

```

1 function output = ZhouK_Prelab7()
2
3 format shortG
4
5
6
7 % ----- TASK 1 -----
8
9
10
11 % initialize variable for the step size
12 h = 0.3;
13
14 % initialize the first column of the table
15 values(:,1) = [0 1 2];
16
17 % initialize the second column of the table
18 values(:,2) = [0 0.3 0.6];
19
20 % fill in the given initial conditions for t=0, t=0.3 and t=0.6
21 values(:,3) = [0.7 0.64670 0.58891];
22
23 % initialize function handle for g
24 g = @(y,t) (1.2*y*t*t) - (0.3*y);
25
26 % initialize the fourth column of the table
27 values(1,4) = g(values(1,3), values(1,2));
28 values(2,4) = g(values(2,3), values(2,2));
29 values(3,4) = g(values(3,3), values(3,2));
30
31 % initialize function handle for the predictor
32 predictor = @(h,y,g2,g1,g0) y + (h*((23/12)*g2 - (4/3)*g1 + (5/12)*g0));
33
34 % initialize function handle for the predictor modifier
35 predictor_modifier = @(p,c_prev,p_prev) p + (0.9*(c_prev - p_prev));
36
37 % initialize function handle for the corrector
38 corrector = @(h,y,g3,g2,g1) y + (h*((5/12)*g3 + (2/3)*g2 - (1/12)*g1));
39
40 % initialize function handle for the corrector modifier
41 corrector_modifier = @(c,p) c - (0.1*(c - p));
42
43 % initialize function handle for the convergence
44 convergence = @(new, old) (new - old)/new;
45
46 % apply the predictor formula to indices 2, 1, 0
47 % p = 0.65702
48 p = predictor(h,values(3,3),values(3,4),values(2,4),values(1,4));
49 old = p;
50
51 % enter in the fourth row to the table
52 values(4,:) = [3 0.9 p g(p,0.9)];
53
54 % apply the corrector formula to indices 3, 2, 1
55 % c = 0.66275
56 c = corrector(h,values(3,3),values(4,4),values(3,4),values(2,4));
57 c_prev = c;
58
59 % apply the corrector modifier formula and store it in the table
60 % c_m = 0.66218
61 c_m = corrector_modifier(c,p);

```

```

62 values(4,3) = c_m;
63 values(4,4) = g(c_m,values(4,2));
64
65 % check for convergence
66 % eps = 0.0077834
67 eps = convergence(c_m, old);
68 old = c_m;
69
70 % apply the corrector formula to indices 3, 2, 1
71 % c = 0.66318
72 c = corrector(h,values(3,3),values(4,4),values(3,4),values(2,4));
73 c_prev = c;
74
75 % apply the corrector modifier formula and store it in the table
76 % c_m = 0.66257
77 c_m = corrector_modifier(c,p);
78 values(4,3) = c_m;
79 values(4,4) = g(c_m,values(4,2));
80
81 % check for convergence
82 % eps = 0.00058808
83 eps = convergence(c_m, old);
84 old = c_m;
85
86 % apply the predictor formula to indices 3, 2, 1
87 % p = 0.87197
88 p_prev = p;
89 p = predictor(h,values(4,3),values(4,4),values(3,4),values(2,4));
90
91 % enter in the fourth row to the table
92 values(5,:) = [4 1.2 p g(p,1.2)];
93
94 % apply the predictor modifier formula and store it in the table
95 % p_m = 0.87751
96 p_m = predictor_modifier(p,c_prev,p_prev);
97 values(5,3) = p_m;
98 values(5,4) = g(p_m,values(5,2));
99 old = p_m;
100
101 % apply the corrector formula to indices 4, 3, 2
102 % c = 0.90631
103 c = corrector(h,values(4,3),values(5,4),values(4,4),values(3,4));
104 c_prev = c;
105
106 % apply the corrector modifier formula and store it in the table
107 % c_m = 0.90288
108 c_m = corrector_modifier(c,p);
109 values(5,3) = c_m;
110 values(5,4) = g(c_m,values(5,2));
111
112 % check for convergence
113 % eps = 0.028091
114 eps = convergence(c_m, old);
115 old = c_m;
116
117 % apply the corrector formula to indices 4, 3, 2
118 % c = 0.91084
119 c = corrector(h,values(4,3),values(5,4),values(4,4),values(3,4));
120 c_prev = c;
121
122 % apply the corrector modifier formula and store it in the table

```

```

123 % c_m = 0.90695
124 c_m = corrector_modifier(c,p);
125 values(5,3) = c_m;
126 values(5,4) = g(c_m,values(5,2));
127
128 % check for convergence
129 % eps = 0.0044926
130 eps = convergence(c_m, old);
131 old = c_m;
132
133 % apply the corrector formula to indices 4, 3, 2
134 % c = 0.91156
135 c = corrector(h,values(4,3),values(5,4),values(4,4),values(3,4));
136 c_prev = c;
137
138 % apply the corrector modifier formula and store it in the table
139 % c_m = 0.90761
140 c_m = corrector_modifier(c,p);
141 values(5,3) = c_m;
142 values(5,4) = g(c_m,values(5,2));
143
144 % check for convergence
145 % eps = 0.00072122
146 eps = convergence(c_m, old);
147 old = c_m;
148
149 values
150
151 % values =
152 %
153 %          0          0          0.7          -0.21
154 %          1          0.3        0.6467        -0.12417
155 %          2          0.6        0.58891        0.077736
156 %          3          0.9        0.66257        0.44525
157 %          4          1.2        0.90761        1.2961
158
159
160
161 % ----- TASK 2 -----
162
163
164
165 % 1) the predicted value
166 % ANSWER: 2
167 % You would need both the n_y and the n+1_y of the predicted
168 % value, since you need first calculate and store the n+1_y value as the
169 % new predicted value before calculating the n+1_y modified predicted value
170 % using n_y.
171
172 % 2) the modified predicted value
173 % ANSWER: 1
174 % This value can be rewritten after each iteration, as you never need to
175 % retrieve two consecutive instances of it.
176
177 % 3) the corrected value (multiple times)
178 % ANSWER: 1
179 % This value can be rewritten after each iteration, as you never need to
180 % retrieve two consecutive instances of it.
181
182 % 4) the modified corrected value (once for each corrected value)
183 % ANSWER: 2

```

```
184 % You would only need both the n_y and n+1_y of the modified corrected
185 % value, since in order to calculate the error you need both the current
186 % and previous approximations.
187
```