# Revision Worksheet

You have been engaged in a team developing an intriguing Tank assault game. Other developers in your team have supplied you with the assets required and an initial game setup. You are required to follow the steps below to complete a robust and scalable OOP implementation of the game.

# Part 1 – Game Setup and Movement

1. Extract and open in Unity the project folder **A01\_Tanks\_Materials**.
2. Ensure an aspect ratio of 16:9 with an Orthographic Main Camera of Size=5
3. Create a Sprites folder. Import the Green Grass Background image in P:\BSc\DS\OOPMM and add it to your scene. Resize the Sprite using Pixels-per-Unit so it covers the whole Camera Viewport.
4. Add the **PlayerTank** Prefab to the Scene and add the **PlayerMovement** script as a new component.
5. Add the **PlayerGun** script to the PlayerTank gameobject and populate appropriately the script’s inspector attributes with **PlayerMissile** prefab and **FireTransform** child gameobject.
6. The Player tank can be moved using the W and S key. You are required to implement tank rotation using A and D keys. Refer to code from GetMovementInput() to add a similar body to a new method named: GetRotationInput().
7. Change the Key Codes from **W** and **S** to:
   1. **D** for positiveKey
   2. **A** for negativeKey.
8. Test out your code by making sure the PlayerTank moves and rotates using A,W,S,D keys.
9. Create a new **DestroyExplosion** script attached to the Explosion prefab. Write a simple Coroutine to destroy the game object itself (Explosion) after 3 seconds of being spawned in the scene.

# Part 2 - Add a Rotating Fixed Enemy (Turret)

Modify the **FixedEnemy** **GameObject** and **Script** according to the following guidelines:

1. For the FixedEnemy game object, replace the current sprite with **tank\_darkLarge** from the Sprites folder.
2. Add a **Rigidbody2D** component which would allow the tank to rotate, but is **not effected** by any Physics forces.
3. Add an appropriate Collider2D component.
4. Create an EnemyScript script component and implement the following functionality:
   1. Create attributes which are visible and can be changed from the inspector as follows:
      1. **speed** with a default of **2f**
      2. **rotationSpeed** with a default of **180f**
   2. Create a **protected** attribute **playerTank** of type **GameObject** to allow the class to make reference to the PlayerTank.   
        
      (Hint: You can create a “Player” Tag, assign it to PlayerTank and then use FindGameObjectWithTag("Player") in FixedEnemy.cs Update().
   3. Research and use **LookRotation()** and **Slerp()** to rotate the Fixed Enemy towards the PlayerTank on the Z-Axis.
5. Open the EnemyGun Script and change the Coroutine to spawn an Enemy Missile at a random time between 0.5 and 4 seconds.
6. Add the EnemyGun script as a component in your newly created fixed enemy.

# Task 6 – Add a moving Enemy Tank and Player Score

1. Modify the Start() and Update() methods in the **FixedEnemy** class to **virtual**.
2. Create a new GameObject named EnemyTank. Use **tank\_red** sprite for your EnemyTank.
3. Add a child GameObject making use of the **tankRed\_barrel3** sprite. Position the child sprite on top of the parent to simulate a barrel for your enemy.
4. Add appropriate Rigidbody2D and colliders to the newly created EnemyTank.
5. Add a new **EnemyTank** script and class which **inherits** from FixedEnemy class. Start() and Update() methods need to be set to **override** the same base class methods.
6. Call the parent’s (FixedEnemy) Start() and Update() methods **from** the respective Start() and Update() child methods.
7. Apart from calling the parent Update() method, add code to make the enemy tank seek and move in the direction of the player. Your new line of code should transform the turret behaviour into a moving tank. Hint: Research **Vector3.MoveTowards()**
8. Add the EnemyGun script as a component to the EnemyTank game object, and fill in the inspector fields appropriately.
9. Create a UI text and a scalable Canvas to display the Score.
10. Add the Health script as a component to PlayerTank, FixedEnemy and EnemyTank.
11. Add an empty gameobject to your scene. Rename it to: GameController.
12. Add the GameManager script as a component to the new GameManager gameobject.
13. In the inspector, link the ScoreText to your GameManager script.
14. Make sure you save your work, copy to a compressed ZIP file and submit through Moodle.