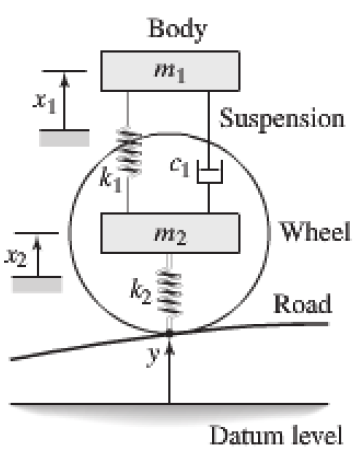
***Read all of the following information before starting the exam:***

* The exam is open book, open notes, open Python documentation, open internet, etc.
* You **MAY NOT** use any form of technology to communicate with, send to, or receive information from another person (e..g., classmates, other instructors, anonymous persons on the internet). HOWEVER, you may submit written questions to the professor or TA by email.
* You will be uploading your exam solutions as a single .zip file to the Canvas, ***EX3*** 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.
* You may use any Python packages you wish!
* You should ensure that your programs accept reasonable user inputs and can display new results if the user decides to change a GUI value (within reason).
* The graders have the discretion to award up to 10 bonus points for well-organized and documented code. Please be organized and detailed and you shall be rewarded!

1. (35 points)

Write a program to model the vertical position of a car that is rolling along at constant speed and hits a linear ramp to a new level of y=6 inches.

* 1. Create an object-oriented python program to model the vertical position of the car body as a function of time using a single wheel and suspension as illustrated. Note: we have nearly solved this problem in Quarter Car Model.py of Week 6 lecture, but not in OOP and not for a linear ramp.



* 1. Create a graphical user interface that allows the user to modify: the car speed, the wheel and car body masses, the spring constants, the damping coefficient of the shock absorber and the angle of the linear ramp to the new level. Initialize your GUI with reasonable values for the masses, spring constants, damping coefficient and ramp angle. Your GUI should display a matplotlib plot of the car body vertical position as a function of time (of sufficient length to capture all the bouncing of the car that may occur). Changes to the physical parameters through the GUI should be incorporated into the model and the plot updated with, say, the push of a calculate button.

1. (35 pts)

In the week 11 materials on canvas, you were given HW8-final.zip. In this archive is a folder named ‘Rankine-rev3-final’ that contains the GUI and OOP version of our Rankine program. Re-write the Rankine program in the **Model-View-Controller design pattern** by writing a controller class that contains a rankine object (i.e., the model) and a view object, which is responsible for displaying the data from the model. Modify the GUI, as necessary, such that only the controller is instantiated and used in the application.

Hints for possible structure of your program:

The rankine class should only contain a constructor with the necessary class variables.

The rankineCycleController class should have these objects defined in its constructor:

self.model=rankine()

self.view=rankineCycleView()

and the functions that operate on the model:

def set(…

def calc\_efficiency(…

def buildVaporDomeData(…

def buildDataForPlotting(…

and the functions that operate on the view:

def plot\_cycle\_XY(…

def print\_summary(…

def get\_summary(…

def updateView(self, displayWidgets=None)

The rankineCycleView class should have these functions:

def print\_sumarry(…

def get\_summary(…

def plot\_cycle\_XY(…

def updateView(self, displayWidgets=None)

1. (30 pts)

Write a program to read and display information from a truss design file. The design file will be provided on canvas along with this assignment. Your program should look something like the figure below, but this one was made with OpenGL graphics, whereas you should use the QGraphicsView etc. for the figure. Note: you may want to use/reuse the code from our pipe network program.

