Appendices (Criterion D)

First Consultation with **client** (Identifying Problem and General Solution) - April 10/11th, 2021

- The client outlined his general interests in the stock market, as well as some specific measurements that he was interested in as a foundational way to look into a company before he would decide that he'd like to research a company further. We discussed which companies we'd like to focus on, specifically high profile ones because those carry historical data that is more easily accessible.

Second Consultation with **advisor** (Identifying a Solution to the Problem) - April 13th, 2021

- The advisor helped the student with determining the proper ways to plan the product solution for criterion B, with hand sketches of a simplified flow chart of how the user would navigate the class. It was determined at this meeting that the flow chart would mostly consist of boxes, because the user doesn't necessarily face "yes or no" choices, rather the more specific choice of menu for the user.

Mid-Way Consultation with **advisor** (Demonstrations) - April 24th, 2021

- The student presented to the advisor their rough drafts of planning stage, criterion B, which included a flow chart from draw.io and hand drawn screen sketch, demonstrating what the command line interface should represent in the final product. The advisor was overall satisfied with the progress of planning and understanding of the class methods structure, and approved moving forward with the project and fully realizing the final product code.

Final Presentation/Consultation with Client/Advisor - May 11th, 2021

- The client was overall very satisfied with the clear presentation of each option and analysis within the final product. They indicated that the program was moderately intuitive with the ability to switch between menus. In terms of recommendations for improvements, the client indicated an interest in seeing even larger long term calculations from the data files (since the .csv files extend up to five years worth of data). They also mentioned an interest in seeing a price to earnings ratio calculation in a future development of the product.

Copy of Final Coded Solution (See Below)

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;

public class dataAnalyzer
{

// instance variables - replace the example below with your own
   /**
    * Constructor for objects of class hpeData
    */
  public dataAnalyzer()
   {

   }

   /**
    * Method to read in the csv File.
    */
   public static ArrayList<String[]> readDataFromFile(String filename) throws
FileNotFoundException
   {
      Scanner myFile = new Scanner(new File(filename)); // Open the File
      // System.out.println ("Read Data 1");

      // ArrayList<String[]> dataHPE = new ArrayList<String[]>(); // Array of Row
Data
```

```
        // String[] row; // Array of Row Contents

        ArrayList <String[]> data = new ArrayList<String[]>(); // Array of Row Data
                String[] row; // Array of Row Contents



        // loop through all the lines in the file
        while (myFile.hasNextLine())
        {
          String line = myFile.nextLine();
          row = line.split(",");
          data.add(row);
        }

        myFile.close();
        // System.out.println(data);
        return data;

      }

      // Method to take a substring from a string in the csv file, remove $, and convert
the string to a double and return that value.
      public static double parsePrice(String dollarPrice)
      {
        String price = dollarPrice.substring(1);
        double newPrice = Double.parseDouble(price);
        return newPrice;
      }

      public static void clear()
      {
        for(int i = 0; i<50;i++)
        {
          System.out.println();
        }
      }
      public static void main(String[] args)
      {
        // Arrays for Stock Data
```

```
            ArrayList<String[]> dataHPE = new ArrayList<String[]>();
            ArrayList<String[]> dataAMZN = new ArrayList<String[]>();
            ArrayList<String[]> dataGOOGL = new ArrayList<String[]>();
            ArrayList<String[]> dataTSLA = new ArrayList<String[]>();
            ArrayList<String[]> dataFB = new ArrayList<String[]>();


            ArrayList<String[]> currentStockData = new ArrayList<String[]>();

          // String currentStockData = ""; // Current Stock Data Record

          Scanner keyboard = new Scanner(System.in);
          String input = "";

          while(!input.equals("Q"))
          {
            // clear();
            System.out.println("1. Read Data From File");
            System.out.println("2. Average of Last Month of Trade Days");
            System.out.println("3. Average of Last Year of Trade Days");
            System.out.println("4. Display Data");
            System.out.println("5. Lowest Trade Date in the Last Month of Trade Days");
            System.out.println("6. Lowest Trade Date in the Last Year of Trade Days");
            System.out.println("7. Highest Trade Date in the Last Month of Trade Days");
            System.out.println("8. Highest Trade Date in the Last Year of Trade Days");
            System.out.println("Q. Quit");

            input = keyboard.nextLine();

            if (input.equals("1"))
            {
              clear();
              System.out.println("Please enter the name of the stock to analyze: "); //
Determine the Stock File to Load
              System.out.println("(1.) Hewlett Packard Enterprise, HPE"); // Determine the
Stock File to Load
              System.out.println("(2.) Amazon, AMZN"); // Determine the Stock File to
Load
              System.out.println("(3.) Alphabet, GOOGL"); // Determine the Stock File to
Load
```

