Assignment 6
Vaibhav Sharma
200365101

1.
    a. When driving to a red light or stop Sign, I take foot off gas pedal and let the car go through some engine braking before I press the brake myself. Depending on my current velocity, I will apply breaks around 70m to 100m before the stop for my car ABS to calculate the braking distance and stop the car without locking the wheel. To Summarize, I will allow the car to decelerate first and then hit the breaks X distance before the red light where X depends on how fast I am going. Usually I will let my car to slow down 30 -35 km/hr before braking to make it stop smoothly(Assuming I was initially driving at 50 km/h before braking).
    b.

```
# Apply forces

    target = Vector(0, 0, 5)
    player = Vector(cam.position.x, cam.position.y, cam.position.z)
    proximity = target.sub(player).mag()

    targetfinal =  player.sub(target).mag()

    seek(mobject, target, 100)

    if proximity <= targetfinal:
        seek(mobject, player, 3)
```

```
def mySeek(mobject, target , maxForce):
    zero = Vector(0,0,0)
    engineBraking = zero.sub(own.position)
    engineBraking = engineBraking.div(0.5)



    steer = engineBraking.sub(mobject.vely)
    if steer.x > maxForce:
        steer.x = maxForce
        print(maxForce)
    if steer.y > maxForce:
        steer.y = maxForce
    if steer.z > maxForce:
        steer.z = maxForce

    mobject.applyForce(steer)

    desired = target.sub(own.position)
    desired = desired.div(2.0)
    steer = desired.sub(mobject.vely)

    if steer.x > maxForce:
        steer.x = maxForce
        print(maxForce)
    if steer.y > maxForce:
        steer.y = maxForce
    if steer.z > maxForce:
        steer.z = maxForce

    mobject.applyForce(steer)
```

In my own mySeek function- i am applying engine braking before bringing it to complete stop using reynolds arrival method

2.

a. Good paper!
b. In my Arrival method, I am applying some engine braking as my scenario requires our model to reflect a model of a car. In reynolds method, he is forming a circle around the target and using steering force to slow down the car when it reaches inside that area of the circle. The Desired velocity - current velocity will result into a Vector pointing the opposite direction which will enforce slowing down.

3.

a. Reynold's discussed path following where he describes how the object stays on the correct path or how it comes back to the correct path if it wanders off.
    i. In path following - the object always check it's future location and checks if the future location is on correct path
    ii. IF - the object if on correct path then it does not steer and just go forward
    iii. Else - it finds a perpendicular line to the path and choose the shortest path
        1. We will call the perpendicular joining the path - pointA
        2. From pontA , we then look for the future location of the object while staying on the path, we will call it pointB
        3. Then we mark point B as target and choose our current location as current and simply apply the steering force for out object to get to the correct path.
b. Just writing a rough pseudo code as I dont have the code for the path

```
#furturePosition is a function which returns
    # true if mobject is on the path
    # false if mobject not on the path
#furturePositionlocation is a fucntion which returns Vector
    # the future location of the mobject from a given point
if (futurePosition(mobject)):

    # ...


else:
    pointA = #prependicular on the path
    pointB = Vector(furturePositionlocation(mobject, pointA))
    target = pointB
    seek(mobject, target, 3000)
```

C. my wandering algorithm is very similar to what reynold's explained so if implemented correctly, it should get the object to the correct path. However, I think looking

for two future locations from the target is a good idea in case path includes a turn right after our target position