Read all of the following information before starting the exam:

Permitted on exam:

- Open book, open notes, open Python documentation, open internet (Google, etc.).
- All your code should be your own work. You are **encouraged** to copy/modify your existing code from homework and previous exams as necessary. This includes any example code uploaded to our Canvas site this semester, but not any other code that may have escaped to student files from other semesters.

Not permitted on exam:

- You **MAY NOT** use any form of technology to communicate with, send to, or receive information from another person (other than the instructor), during the exam.

Exam submission:

- You should create a folder called **EXF** to house all your files and then submit a zipped version of that folder.
- During the exam, you should SAVE YOUR WORK **often**!
- You will be uploading your exam solutions to the Canvas, **EXF** dropbox. You may upload multiple versions of your solution (so you can save early and often). We will grade the LAST submitted version of each problem.
- Given the fact that you will be submitting your exam from home, internet service may be variable. If you are having trouble submitting, please let me know with a text message (405)-742-8053. If you can't submit on CANVAS, you may email your .zip file to me at jim.smay@okstate.edu

Grading:

- The points awarded for each problem are specified.

- You should test your output using appropriate test values. We will grade based on code correctness, readability and output results (i.e., print to screen values and plots).

1. (25pts) Frictional losses in pipes:

   In our pipe network problems, we have used the Reynolds number, the Colebrook equation and the Darcy-Weisbach equation to calculate frictional losses in pipes with fully turbulent flow. Create a command line interface (CLI) program to calculate frictional losses in a pipe given the pipe diameter (d), pipe length (l), pipe roughness (e), fluid dynamic viscosity (μ), fluid density (ρ), and volumetric flow rate (Q). The flow may be anywhere from laminar to fully turbulent.

   The program should meet the following requirements:
   i) It should first prompt the user to select a units system (English or Metric).
   ii) It should prompt the user to specify fluid viscosity with water as default.
   iii) It should prompt the user to specify fluid density with water as default.
   iv) It should prompt the user to specify pipe diameter, length and volumetric flow rate.
   v) It should return Reynolds number, flow regime (laminar, transition, or turbulent), and the head loss.

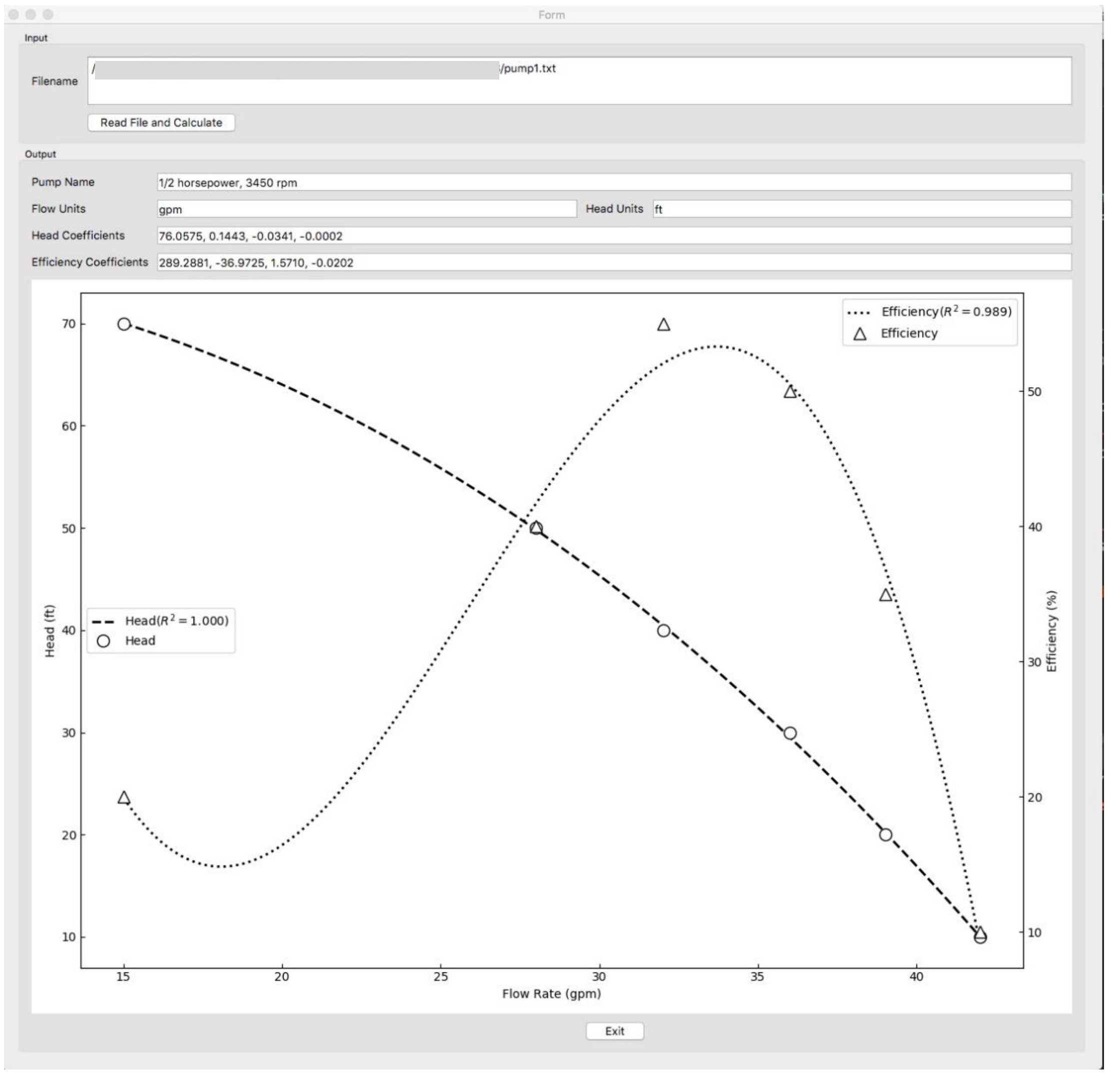
   All prompts and return values should be clearly labeled with proper units.

