Daily Practice Program Date 21/12/2023

Program 1 : Fraction addition and substraction

```
package ishwarchavan.com;

public class FractionAddSub {   //class created

    public String fractionAddition(String expression) { //function created
        int[] fraction = new int[]{0, 1, 0}; // {numerator, denominator, next unvisited
index}
        if (expression.charAt(0) != '-') expression = "+" + expression;
        while (fraction[2] < expression.length()) {   //condition checking
            int[] nextFraction = getNextFraction(expression, fraction);
            sum(fraction, nextFraction);
        }
        String res = fraction[0] + "/" + fraction[1];
        return res;
    }

    public int[] getNextFraction(String expression, int[] lastFraction) {   //function
created
        int nextIdx = lastFraction[2];
        int sign = expression.charAt(nextIdx++) == '+' ? 1 : -1;

        int numerator = 0;
        while (expression.charAt(nextIdx) != '/') {       //condition checking
            numerator = numerator * 10 + (expression.charAt(nextIdx++) - '0');
        }
        numerator *= sign;
        nextIdx++;          // skip '/'

        int denominator = 0;
        while (nextIdx < expression.length() && expression.charAt(nextIdx) != '-' &&
expression.charAt(nextIdx) != '+') {
            denominator = denominator * 10 + (expression.charAt(nextIdx++) - '0');
        }

        lastFraction[2] = nextIdx;
        return new int[]{numerator, denominator};
    }

    public void sum(int[] lastFraction, int[] curFraction) {  //function created
        lastFraction[0] *= curFraction[1];
        curFraction[0] *= lastFraction[1];
        lastFraction[1] = lastFraction[1] * curFraction[1];
        curFraction[1] = lastFraction[1] * curFraction[1];
        lastFraction[0] += curFraction[0];

        int gcd = gcd(Math.abs(lastFraction[0]), Math.abs(lastFraction[1]));
        lastFraction[0] /= gcd;
        lastFraction[1] /= gcd;
    }

    public int gcd(int a, int b) {     //gcd function created
        if (a == 0) return b;  //if true then return b
        while (b != 0) {   //condition checking
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;   //returning a
    }
}
```

Program 2 : Longest harmonious subsequence

```java
package ishwarchavan.com;

import java.util.Arrays;

public class LongestSubsequence {  //class created
      public static void main(String[] args) {
            int[] nums = {1,2,2,2,2,3,3,5};
            System.out.println(findLHS( nums));
      }

    public static int findLHS(int[] nums) {   //function created
        Arrays.sort(nums);
        int prev = Integer.MIN_VALUE;   //type casting
        int prevN = 0;
        int curr = nums[0];
        int currN = 0;
        int max = 0;
        for (int i = 0; i < nums.length; i++) {   //loop iterating
            if (nums[i] != curr) {   //condition checking
                if (prev+1 == curr)
                    max = Math.max(prevN + currN, max);  //storing the result
                prev = curr;
                prevN = currN;
                curr = nums[i];
                currN = 1;
            } else {
                currN++;   //incrementing
            }
        }
        if (prev+1 == curr)   //condition checking
            max = Math.max(prevN + currN, max);
        return max;
    }
}
```