MEE 340 Group Task 2 (Team 10)

Problem 1

In [1]:
```
import numpy as np
import math

def grouptask2(maxh=0.675,minh=0.03,dh=0.015,height=0.75,top=1.5,bottom=0.8,g=9.81,nu=1.05*(10**-6),
               minorLoss=0.35 + 0.5 + 1000*0.05/9.81,alpha=0.05):

    #Minor loss is a sum of losses from area contraction, sharp entrance, and ball valve, respectively
    #Default values are currently set to solve Problem 1
    #For Problem 3, change in dh is required when calling the function

    minh = minh - 0.001                     #correction for np.arange to include endpoint
    h = np.arange(maxh,minh,-dh)            #Define array to contain all values of height (h) given dh

    deltaRMax = (top-bottom)/2              #Define array to contain radii toward finding volume in each subdivision
    radii = []

    for i in h:
        radii.append((bottom/2)+((deltaRMax/height)*i))


    Vol = []                                #Calculate Volume for each height division

    for i in radii:
        Vol.append(math.pi*(i**2)*dh)


    #Find Vave (Velocity at exit of piping) for each volume division
    #Note that major losses and pipe dimensions are only coming from the problems
    #May need adjustments for actual lab

    L1 = 8.00
    D1 = 0.05
    L2 = 12.00
    D2 = 0.03

    Vave = []

    #Calculate major losses and select Vave for each h and volume division

    for x in h:
        Vguess = math.sqrt(2*g*x)    #start off with an ideal jet exit velocity
        flag = 1

        while (flag == 1):
            Re1 = Vguess*D1/nu
            Re2 = Vguess*D2/nu

            #For the purposes of shortening time for final calculation, the Haaland Equation will be implemented.
            #Solve for F1,F2

            F1 = (1/-1.8*math.log10(6.9/Re1))**2
            F2 = (1/-1.8*math.log10(6.9/Re2))**2

            Vresult = math.sqrt((2*g*x)/(1+F1*L1/D1+F2*L2/D2+minorLoss))  #Update new V for error testing
            errorTest = (Vguess-Vresult)/(Vresult)

            if (errorTest > -alpha and errorTest < alpha):  #error estimation: if change in height is sufficiently low, use current V as Vave for the height
                Vave.append(Vresult)
                flag = 0
            else:
                Vguess = Vresult #Reset/Update Vguess


    #Obtain deltaT array to compute time to deplete each division

    area = (math.pi/4)*(D2**2)
    deltaT = []
    i = 0  #index counter

    for j in Vave:
        deltaT.append(Vol[i]/(j*area))
        i = i+1


    #Find total time (in seconds) to deplete tank by summing elements in deltaT

    total = sum(deltaT)
    print(total)
```

In [2]:
```
grouptask2()
```

12936.50700712607

The total time to empty the reservoir between the two levels is **12937 sec. (3.6 hours)**

Problem 2

In [3]:
```
#Problem 2 presents a rectangular reservoir
#L = 1.2
#W = 0.7
#H = 0.6
#let dh be the same (0.015)

def grouptask2_rect(maxh=0.6,minh=0,dh=0.015,length=1.2,width=0.7,g=9.81,nu=1.05*(10**-6),
               minorLoss=0.35 + 0.5 + 1000*0.05/9.81,alpha=0.05):

    #Minor loss is a sum of losses from area contraction, sharp entrance, and ball valve, respectively
    #Default values are currently set to solve Problem 2

    minh = minh                     #correction for np.arange to include endpoint
    h = np.arange(maxh,minh,-dh)            #Define array to contain all values of height (h) given


    Vol = []                                #Calculate Volume for each height division

    for i in h:
        Vol.append(length*width*dh)


    #Find Vave (Velocity at exit of piping) for each volume division
    #Note that major losses and pipe dimensions are only coming from the problems
    #May need adjustments for actual lab

    L1 = 8.00
    D1 = 0.05
    L2 = 12.00
    D2 = 0.03

    Vave = []

    #Calculate major losses and select Vave for each h and volume division

    for x in h:
        Vguess = math.sqrt(2*g*x)    #start off with an ideal jet exit velocity
        flag = 1

        while (flag == 1):
            Re1 = Vguess*D1/nu
            Re2 = Vguess*D2/nu

            #For the purposes of shortening time for final calculation, the Haaland Equation will be implemented.
            #Solve for F1,F2

            F1 = (1/-1.8*math.log10(6.9/Re1))**2
            F2 = (1/-1.8*math.log10(6.9/Re2))**2

            Vresult = math.sqrt((2*g*x)/(1+F1*L1/D1+F2*L2/D2+minorLoss))  #Update new V for error testing
            errorTest = (Vguess-Vresult)/(Vresult)

            if (errorTest > -alpha and errorTest < alpha):  #error estimation: if change in height is sufficiently low, use current V as Vave for the height
                Vave.append(Vresult)
                flag = 0
            else:
                Vguess = Vresult #Reset/Update Vguess


    #Obtain deltaT array to compute time to deplete each division

    area = (math.pi/4)*(D2**2)
    deltaT = []
    i = 0  #index counter

    for j in Vave:
        deltaT.append(Vol[i]/(j*area))
        i = i+1


    #Find total time (in seconds) to deplete tank by summing elements in deltaT

    total = sum(deltaT)
    print(total)
```

In [4]:
```
grouptask2_rect()
```

11732.694329380374

Total time to drain in rectangular reservoir is decreased compared to Problem 1. (~3.3 hours)

Problem 3

In [5]:
```
#Adjust Problems 1 & 2 to obtain draining times with various time increments

#Max height in Problem 1 is 0.75 m
grouptask2(dh=0.75*0.005)
```

12650.567615681995

Problem 1 w/ 0.5% height increments yields **12651 sec (~3.5 hours)**.

In [6]:
```
grouptask2(dh=0.75*0.018)
```

12602.680225301023

Problem 1 w/ 1.8% height increments yields **12603 sec (~3.5 hours)**.

In [7]:
```
#Max height in Problem 2 is 0.6 m
grouptask2_rect(dh=0.6*0.005)
```

12269.34069536582

Problem 2 w/ 0.5% height increments yields **12269 sec (~3.4 hours)**.

In [8]:
```
grouptask2_rect(dh=0.6*0.018)
```

12356.491369765543

Problem 2 w/ 1.8% height increments yields **12356 sec (~3.4 hours)**.