the screen
        //Also add some spawn protection for 3 seconds
        //Also Decrease the lives
        if (collision.gameObject.CompareTag("Spaceship") && (Time.time -
timeOfLastSpawn > 3f))
        {
            timeOfLastSpawn = Time.time;

            Destroy(collision.gameObject);

            bool stillAlive = GameManager.gamemanager.DecreaseLives();

            if (stillAlive == true)
            {
                GameManager.gamemanager.CreatePlayerSpaceship();
            }
            else
            {
                return;
            }
        }
```

```
        //If the asteroid hits the bullet, destroy the bullet and asteroid and create
small debris fragments
        if (collision.gameObject.CompareTag("Bullet"))
        {
            //Destroy the bullet
            Destroy(collision.gameObject);

            //If the asteroid hit is a large asteroid
            //Only create small asteroids
            //And increase the score by 50
            if (this.isLarge == true)
            {
                //Create small asteroids (asteroid fragments)
                for (int i = 0; i < 2; i++)
                {
                    //Spawn the fragments
                    GameObject fragmentAsteroid = Instantiate(asteroidObject);

                    //Add the fragment asteroids to the active asteroid list
                    GameManager.gamemanager.activeAsteroids.Add(fragmentAsteroid);

                    //Position the fragments where the large asteroid was
                    fragmentAsteroid.transform.position = this.transform.position;

                    //Set the scale of the fragments
                    fragmentAsteroid.transform.localScale = new Vector3(0.045f,
0.045f, 0.045f);

                    //Set the fragments isLarge boolean to false
                    fragmentAsteroid.GetComponent<Asteroid>().isLarge = false;

                    //Set the fragments velocity in a random direction
                    fragmentAsteroid.GetComponent<Rigidbody>().AddForce(new
Vector3(Random.Range(-300f, 300f), 0, Random.Range(-300f, 300f)));
                }

                //Hit larger asteroid
                //Increase the score by 50
                //Argument is Large asteroid = true
                GameManager.gamemanager.IncreaseScore(true);
            }
            else
            {
                //Hit small asteroid
                //Increase the score by 100
                //Argument is Large asteroid = false
                GameManager.gamemanager.IncreaseScore(false);
            }

            //Destroy the asteroid (large or small) that was hit by the bullet
            Destroy(this.asteroidObject);
        }
    }

    // Update is called once per frame
    void Update()
    {

    }
}
```

Spaceship (unchanged this lab)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Spaceship : MonoBehaviour
{
    public static Spaceship mySpaceship;

    public GameObject bulletPrefab;

    public float fireRate = 0.25f;

    private float nextFire = 0.0f;

    // Start is called before the first frame update
    void Start()
    {
        mySpaceship = this;
    }

    // Update is called once per frame
    void Update()
    {
        if(Input.GetKey(KeyCode.UpArrow))
        {
            this.GetComponent<Rigidbody>().AddRelativeForce(Vector3.forward * 20);
        }

        if (Input.GetKey(KeyCode.LeftArrow))
        {
            this.GetComponent<Rigidbody>().AddTorque(transform.up * -60);
        }

        if (Input.GetKey(KeyCode.RightArrow))
        {
            this.GetComponent<Rigidbody>().AddTorque(transform.up * 60);
        }

        //The spaceship shoots bullets
        //Spawn the bullet in front of the spaceship
        //Set the rotation
        //Set the velocity
        if (Input.GetKeyDown(KeyCode.Space) && Time.time > nextFire)
        {
            //Set the next in game time that the bullet can be fired
            nextFire = Time.time + fireRate;

            //Spawning the bullet

            //Get the spaceship's position
            Vector3 spaceshipPos = mySpaceship.transform.position;

            //Get the spaceship's forward direction
```

```
            Vector3 spaceshipDirection = mySpaceship.transform.forward;

            //Get the spaceship's rotation
            Quaternion spaceshipRotation = mySpaceship.transform.rotation;

            //Choose a distance in front of the player to spawn the bullet
            float spawnDistance = 2.05f;

            //Choose a bullet speed
            float bulletSpeed = 25f;

            //Get the bullet velocity
            Vector3 bulletVelocity = spaceshipDirection * bulletSpeed;

            //Get the position in front of the spaceship
            Vector3 bulletSpawnPos = spaceshipPos + spaceshipDirection *
spawnDistance;

            //Spawn bullet in calculated position and spaceship rotation
            GameObject bullet = Instantiate(bulletPrefab, bulletSpawnPos,
spaceshipRotation);

            //Set the velocity of the bullet
            bullet.GetComponent<Rigidbody>().velocity = bulletVelocity;
        }
    }
}
```

CheckOutOfScreen (unchanged this lab)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CheckOutOfScreen : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {
        //Wrap object to other side of screen, check every 0.2 seconds. 5 times a
second
        InvokeRepeating("CheckOutOfScreenMethod", 0.2f, 0.2f);
    }

    // Update is called once per frame
    void Update()
    {

    }
```

```
    void CheckOutOfScreenMethod()
    {
        Vector3 currentWorldPos = this.transform.position;
        Vector3 viewPos = Camera.main.WorldToViewportPoint(currentWorldPos);
        if (viewPos.x > 1f)
        {
            //If the object is a bullet destroy it, the bullet has the "Bullet" tag,
set in inspector
            if (this.CompareTag("Bullet"))
            {
                Destroy(this.gameObject);
                return;
            }

            //Wrap to other side of screen
            this.transform.position = new Vector3(-currentWorldPos.x + 1,
currentWorldPos.y, currentWorldPos.z);
            return;
        }

        if (viewPos.x < 0f)
        {
            //If the object is a bullet destroy it, the bullet has the "Bullet" tag,
set in inspector
            if (this.CompareTag("Bullet"))
            {
                Destroy(this.gameObject);
                return;
            }

            //Wrap to other side of screen
            this.transform.position = new Vector3(-currentWorldPos.x - 1,
currentWorldPos.y, currentWorldPos.z);
            return;
        }

        if (viewPos.y > 1f)
        {
            //If the object is a bullet destroy it, the bullet has the "Bullet" tag,
set in inspector
            if (this.CompareTag("Bullet"))
            {
                Destroy(this.gameObject);
                return;
            }

            //Wrap to other side of screen
            this.transform.position = new Vector3(currentWorldPos.x,
currentWorldPos.y, -currentWorldPos.z + 1);
            return;
        }

        if (viewPos.y < 0f)
        {
            //If the object is a bullet destroy it, the bullet has the "Bullet" tag,
set in inspector
            if (this.CompareTag("Bullet"))
            {
                Destroy(this.gameObject);
                return;
            }
```

```
            //Wrap to other side of screen
            this.transform.position = new Vector3(currentWorldPos.x,
currentWorldPos.y, -currentWorldPos.z - 1);
            return;
        }
    }
}
```

Bullet (unchanged this lab) (Yes it's empty, I coded Bullet's features elsewhere)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Bullet : MonoBehaviour
{
    // Start is called before the first frame update
    void Start()
    {

    }

    // Update is called once per frame
    void Update()
    {

    }
}
```

AutoDestroy (unchanged this lab)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class AutoDestroy : MonoBehaviour
{
    private float lifetime = 2.85f;

    // Start is called before the first frame update
    void Start()
    {
        lifetime = Random.Range(1.15f, 3.45f);
```

```
    }

    void Awake()
    {
        StartCoroutine(ProcessLifetime());
    }

    private IEnumerator ProcessLifetime()
    {
        yield return new WaitForSeconds(lifetime);
        Destroy(this.gameObject);
    }
}
```