**Project 3 Grading Guide**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item #.** | **Description** | **Correctness** | **Style** |
| PROBLEM 1 | DrawTennisCourt.m, CornellTennis.m | Total = 6 | Total = 1 |
| 1 | DrawTennisCourt calls DrawRectNoBorder correctly to draw the following (1 point for at least 3 bullets drawn correctly, 2 points for all):   * Green court background. * Border white lines. * Singles sidelines. * Net line. * Service lines.   See Insight P5.3.7 or solutions for dimensions.  **NOTE:** Allow small variation in dimensions due to placement/thickness of lines.  **NOTE:** Accept reasonable alternative approaches (e.g. green rectangles on a white background) as long as DrawRectNoBorder is used for the drawing. | 2 |  |
| 2 | DrawTennisCourt draws with the lower left corner at the parameters given to the function (1 point), and only draws one tennis court (1 point). | 2 |  |
| 3 | CornellTennis draws 3 rows of 4 tennis courts (1 point) spaced 40 apart (1 point). | 2 |  |
| 4 | CornellTennis uses loops and calls to DrawTennisCourt. No direct drawing should take place in CornellTennis.  **NOTE:** Comment but award the point if the call to DrawTennisCourt appears more than once, which is unnecessary. |  | 1 |
| PROBLEM 2.1 | SPModel.m | Total = 11 | Total = 2 |
| 5 | While loop exits with appropriate condition comparing simulation time to tf. | 1 |  |
| 6 | C(t) is computed and recorded in do correctly. (-1 for each error; -2 max). | 2 |  |
| 7 | Update vector time correctly (-1 for each error; -2 max). | 2 |  |
| 8 | The last element of vector time is close to tf (within dt). If the anerobic phase extends beyond tf, then the last value in time is near tf and the last value in do is 0. | 1 |  |
| 9 | Correct if condition to find when anaerobic condition begins | 1 |  |
| 10 | L\_i, T\_anaerobic, L\_f are computed correctly (-1 for each error; -2 max) | 2 |  |
| 11 | After the anerobic period, parameters are reset for the start of the second aerobic period:  L = L\_f, ct = c0 = 0, sim time = t + T\_an, t = 0 | 1 |  |
| 12 | Do = 0 and the simulation time is recorded in their respective vectors at the start of the anerobic conditions. | 1 |  |
| 13 | A single loop is used for the whole function, or  If the structure of the function requires two loops be used for the two aerobic phases, they should be organized as calls to a subfunction to avoid code repetition.  (Award 2 or 0 points.) |  | 2 |
|  | TEST CASES:  Base values:  dt = 0.5, tf = 20, cs = 8, c0 = 7, L = 30  1. Kr = .35 Kd = .35 Ka = .8: One aerobic phase  2. Kr = .25 Kd = .25 Ka = .2: Two aerobic and one anaerobic phases  3. Kr = .15 Kd = .15 Ka = .1: One aerobic and one anaerobic phase (plot should be cut off at tf with do still at 0). |  |  |
| PROBLEM 2.2 | DOAnalysis.m | Total = 3 | Total = 1 |
| 14 | Use different variables for plot handles (hka) in each plot. Or hka should be reset to [] before the second and third sensitivity analyses. |  | 1 |
| 15 | Successfully display three plots for three parameters. | 1 |  |
| 16 | Labels, legends are consistent with problem description. | 1 |  |
| 17 | Range of changes in three parameters are appropriate | 1 |  |
|  | Sample values and ranges; base values:  dt = 0.5, tf = 20, cs = 8, c0 = 7, L = 30,  Kd = Kr = 0.35, Ka = 0.8.  Varying Ka = 0.2:0.2:0.8  Varying Kr: Set Kd = 0.1, vary Kr from 0.1 to 0.5  Varying L: 20 to 80 |  |  |
| GENERAL |  |  | Total=10 |
| G1 | Script starts with a concise comment describing the program.  Function comment follows function header. |  | 1 |
| G2 | Code is sufficiently (but not excessively) commented. |  | 1 |
| G3 | Line lengths are not excessively long (80 columns).  **NOTE**: It's ok if a couple lines are a little too long, especially if it’s due to having to print a very long string**.** |  | 1 |
| G4 | No extra output (debugging output) produced |  | 1 |
| G5 | Proper indentation is always used. |  | 1 |
| G6 | Use meaningful variable names. Do not overwrite MATLAB keywords. |  | 1 |
| G7 | Name important parameters as variables (constants). |  | 1 |
| G8 | No superfluous code (e.g., an empty if or else branch or a useless loop). Of course some students will have code that is awkward or unclear or inefficient. This point is specifically for not having code that does literally nothing. |  | 1 |
| G9 | Reasonably efficient code. |  | 1 |
| G10 | Does NOT put semicolon at wrong places, e.g., at the end of these lines: "if", "elseif", "else"," for","while", "function". |  | 1 |
| TOTAL |  | 20 | 14 |

**Penalties**

|  |  |  |
| --- | --- | --- |
| P1 | Student's code does not execute (or student provides a script when a function is required and vice-versa) | -1 from final score |
| P2 | Student's code crashes or does not terminate (infinite loop) for normal cases. | -1 from final score |
| P3 | All function headers and file names match those specified in the project description exactly. All input and output variables should be of the correct type. | -1 from final score |

**Grade Calculation**

|  |  |
| --- | --- |
| Total Possible Correctness Points | TC = 20 |
| Total Possible Style Points | TS = 14 |
| Student Correctness Points | C = min( \_\_\_ + 1 freebie point, TC) |
| Student Style Points | S = min( \_\_\_ + 1 freebie point, TS) |

Exceptions: If any file is missing/unacceptable, no freebie points can be applied to that file and subtract 3 style points for each missing/unacceptable file.

|  |  |
| --- | --- |
| Student’s final score | ([(C/TC)+(S/TS)] X 5) - Penalties  (Out of 10; 1 decimal; no negative score; round to NEAREST) |