```
                    System.out.println("(4.) Tesla, TSLA"); // Determine the Stock File to Load
                    System.out.println("(5.) Facebook, FB"); // Determine the Stock File to Load

                    // String stockName = keyboard.nextLine();
                    String inputStockType = keyboard.nextLine(); // Record the Stock Type to
Load
                    // currentStockType = integer.parseInt(inputStockType); // Set the Inputed
Stock (Company) (Loaded) as the current Stock Company

                    try
                    {
                      if (inputStockType.equals("1"))
                      {
                         dataHPE = readDataFromFile("hpeDataNew.csv");
                         clear();
                         System.out.println("Succesfully imported data for " + inputStockType);
                         currentStockData = dataHPE; // Set: Current Stock Data (Array)
                                        System.out.println();


                      }
                      else if (inputStockType.equals("2"))
                      {
                         // System.out.println("Amazon");
                         dataAMZN = readDataFromFile("amazonData.csv");
                         clear();
                         System.out.println("Succesfully imported data for " + inputStockType);
                         currentStockData = dataAMZN;
                                             // System.out.println("Test2");

                         System.out.println();


                      }
                      else if (inputStockType.equals("3"))
                      {
                         dataGOOGL = readDataFromFile("alphabetData.csv");
                         clear();
                         System.out.println("Succesfully imported data for " + inputStockType);
                         currentStockData = dataGOOGL;
                    System.out.println();


            }
            else if (inputStockType.equals("4"))
            {
               dataTSLA = readDataFromFile("teslaData.csv");
               clear();
               System.out.println("Succesfully imported data for " + inputStockType);
               currentStockData = dataTSLA;

                    System.out.println();


            }
            else if (inputStockType.equals("5"))
            {
               dataFB = readDataFromFile("facebookData.csv");
               clear();
               System.out.println("Succesfully imported data for " + inputStockType);
               currentStockData = dataFB;

                    System.out.println();


            }
            else
            {
               clear();
               System.out.println("Invalid input");
               System.out.println();

            }
        }
        catch(FileNotFoundException e)
        {
           System.out.println("No stock data for " + inputStockType);
        }
    }
```

```
                else if (input.equals("2"))
                {
                    double totalTwenty = 0;
                    double finalAverage = 0;
                    for (int i = 1; i < 21; i++)
                    {
                        double dailyClose = parsePrice(currentStockData.get(i)[1]);
                        totalTwenty += dailyClose;
                    }
                    if(currentStockData.size() > 1)
                    {

                        double average = (double)totalTwenty / 20;

                        // average = average * 100;
                        // finalAverage = Math.round(average)/100;
                        // for rounding: convert to string and take substring with only two values
after the decimal
                        clear();
                        System.out.println();
                        System.out.println("The average trading price over the last month of trade
days is $" + String.format("%.2f", average));
                    }
                    System.out.println("Press 'Enter' when ready to return to main menu");
                    System.out.println();
                    input = keyboard.nextLine();
                }
                    else if (input.equals("3"))
                    {
                        double totalYear = 0;
                        double finalAverage = 0;
                        for (int i = 1; i < 253; i++)
                        {
                            double dailyClose = parsePrice(currentStockData.get(i)[1]);
                            totalYear += dailyClose;
                        }
                        if(currentStockData.size() > 1)
                        {
                            double average = (double)totalYear / 252;
                    // average = average * 100;
                    // finalAverage = Math.round(average)/100;
                    clear();
                    System.out.println("The average trading price over the last year of trade
days is $" + String.format("%.2f", average));
                  }
                  System.out.println("Press 'Enter' when ready to return to main menu");
                  input = keyboard.nextLine();
                }

                else if (input.equals("4"))
                {

                clear();
                System.out.println("How much data would you like to display from the current
company? Selected below.");
                System.out.println("1. Print data for the last month of trade days."); //
Determine the Stock File amount to Load
                System.out.println("2. Print data for the last six months of trade days"); //
Determine the Stock File amount to Load
                System.out.println("3. Print data for the last year of trade days"); // Determine
the Stock File amount to Load
                String inputStockType = keyboard.nextLine();
                // try
                // {
                  if (inputStockType.equals("1"))
                  {
                    int count = 0;
                    for (String[] row : currentStockData)
                    {
                      if (count < 23)
                      {
                      System.out.println(Arrays.toString(row));
                      }
                      count++;
                    }
                    System.out.println("Press 'Enter' when ready to return to main menu");
                    System.out.println();
                    input = keyboard.nextLine();
                  }
```

