Numerical Applications to Differential Equations

Assignment 2

This assignment is largely intended to introduce you to the debugging using Matlab.

1. Enter the MATLAB program of Figure 1 using the MATLAB editor. Remember to add clear all, clc, and close all statements at the beginning of all programs. The program is supposed to display all the elements in the array one element at a time.

```
%array to display
 values = [23, 17, 46, 15, 88, 34];

for values
  disp(values)
end
```

Figure 1, MATLAB Program to Display the Elements of an Array Containing Errors

   (a) Run the program and correct any syntax errors.

   (b) Set a breakpoint at the line: disp(values).

   (c) Debug the program by stepping through the program and watching the workspace and command window for incorrect behavior. Correct any logic errors. Reset the breakpoints if necessary and rerun the corrected script watching the workspace and command window.

   (d) Correct any style errors.

2. Enter the MATLAB program of Figure 2 using the MATLAB editor. Remember to add clear all, clc, and close all commands at the beginning of all programs.

```
% thirty random integers between -10 and 100
 v = round(-10.0 + 110*rand(1,30));
% sum all positive elements of array v

for k = 1:1:length(v)
  if (v >= 0)
  result = result + v
  end
end

disp('The sum of the positive elements in v is:  '),disp(result);
```

Figure 2, MATLAB Program to Sum positive Elements of an Array Containing Errors

   (a) Run the program and correct any syntax errors.

   (b) Using the MATLAB Editor, set breakpoint at the lines with the statements:
   for k = 1:1:length(v)
   result = result + v

   (c) Debug the program by stepping through the program and watching the workspace and command window for incorrect behavior. Correct any logic errors. Reset the breakpoints if necessary and rerun the corrected script watching the workspace and command window.

   (d) Correct any style errors.

3. Enter the MATLAB program of Figure 3 using the MATLAB editor. Set breakpoints and debug the program correcting any syntax, logic, and style errors.

```
% program reads in dollar amounts of each transaction and
% keeps track of total dollar amount of all transactions

totalAmount = 0.0;
numberTransactions = input('How many transactions:  ');

while (k <= numberTransactions)
  transcationAmount = input('Transaction amount (in dollars):  ')
  totalAmount = transcationAmount
  k = k + 1;
end

disp('The total transaction amount is:  '),didp(totalAmount);
```

Figure 3, MATLAB Program to Keep Track of Amount of all Transactions Containing Errors

4. The function of Figure 4a is supposed to evaluate a quadratic polynomial $g(x) = a_2x^2 + a_1x + a_0$

```
function gg = evaluatePoly(aa, xx)
% ------------------------------------------------------
% evaluatePoly.m
% ------------------------------------------------------
% evaluatePoly evaluates a quadratic polynomial function
% with polynomial coefficients aa = [a2, a1, a0] for the
% value(s) in xx
% ------------------------------------------------------
% syntax:  gg = evaluatePoly(a2, a1, a0, xx)
% aa is a 1 by 3 vector if polynomial coefficients
% xx is the values to evalaute the polynomial for
% gg is the polynomial function values
% ------------------------------------------------------
clear all;
clc;
close all;
gg = zeros(1,xx);
gg(xx) = a2*xx^2 + a1*xx + a0;
end
```

Figure 4a, evaluatePoly Function Containing Errors

Using the test program of Figure 4b, the function was determined to have errors.

Using the MATLAB debugger and the test program of Figure 4b, identify and correct the errors in the evaluatePoly Function of Figure 4a.

```
clear all;
clc;
close all;
coef = [1, -4, 2];
g = evaluatePoly(coef, 0);
disp(g);
g = evaluatePoly(coef, 5);
disp(g);
x = [-5:0.1:5];
g = evaluatePoly(coef, x);
figure(1), plot(x,g);
```

Figure 4b, Test Program for evaluatePoly Function of Figure 4a