ENGR 16100
Homework 3 Python Programming Exercises

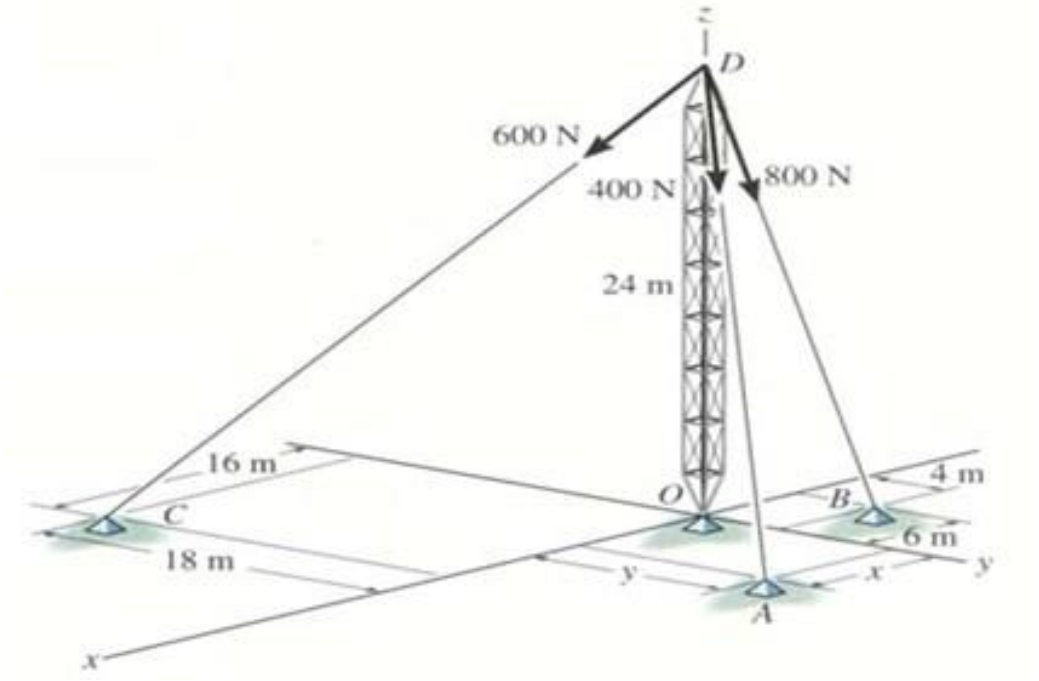
Individual Assignment: See the course syllabus for a definition of what this constitutes.

Notes:
1) Failure to submit a programming assignment with the correct filename may result in no points earned.
2) Be aware that the examples are NOT meant to be comprehensive; there are other legitimate output requirements that are not illustrated in the examples. It is recommended that you create your own additional examples for testing purposes.

Programming Problem 1 (of 2)

Learning Objectives: Utilize conditional if-elif-else statements while programming in Python; Output data from a function to the screen in Python; Import modules; Apply course code standard.

Background:
When structural engineers design a restraining system for a tall, thin structure such as a cell tower, there are several conditions that must be checked to ensure safe operation of the restraints. One such condition is called excessive overturning moment. To understand this condition, consider the restraining system shown below, where three cables each exert a tensile force on point *D* at the top of the tower.



Let $\vec{F}_{total}$ be the total force exerted on the tower at point *D* by the cables. You know from previous class work that $\vec{F}_{total}$ has a vector component that is parallel to the tower axis *OD*; you also know that $\vec{F}_{total}$ has a vector component that is perpendicular to this axis. (Recall the fire truck problem from Homework 2 for another example of this idea.) Suppose that $\left|\vec{F}_{perp}\right|$ is the magnitude of the vector component that is perpendicular to the tower axis; also, suppose that *h* is the height of the tower, *i.e.,* the distance from *O* to *D*. (Note that $\left|\vec{F}_{perp}\right|$ is the magnitude of the vector component $\vec{F}_{perp}$; by definition, this magnitude cannot be less than zero.) Define the *overturning moment magnitude* as follows:

$$M_{overturning} = \left|\vec{F}_{perp}\right| * h$$

As $M_{overturning}$ becomes large, it becomes likely that the tower's connection to the ground at point *O* will break, causing the entire tower to fall over ("overturn"). Thus, when designing the restraining system, engineers must make sure that the overturning moment magnitude stays below an upper bound M_{max}.

