Math 168 Weekly assignment #2 Due Friday, Sep 21

About You

For this section only, if you prefer your responses to come just to the instructor, you may email your answers to *whitney@math.harvard.edu* *any time prior to class on the 21st.*

1. From time to time, photographs may be taken in class, especially when we are doing a building project or displaying physical items that we have made. Please indicate whether I **do** or **do not** have permission to:
   a) Post pictures showing your likeness on sites accessible to the Harvard community?
   b) Post pictures showing your likeness on sites accessible to the general public, which may or may not be affiliated with Harvard?
   In addition, please indicate for each setting in which you grant permission for pictures to be posted, whether you (i) do not want to be identified by name, (ii) have no preference whether or not you are identified by name, or (iii) always wish to be identified by name if your likeness is posted. Thank you very much.

About modeling

This is an example of a "modeling" assignment in this class. Here are the ground rules for all such assignments: Some of the questions may require written answers, which should be narrative/descriptive in nature. These should be typed into any generally readable text document format (even just plain text ".txt" or Markdown ".md" is fine, or any commonly readable word processing format, or PDF). Other questions require saved files from the modeling software being used. You should collect up a single written file, and all of the saved files, and submit them electronically all at once.

1. Obtain access to one of the two-dimensional dynamic geometry packages you may use for submitting assignments in this class: GeoGebra (geogebra.org, use online or download an application), Cinderella (beta.cinderella.de, download an application), or CaRMetal (carmetal.org/index.php/fr/telecharger, download an application) and open up a new session. [No response required for this item.]
2. Draw two line segments, not connected to each other, fairly far apart on your construction space. I will call the first segment *r*, and the second one *s*. Make sure *r* is not less than half the length of *s*. Now figure out how to draw two circles, centered at the two endpoints of *s*, the radius of each of which is the same as the length of *r*. These two circles should intersect at two points. Figure out how to draw a line *t* that goes through both of these intersection points, and figure out how to specify the point *M* where *s* and *t* intersect. [No response required for this item.]
3. Figure out how to get the software to continuously display two distances: the distance from each endpoint of *s* to *M*. Now move one or the other or both of the endpoints of *s* around in the neighborhood of where it started. What do you observe about these two distances? Explain why that behavior occurs.
4. Describe what happens if you either stretch *s* out or make *r* smaller to the point where *r* **is** less than half of the length of *s*, and then move it back again (so that the original relationship, r longer than half *s*) holds again. Is the behavior the same in each case?
5. Now create a triangle of which *s* is one side, and (by whatever method you like within the package) construct the perpendicular bisectors of the two new sides. Try moving any/all of the

vertices of this triangle around. What do you observe? What geometrical fact or phenomenon does this illustrate?

6. Save your session, making sure the two distances are continuously displaying, and submit your saved file as part of this assignment.

About the Pythagorean Theorem

1. The Pythagorean Theorem has an incredibly rich array of different proofs and demonstrations. Our goal is to build a structure or structures embodying one or more of them. To do that, we need to create software models of the proofs and how they might work as a mechanism. So, take a look at the web page cut-the-knot.org/pythagoras/ – it lists over 100 (!) proofs of the Pythagorean Theorem. Choose one of them, and create a dynamic geometry model of that proof. Make sure the "original" right triangle which the model is illustrating the Pythagorean Theorem for is clearly indicated in your file, and that you can modify that right triangle, at least over some range of possibilities, and the diagram/illustration/mechanism will update accordingly. Also clearly indicate, either in a caption in your session file or in your written responses, which of the proofs from that web page you have modeled. Save your session and submit this model as part of your assignment.
2. Does your model allow *any* right triangle to be illustrated as a case of the Pythagorean Theorem? Describe how you know and/or what goes wrong if there are cases in which your model stops working/shows improper results.
3. Describe whether or not you feel your model shows something that could be built physically to demonstrate the Pythagorean Theorem and/or illuminate one of its proofs. Describe why you feel that's the case. [Note that your credit for this section of the assignment will not depend on whether you model something that can actually be built, only on whether you've created a clear software model of a proof of the Pythagorean Theorem from the specified web page, and answered thoughtfully about the characteristics of the model.]

Challenge problems

Modeling challenge problems will involve a greater depth and/or creativity in the use of the relevant software packages. In addition, you should submit program files that adhere to a high standard of clarity and self-documentation, with labels and comments in the files. As with theoretical challenge problems, you can do any, all, or none of the challenge problems; each successful effort on a problem will reduce the weight of all of the other elements of the course.

1. **A Tale of Four Dynamic Geometers**. Even just strictly limiting to two-dimensional Euclidean geometry, the three dynamic geometry packages listed above and Geometry Expressions online (geometryexpressions.com) each have their own special capabilities and weaknesses. Find (at least) one task/construction/activity that each is well-suited to, but which cannot be done well/easily/at all (that's the best kind of example) in any of the others. Submit a save file from each of the first three illustrating its "special power" in this regard, and an annotated screenshot from Geometry Expressions illustrating where it shines. (Note, sadly, that the free on-line version of Geometry Expressions has no way to save your work, so be careful when you are creating your demonstration and be aware of this limitation.) Note that all four programs are quite solid on basic geometric constructions, so you will have to be creative and push the envelope of using dynamic geometry to find the unique capabilities of each.
2. Also, find one aspect of each in which you find it weaker than the other programs. This aspect may be subjective, and you don't need to (but you may) submit a session file or screenshot for these issues; you should submit a narrative description of each issue, and feel free to optionally

include a session file or screenshot if it will make your narrative easier to write/understand. Note that “fonts are ugly” or “can’t save files” are not acceptable as the weaknesses for this assignment; you should focus on the geometrical capabilities and how you can execute them in your response.

3. Create and submit a “feature matrix” comparing all four software packages, including all of the aspects identified in items 1 and 2, together with any other interesting features that you may have noted in your efforts to compare the four systems.