CS100M Fall 2007 Grading Guide: Project 2

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

- c ands stand for correctness and style; see table below.
- parts with ** next to them means that they are double the value, *** for triple, etc.

Score	5	4	3	2	1	0
#correctness errors	0-1	2-3	4-6	7-11	12-16	>17
#style errors	0-1	2-3	4-6	7-11	12-16	>17

General

(s0a) Use meaningful variable names

(s0b) Appropriate indentation

(s0c) Appropriate comment header in each script/function file

(s0d) Appropriate and concise comments throughout

(s0e) Reasonable line lengths; no horizontal scrolling

(s0f) [up to **] No superfluous code

(s0g) [up to **] Reasonably efficient and concise code; a little inefficient is OK

(s0h) No debugging output

(c0a) [2* max]Program successfully executes without crashing.(*for occasional,**for persistent)

Problems

1. Flipping an unfair coin

(c1a) [2* max, 1* for each mistake found] Correctly simulates the toss of the unfair coins, which typically includes:

(i) the rand function

(ii) the expression to decide if a coin is heads or tails

(iii) we accept equivalent ways such as rand ∗ 3 > 1,or floor(rand ∗ 3)==1, etc.

(c1b) [2* max, 1* for each mistake found] Correctly decides the result of each trial, Ann wins, Bob wins, or tie? It typically includes:

(i) the expressions that decide if Ann and Bob win when we consider them separately

(ii) who wins, or is it a tie, when we consider them together

(iii) it is OK if they merge the above steps together into some huge and complicated expressions, as long as the variable names are clear

(c1c) [2*max,1*for each mistake found, Correctly writes the loop that stops correctly.It typically needs:

(i) to keep track of necessary information, such as number of total trials, number of trials in which Ann won, etc.

(ii) to write the expression that decides if the game ends

(s1a) Displays the result neatly.

2. Betsy Ross Flag

(c2a) Correctly calculates the positions and sizes of stripes

(c2b) Correctly calculates the position and size of the blue field in the corner

(c2c) Correctly calculates the positions of the stars

(c2d) Correctly uses the functions DrawStar and DrawRect

(s2a) The resulting picture looks clear; the parameters are chosen reasonably

(s2b) Uses a loop to draw the stripes; it is NOT OK if they write code for each stripe; don't double count with (s0f) or (s0g)

(s2c) Uses a loop to draw the stars; the same as the above, we DON'T accept superfluous code.

3. Pythagoras rules!!

(c3a) [2* max, 1* for each mistake] Correctly uses the nested loops (and the sqrt function) to find out the value of c. This includes setting the right ranges for a, b and the right value for M. Any more than 2 loops should be considered superfluous code.

(c3b) Correctly uses the if statement and the floor function to check if a, b, c are Pythagorean numbers

(c3c) Correctly increments the counter to count the number of Pythagorean triplets

(s3a) Displays the output neatly