Program

StockPrice (main) Class:

```
import java.util.ArrayList;
import java.util.Scanner;

public class StockPrice {

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        // initialize a targetPrice variable to save the user entered target
price value
        float targetPrice;

        // Array of stock price;
        Float[] stock_Prices = { 540.5f, 920.2f, 811.8f, 103.4f, 1002.9f, 700.3f,
713.1f, 1600.0f, 139.9f, 680.3f };

        // Initializing the arrayList same as array of stock price
        ArrayList<Float> stock_Prices_Array_List = new ArrayList<>();
        for (float price : stock_Prices) {

            // add methods add the value to the arrayList
            stock_Prices_Array_List.add(price);
        }

        // calling calculateAveragePrice function to get the average of the stock
prices
        // with array and arrayList
        float averagePriceArray = Helper.calculateAveragePrice(stock_Prices);
        float averagePriceArrayList =
Helper.calculateAveragePrice(stock_Prices_Array_List.toArray(new Float[0]));

        System.out.println("Average Stock Price (Array) = " + averagePriceArray);
        System.out.println("Average Stock Price (ArrayList) = " +
averagePriceArrayList);

        /*
         * calling findMaximumPrice function to get the maximum price from the
stock
         * prices with array and arrayList
         */
        float maxPriceArray = Helper.findMaximumPrice(stock_Prices);
        float maxPriceArrayList =
Helper.findMaximumPrice(stock_Prices_Array_List.toArray(new Float[0]));

        System.out.println("Maximum Stock Price (Array) = " + maxPriceArray);
        System.out.println("Maximum Stock Price (ArrayList) = " +
maxPriceArrayList);

        System.out.print("Enter Target Price: ");

        // handling error using try catch because may be user enter any wrong
value i.e.
        // String
        try {
            // getting target price value from user
            targetPrice = scan.nextFloat();
        } catch (Exception e) {
            scan.close();
            throw new Error("Invalid Input!");
        }
        scan.close();

        /*
         * calling countOccurrences function to get number of occurance of target
price
         * value entered by user with array and arrayList
         */
        int occurrenceCountArray = Helper.countOccurrences(stock_Prices,
targetPrice);
        int occurrenceCountArrayList =
Helper.countOccurrences(stock_Prices_Array_List.toArray(new Float[0]),
                targetPrice);

        System.out.println("Occurrence Count of " + targetPrice + " (Array) = " +
occurrenceCountArray);
        System.out.println("Occurrence Count of " + targetPrice + " (ArrayList) =
" + occurrenceCountArrayList);

        /*
         * Calling helper's computeCumulativeSum function to get the cummulative
sum
         * arrayList
         */
        ArrayList<Float> cumulativeSumArrayList =
Helper.computeCumulativeSum(stock_Prices_Array_List);
```

```
        System.out.println("Cumulative Sum of Stock Prices (ArrayList) = " +
cumulativeSumArrayList);
    }
}
```

Helper Class:

```
import java.util.ArrayList;

//This is a helper class which contain all the calculation functions
public class Helper {

    // calculateAveragePrice function takes array of type Float as input and
return the average float number (price);
    public static float calculateAveragePrice(Float[] prices) {
        float sum = 0;
        for (float price : prices) {
            sum += price;
        }
        return sum / prices.length;
    }

    // findMaximumPrice takes array of type Float as input and return the maximun
float number (price);
    public static float findMaximumPrice(Float[] prices) {
        float maxPrice = 0;
        for (float price : prices) {
            if (price > maxPrice) {
                maxPrice = price;
            }
        }
        return maxPrice;
    }

    // countOccurrences function takes array of type Float and a target float
number as input and return the number of occurance of that target number;
    public static int countOccurrences(Float[] prices, float targetPrice) {
        int num = 0;
        for (float price : prices) {
            if (price == targetPrice) {
                num++;
            }
        }
        return num;
    }

    // computeCumulativeSum function takes array of type Float as input and
return the cummulative sum of given array;
    public static ArrayList<Float> computeCumulativeSum(ArrayList<Float> prices)
{
        ArrayList<Float> cumSum = new ArrayList<>();
        float sum = 0;
        for (float price : prices) {
            sum += price;
            cumSum.add(sum);
        }
        return cumSum;
    }
}
```

Space and Time Complexity

Calculate the Average Stock Price

- **Time Complexity:**

  n mean number of array in the array or arrayList.

  - Array: O(n)
  - ArrayList: O(n)
- **Space Complexity:**
  - Array: O(1) - Constant space complexity. Only one variable (**sum**) is used.
  - ArrayList: O(1) - Constant space complexity. Only one variable (**sum**) is used.

Find the Maximum Stock Price

- **Time Complexity:**
  - Array: O(n)
  - ArrayList: O(n)
- **Space Complexity:**
  - Array: O(1) - Constant space complexity. Only a single variable (**maxPrice**) is used.
  - ArrayList: O(1) - Constant space complexity. Only a single variable (**maxPrice**) is used.

Determine the Occurrence Count of a Specific Price

- **Time Complexity:**
  - Array: O(n)
  - ArrayList: O(n)
- **Space Complexity:**
  - Array: O(1) - Constant space complexity. Only a single variable (**count**) is used.
  - ArrayList: O(1) - Constant space complexity. Only a single variable (**count**) is used.

Compute the Cumulative Sum of Stock Prices

- **Time Complexity:**
    - Array: O(n)
    - ArrayList: O(n)
- **Space Complexity:**
    - Array: O(1) - Constant space complexity. Only one variable (**sum**) is used.
    - ArrayList: O(n) - Linear space complexity. A new ArrayList is created to store cumulative sums.

The End