While the figure shows a specific situation involving particular values for the cable force magnitudes and locations of the cable anchor points, structural engineers must be able to analyze many different scenarios for force magnitudes and anchor points. Thus, having a program that can quickly check the overturning moment requirement for any given set of forces and locations is valuable.

Instructions:

Your task is to write a Python program that checks whether a given restraining system is safe according the requirement that $M_{overturning} < M_{max}$. You can make the following assumptions about the restraining system:

- The restraining system will always have exactly three cables. Each cable will always run from the ground (the *xy*-plane) to the top of the tower (point *D*).
- The anchor points (call them *A, B, C*) can be anywhere in the *xy*-plane, but no two anchor points will ever be at the same point in the plane.
- The tensile force in each cable will always be a positive value.

Your program should accept the following input values from the user:

- The height *h* of the cell tower (in meters);
- For each anchor point, the *x*-coordinate and *y*-coordinate (in meters) of the anchor point should be input individually, but **you must use a single list to store the coordinate values**. In other words, you cannot use six distinct floating point variables to store these values; all values should be contained within one list. Failure to use a list for this will result in loss of points.
    - There is no need for user input of the *z*-coordinates, since these will always be zero.
    - You may organize the list of coordinate values in any manner you deem appropriate, so long as all of the values are contained within a single list.
    - Hint: there are several different ways to create this list. You may wish to do an internet search for "python list methods" to get some ideas for doing this; there are also ways of doing this that do not use list methods. Any technique that creates an appropriate list will be accepted.
- For each cable, the magnitude of the tensile force that the cable exerts on the tower (in newtons).
- The upper bound M_{max} that the overturning moment magnitude must stay below to be safe (in newton-meters).

No error checking of the inputs is necessary; you may assume the user always inputs appropriate values. Your program should output the following information to the console:

- The value of the overturning moment magnitude $M_{overturning}$ (in newton-meters).
- A message that states whether or not the restraining system is safe.

See the example below for an illustration of how your inputs and outputs should look. Note that there are no rounding requirements on your outputs.

Save this program as `HW3_Prob1_login.py` (where *login* is <u>your</u> username) and submit it to the appropriate programming dropbox on Gradescope. Also, create a flowchart of the relevant algorithm for this problem, and include this flowchart in your PDF submission of the written portion of this assignment. Please note that the PDF for the written portion of the assignment will be submitted to its own dropbox -- do not submit this PDF to the programming dropbox or vice versa.

Example: (User inputs in bold)

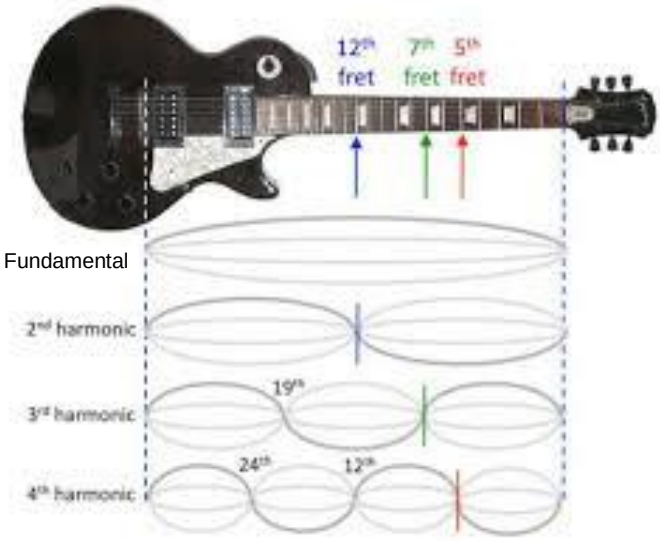
```
Enter the height (in meters): 24.0
Enter the x coordinate of anchor point A (in meters): 8.0
Enter the y coordinate of anchor point A (in meters): 8.0
Enter the x coordinate of anchor point B (in meters): -6.0
Enter the y coordinate of anchor point B (in meters): 4.0
Enter the x coordinate of anchor point C (in meters): 16.0
Enter the y coordinate of anchor point C (in meters): -18.0
Enter the tensile force in cable DA (in newtons): 400
Enter the tensile force in cable DB (in newtons): 800
Enter the tensile force in cable DC (in newtons): 600
Enter the maximum value for the overturning moment (in N-m): 11000

The overturning moment magnitude is 5339.9 N-m.
This restraining system is safe.
```

Programming Problem 2 (of 2)

Learning Objectives: Manipulate Python lists; Utilize simple repetition loops and conditional if-elif-else statements while programming in Python; Output data from a function to the screen in Python; Apply course code standard.