```
                    if (inputStockType.equals("2"))
                    {
                        int count = 0;
                        for (String[] row : currentStockData)
                        {
                            if (count < 177)
                            {
                            System.out.println(Arrays.toString(row));
                            }
                            count++;
                        }
                        System.out.println("Press 'Enter' when ready to return to main menu");
                        System.out.println();
                        input = keyboard.nextLine();
                    }
                    if (inputStockType.equals("3"))
                    {
                        int count = 0;
                        for (String[] row : currentStockData)
                        {
                            if (count < 253)
                            {
                            System.out.println(Arrays.toString(row));
                            }
                            count++;
                        }
                                System.out.println("Press 'Enter' when ready to return to main
menu");
                                System.out.println();
                                input = keyboard.nextLine();
                }
            // }

            // catch(FileNotFoundException e)
            // {
                // System.out.println("No stock data for " + inputStockType);
            // }
        }
            else if (input.equals("5"))
            {
                double lowestDay = parsePrice(currentStockData.get(1)[1]);

                for (int i = 1; i < 21; i++)
                {
                   double dailyClose = parsePrice(currentStockData.get(i)[1]);
                   if (dailyClose < lowestDay)
                   {
                     lowestDay = dailyClose;
                   }
                   // else lowestDay = lowestDay;
                }
                if(currentStockData.size() > 1)
                {
                   clear();
                   System.out.println();
                   System.out.println("The lowest trading price over the last month of trade days
is $" + lowestDay);
                }
                System.out.println("Press 'Enter' when ready to return to main menu");
                System.out.println();
                input = keyboard.nextLine();
                }
                else if (input.equals("6"))
                {
                double lowestDay = parsePrice(currentStockData.get(1)[1]);

                for (int i = 1; i < 253; i++)
                {
                   double dailyClose = parsePrice(currentStockData.get(i)[1]);
                   if (dailyClose < lowestDay)
                   {
                     lowestDay = dailyClose;
                   }
                   // else lowestDay = lowestDay;
                }
                if(currentStockData.size() > 1)
                {
                   clear();
                   System.out.println();
```

```
                        System.out.println("The lowest trading price over the last year of trade days is
$" + lowestDay);
                    }
                    System.out.println("Press 'Enter' when ready to return to main menu");
                    System.out.println();
                    input = keyboard.nextLine();
                    }
                    else if (input.equals("7"))
                    {
                    double highestDay = parsePrice(currentStockData.get(1)[1]);

                    for (int i = 1; i < 21; i++)
                    {
                        double dailyClose = parsePrice(currentStockData.get(i)[1]);
                        if (dailyClose > highestDay)
                        {
                          highestDay = dailyClose;
                        }
                        // else lowestDay = lowestDay;
                    }
                    if(currentStockData.size() > 1)
                    {
                        clear();
                        System.out.println();
                        System.out.println("The highest trading price over the last month of trade days
is $" + highestDay);
                    }
                    System.out.println("Press 'Enter' when ready to return to main menu");
                    System.out.println();
                    input = keyboard.nextLine();
                    }
                    else if (input.equals("8"))
                    {
                    double highestDay = parsePrice(currentStockData.get(1)[1]);

                    for (int i = 1; i < 253; i++)
                    {
                        double dailyClose = parsePrice(currentStockData.get(i)[1]);
                        if (dailyClose > highestDay)
                        {
```

```
                        highestDay = dailyClose;
                      }
                      // else lowestDay = lowestDay;
                    }
                    if(currentStockData.size() > 1)
                    {
                      clear();
                      System.out.println();
                      System.out.println("The highest trading price over the last year of trade days is
$" + highestDay);
                    }
                    System.out.println("Press 'Enter' when ready to return to main menu");
                    System.out.println();
                    input = keyboard.nextLine();
                    }
          }
      }
}
```