Name: Andrew Brown	**Lab Time:** T 12:00
People Worked With:	**Websites Used:**
Time spent on zyBooks (hrs):2	**Time spent on lab (hrs):5**
Submission Instructions Turn all work in to Lab 5 on Gradescope (PDF) and Canvas (.zip file), even if it is not complete yet. If you are not finished, complete the assignment outside of lab and re-submit to Lab 5 on Gradescope and Canvas. All labs are typically due at the same time on Monday every week, but check Canvas if in doubt.	

Learning objectives:

- Use for loops to analyze data in a 2D array of grading data
    - By row (trends across grading item type)
    - By column (trends across class)
- Use a while loop to find the smallest value in a quadratic function

New MATLAB commands

These are highlighted in **bold** in the instructions below.

- `zeros( nRows, nCols )` – you've seen this before, but now we'll actually make a 2D array
- `m(j,k)` – access elements of a 2D array (jth row, kth column)
- `size( m, 1 )` – number of rows in a 2D array m
- `size( m, 2 )` – number of columns in a 2D array m
- `ASorted = sort(A)` – sort an array A

Lab Problems

Files to download to your Lab 5 Folder

- *AddNoiseData.m*
- *Lab5FindMinStart.m*
- *DataClass.csv – do a download, not a copy and paste, in order to keep formatting*

Getting Started

Anything for experimentation goes here with bullet points

- Try typing m = zeros(3, 5)
    - Double click on m in the variable window
    - How many rows does m have? How many columns?
- Type size(m). What do you get?
    - Try size(m, 1) and size(m,2).
- Try length(m) – what do you get?
    - DON'T use length with 2D arrays
- Run the script AddNoiseData.m and make sure it works
- Open up DataClass with Excel so you can see what it looks like
    - One row for each student
    - One column for each grade item (labs, homeworks, etc)
    - See AddNoiseData.m for column names
    - This is legitimate data, but I have added noise to make it unidentifiable

Problem 1

Plot last year's ENGR112 data by type.

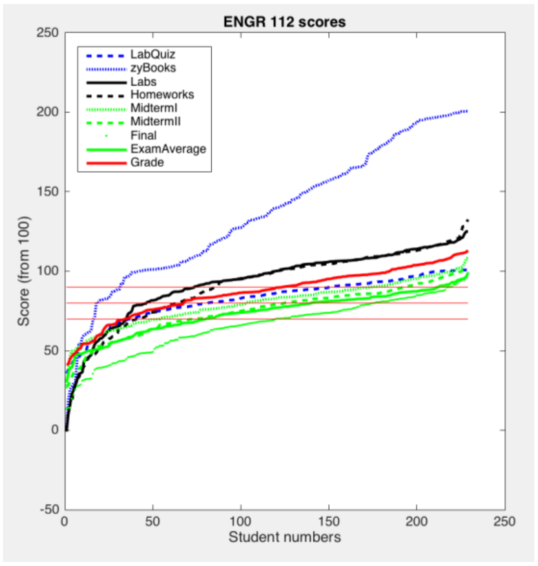
Deliverables:

1. Plot of data
   a. Each type of data
   b. Lines for grade divisions (70, 80, 90)
   c. Legend
2. Script
3. Command window output

Step by Step Instructions:

- Start a new script and copy the first lines from AddNoiseData.m (the ones that read in the data and create the string arrays that say what the column headers are and provide colors for the plot).
  - Run and make sure that you are reading data correctly – what size is dataN?
- Plot one column of data
  - colData = dataN(:, 1); % Gets the first column of the data
  - What happens if you just plot it? Try plotting it with markers…
  - You want sort().
    - Sort the data and plot it.
    - In this case you don't need to make a dependent variable t. You can just call plot(colData, '-b') and it will default to plotting 0, 1, 2, etc
- Wrap the code for the bit above in a **for** loop in order to plot all of the columns
  - Remember that size(dataN, 2) will give you the number of columns to plot over
  - Remember to change 1 to your loop index variable
- Pretty colors
  - You can use the strLineStyles to set a different color for each plot. Remember to use {}
  - You can use the strColumns in the legend as-is; to put the legend in the upper left hand corner, use the 'Location' option (google matlab legend).
- Adding the horizontal stripes
  - Use a for loop to plot one stripe at 70, one at 80, one at 90. Remember that the x values start at 0 and go to the number of students == number of rows.
    - plot([xStart, xEnd], [height, height], '-r');
      - where xStart is the first value (0), xEnd is the number of elements you plotted, and height is one of 70, 80, or 90
- Don't forget to add the titles, legend, and labels

