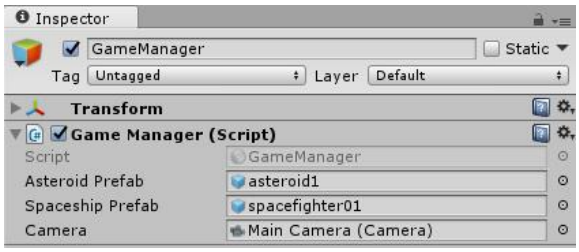
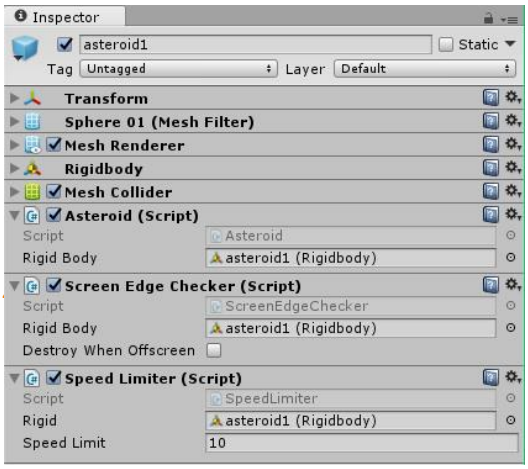
CT3536 Unity3D Lab 6
Sample Solution




```
public class GameManager : MonoBehaviour {

    // inspector settings
    public GameObject asteroidPrefab, spaceshipPrefab;
    //

    // class-level statics
    public static GameManager instance;
    public static int currentGameLevel;
    public static Vector3 screenBottomLeft, screenTopRight;
    public static float screenWidth, screenHeight;
    //

    // Use this for initialization
    void Start () {
        instance = this;
        Camera.main.transform.position = new Vector3 (0f, 30f, 0f);
        Camera.main.transform.LookAt (Vector3.zero, new Vector3 (0f, 0f, 1f));
        currentGameLevel = 0;

        StartNextLevel ();
        CreatePlayerSpaceship ();
    }

    public static void StartNextLevel() {
        currentGameLevel++;

        // find screen corners and size, in world coordinates
        // for ViewportToWorldPoint, the z value specified is in world units from the camera
        screenBottomLeft = Camera.main.ViewportToWorldPoint(new Vector3(0f,0f,30f));
        screenTopRight = Camera.main.ViewportToWorldPoint (new Vector3(1f,1f,30f));
        screenWidth = screenTopRight.x - screenBottomLeft.x;
        screenHeight = screenTopRight.z - screenBottomLeft.z;
        Debug.Log ("BottomLeft: "+screenBottomLeft);
        Debug.Log ("TopRight: "+screenTopRight);
        Debug.Log ("Width: " + screenWidth);
        Debug.Log ("Height: " + screenHeight);

        // instantiate some asteroids near the edges of the screen
        for (int i = 0; i < currentGameLevel * 2 + 3; i++) {
            GameObject go = Instantiate (instance.asteroidPrefab) as GameObject;
            float x, z;
            if (Random.Range (0f, 1f) < 0.5f)
                x = screenBottomLeft.x + Random.Range (0f, 0.15f) * screenWidth; // near the left edge
            else
                x = screenTopRight.x - Random.Range (0f, 0.15f) * screenWidth; // near the right edge
            if (Random.Range (0f, 1f) < 0.5f)
                z = screenBottomLeft.z + Random.Range (0f, 0.15f) * screenHeight; // near the bottom edge
            else
                z = screenTopRight.z - Random.Range (0f, 0.15f) * screenHeight; // near the top edge
            go.transform.position = new Vector3(x, 0f, z);
            go.GetComponent<Asteroid> ().SetScale (0.08f, 0.12f);
        }

    }

    public static void CreatePlayerSpaceship() {
        // instantiate the player's spaceship
        GameObject go = Instantiate (instance.spaceshipPrefab) as GameObject;
        go.transform.position = Vector3.zero;
    }
}
```

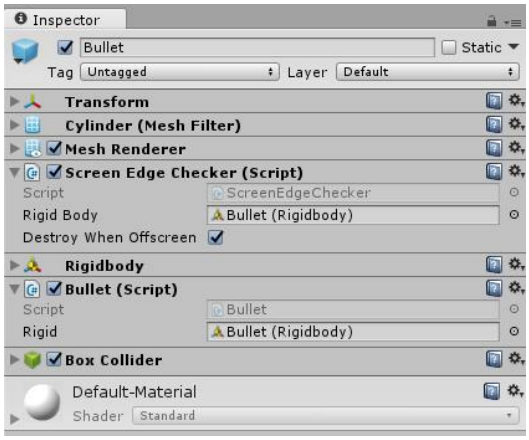
```
public class Asteroid : MonoBehaviour {

    // inspector settings
    public Rigidbody rigidBody;
    //

    // Use this for initialization
    void Start () {
        // randomise velocity
        rigidBody.velocity = new Vector3(Random.Range(-10f,10f),
          0f, Random.Range (-10f, 10f));
        rigidBody.angularVelocity = new Vector3(Random.Range(-4f,
           Random.Range (-4f, 4f), Random.Range (-4f, 4f));
    }

    public void SetScale(float min, float max) {
        transform.localScale = new Vector3(Random.Range(min,max),
           Random.Range(min,max), Random.Range(min,max));
        rigidBody.mass = transform.localScale.x *
           transform.localScale.y * transform.localScale.z;
    }

    void OnCollisionEnter(Collision collision) {
        if (!collision.gameObject.name.Contains("asteroid")) {
            Spaceship ss = collision.gameObject.GetComponent<Spaceship> ();
            if (ss != null && ss.isInvulnerable)
                return;

            // we've collided with something other than another asteroid
            Destroy(collision.gameObject); // if it's the player spaceship, the Spaceship script's OnDestroy
will look after re-creating it
