```
function output = ZhouK_Prelab7()

format shortG



% ---------------------- TASK 1 ----------------------



% initialize variable for the step size
h = 0.3;

% initialize the first column of the table
values(:,1) = [0 1 2];

% initialize the second column of the table
values(:,2) = [0 0.3 0.6];

% fill in the given initial conditions for t=0, t=0.3 and t=0.6
values(:,3) = [0.7 0.64670 0.58891];

% intialize function handle for g
g = @(y,t) (1.2*y*t*t) - (0.3*y);

% initialize the fourth column of the table
values(1,4) = g(values(1,3), values(1,2));
values(2,4) = g(values(2,3), values(2,2));
values(3,4) = g(values(3,3), values(3,2));

% initialize function handle for the predictor
predictor = @(h,y,g2,g1,g0) y + (h*((23/12)*g2 - (4/3)*g1 + (5/12)*g0));

% initalize function handle for the predictor modifier
predictor_modifier = @(p,c_prev,p_prev) p + (0.9*(c_prev - p_prev));

% initialize function handle for the corrector
corrector = @(h,y,g3,g2,g1) y + (h*((5/12)*g3 + (2/3)*g2 - (1/12)*g1));

% intialize function handle for the corrector modifier
corrector_modifier = @(c,p) c - (0.1*(c - p));

% initialize function handle for the convergence
convergence = @(new, old) (new - old)/new;

% apply the predictor formula to indices 2, 1, 0
% p = 0.65702
p = predictor(h,values(3,3),values(3,4),values(2,4),values(1,4));
old = p;

% enter in the fourth row to the table
values(4,:) = [3 0.9 p g(p,0.9)];

% apply the corrector formula to indices 3, 2, 1
% c = 0.66275
c = corrector(h,values(3,3),values(4,4),values(3,4),values(2,4));
c_prev = c;

% apply the corrector modifier formula and store it in the table
% c_m = 0.66218
c_m = corrector_modifier(c,p);
```

```
values(4,3) = c_m;
values(4,4) = g(c_m,values(4,2));

% check for convergence
% eps = 0.0077834
eps = convergence(c_m, old);
old = c_m;

% apply the corrector formula to indices 3, 2, 1
% c = 0.66318
c = corrector(h,values(3,3),values(4,4),values(3,4),values(2,4));
c_prev = c;

% apply the corrector modifier formula and store it in the table
% c_m = 0.66257
c_m = corrector_modifier(c,p);
values(4,3) = c_m;
values(4,4) = g(c_m,values(4,2));

% check for convergence
% eps = 0.00058808
eps = convergence(c_m, old);
old = c_m;

% apply the predictor formula to indices 3, 2, 1
% p = 0.87197
p_prev = p;
p = predictor(h,values(4,3),values(4,4),values(3,4),values(2,4));

% enter in the fourth row to the table
values(5,:) = [4 1.2 p g(p,1.2)];

% apply the predictor modifer formula and store it in the table
% p_m = 0.87751
p_m = predictor_modifier(p,c_prev,p_prev);
values(5,3) = p_m;
values(5,4) = g(p_m,values(5,2));
old = p_m;

% apply the corrector formula to indices 4, 3, 2
% c = 0.90631
c = corrector(h,values(4,3),values(5,4),values(4,4),values(3,4));
c_prev = c;

% apply the corrector modifier formula and store it in the table
% c_m = 0.90288
c_m = corrector_modifier(c,p);
values(5,3) = c_m;
values(5,4) = g(c_m,values(5,2));

% check for convergence
% eps = 0.028091
eps = convergence(c_m, old);
old = c_m;

% apply the corrector formula to indices 4, 3, 2
% c = 0.91084
c = corrector(h,values(4,3),values(5,4),values(4,4),values(3,4));
c_prev = c;

% apply the corrector modifier formula and store it in the table
```

```
% c_m = 0.90695
c_m = corrector_modifier(c,p);
values(5,3) = c_m;
values(5,4) = g(c_m,values(5,2));

% check for convergence
% eps = 0.0044926
eps = convergence(c_m, old);
old = c_m;

% apply the corrector formula to indices 4, 3, 2
% c = 0.91156
c = corrector(h,values(4,3),values(5,4),values(4,4),values(3,4));
c_prev = c;

% apply the corrector modifier formula and store it in the table
% c_m = 0.90761
c_m = corrector_modifier(c,p);
values(5,3) = c_m;
values(5,4) = g(c_m,values(5,2));

% check for convergence
% eps = 0.00072122
eps = convergence(c_m, old);
old = c_m;

values

% values =
%
%             0            0          0.7        -0.21
%             1          0.3       0.6467     -0.12417
%             2          0.6      0.58891     0.077736
%             3          0.9      0.66257      0.44525
%             4          1.2      0.90761       1.2961


% ----------------------- TASK 2 ----------------------


% 1) the predicted value
% ANSWER: 2
% You would need both the n_y and the n+1_y of the predicted
% value, since you need first calculate and store the n+1_y value as the
% new predicted value before calculating the n+1_y modified predicted value
% using n_y.

% 2) the modified predicted value
% ANSWER: 1
% This value can be rewritten after each iteration, as you never need to
% retrieve two consecutive instances of it.

% 3) the corrected value (multiple times)
% ANSWER: 1
% This value can be rewritten after each iteration, as you never need to
% retrieve two consecutive instances of it.

% 4) the modified corrected value (once for each corrected value)
% ANSWER: 2
```

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

```