Self-check:



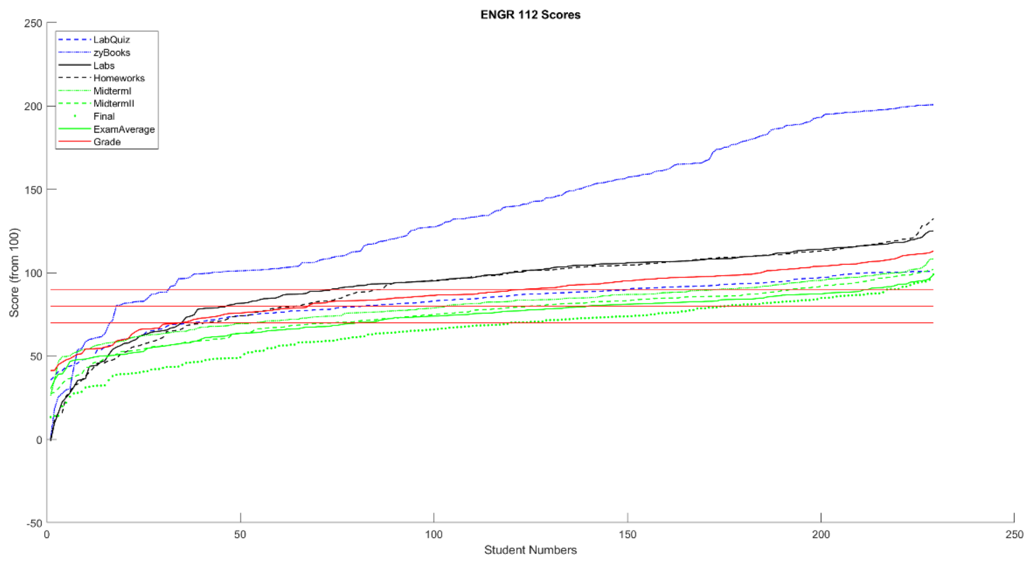
Grading Criteria:
[20 pts] [*Extracted column of data and sorted it]*
[20 pts] [*Used for loop to plot each type of data]*
[20 pts] [For loops use size of dataN (not hard-wired)]
[15 pts] [Red lines at 70, 80, 90, plotted using a for loop]
[15 pts] [Titles, labels, and legend]
[10 pts] [Comments]

Answer script here:

```
%Andrew Brown Lab 5 Problem 1

clc
clear
close all

%Plot last year's ENGR112 data by type

% Script to read in 2D array of grade data
% Also provides column names
dataN = csvread('DataClass.csv');
strColumns = {'LabQuiz', 'zyBooks', ...
    'Labs', 'Homeworks', ...
    'MidtermI', 'MidtermII', 'Final', ...
    'ExamAverage', 'Grade'};

strLineStyles = {'--b', ':b', '-k', '--k', ':g', '--g', '.g', '-g', '-r'};
%Line Styles
for i=1:size(dataN,2)
    hold on
    colData=dataN(:,i); %Gets the first column of the data.
    plot(sort(colData),strLineStyles{i},'linewidth',1) %plot each of the 9
columns sorted from high to low

end

counter=1:length(dataN); %make a counter from 1 to the length of dataN
```

```
for j=70:10:90 %count in intervals of 10
    hold on
    hLine=ones(1,length(dataN));%create an array of ones length dataN long
    plot(counter,j.*hLine,'-r') %plot 3 horizontal lines at 70 80 and 90
end

legend(strColumns,'Location','NW') %plot the legend with the given names in
the top right corner
title('ENGR 112 Scores') %title the graph
xlabel('Student Numbers') %add x label
ylabel('Score (from 100)') %add y label
```

Plot here:



Problem 2

Print out the following statistics from the data in problem 1:

- **Number of students who did better on the labs than the homeworks**
- **Number of students who did better on the homeworks than the final exam**
- **Average score for each category**
- **[Extra credit] Average difference between labs & homeworks and homeworks & final exam**
- **[Extra credit] Do the number counts with relational operators. You can calculate those numbers with a single line of code…**

Deliverables:

4. Script for calculating the above
5. Command window output
6. Must use for loops

Step by Step Instructions:

