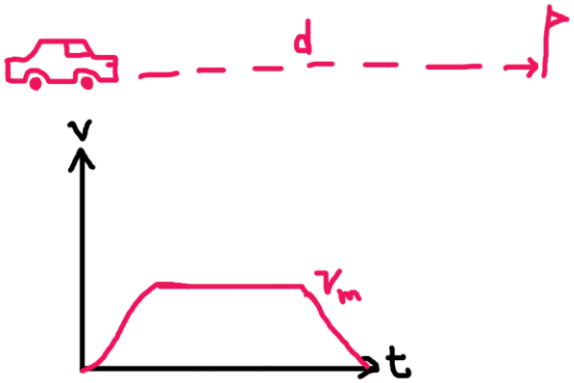
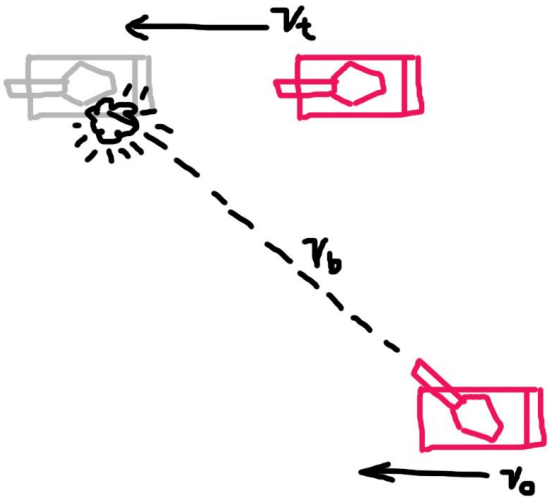
Test 1: Drive a vehicle to the target, all movement is done on a single axis. Accelerate the vehicle up to maxSpeed and come to a full stop upon reaching. Fill up the method with your logic.



