solution@21-12-23

592. Fraction Addition and Subtraction

```
import java.util.*;

public class FractionAdditionSubtraction {
  // Declare lists at the class level
  private static List<Character> signs;
  private static List<Integer> numerators;
  private static List<Integer> denominators;

  public static String fractionAddition(String expression) {
    signs = new ArrayList<>();
    numerators = new ArrayList<>();
    denominators = new ArrayList<>();

    // Parse the input expression
    parseExpression(expression);

    // Find the common denominator
    int commonDenominator = findCommonDenominator();

    // Perform addition or subtraction
    int resultNumerator = calculateNumerator(commonDenominator);

    // Simplify the result
    int gcd = gcd(Math.abs(resultNumerator), commonDenominator);
    resultNumerator /= gcd;
    commonDenominator /= gcd;

    // Format the result as a string
```

```
  if (commonDenominator == 1) {
    return Integer.toString(resultNumerator);
  } else {
    return resultNumerator + "/" + commonDenominator;
  }
}

private static void parseExpression(String expression) {
  int index = 0;
  int length = expression.length();

  while (index < length) {
    char sign = expression.charAt(index++);
    signs.add(sign);

    int numerator = 0;
    while (index < length && Character.isDigit(expression.charAt(index))) {
      numerator = numerator * 10 + (expression.charAt(index++) - '0');
    }
    numerators.add(numerator);

    index++; // skip '/'

    int denominator = 0;
    while (index < length && Character.isDigit(expression.charAt(index))) {
      denominator = denominator * 10 + (expression.charAt(index++) - '0');
    }
    denominators.add(denominator);
  }
}
```

```
private static int findCommonDenominator() {
  int result = 1;
  for (int denominator : denominators) {
    result *= denominator;
  }
  return result;
}

private static int calculateNumerator(int commonDenominator) {
  int result = 0;
  int n = numerators.size();

  for (int i = 0; i < n; i++) {
    int numerator = numerators.get(i);
    int denominator = denominators.get(i);
    int factor = commonDenominator / denominator;

    if (signs.get(i) == '+') {
      result += numerator * factor;
    } else {
      result -= numerator * factor;
    }
  }

  return result;
}

private static int gcd(int a, int b) {
  while (b != 0) {
    int temp = b;
    b = a % b;
```

```
    a = temp;
   }
   return a;
  }

  public static void main(String[] args) {
   // Read input expression from the user
   Scanner scanner = new Scanner(System.in);
   System.out.print("Enter the expression: ");
   String expression = scanner.nextLine();

   // Call the fractionAddition method with user input
   String result = fractionAddition(expression);
   System.out.println("Result: " + result);

   // Close the scanner
   scanner.close();
  }
}
```

Output:-

PS C:\Users\Ajeet\Desktop\java> -2+(4/5)+6

4.8

594. Longest Harmonious Subsequence

```
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class LongestHarmoniousSubsequence {
```

```
public static int findLHS(int[] nums) {
  // Count the occurrences of each number
  Map<Integer, Integer> countMap = new HashMap<>();
  for (int num : nums) {
    countMap.put(num, countMap.getOrDefault(num, 0) + 1);
  }

  int maxLen = 0;

  // Iterate through unique numbers in the array
  for (int num : countMap.keySet()) {
    // Check if there exists a number whose difference with the current number is 1
    if (countMap.containsKey(num + 1)) {
      int currentLen = countMap.get(num) + countMap.get(num + 1);
      maxLen = Math.max(maxLen, currentLen);
    }
  }

  return maxLen;
}

public static void main(String[] args) {
  // Read input array from the user
  Scanner scanner = new Scanner(System.in);
  System.out.print("Enter the length of the array: ");
  int n = scanner.nextInt();

  int[] nums = new int[n];
  System.out.println("Enter the elements of the array:");
  for (int i = 0; i < n; i++) {
    nums[i] = scanner.nextInt();
```

```
    }

    // Call the findLHS method with user input
    int result = findLHS(nums);
    System.out.println("Length of the longest harmonious subsequence: " + result);

    // Close the scanner
    scanner.close();
  }
}
```

Output

Enter the elements of the array:

1 2 2 2 3 3 3 3 5 6 7 77777

Length of the longest harmonious subsequence: 7