- Start a new script
  - Copy the first few lines from your Lab 1 problem (reading dataN and the strColumns)
  - As in problem 1 you'll need a for loop over each column. You can either copy your for loop from problem 1 or just re-type it
- Printing out the means of each column
  - There are two ways to do this:
    - One is to call mean with the entire matrix (refer to help on mean to see what will happen when you do this)
    - Use the colon operator to get out **all** of the elements for the column IN the for loop and then call mean on that (column) array.
      - data(:,1) – returns all the data from column 1
  - Either way you'll need a for loop that goes over each column (use size() command)
    - Use strColumns to print out the name of the column along with the mean
  - Note: The first method is more efficient.
- Counting students
  - First thing to do is declare an index variable for each of the columns you're interested in; that way, if they change later you don't have to re-write your code. Also makes it easier to read the code. Declare these before the for loop
    - indexLabColumn = 3;
    - …

- Create a for loop that goes over each STUDENT this time (i.e., number of rows…)
- Make a counting variable, eg, betterLabThanHomework = 0;
    - Again, this goes before the for loop starts
- Do just the homework score better than the lab score first. Hint: You'll need an if statement in the for loop
    - If you're struggling with either how to write the if or how to increment the counter variable, grab a TA
- Add another if statement for the homeworks and exams
- Overall this should look like:
    - Set the counting and index variables
    - for number of students
        - if student's lab score is bigger than their homework score
            - Add one to the betterLabThanHomework score
        - if …
    - Print out the counting variables

- Extra credit I
    - This should look like mean(lab – homework) – how do you get out ALL of the student lab scores all at once? Hint: colon operator
        - You should do this without a for loop for credit.
- Extra credit II
    - This should look like sum(lab > homework) – and like extra credit I you want to perform the comparison on ALL of the students all at once (not using a for loop)
        - lab > homework will return an ARRAY of zeros and ones, zeros for false, ones for true. Then just count the ones…

Self-check:

```
Average 82.X4 for item LabQuiz
Average 135.X2 for item zyBooks
Average 93.X0 for item Labs
Average 91.X9 for item Homeworks
Average 80.X1 for item MidtermI
Average 75.X9 for item MidtermII
Average 65.X6 for item Final
Average 73.X9 for item ExamAverage
Average 86.X4 for item Grade

Number of people with lab scores higher than homeworks: 1X5
 Average difference: 1.X0
Number of people with homework scores higher than final exam: 2X5
 Average difference: 25.X3
Total number of students: 2X9
```

Grading Criteria:
[20 pts] [Printing means with a for loop]
[10 pts] [Printing item name with mean]
[20 pts] [Calculating number of people Labs > homeworks]

[20 pts] [Calculating number of people homeworks > final exams]
[20 pts] [Average difference for each]
[10 pts] [Use size to get number of rows/columns]
[20 pts] [Extra credit I – must do without for loop]
[20 pts] [Extra credit II – must do without for loop]

Answer script here:

```
%Andrew Brown Lab 5 Problem 2

clc
clear

%Print out and calculate statistics from script 1

% Script to read in 2D array of grade data
% Also provides column names
dataN = csvread('DataClass.csv');
strColumns = {' LabQuiz', ' zyBooks', ...
    ' Labs', ' Homeworks', ...
    ' MidtermI', ' MidtermII', ' Final', ...
    ' ExamAverage', ' Grade'};

for i=1:size(dataN,2)
    colData=dataN(:,i); %Gets the first column of the data.
    fprintf(strcat('Average %0.2f for
item',strColumns{i},'\n'),mean(colData)); %Print out individual averages
end

%Calculate number of people with lab>HW and HW>Final
indexLabColumn=3; %index lab column
indexHWColumn=4; %index HW column
indexFinalColumn=7; %index Final exam column
betterLabThanHomework=0; %initialize Lab>HW variable
betterHWThanFinal=0; %initialize HW>Final exam variable
for j=1:size(dataN,1)
    if dataN(j,indexLabColumn)> dataN(j,indexHWColumn) %if Lab>HW
        betterLabThanHomework=betterLabThanHomework+1; %add 1 to variable
    end
    if dataN(j,indexHWColumn)> dataN(j,indexFinalColumn) %if HW>Final
        betterHWThanFinal=betterHWThanFinal+1; %add 1 to variable
    end
end

%Average difference between the labs and HWs and the HWs and final
lab=dataN(:,3); % all of the lab grades
HW=dataN(:,4); %all of the HW grades
final=dataN(:,7); % all of the final exam grades
LabHW=mean(lab-HW); %average difference of labs vs. HW
HWFinal=mean(HW-final); %average difference of HWs vs. final exam

%Print out results
fprintf('\nNumber of people with lab scores higher than homeworks:
%0.0f\n',betterLabThanHomework) %Print how many did better on labs than on
HW
```

```
fprintf('Average Difference: %0.2f\n',LabHW) % print average difference of
labs vs hw
fprintf('Number of people with homework scores higher than final exam:
%0.0f\n',betterHWThanFinal) %Print how many did better in HW than the final
exam
fprintf('Average Difference: %0.2f\n',HWFinal) %print average difference of
HW vs final exam
fprintf('Total number of students: %0.0f\n', length(colData)) %Total number
of students
```