   Notes:
   1. User input from the command line can be achieved with `input()` method.
   2. The f for laminar flow can be found by: $f=64/Re$ ($Re<2000$)
   3. The f for transitional flow can be found by: $f=0.316 \cdot Re^{-0.25}$ (the Blasius equation, $3000<Re<10^5$)
   4. The f for turbulent flow can be found by the Colebrook equation ($Re>4000$)

2. (25 pts) Write a program to read and display information from a pump file.

Your program will have the following features:

i. It will have a graphical user-interface (GUI) for selecting the file and presenting the output.
ii. The `clicked` signal of the *Read File and Calculate* button will open a dialog box for searching the directories of your hard drive to navigate to location of the data file.
iii. Upon selecting the data file, the path to the data file is stored, the file is ***read, parsed*** **and** ***processed*** to produce the required output (as demonstrated in the screen capture below).
iv. The curves fitted to the pump capacity and efficiency data are third order polynomials showing minimal sum of squared errors with the coefficients displayed in the line edit widgets.
v. The plot should have labels on both y-axes, the x axis, and legends for each curve.



Filename: /[redacted]/pump1.txt

Pump Name: 1/2 horsepower, 3450 rpm

Flow Units: gpm   Head Units: ft

Head Coefficients: 76.0575, 0.1443, -0.0341, -0.0002

Efficiency Coefficients: 289.2881, -36.9725, 1.5710, -0.0202

3. (50 pts) The Air Standard Otto Cycle:
We have used steam tables to calculate properties of Rankine cycles where condensation and vaporization are inherent. In this problem, we want to use the ideal gas behavior of air as the working fluid in an air standard Otto cycle consisting of 4 thermodynamic processes:

1→2: an isentropic compression of the air as the piston moves from bottom dead center to top dead center. Recall from thermo: $\Delta_1 s_2 = 0 = \int_{T_1}^{T_2} \frac{c_v}{T} dT + R \ln \frac{v_2}{v_1}$; $\Delta_1 u_2 = \int_{T_1}^{T_2} c_v dT$ for ideal gas

2→3: a constant-volume heat transfer to the air from an external source while the piston is at top dead center. ${}_2 q_3 = \Delta_2 u_3 = \int_{T_2}^{T_3} c_v dT$ for ideal gas at constant volume

3→4: an isentropic expansion (power stroke). $\Delta_3 s_4 = 0 = \int_{T_3}^{T_4} \frac{c_v}{T} dT + R \ln \frac{v_4}{v_3}$; $\Delta_3 u_4 = \int_{T_3}^{T_4} c_v dT$

4→1: a constant-volume heat rejection while the piston is at bottom dead center. ${}_4 q_1 = \Delta_4 u_1 = \int_{T_1}^{T_4} c_v dT$

Always, for air as an ideal gas: $PV = nRT$

Write an object-oriented program with GUI in the Model-View-Controller pattern to model the air standard Otto cycle given inputs of: *Cylinder Volume, Initial Pressure, Compression Ratio,* and *Maximum Temperature*. Your program should allow the user to specify English or Metric units and display results and inputs with appropriate units. You may, however, choose to always work in Metric units in the model and only modify the view to display English or Metric units.

Notes:
Your program should output T_1, T_2, T_3, T_4 and cycle efficiency in addition to plots of *p* vs. *v* and *T* vs. *s*.
In the air standard Otto cycle, air behaves as an ideal gas with variable specific $c_p(T)=c_v(T)+R$.