CS1112 Spring 2015 Project 1 due Thursday 2/5 at 11pm

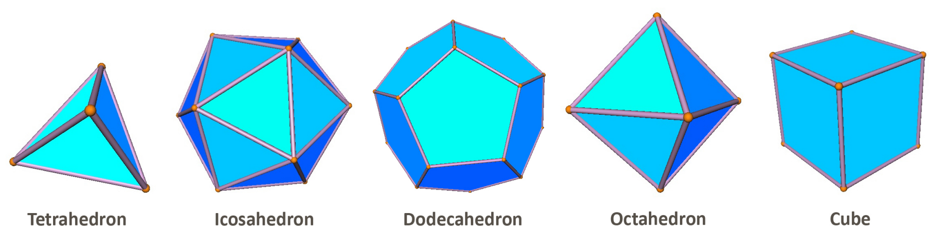
You must work either on your own or with one partner. If you work with a partner you must first register as a group in CMS and then submit your work as a group. *Adhere to the Code of Academic Integrity.* For a group, "you" below refers to "your group." You may discuss background issues and general strategies with others and seek help from the course staff, but the work that you submit must be your own. In particular, you may discuss general ideas with others but you may not work out the detailed solutions with others. It is not OK for you to see or hear another student's code and it is certainly not OK to copy code from another person or from published/Internet sources. If you feel that you cannot complete the assignment on you own, seek help from the course staff.

Objectives

Completing this project will help you learn about MATLAB scripts, branching, and some MATLAB built-in functions. You will also start to explore MATLAB graphics.

1 Which Platonic Solid is Most Spherical? Surprise, surprise ...

A solid is a *Platonic solid* if each face is identical in size and shape. There are only five:



The number of faces for the solids range from four—the tetrahedron—to twenty—the icosahedron. Exercise **P1.1.5** in *Insight* (page 13) gives more details on Platonic solids.

First let's consider the two-dimensional case. Suppose I ask which regular polygon is more circular, the square or the hexagon? Likely you would say the hexagon, and you can draw a circle with an inscribed square and an inscribed hexagon showing that the hexagon's area is bigger than that of the square (and therefore is closer to the circle and therefore is "more circular").

Going to three dimensions, we can answer the question "which Platonic solid is most spherical?" by comparing the volume of inscribed Platonic solids. Using some of the formulae from Exercise **P1.1.5** in *Insight*, write a script `platonicSol.m` to print a table showing the volumes of the five Platonic solids when *inscribed* in a unit sphere (a sphere with radius one). To print a "table" simply use `fprintf` with an appropriate format sequence so that the values line up neatly in a column. See scripts `Eg1_2` and `Eg1_1` in *Insight* for examples on using `fprintf` and see page 54 Exercise M3.1.4 for an example of the table format. In your script, write the following comment and fill in the blank based on your observation:

```
% Comparing the inscribed Platonic solids, I think the __________ is
% most spherical!
```

Does the result match your intuition? Let's try another criterion for comparison. Add more code to `platonicSol.m` to calculate and print the volumes of the Platonic solids that *circumscribe* a unit sphere. Organize your code so that the circumscribed volumes are printed as another "column" of the table started above. Write the following comment and fill in the blank based on your observation:

```
% Comparing the circumscribed Platonic solids, I think the __________ is
% most spherical!
```

[Think about this question for fun but you do not need to submit an answer: what may be a better criterion for determining which platonic solid is most spherical?]

Errata: In the example below the table of formulae in *Insight* p.13, the divisor for calculating `Vol20` should be 12, not 4, i.e., `Vol20 = ((15 + 5*sqrt(5))/12)*E^3;` Later please look at the *Insight* page on the course website to see the erratum list for the rest of the book.

2 Construct a Parallelogram

Download the program `parallelogram.m` and run it. A graphics window showing a black dotted line connecting two blue asterisks will pop up. The message near the top (the title area) says to click in the window. After you click, a red circle marks the clicked point and its coordinates are given in the title area.