Command window output

```
Average 82.44 for item LabQuiz
Average 135.62 for item zyBooks
Average 93.10 for item Labs
Average 91.19 for item Homeworks
Average 80.01 for item MidtermI
Average 75.19 for item MidtermII
Average 65.56 for item Final
Average 73.59 for item ExamAverage
Average 86.64 for item Grade

Number of people with lab scores higher than homeworks: 125
Average Difference: 1.90
Number of people with homework scores higher than final exam: 205
Average Difference: 25.63
Total number of students: 229
```

Problem 3

Find the (approximate) minimum of a function y(t) = $3t^2 + 2t + 40$. Start at t=-10 and search in increments of t = t+1 until you hit the bottom (next y value is bigger than the previous, instead of smaller), then keep cutting the search increments in half (0.5, 0.25, etc) until your divisions are less than 0.001.

Question 1: Once you hit the bottom how many steps (at most) will you take at each of the smaller step values (0.5, 0.25, etc?) Why?

Question 2: Why do you get different answers when you start at t = -10 and t = -9.5?

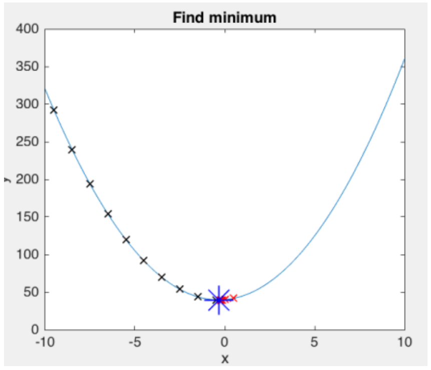
Deliverables:

7. Plot the next point in RED if it is bigger than the previous point
8. Plot the next point in BLACK if it is smaller than the previous point
9. Provide the plot for a starting value of -10
10. Print out the final t that you found for a starting value of t of -10 and -9.5
11. Answer the question

Step by Step Instructions:

- Start with the script Lab5FindMinStart.m
- STEP 1
  - Fill in the function and make sure it plots ok (looks like plot below, but without the X's)
- STEP 2
  - The while/if structure is already set up for you; you just have to edit the conditions.
    - Fill in the while condition
    - Fill in the if condition
    - Change t where indicated
    - Change deltaT where indicated
- Run the script with startT set to -9.5 and -10.0
  - Set break points to make sure t changes the way you expect
  - Plot
- The Questions
  - Set a break point in the while loop and run it to see what happens
    - Set a break point in each branch of the if statement
    - To answer the question, step through the code and see what happens to the values of t, delta t, yPrev and yNext.

Self-check: (plot is for starting t of -9.5)



```
Starting t is -9.50, minimum t is -0.2X0000, y is 39.XX7500, fminsearch -
0.333301, y is 39.666667
Starting t is -10.00, minimum t is 0.XX0000, y is 40.XX0000, fminsearch -
0.333313, y is 39.666667
```

Grading Criteria:
[10 pts] [*Function declaration]*
[10 pts] [*While condition]*
[10 pts] [*If statement condition]*
[10 pts] [deltaT set correctly in if statement*]*
[10 pts] [*t set correctly in if statements]*
[10 pts] [*Ran with starting value of -10.0]*
[10 pts] [*Ran with starting value of -9.5]*
[15 pts] [*Question 1 answer]*
[15 pts] [*Question 2 answer]*

Answer script here:

```
% Simplified version of find the minimum of a function
%  Basically walk down the gradient from one side
%  Start at t = -10 and keep adding deltaT = 1 to t until the next
%    value is bigger
%  At which point divide deltaT by 2 and continue...
%     until deltaT is smaller than 0.001

clear
clc
clf

% A parabola
% STEP 1 - write the anonymous function from the lab here
y = @(t) [(3*t^(2))+(2*t)+40];

% Plot it just to see what it looks like
fplot( y, [-10,10] );
hold on;
xlabel('t');
ylabel('y');
title('Find minimum');
```

