**Project 2 Grading Guide**

|  |  |  |  |
| --- | --- | --- | --- |
| **Item #.** | **Description** | **Correctness** | **Style** |
| PROBLEM 1 | starArea.m | Total = 5 | Total = 1 |
| 1 | Loops through proper values of k (1->20)  **NOTE:** the upper bound of k CAN be given via input | 1 |  |
| 2 | Correct n values are used (10\*k and 10\*k + 1) | 1 |  |
| 3 | Correctly choosing the formula to use under each condition. | 1 |  |
| 4 | Theta correctly computed  **NOTE:** Do NOT take points off if theta is not used as an intermediate variable but the final results of A(n) and A(n+1) are correct. | 1 |  |
| 5 | A(n) and A(n+1) correctly computed  **NOTE:** If the formulas are correct for A(n) and A(n+1), but the values computed are wrong due to incorrect n, then only subtract from (#2)  **NOTE:** If the formulas are correct, but not correctly used because of incorrect if conditions, then only subtract from (#3) | 1 |  |
| 6 | Results printed in neat table format  Do not take points off if perimeter is outputted as well |  | 1 |
| PROBLEM 2 | o2curve.m | Total = 11 | Total = 1 |
| 7 | While loop used with the correct three exit conditions (-1C for each error)  **NOTE:** The three exit conditions are  a. Oxygen level returns to original level (within a tolerance), after the initial decline. They can use either of the following  i. c > co - tolerance && (no longer declining)  ii. c > co  b.Time has exceeded the maximum simulation time  t >= uBound  c. DO level reaches 0  c <= 0  **NOTE:** If the student used a for loop running until the max time, instead of a while loop, then subtract the point for efficiency (#28), and 1 point from (#7) | 3 |  |
| 8 | t is incremented by delta t | 1 |  |
| 9 | c is recalculated for every delta t | 1 |  |
| 10 | c is plotted for every delta t | 1 |  |
| 11 | The minimum DO level and the time it occurred is computed correctly | 1 |  |
| 12 | if c < 0 : display message one (Anaerobic condition at X days) | 1 |  |
| 13 | if full recovery occurs within t = 10: display message two (Lowest oxygen level of Y mg/L occurs at Z days Full recovery X days after pollution event) | 1 |  |
| 14 | if full recovery does not occur within t = 10 but is no longer decreasing: display message three (Lowest oxygen level of Y mg/L occurs at Z days Full recovery not reached after X days) | 1 |  |
| 15 | if still decreasing after t = 10: display message four (Oxygen level continues to drop after X days) | 1 |  |
|  | TEST CASES:  1. Kr = .15 Kd = .15 Ka = .2: message one: anaerobic at 2.8 days  2. Kr = .35 Kd = .35 Ka = .8: message two: lowest level of 0.9 mg/L at 1.8days, full recovery at 8.8 days  3. Kr = .25 Kd = .25 Ka = .6: message three: lowest level of 1.1 mg/L at 2.3 days, full recovery not reached  4. Kr = .05 Kd = .05 Ka = .1: message four: keeps dropping  **NOTE:** The minimum value and time may be different due to different choice of delta\_t. Don’t take points off in this case. |  |  |
| 16 | All specified parameters are assigned to variables  **NOTE:** If points are subtracted for #16, then only subtract from #26 if they don’t name important parameters in stars or sequence  **NOTE:** Delta\_T should be a variable, not just a magic number |  | 1 |
| PROBLEM 3 | sqrtSequence.m | Total = 3 | Total = 1 |
| 17 | Script is organized into nested loops  **NOTE:** If the students use only one loop but is correct, don’t take points off. | 1 |  |
| 18 | The correct values of t\_k are computed.  **NOTE:** The points for #18 should be all or nothing  **NOTE:** There are two versions of this question. One asks for the value of t1 to t26 and the other one asks for t0 to t26. We accept solutions to both versions.  **NOTE:** Make a comment, but do not deduct points, if the students sequence output converges to three for the last few iterations | 2 |  |
| 19 | The results are displayed in a neat format  **NOTE:** Neat format means anything reasonable that was done using fprintf (a table or just a list is fine), but unsuppressed output is not okay |  | 1 |
| GENERAL |  |  | Total=10 |
| 20 | Script starts with a concise comment describing the program.  Function comment follows function header. |  | 1 |
| 21 | Code is sufficiently (but not excessively) commented. |  | 1 |
| 22 | 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 |
| 23 | No extra output (debugging output) produced |  | 1 |
| 24 | Proper indentation is always used. |  | 1 |
| 25 | Use meaningful variable names. Do not overwrite MATLAB keywords. |  | 1 |
| 26 | Name important parameters as variables (constants). |  | 1 |
| 27 | 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 |
| 28 | Reasonably efficient code. |  | 1 |
| 29 | Does NOT put semicolon at wrong places, e.g., at the end of these lines: "if", "elseif", "else"," for","while", "function". |  | 1 |
| TOTAL |  | 19 | 13 |

**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 = 19 |
| Total Possible Style Points | TS = 13 |
| 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) |