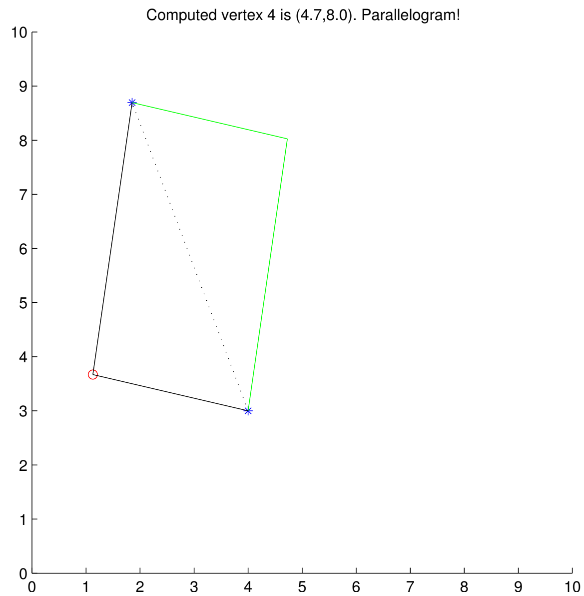
Read the program to make sure you understand what it does. Don't worry about the early commands to set up the figure window, but here's how the `plot` statement works: `plot(x, y, 'b*')` draws a marker at the point (x,y) with the format "blue asterisk"; `plot([x1 x2], [y1 y2]), 'k:'` draws a line from the point (x1,y1) to (x2,y2) with the format "black dotted line." The statement `title('hello there')` displays the text 'hello there' as the title of a figure. The `sprintf` statement works just like `fprintf` in formatting text, but instead of printing directly to the Command Window, `sprintf` allows the text to be saved under a variable name. Then this text (string) variable can be used in other statements, such as the `title` statement as shown in the program.

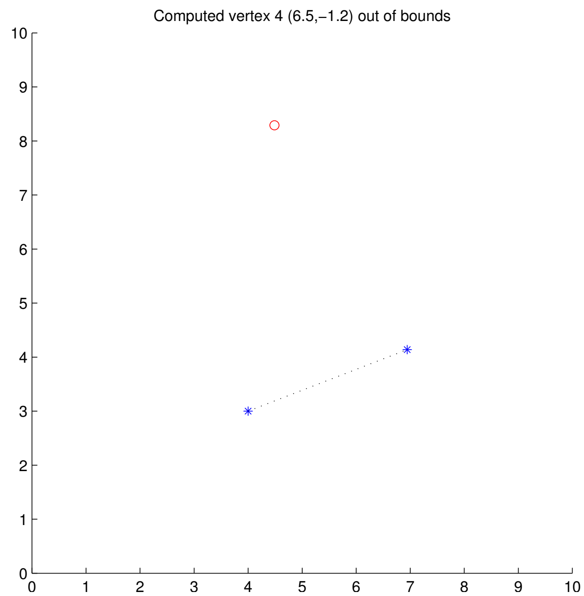
Your job is to modify this program to possibly draw a parallelogram:

1. Change the fixed location of the second vertex (see the comments in the code) to randomly generated coordinates: the x- and y-coordinates should be equally likely to be any value in the range of 1 to 9. *Hint*: The statement `v = rand` assigns to variable `v` a random number in the range of 0 to 1 such that any value between 0 and 1 is equally likely to occur. How do you get a random value within a different range? First, the statement `v = rand` gets you a real number in the range of 0 to 1. Next, scale (think multiply) and shift (think add) the value `v` as necessary to get the range you need.

2. Given three points (fixed vertex 1, random vertex 2, and user-selected vertex 3), there are different ways to compute a fourth point to make a parallelogram. One simple way is to consider two pairs of *opposing* vertices instead of four *adjacent* vertices: *we will take vertices 1 and 2 to be opposing vertices and then compute a vertex that opposes vertex 3.* Draw some diagrams to help yourself think! One way to think about it is to start by drawing a triangle with the first three vertices. Then add a triangle on the other side of the dotted line.

3. Draw the parallelogram only if the computed fourth vertex is within the original axis limit (0 to 10). Use the title area to display an appropriate message that includes the coordinates of the computed vertex. See example figures below (vertices 1 and 2 are the blue asterisks). Experiment with different marker formats! Use 'g' for green, 'y' for yellow, 'x' for an x marker, 'd' for a diamond marker, etc. To get a solid line instead of a dotted line use a dash (-) instead of a colon (:).





3 Quadratic Function

Complete exercise **P1.2.8** in Chapter 1 of *Insight*. Save the file as `myParabola.m`.

Submit your files `platonicSol.m`, `parallelogram.m`, and `myParabola.m` in CMS.