CS100J Spring 2001: Project 5 Grading Guide

Notes

- Please carefully review all notes written on your grading form and project.
- Find the codes for these notes below.
- Try to understand why you received the note so that you may avoid it on your next project.
- * means the item is worth twice

Scores

- Bonus may be applied for exemplary work or doing additional tasks
- Give 1 bonus point for each valid and justified response to Question 5 and 6.
- Let c and s be the number of correctness and style: see table, below
- For each program not included, remove one correctness and style point

category	Points					
	0	1	2	3	4	5
correctness	nothing turned in	$c \geq 4$	$c = 3$	$c = 2$	$c = 1$	$c = 0$
style	nothing turned in	$s \geq 4$	$s = 3$	$s = 2$	$s = 1$	$s = 0$

1. General

(s1a) correctly filled out grading form provided for each partner as coversheet
(s1b) lines of text/code not chopped off and proper code indentation
(s1c) * appropriate code comments
(s1d) title sheet and table of contents provided
(s1e) pages typed, numbered and properly bound

2. Tic Tac Toe

(c2a) * no code that deals with individual squares on board case-by-case
(c2b) * single 2D array field for board (may be **`int`**s, **`char`**s, **`String`**s)
(c2c) program generalized for any **`SIZE`** "tic tac toe"
(s2a) winning rules simplified with loops
(s2b) output runs: one with a tie, and another with a win

3. Cafeteria simulation (Q3)

(c3a) * arrays of trays and of workers have correct syntax and purpose
(c3b) * efficiency calculation in terms of worker position (only two cases: last, not last)
(c3c) create **`n`** workers/trays in loop
(c3d) shifting trays is generalized
(c3e) processing trays is generalized
(c3f) printing belt is generalized
(c3g) correctly read length of belt, **`n`**
(c3h) output run with 10 workers

4. Insertion sort (Q4)

(c4a) * correct insertion sort algorithm (not select sort!)
(c4b) used nested loops
(c4c) correctly finds smallest number / insertion point
(c4d) correctly performs swap
(c4e) correctly shifts numbers in the array
(c4f) demo code shows the three requested test cases
(s4a) sorts as one method or split into methods that find insertion point, shift numbers in the array, swap, etc.

5. Matrices with complex values (Q5)

(c5a) * correct addition of complex numbers
(c5b) private instance fields of real and imaginary parts
(c5c) accessor methods for real and imaginary parts
(c5d) constructor creates random complex value
(c5e) constructor creates complex value with given real and imaginary parameters
(c5f) **`toString`** works properly
(c5g) generate ragged column-major upper-triangular matrix with random complex values
(c5h) * correctly add two matrices
(c5i) arrays print as upper triangular
(c5j) align values in columns and rows (does not to print zeros of lower triangle)
(c5k) output showing addition of upper triangular matrices of sizes 1, 2 and 3

6. Miscellaneous

(c6a) miscellaneous
(s6a) miscellaneous