**Coding Questions**

**Write a Java program to count and display the frequency of each element in an arraylist.**

**Write a Java program to find the maximum frequency of any element in an integer arraylist.**

**Write a Java program to find the smallest (minimum) element in the stack.**

**Write a Java program to remove all duplicate elements from an integer array. Print the array after removing duplicates.**

**Write a Java program to convert each decimal number in an integer array to its hexadecimal equivalent and print both values for each element.**

**Write a Java program to convert each decimal number in an integer array to its octal equivalent and print both the decimal and octal values for each element.**

**Create a class with two integer fields: Age and Marks. Implement the Comparable interface so that objects of this class are compared based on the Age field in ascending order.  
Write a Java program to:**

* **Store a list class objects in a LinkedList.**
* **Add at least three objects with different ages and marks.**
* **Sort the list based on the Age of the objects (from youngest to oldest)**
* **Print the ages of all objects in the sorted order.**

**Write a Java program that validates user age for registration. If the age is below 18, throw a custom checked exception named UnderageException. Handle the exception gracefully and display an appropriate error message.**

**Write a Java program to reverse the order of elements in a given list of integers and display the reversed list.**

**Write a Java program to find and print the second smallest distinct element in a given integer array.  
If no such element exists (e.g., all elements are equal or there is only one element), print an appropriate message.**

**Define a functional interface in Java for adding two numbers. Use a lambda expression to implement this interface and add two integers. Print the result.**

**Write a Java program that uses the built-in Predicate functional interface to determine whether numbers in an array are even or odd.**

**Write a Java program that uses the built-in Function functional interface to calculate the length of a given string using a lambda expression. Print the length of the string "LambdaExpression" as output.**

**Write a Java program to check whether each string stored in a Stack ends with the uppercase letter "A", using a lambda expression with the built-in Function functional interface.**

**Write a Java program to check whether each element in a list of integers is a multiple of 5 using the built-in Predicate functional interface and a lambda expression. Print the result for each element.**

**Write a Java program that uses multiple try-catch blocks to handle different exceptions. In your program:**

* **Try to divide two numbers, where the denominator might be zero, and handle ArithmeticException.**
* **Then, try to access an out-of-bounds index in an array, and handle ArrayIndexOutOfBoundsException.**
* **Use separate try-catch blocks for each exception, printing appropriate error messages.**

**Write a Java program to sort a Vector of integers in ascending order and print both the original and the sorted Vector.**

**Write a Java program to check whether the parentheses and brackets in a given string are balanced. Use a stack to perform this check. The string may contain any combination of '(', ')', '{', '}', '[' and ']'. Print "Balanced" if all opening brackets are properly closed in correct order, or "Not Balanced" otherwise. Test your code with several inputs.**

**Write a Java program to reverse a string using a stack. Display both the original and the reversed string.**

**Write a Java program that creates a Map (such as HashMap) with some key-value pairs. Then, check if a given key is present in the map. If the key exists, print its corresponding value; otherwise, print "Key not found."**

**Write a Java program that stores several string values in a Map (with any key type). Iterate through the map and find all values that end with the uppercase letter 'H' (case-sensitive). Collect these values into a list and print the list of such strings.**

**Write a menu-driven Java program to implement stack operations using the java.util.Stack collection class. The program should allow the user to perform the following operations:**

1. **Push an element onto the stack.**
2. **Pop (remove) the top element from the stack.**
3. **Peek (view) the top element of the stack without removing it.**
4. **Display all elements of the stack.**
5. **Exit the program.**

**Write a Java program to reverse a stack using another stack (auxiliary stack) and standard stack operations.**