Background:



When you pluck a string on your guitar (or any other type of stringed instrument), you are creating *standing waves* in the string that in turn cause the air around the string to vibrate, eventually causing a pressure wave to travel to your ear to create the sensation of a sound. In reality, a simple string pluck creates many, many standing waves, each with a different frequency. The lowest frequency standing wave is called the *fundamental* (or *first harmonic)*, and the higher frequency waves are the 2nd harmonic, 3rd harmonic, 4th harmonic, etc. It can be shown that, if the first harmonic of a given guitar string has a fundamental frequency of f_1, the *n*'th harmonic has a frequency $f_n = n * f_1$.

When a sound engineer makes a recording of a musical performance, the sound often contains waves coming from wide variety of sources in addition to the musical instrument (*e.g.,* low-frequency vibrations of the building, high-frequency clicks coming from electrical equipment, etc.) Since these sounds are undesirable, the engineer will remove them by applying a *band-pass filter* to the recording. A band-pass filter essentially eliminates any sounds whose frequencies are below a lower bound f_{low} or above an upper bound f_{high}; the resulting recording will only contain sounds whose frequencies *f* satisfy the requirement $f_{low} \leq f \leq f_{high}$. This means that the band-pass filter will get rid of some of the harmonics from the stringed instrument being recorded, and it is important that the engineer know which harmonics remain after the filtering is done to ensure that the sound quality is acceptable.

However, band-pass filters are not as perfect as the requirement $f_{low} \leq f \leq f_{high}$ implies. Sounds whose frequencies are within the range $f_{low} \leq f \leq f_{high}$ but close to f_{low} or f_{high} will receive partial filtering; those sounds will still remain in the recording, but their contributions will be reduced due to the effects of filtering. For purposes of this problem, you can interpret "close" to mean that, if a sound has a frequency such that $f_{low} \leq f < 1.10 * f_{low}$ or such that $0.90 * f_{high} < f \leq f_{high}$, that sound will receive partial filtering.

Instructions:

Your task is to write a program allows a sound engineer to determine if the harmonics of a guitar string with known properties will be completely filtered, partially filtered, or not filtered at all due to application of a certain band pass filter. You should assume that the user will enter the following information:

- The fundamental frequency f_1 of a certain guitar string (in Hertz).
- The values of f_{low} and f_{high} (in Hertz) that define the band-pass filter.
- An integer *N* that indicates how many harmonics the sound engineer wishes to investigate, starting from the first harmonic. So, for example, if *N* = 4, this means that the sound engineer wishes to investigate the first, second, third, and fourth harmonics of the guitar string.

No error checking of the inputs is necessary; you may assume the user always inputs appropriate values.

For each of the harmonics that the sound engineer is interested in, your program should output one of three messages to the console: a) the harmonic is completely filtered, b) the harmonic is partially filtered, or c) the harmonic is not filtered. See the example below for an illustration of how your inputs and outputs should look.

Save this program as `HW3_Prob2_login.py` (where *login* is your username) and submit it to the appropriate programming dropbox on Gradescope. Also, create a flowchart of the relevant algorithm for this problem, and include this flowchart in your PDF submission of the written portion of this assignment. Please note that the PDF for the written portion of the assignment will be submitted to its own dropbox -- do not submit this PDF to the programming dropbox or vice versa.

Example: (User inputs in bold)

```
Enter the fundamental frequency for the string (in Hertz): 110
Enter the lowest frequency of the band pass filter (in Hertz): 210
Enter the highest frequency of the band pass filter (in Hertz): 440
How many harmonics do you want to investigate? 4

Harmonic 1 is completely filtered.
Harmonic 2 is partially filtered.
Harmonic 3 is not filtered.
Harmonic 4 is partially filtered.
```