**README.md**

**Take-Home Tana Test – Java Solutions**

**Environment**

* **Programming Language:** Java 21 (OpenJDK)
* **Compiler:** javac 21.0.7
* **Runtime:** Java SE Runtime Environment (OpenJDK)

**Project Structure**

take-home-tana-test

* Question1\_GetLongestString.java
* Question2\_UniqueSumCombinations.java
* Question3\_ArrayReduction.java
* Question4\_FirstUniqueProduct.java
* Question5\_ClosestMinimumDistance.java
* Question6\_TopThreeCommonWords.java
* Question7\_RotateList.java
* README.md

**How to Run**

1. Open a terminal in the project directory.
2. Compile the Java file:

javac Question1\_GetLongestString.java

1. Run the program:

java Question1\_GetLongestString

1. Repeat steps 2 and 3 for each question file.

**Solutions Summary**

**Question 1 – Longest Valid String**

* **Approach:** Checks each string to ensure no consecutive identical characters and only valid characters are allowed. Returns the longest valid string.
* **Run:**

javac Question1\_GetLongestString.java

java Question1\_GetLongestString

**Question 2 – Unique Sum Combinations**

* **Approach:** Uses backtracking to count all unique combinations of numbers in the array that sum to the target result.
* **Run:**

javac Question2\_UniqueSumCombinations.java

java Question2\_UniqueSumCombinations

**Question 3 – Array Reduction**

* **Approach:** Checks if each element (from the second onward) is divisible by its previous element. If so, the array can be reduced to zeros except the first element.
* **Run:**

javac Question3\_ArrayReduction.java

java Question3\_ArrayReduction

**Question 4 – First Unique Product**

* **Approach:** Uses a LinkedHashMap to maintain insertion order while counting occurrences. Returns the first product appearing only once.
* **Run:**

javac Question4\_FirstUniqueProduct.java

java Question4\_FirstUniqueProduct

**Question 5 – Closest Minimum Distance**

* **Approach:** Finds all indices of the minimum value and computes the smallest gap between consecutive occurrences.
* **Run:**

javac Question5\_ClosestMinimumDistance.java

java Question5\_ClosestMinimumDistance

**Question 6 – Top Three Common Words**

* **Approach:** Splits the sentence into words, counts frequency using a HashMap, sorts by frequency and then alphabetically, returns top 3 in ascending order.
* **Run:**

javac Question6\_TopThreeCommonWords.java

java Question6\_TopThreeCommonWords

**Question 7 – Rotate Linked List**

* **Approach:** Calculates list length, makes it circular, adjusts rotation steps, and breaks the circle at the new tail.
* **Run:**

javac Question7\_RotateList.java

java Question7\_RotateList

**Test Files**

**Test\_Question1.java**

import java.util.\*;

public class Test\_Question1 {

public static void main(String[] args) {

List<Character> validChars = Arrays.asList('A','B','C','D');

// Test 1

String[] arr1 = {"AABCDA","ABCDZADC","ABCDBCA","ABCDABDCA"};

System.out.println("Expected: ABCDABDCA | Got: " +

Question1\_GetLongestString.getLongestString(validChars, arr1));

// Test 2 - Edge case

String[] arr2 = {"ZZZ","YYY"};

System.out.println("Expected: '' | Got: " +

Question1\_GetLongestString.getLongestString(validChars, arr2));

}

}

**Test\_Question2.java**

import java.util.\*;

public class Test\_Question2 {

public static void main(String[] args) {

int[] arr1 = {1,2,3,4,5};

System.out.println("Expected: 3 | Got: " +

Question2\_UniqueSums.countCombinations(arr1, 10));

int[] arr2 = {1,2,4,7,5};

System.out.println("Expected: 1 | Got: " +

Question2\_UniqueSums.countCombinations(arr2, 17));

}

}

**Test\_Question3.java**

public class Test\_Question3 {

public static void main(String[] args) {

int[] a1 = {1,2,3};

int[] a2 = {2,4,8};

int[] a3 = {3,6,14};

System.out.println("Expected: 1 | Got: " + Question3\_ArrayReduction.canMakeZero(a1));

System.out.println("Expected: 1 | Got: " + Question3\_ArrayReduction.canMakeZero(a2));

System.out.println("Expected: 0 | Got: " + Question3\_ArrayReduction.canMakeZero(a3));

}

}

**Test\_Question4.java**

public class Test\_Question4 {

public static void main(String[] args) {

String[] products = {"Apple","Computer","Apple","Bag"};

System.out.println("Expected: Computer | Got: " +

Question4\_FirstUniqueProduct.firstUniqueProduct(products));

String[] products2 = {"Pen","Pen","Pen"};

System.out.println("Expected: null | Got: " +

Question4\_FirstUniqueProduct.firstUniqueProduct(products2));

}

}

**Test\_Question5.java**

public class Test\_Question5 {

public static void main(String[] args) {

int[] arr = {1,2,3,1,4,5,2};

System.out.println("Expected: 3 | Got: " +

Question5\_ClosestMinDistance.closestMinDistance(arr));

}

}

**Test\_Question6.java**

import java.util.\*;

public class Test\_Question6 {

public static void main(String[] args) {

String sentence = "hi there care to discuss algorithm basis or how to solve algorithm or";

List<String> result = Question6\_TopThreeWords.topThreeWords(sentence);

System.out.println("Expected: [algorithm, or, to] | Got: " + result);

}

}

**Test\_Question7.java**

public class Test\_Question7 {

public static void main(String[] args) {

Node head = new Node("ID\_A01");

head.next = new Node("ID\_A02");

head.next.next = new Node("ID\_A03");

head.next.next.next = new Node("ID\_A04");

head.next.next.next.next = new Node("ID\_A05");

head.next.next.next.next.next = new Node("ID\_A06");

Node rotated = Question7\_RotateList.rotateRight(head, 2);

Question7\_RotateList.printList(rotated);

// Expected Output: ID\_A05 -> ID\_A06 -> ID\_A01 -> ID\_A02 -> ID\_A03 -> ID\_A04 -> null

}

}