Questions

In this assignment you will be modelling a Water Tower problem. However the main goal is for you to try some introductory programming using Python. You will have access to much test data however your solution will be tested against additional trials. Your code will need to pass all tests to achieve full marks.

Water towers are among the simplest yet ingenious methods to distribute water having been in use in some form since ancient times. More than simply calculating capacity or flow rate however, we can model how a tank can be filled and emptied. We will model a scaled down version of this problem. In our model, our water tank will have a single water flow inward, and a single flow outward and be perfectly cylindrical. The equation (1) below describe the volume of the water tower and difference equations (2) (3) mentioning the changes for volume and height of the water tower.

$$V_t = \pi H r^2 \qquad (1)$$

$$V_{n+1} = V_n + (f_{in} - f_{out})\Delta t \qquad (2)$$

$$h_{n+1} = h_n + \frac{(f_{in} - f_{out})\Delta t}{\pi r^2} \qquad (3)$$

Where

- V_t — total volume of the water tower in cubic meters
- V_{n+1} , h_{n+1} — volume and height of water in tower at (n+1) in cubic meters and meter respectively
- f_{in} , f_{out} — the flow rate in and out of water tower respectively in cubic meters per second
- H — total height of the water tower in meters
- $\Delta t = (t_{n+1} - t_n)$ — time step in seconds

You are tasked to write a function called `trackFlow` which when given some initial conditions iteratively simulates the contents of the tower. By 'iteratively' we mean you will update the volume of water in the tank repeatedly over numerous time-intervals.

More specifically, the tank will be filled up until some time t_{open} at which point the outward flow is activated (instantly).

Your function will accept the following arguments (in the following order):

- f_{in} — The flow-rate in (cubic meters per second)
- f_{out} — The flow rate out (cubic meters per second)
- r — The radius of the tower (meters)
- H — The height of the tower (meters)
- h — The height of the initial water-level (meters)
- t_{max} — The maximum time allowed to simulate the system (seconds)
- t_{open} — The time when the outwards flow opens (seconds)

Your calculations should continue until either $t > t_{max}$, $h > H$ *or* $h < 0$ whichever comes first.

Assume
- The time-step = 0.1 sec
- The inward flow is open at t = 0.0 sec

Your function should return the following outputs in the same order
- A list of volume values (in cubic meters)
- A list of water-height values (in meters)
- A list of time-stamps at which the values were calculated (in seconds)

Note: *Do not print the lists in a formatted manner, simply return them at the end of calculations.*

All values during the computation must be rounded to 10 decimal places to avoid approximation problem. See details at:
https://docs.python.org/2/tutorial/floatingpoint.html

In addition, all output values are to be rounded to 2 decimal places when they are appended to the output result-arrays. See the `round()` function for more information.

Your submission will be tested against a sample solution with numerous test-cases in addition to those presented below.

Sample testing data:

```
#Steady-state after 1s
Volumes, Heights, Times = testFlow(1, 1, 1, 10, 0, 3, 1)

Volumes = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8,
0.9, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,
1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]

Heights = [0, 0.03, 0.06, 0.1, 0.13, 0.16, 0.19, 0.22,
0.25, 0.29, 0.32, 0.32, 0.32, 0.32, 0.32, 0.32, 0.32, 0.32,
0.32, 0.32, 0.32, 0.32, 0.32, 0.32, 0.32, 0.32, 0.32, 0.32,
0.32, 0.32, 0.32]

Times = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,
1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1,
2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0]

#Decreases until tank is empty
Volumes, Heights, Times = testFlow(1, 5, 1, 10, 0, 5, 2)

Volumes = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8,
0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0,
1.6, 1.2, 0.8, 0.4]

Heights = [0, 0.03, 0.06, 0.1, 0.13, 0.16, 0.19, 0.22,
0.25, 0.29, 0.32, 0.35, 0.38, 0.41, 0.45, 0.48, 0.51, 0.54,
0.57, 0.6, 0.64, 0.51, 0.38, 0.25, 0.13]

Times = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,
1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1,
2.2, 2.3, 2.4]

#Increases after out-flow is open to maximum time
Volumes, Heights, Times = testFlow(1, 0.5, 1, 10, 0, 3, 1)

Volumes = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8,
0.9, 1.0, 1.05, 1.1, 1.15, 1.2, 1.25, 1.3, 1.35, 1.4, 1.45,
1.5, 1.55, 1.6, 1.65, 1.7, 1.75, 1.8, 1.85, 1.9, 1.95, 2.0]

Heights = 0, 0.03, 0.06, 0.1, 0.13, 0.16, 0.19, 0.22, 0.25,
0.29, 0.32, 0.33, 0.35, 0.37, 0.38, 0.4, 0.41, 0.43, 0.45,
0.46, 0.48, 0.49, 0.51, 0.53, 0.54, 0.56, 0.57, 0.59, 0.6,
0.62, 0.64]

Times = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,
1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1,
2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0]
```

Marks distribution:

- Function definition including name, order of inputs and outputs, validation: 10 marks
- Functionality of the function: 30 marks