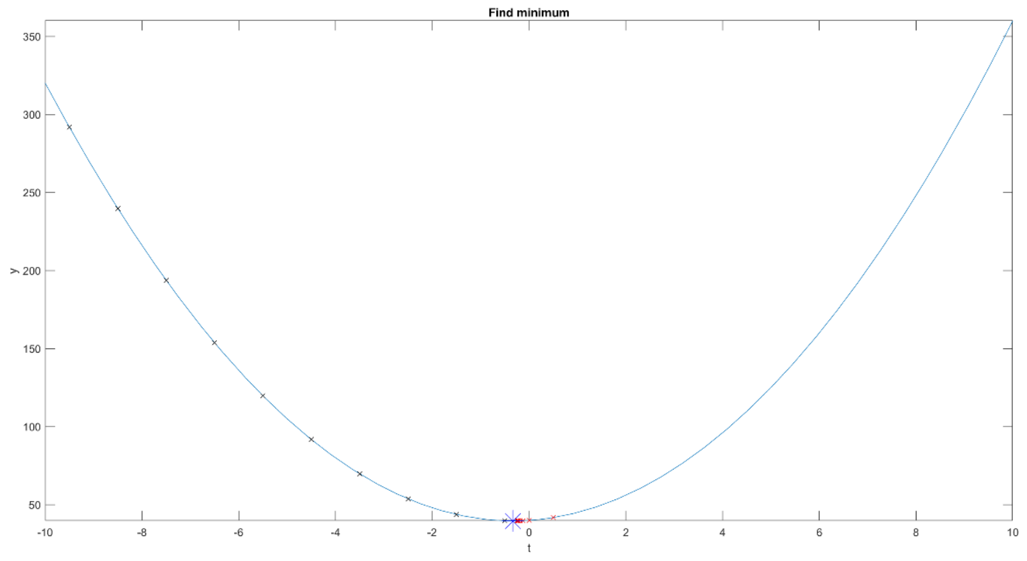
```
% Setting up before the while loop
% t to start at (save it for fprintf below)
tStart = -9.5;

% How much to keep incrementing t by
deltaT = 1;

% t will change in the while loop
t = tStart;

% Evaluate at t and t+deltaT
yPrev = y(t);
yNext = y(t + deltaT);

% Plot starting point
plot( t, yPrev, 'Xk');

% The while loop structure
% you have to fill in the actual conditions (where it says CHANGE)
%   and set t and deltaT
while deltaT>0.001
    if yNext>yPrev
        % plot in red
        plot( t+deltaT, yNext, 'Xr');
        % CHANGE: Cut the step size in half
        deltaT = deltaT/2;
    else
        % Next step is SMALLER than previous - take one deltaT step
        % plot in black
        plot( t+deltaT, yNext, 'Xk');
        % CHANGE: Take next step...
        t = t+deltaT;
    end
    % Re-evaluate yPrev and yNext with the changed t or deltaT values
    yPrev = y(t);
    yNext = y(t + deltaT);
end

% The t and y you found
fprintf('Starting t is %0.2f, minimum t is %0.6f, y is %0.6f, ', tStart, t,
y(t) );

% fminsearch does a much more sophisticated version of this, but basically
%  the same idea - call here to see what *it* would return
tMin = fminsearch(y, t );
fprintf('fminsearch %0.6f, y is %0.6f\n', tMin, y(tMin) );
% Add a blue * where the minimum actually is
plot( tMin, y(tMin), '*b', 'MarkerSize', 20);
```

Answer questions here:

Question 1: When it hits the bottom of the loop, you will take one step at most of the small er values because deltat is changing every time through the loop

Question 2: You get different outputs because you are giving a slightly different input. By starting at a different number, you get a slightly different output by the nature of this loop.

