**Question 1 (ArrayLists)**
Write a function that takes an ArrayList of integers and removes any duplicate values, keeping only the first occurrence.

**Question 2 (ArrayLists)**
Create a function with two ArrayLists of integers as parameters. Merge the two lists into a third ArrayList without duplicates. Return the resulting ArrayList

**Question 3(ArrayLists)**
Create a function with two ArrayLists of integers as parameters. Return an ArrayList that contains integers that exist in both ArrayLists.

**Question 4(ArrayLists)**
Write a program that first creates an integer ArrayList with integers from 0 to 100 ( **[0, 1, 2, …,100]** ), then filters the ArrayList of integers keeping only the even numbers, and print the result.

***Questions 5, 6, 7.and 8 are follow-up questions***
**Question 5(ArrayLists, this, classes)**
Define a class "Book" with attributes title(String), author(String), and price(double). Create a constructor using **this** keyword which initializes all the given attributes. Write a function that takes maxPrice (double) and a books list (ArrayList<Book>) as parameters. This function should filter out all the Books that have their price attribute bigger than maxPrice.

**Question 6 (ArrayLists, sorting)**
Write a function that returns a sorted ArrayList based on price (low to high).

**Question 7(ArrayLists)**
Write a function that returns a sorted ArrayList based on price (high to low). Do not sort the ArrayList. Instead, call the function from the question 6, and reverse the arraylist.

**Question 8(ArrayLists, classes)**
Write a function with the parameter requestedAuthor(String) that returns all the books with authors matching the requestedAuthor parameter.

**Question 9(ArrayLists)**
Write a program to compare two ArrayLists and determine if they are equal (contains the same elements in the same order).

**Question 10(ArrayLists)**
Write a function that counts the frequency of each element in the list. Print the frequency information. (Example output: 1 occurs 3 times, 5 occurs 2 times, … )

**Question 11(ArrayLists)**
Create a 2D ArrayList (an ArrayList of ArrayLists) to store a matrix of integers. Write a function that prints it in matrix form.

**Question 12(ArrayLists)**
Write two functions that finds the maximum and minimum elements in an ArrayList of integers.

**Question 13(ArrayLists)**
Implement a function shiftList(ArrayList<Integer> list, int k) that shifts the elements of the ArrayList to the right by k positions. The resulting ArrayList should have the same size as the initial array. Shifted array elements that exceed the size limit should appear in the front of the ArrayList.

**Question 14(ArrayLists)**
Implement a function removeElement(ArrayList<String> list, String element) that removes all occurrences of a specified string from the ArrayList.

**Question 15(ArrayLists)**
Implement a function splitList(ArrayList<Integer> list, int threshold) that splits the ArrayList into two separate ArrayLists: one containing values less than threshold, and the other containing values greater than or equal to threshold. Print both lists.

**Question 16 (ArrayLists)**
Implement a function concatenateLists(ArrayList<String> list1, ArrayList<String> list2) that concatenates two ArrayLists of strings into a single list and returns it.

**Question 17(ArrayLists)**
Implement a function findDuplicates(ArrayList<Integer> list) that returns an ArrayList of elements that appear more than once in the input list.

**Question 18(ArrayLists)**
Implement a function isPalindrome(ArrayList<Character> list) that checks whether the list of characters forms a palindrome (same forwards and backwards) and returns true or false.

**Question 19(ArrayLists)**
Implement a function longestIncreasingSubsequence(ArrayList<Integer> list) that returns a new ArrayList containing the longest increasing subsequence of numbers from the input list.

**Question 20(ArrayLists)**
Implement a function findMode(ArrayList<Integer> list) that returns the element that appears most frequently in the list. If multiple elements have the same highest frequency, return any one of them.