Array Assignments.

1. Write the code for the following problem. Assign 10 last names to an array. Write a function to display the names. Write another function to display the names in reverse order.

| Component | Description |
|---|---|
| **Input** | 1. **Array of Last Names**: Predefined array containing 10 last names (e.g., `["Smith", "Johnson", ...]`). |
| **Process** | 1. **Function to Display Names**: Loop through the array and display each name in the original order. |
| | 2. **Function to Display Names in Reverse**: Loop through the array in reverse order to display names. |
| **Output** | 1. **Original Order**: Display all names in the array in the original order. |
| | 2. **Reverse Order**: Display all names in reverse order. |

2. Write the code for the following problem. Add another array to problem 1 above. This array should contain exam score for the respective students. That is, the first name goes with the first score etc. These are called parallel arrays. Also modify the display functions to include exam score array in addition to the last name array.

| | |
|---|---|
| **Input** | 1. **File Data**: Read from a file where each line contains:<br>- `Student Last Name` (string)<br>- `Score` (integer)<br>- `Exam Score` (integer). |
| **Process** | 1. **Load Data**: Read the file line by line and split each line into three arrays/lists:<br>- `last_names` for student names.<br>- `scores` for scores.<br>- `exam_scores` for exam scores. |
| | 2. **Display Function**: Modify the display function to show data from all three arrays, ensuring corresponding data is displayed together (parallel arrays). |
| | 3. **Search Function**: Use the `last_names` array for searching.<br>- If the name is found, retrieve the index and display the corresponding score and exam score from the other arrays.<br>- If not found, display an appropriate message. |

| | |
|---|---|
| **Output** | 1. **All Data**: Display all student last names, scores, and exam scores. |
| | 2. **Search Results**: Display the last name, score, and exam score for the searched student if found, or a message indicating the name is not in the list. |

3. Write the code for the following problem. The data to load is lastname and score. You can do this from a file. Add a function to problem to display the last name and highest, last name and lowest. Hint: for highest initialize a variable to 0 (high_var). If the array value is higher than the high_var then set high_var to the array value and set high_index to the position of the array. Proceed through the array until you get to the end. Do the same for finding the lowest using low_var set to 999 (higher than the highest value).

| | |
|---|---|
| **Input** | 1. **File Data**: Read from a file where each line contains:<br>- `Last Name` (string)<br>- `Score` (integer). |
| **Process** | 1. **Load data from file**: Read each line of the file, split the data into `last name` and `score`, and store them in separate arrays/lists. |
| | 2. **Initialize variables**:<br>- `high_var` to 0 to track the highest score.<br>- `low_var` to 999 to track the lowest score. |
| | 3. **Loop through scores**:<br>- Compare each score with `high_var`. If higher, update `high_var` and store the corresponding index. |
| | - Compare each score with `low_var`. If lower, update `low_var` and store the corresponding index. |
| | 4. **Retrieve names**: Use the stored indices of `high_var` and `low_var` to get the corresponding last names from the list. |
| **Output** | 1. **Highest Score**: Display the last name and the highest score.<br>2. **Lowest Score**: Display the last name and the lowest score. |

4. Load list of 10 Player Names and Batting Averages from a file into arrays. (Create your own file with two items: player last name and batting average, i.e. 0.267, 0.300 etc). Write a function to display the arrays. Then use a while loop to repeatedly ask the user for a last name. Write another function to search for the last name in the array and then display last name and batting average when found.

| | |
|---|---|
| **Input** | 1. **File Data**: Read from a file where each line contains:<br>- `Player Last Name` (string)<br>- `Batting Average` (float). |
| | 2. **User Input**: A last name entered by the user (string) for searching. |
| **Process** | 1. **Load Data**: Read each line from the file, split the data into two arrays/lists: one for `last names` and one for `batting averages`. |
| | 2. **Function to Display Data**: Write a function to print both arrays, showing all player names and their batting averages. |
| | 3. **Search Function**:<br>- Accept the `last name` as input.<br>- Search for the name in the `last names` array using a loop. |
| | - If found, retrieve the index and display the corresponding batting average.<br>- If not found, display an appropriate message. |
| | 4. **Repeat Search**: Use a `while` loop to allow repeated searches for last names until the user decides to quit. |
| **Output** | 1. **Player Data**: Display all player names and their batting averages. |
| | 2. **Search Results**: Display the searched last name and batting average if found, or a message indicating the name is not in the list. |

5. Modify 4 above to display a message, “Name not found” when the name is not in the list.

Example to be provided.

Load 10 employee last names and salaries into parallel arrays. Write a function to display the last names and salaries. Display the last names in reverse order. Write a function to find the employee with the highest salary. Write a loop to sum and display total of all salaries. Repeatedly ask the user for a name. Display the name and salary when found. Display message “Employee Not Found” when the last name is not in the list.