1. import java.util.HashMap;

public class Performance {

private int[] marks;

public Performance() {

marks = new int[10];

}

public void readMarks(int[] studentMarks) {

if (studentMarks.length == 10) {

marks = studentMarks.clone();

} else {

System.out.println("Invalid input. Please provide marks for all 10 students.");

}

}

public int highestMark() {

int max = marks[0];

for (int i = 1; i < marks.length; i++) {

if (marks[i] > max) {

max = marks[i];

}

}

return max;

}

public int leastMark() {

int min = marks[0];

for (int i = 1; i < marks.length; i++) {

if (marks[i] < min) {

min = marks[i];

}

}

return min;

}

public int getMode() {

HashMap<Integer, Integer> frequencyMap = new HashMap<>();

for (int mark : marks) {

frequencyMap.put(mark, frequencyMap.getOrDefault(mark, 0) + 1);

}

int mode = marks[0];

int maxFrequency = frequencyMap.get(marks[0]);

for (int mark : frequencyMap.keySet()) {

if (frequencyMap.get(mark) >= maxFrequency && mark >= mode) {

mode = mark;

maxFrequency = frequencyMap.get(mark);

}

}

return mode;

}

public int getFreqAtMode() {

HashMap<Integer, Integer> frequencyMap = new HashMap<>();

for (int mark : marks) {

frequencyMap.put(mark, frequencyMap.getOrDefault(mark, 0) + 1);

}

int mode = marks[0];

int maxFrequency = frequencyMap.get(marks[0]);

for (int mark : frequencyMap.keySet()) {

if (frequencyMap.get(mark) >= maxFrequency && mark >= mode) {

mode = mark;

maxFrequency = frequencyMap.get(mark);

}

}

return maxFrequency;

}

public void display() {

System.out.println("Mode: " + getMode());

System.out.println("Mode Frequency: " + getFreqAtMode());

System.out.println("Highest Mark: " + highestMark());

System.out.println("Least Mark: " + leastMark());

}

public static void main(String[] args) {

Performance performance = new Performance();

int[] studentMarks = {75, 90, 85, 75, 92, 90, 85, 75, 92, 88};

students

performance.readMarks(studentMarks);

performance.display();

}

}

Output:

Mode: 75

Mode Frequency: 3

Highest Mark: 92

Least Mark: 75

2. public class AlphabetWarGame {

private static final String LEFT\_SIDE = "wpbs";

private static final String RIGHT\_SIDE = "mqdz";

private String leftSideStrengths;

private String rightSideStrengths;

public AlphabetWarGame() {

this.leftSideStrengths = LEFT\_SIDE;

this.rightSideStrengths = RIGHT\_SIDE;

}

public AlphabetWarGame(String leftStrengths, String rightStrengths) {

this.leftSideStrengths = leftStrengths;

this.rightSideStrengths = rightStrengths;

}

private int calculateStrength(String word, String strengths) {

int totalStrength = 0;

for (char letter : word.toCharArray()) {

if (strengths.indexOf(letter) != -1) {

totalStrength += strengths.indexOf(letter) + 1;

}

}

return totalStrength;

}

public String AlphabetWar(String word) {

int leftStrength = calculateStrength(word, leftSideStrengths);

int rightStrength = calculateStrength(word, rightSideStrengths);

if (leftStrength > rightStrength) {

return "Left side wins!";

} else if (rightStrength > leftStrength) {

return "Right side wins!";

} else {

return "Let's fight again!";

}

}

public String AlphabetWar(String leftWord, String rightWord)

{

int leftStrength = calculateStrength(leftWord, leftSideStrengths);

int rightStrength = calculateStrength(rightWord, rightSideStrengths);

if (leftStrength > rightStrength) {

return "Left side wins!";

} else if (rightStrength > leftStrength) {

return "Right side wins!";

} else {

return "Let's fight again!";

}

}

public static void main(String[] args) {

AlphabetWarGame game = new AlphabetWarGame();

System.out.println(game.AlphabetWar("z"));

System.out.println(game.AlphabetWar("zdqmwpbs"));

System.out.println(game.AlphabetWar("wwwwwwz"));

}

}

Output:

Right side wins!Let's fight again!

Left side wins!