CS100M Grading Guide: Project 1

The coded items below (e.g., *c1e*, *s2a*) indicate what a student's solution should accomplish. Codes that begin with the letter 'c' deal with correctness; codes that begin with 's' deal with style.

Grader: If a student's solution does not accomplish task *c1a*, for example, then write the task code 'c1a' along with any diagnostic remarks you can give. Count the number of correctness and style errors separately. Items marked with ** count as two errors. In the table below, the top row lists the possible scores (1 to 5). The next row lists the number of correctness errors corresponding to every score category. The style score is determined similarly. Enter the total score (maximum of 10) in CMS as the project score. If there are bonus questions, enter any bonus points separately in the "Bonus Bucket," separate from the project score.

Student: Read the grading guide for every project, even if you get a perfect score! Notice from the table below that we often give one or two "freebies," i.e., mistakes that don't cost you any points. Learn from working on the project, and learn from any mistakes.

Scores

Score	0	1	2	3	4	5
No. of correctness errors	Not handed in	> 7	$5-7$	$3-4$	2	$0-1$
No. of style errors	Not handed in	> 7	$6-7$	$4-5$	$2-3$	$0-1$

0. General

(c0a) All programs are written in the MATLAB programming language.

(s0a) Variable names are appropriately chosen

(s0b) Code lines are properly indented

(s0c) All extraneous output is suppressed with semi-colons

(s0d) Reasonable line lengths, no horizontal scrolling

(s0e) Appropriate header comment in each script file

(s0f) New lines are printed when output is produced.

(s0g) No superfluous code

1. Questions about ASCII and course policies

(c1a) Answers to questions 1-2 are correct (there are various possible answers for 2)

(c1b) Answers to questions 6-7 are correct.

(c1c) Answers to questions 8-10 are correct.

(c1d) ** The file is in ASCII format.

Also, consider the general style point about line lengths.

2. Divided difference approximation

The exercise did not require the student to include input statements (but we are not going to discount points for doing that.)

(c2a) The right analytical formula for the derivative is used.

(c2b) The right formula for $f(x+h)$ is used in both cases.

(c2c) Both values of h are used.

(c2d) The right formula for the divided difference approximation is used.

(c2e) All results are displayed correctly (to at least three decimal places).

```
f'(-2)= -18.00,  f'(-2) = -13.5937, f'(-2) = -17.9002
```

(s2a) Variables are used to store the values of x, h and other intermediate results in the calculation.

(s2b) Explanatory messages are displayed with the output.

Note about(s2a): It is good practice to use a variables like x, h, or $h1, h2$ to store values that are going to be used multiple times in a formula. This serves three purposes: 1) It improves readability. 2) It minimizes the chance to make mistakes. 3) It makes it easy to modify the program in case we decide to use another value for $h1$, say.

3. Area of a random triangle

(c3a) All coordinates generated are real numbers in the range $(1, 10)$, computed using the function **rand**. The formula used should generate any number in this range as well.

(c3b) Every coordinate gets a different random value (i.e. a separate call to rand is used for each one).

(c3c) The lengths of the sides are calculated correctly.

(c3d) The half perimeter is calculated correctly.

(c3e) The area is calculated correctly, using Heron's formula.

(s3a) Variables are used to store intermediate results (like side lengths and semiperimeter)

(s3b) Code is broken into appropiately commented sections.

4. Rays in the Cartesian Plane

4..1 First Solution (Determine the quadrant for a, determine the quadrant for b and then compare

(c4a) The values of a and b are mapped into $[0, 360)$ with the right formula. (Doesn't have to be `mod(a,360)`, though).

(c4b) The quadrants for a and b are appropiately determined via **if-elseif-else** statements and then compared appropiately (using `==` instead of `=`)

(s4a) The condition is not too cumbersome or redundant (e.g. Splitting into 16 cases is cumbersome.)

4..2 Second Solution (single compound condition)

(c4a) The values of a and b are mapped into $[0, 360)$ with the right formula.

(c4b) The right single compound condition is used for determining at once whether both rays are in the same quadrant.

(s4a) The condition is not too cumbersome or redundant (e.g. Splitting into 16 cases is cumbersome.)