            Destroy(this.gameObject);

            if (rigidBody.mass > 0.00015f) {
                float minScale = rigidBody.mass * 50f;
                float maxScale = minScale * 2f;
                for (int i = 0; i < 3; i++) {
                    GameObject go = Instantiate (GameManager.instance.asteroidPrefab) as GameObject;
                    go.transform.position = transform.position;
                    go.GetComponent<Asteroid> ().SetScale (minScale, maxScale);
                }
            }
        }
    }
}
```



```
public class ScreenEdgeChecker : MonoBehaviour {

    // inspector settings
    public Rigidbody rigidBody;
    public bool destroyWhenOffscreen = false;
    //

    // Use this for initialization
    void Start () {
        // start periodically checking for being off-screen
        InvokeRepeating ("CheckScreenEdges", 0.1f, 0.1f);
    }

    private void CheckScreenEdges() {
        Vector3 pos = transform.position;
        Vector3 vel = rigidBody.velocity;
        float xTeleport = 0f, zTeleport = 0f;

        if (pos.x < GameManager.screenBottomLeft.x && vel.x <= 0f)
            xTeleport = GameManager.screenWidth;
        else if (pos.x > GameManager.screenTopRight.x && vel.x >= 0f)
            xTeleport = -GameManager.screenWidth;

        if (pos.z < GameManager.screenBottomLeft.z && vel.z <= 0f)
            zTeleport = GameManager.screenHeight;
        else if (pos.z > GameManager.screenTopRight.z && vel.z >= 0f)
            zTeleport = -GameManager.screenHeight;

        if (xTeleport != 0f || zTeleport != 0f) {
            if (destroyWhenOffscreen)
                Destroy (this.gameObject);
            else
                transform.position = new Vector3 (pos.x + xTeleport, 0f, pos.z + zTeleport);
        }

    }
}
```

```
public class SpeedLimiter : MonoBehaviour {

    // inspector settings
    public Rigidbody rigid;
    public float speedLimit = 5f;
    //

    // Update is called once per frame
    void FixedUpdate () {
        float spd = rigid.velocity.magnitude;
        if (spd > speedLimit)
            rigid.velocity *= speedLimit / spd;
    }
}
```

```
public class Bullet : MonoBehaviour {

    // inspector settings
    public Rigidbody rigid;
    //

    // Use this for initialization
    void Start () {
        rigid.velocity = transform.forward * 30f;
    }

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

    }
}
```



```
public class Spaceship : MonoBehaviour {

    // inspector settings
    public Rigidbody rigidBody;
    public GameObject bulletPrefab;
    //

    // public member data
    [HideInInspector] public bool isInvulnerable = true;
    //

    // private member data
    private float lastFiredTime = 0f;
    //

    void Start() {
        Invoke ("MakeVulnerable", 2f);
    }

    private void MakeVulnerable() {
        isInvulnerable = false;
    }

    // Update is called once per frame
    void FixedUpdate () {
        if (Input.GetKey(KeyCode.UpArrow))
            rigidBody.AddForce(transform.forward * (rigidBody.mass * Time.fixedDeltaTime *
500f));

        if (Input.GetKey(KeyCode.LeftArrow))
            rigidBody.AddTorque(-transform.up * (rigidBody.mass * Time.deltaTime * 500f));
        else if (Input.GetKey(KeyCode.RightArrow))
            rigidBody.AddTorque(transform.up * (rigidBody.mass * Time.deltaTime * 500f));

        // firing is only allowed at most once per 0.25 seconds
        if (Input.GetKey (KeyCode.Space) && lastFiredTime + 0.25f <= Time.time) {
            lastFiredTime = Time.time;
            FireBullet ();
        }
    }

    void OnDestroy() {
        GameManager.CreatePlayerSpaceship();
    }

    private void FireBullet() {
        GameObject go = Instantiate(bulletPrefab);
        go.transform.position = transform.position + transform.forward*3f;
        go.transform.rotation = transform.rotation;
    }
}
```