READ CODE:

1. *(9 pts)* Given the following code:

```
A = [8,7,2,7;7,2,1,2;6,2,6,4;7,10,9,5];
disp('Before');
disp(A);

%Code that alters A goes here
[redacted]

disp('After');
disp(A);
```

Write down the code that performs the following tasks (you only need to write the portion of the code that would go in the rectangle above (could be one or many lines of code):

a. Replace the first and third rows of the A array each with the second row of the A array (see expected result below).

```
A([1 3], :) = A(2, :)
```

b. Set all interior points to zero (do not hard code for this particular A array, write generally).

```
A(2:end-1, 2:end-1) = 0
```

c. Set the first column equal to the element by element product of the first and second columns. Set the last column equal to the element by element product of the last and second to last columns (do not hard code for this particular A array, write generally).

```
A(:, 1) = A(:, 1) .* A(:, 2);
A(:, end) = A(:, end) .* A(:, end-1);
```

a. Expected result	b. Expected result	c. Expected result
Before 8 7 2 7 7 2 1 2 6 2 6 4 7 10 9 5 After 7 2 1 2 7 2 1 2 7 2 1 2 7 10 9 5	Before 8 7 2 7 7 2 1 2 6 2 6 4 7 10 9 5 After 8 7 2 7 7 0 0 2 6 0 0 4 7 10 9 5	Before 8 7 2 7 7 2 1 2 6 2 6 4 7 10 9 5 After 56 7 2 14 14 2 1 2 12 2 6 24 70 10 9 45

WRITE CODE:

8. *(15 pts)* Write a function code. The function name should be **`processData`**. There should be three inputs: an array or radii in units of ft, a calculation flag, and a conversion flag. There should be one output: the calculation based on the inputted array of radii. See variable names in the function header below. The function should process the **`radii`** array in three ways:

(1) **`flagCalc = "area": calc`** should be filled with areas corresponding to the **`radii`** array. Use the formula for area of a circle.
(2) **`flagCalc = "circumference": calc`** should be filled with circumferences corresponding to the **`radii`** array.
(3) **`flagCalc = "diameter": calc`** should be filled with diameters corresponding to the **`radii`** array.

Finally, if the third input, **`flagConvert`** is passed in as true, then you should do the above calculations in metric units (final units for area should be m^2, for circumference should be m, and for diameter should be m). Otherwise, keep the units in English.

```
%1 meter = 3.281*(1 foot)
function [calc] = processData(radii, flagCalc, flagConvert)
    if sum(radii<0)>1 % Check radii for negative values.
        disp("Array radii contains a negative value. Check
              input and try again.");
        calc = 0;
        return;
    end
    if flagConvert
        radii = radii * 3.281;
    end
    switch
        case "area"
            calc =(radii.^2)* pi;
        case "circumference"
            calc = radii * 2 * pi;
        case "diameter"
            calc= radii * 2;
        otherwise
            disp("Bad input. Try again.");
    end
end
```

WRITE CODE

9. *(8 pts)* For the previous problem, call your function **`processData`** four times. Your calls should demonstrate that you understand how to test the functionality of the code you wrote. Therefore, you will only receive credit if the four function calls test a wide range of functionality.

```
% check area with conversion
test_1 = processData(0:0.5:10, "area", true);
% check circumference
test_2 = processData(0:0.5:10, "circumference", false);
% check diameter
test_3 = processData(0:0.5:10, "diameter", false);
%Check negative radii
test_4 = processData(-5:0.5:5, "area", false);
% check bad flagCalc
test_5 = processData(0:0.5:10, "badinput", false);
```