Plot here:



Command window output

```
Starting t is -9.50, minimum t is -0.250000, y is 39.687500, fminsearch -
0.333301, y is 39.666667
Starting t is -10.00, minimum t is 0.000000, y is 40.000000, fminsearch -
0.333313, y is 39.666667
>>
```

Extra Credit: Problem 1

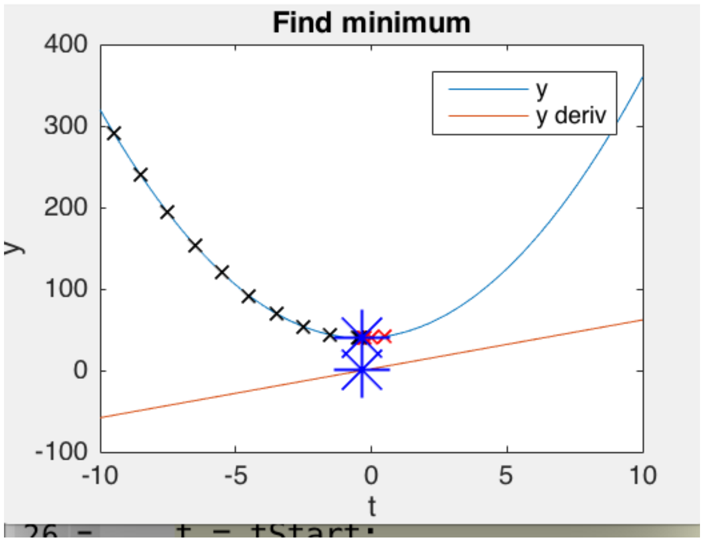
Add in a derivative check to problem 3. Plot the derivative (y'(t) = 6t + 2) as well as the function. Print out the t minimum that you find for a starting value of t = -10 and -9.5.
Question 1: Do you still get different values for different starting t values? Try more than just the t values given
Question 2: What would you have to change to get a more accurate answer?
Question 3: If you started at t = 10 and went to the left (deltaT = -1) would your code work? Why or why not?

Step by Step Instructions:

- Derivatives are simply how the function is changing as t increases. I.e., if you take a small step in t (t + deltaT) does the function increase or decrease? Analytic derivatives (like y'(t) above) let you do this for an infinitely small value of deltaT.
    - Look at the plot in the self-check. Up until the big blue star (the minimum) the derivative is negative. After that, the derivative is positive.
        - To properly check if you've "crossed" the bottom point, check to see if the derivative is still negative at the next t + deltaT value. If it's not, that means you've crossed the minimum and you need to divide deltaT in half.
- TODO
    - Add a new function yDeriv (given above as y') and plot it.
    - Plot the blue star on yDeriv using the t value from fminsearch (Copy the plot from the bottom of Problem 3's script and use yDeriv instead of y)
    - Adjust your if statement to ask if the derivative of y **at the next time step** is positive instead of just checking for the value being bigger
        - Actually, you can take out the value is bigger condition altogether..
- Questions
    - 1) Print out the t value with more decimal places… Also choose a bunch of different starting values
    - 2) Try changing the code to see if you can get a better answer
    - 3) Try it - change startT to 10 and deltaT to -1 – what happens?

Self-check:

```
Starting t is -9.50, minimum t is -0.333984XXXX, y is 39.666668XXX,
fminsearch -0.333332, y is 39.66666XX
Starting t is -10.00, minimum t is -0.3339843XXXX, y is 39.66666XX,
fminsearch -0.333332, y is 39.666667
```



Grading Criteria:
[15 pts] [Derivative function]
[10 pts] [Plot derivative with * at yDeriv(fminsearch result) = 0 as shown]
[10 pts] [Print t values at -9.5 and -10]
[20 pts] [Correct if condition]
[15 pts] [Question 1 answer]
[15 pts] [Question 2 answer]
[15 pts] [Question 3 answer]

Answer script here:

```
% Copy and paste your script here. Make sure your formatting looks the same as MATLAB, with size 10 font.
```

Question answers here

Question 1:
Question 2:
Question 3:

Plot here:

```
Save your plot as a png and paste it here.
```

Command window output

```
Paste output here.
```

zyBooks Challenge Exercises

Do the challenge activities for the following in Week 5. Now is also a good time to go back to the challenge exercises in Week 3 and try them.

1. **More while**
   a. While loop with branching
2. **Nested loops**
   a. Double summation
3. **Loops and arrays**
   a. Populating an array with a for loop
   b. Finding values in arrays
   c. Modifying an array's elements
   d. Modifying an array's elements using other elements
4. **Counting [optional]**
   a. Insect population
   b. While loop: Summation
5. **Remainder and modulus function**
   a. Compute change
6. **2D arrays: Introduction**
   a. 2D array: Stock trends
7. **Indexing an element in a 2D array**
   a. 2D array indexing: Finding the center element of a square array
8. **Indexing rows and columns using a single colon**
   a. Row indexing: Snow fall records
   b. Column indexing: Updating price tables using a single colon
9. **Indexing 2D arrays using the end keyword**
   a. Swapping last matrix columns