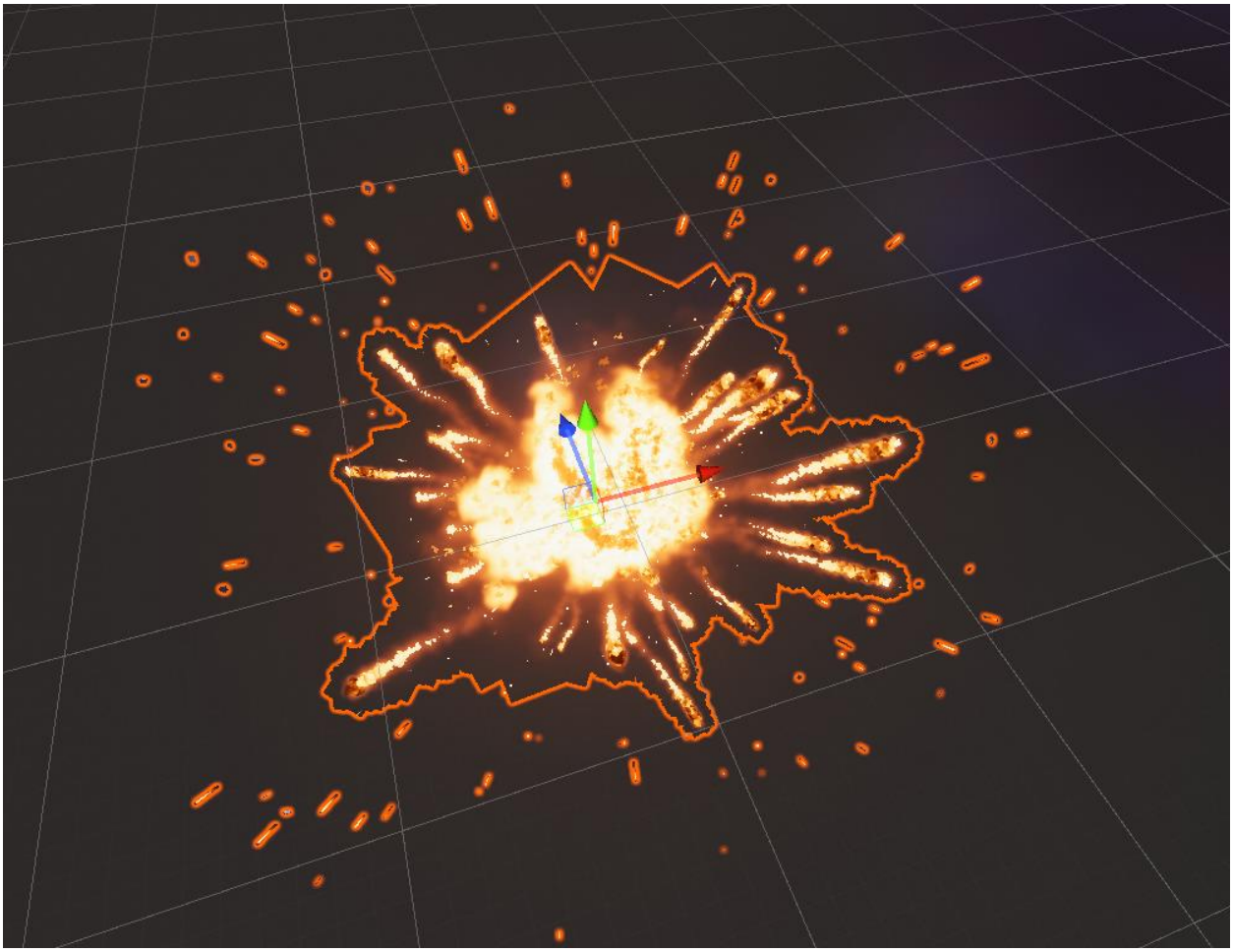

```
enum EngineState
{
    /// <summary>
    /// On returning this, currentSpeed += forwardAcceleration
    /// </summary>
    Accelerate,

    /// <summary>
    /// On returning this, the vehicle maintains currentSpeed.
    /// </summary>
    Idle,

    /// <summary>
    /// On returning this, currentSpeed -= brakeDeceleration. (Up to currentSpeed == 0)
    /// </summary>
    Brake,
}

/// <summary>
/// Called every second. Accelerate to target, and come to a full stop upon reaching.
/// Make sure currentSpeed does not exceed maxSpeed. When distanceToTarget ~ 0, make sure currentSpeed is ~ 0 too.
/// </summary>
/// <param name="distanceToTarget">Distance to target in meters, initially always > 0</param>
/// <param name="currentSpeed">Current speed towards target, in m/s</param>
/// <param name="forwardAcceleration">Acceleration applied when returning EngineState.Accelerate, in m/s^2</param>
/// <param name="brakeDeceleration">Deceleration applied when returning EngineState.Brake, in m/s^2</param>
/// <param name="maxSpeed">Max speed limit, in m/s</param>
/// <returns>EngineState, move and stop the vehicle by returning the right EngineState on each tick</returns>
EngineState DetermineEngineState(float distanceToTarget, float currentSpeed, float forwardAcceleration, float brakeDeceleration, float maxSpeed)
{
    // TODO: Your code here
}
```

Test 2: Calculate the interception point for a projectile towards a moving target. Fill up the method with your logic.




```
/// <summary>
/// Called each time your tank wants to fire. Calculate and return interceptPosition so that it is able to rotate the turret
/// and fire at a moving target.
/// </summary>
/// <param name="selfPosition">Current position of your tank</param>
/// <param name="selfVelocity">Current velocity of your tank</param>
/// <param name="targetPosition">Position of the target</param>
/// <param name="targetVelocity">Velocity of the target</param>
/// <param name="bulletSpeed">When fired, this is the speed at which the bullet moves</param>
/// <param name="interceptPosition">The expected position at which the bullet will impact the moving target. Calculate this.</param>
/// <returns>Return True if it is possible to intercept. False otherwise (if the target is moving too fast)</returns>
bool CalculateInterceptPosition(Vector3 selfPosition, Vector3 selfVelocity, Vector3 targetPosition, Vector3 targetVelocity, float
bulletSpeed, out Vector3 interceptPosition)
{
    // TODO: Your code here
}
```

Test 3: To your best ability, recreate this explosion in VFX Graph



Test 4: To your best ability, recreate this smoldering effect in Shader Graph