1 – Topcoder SRM 146DIV2 250 - YahtzeeScore Problem Statement This task is about the scoring in the first phase of the die-game Yahtzee, where five dice are used. The score is determined by the values on the upward die faces after a roll. The player gets to choose a value, and all dice that show the chosen value are considered active. The score is simply the sum of values on active dice. Say, for instance, that a player ends up with the die faces showing 2, 2, 3, 5 and 4. Choosing the value two makes the dice showing 2 active and yields a score of 2 + 2 = 4, while choosing 5 makes the one die showing 5 active, yielding a score of 5. Your method will take as input a int[] toss, where each element represents the upward face of a die, and return the maximum possible score with these values. Definition Class: YahtzeeScore Method: maxPoints Parameters: int[] Returns: int Method signature: int maxPoints(int[] toss) (be sure your method is public)

Constraints - toss will contain exactly 5 elements. - Each element of toss will be between 1 and 6, inclusive. Examples 0) { 2, 2, 3, 5, 4 }

Returns: 5 The example from the text.

1) { 6, 4, 1, 1, 3 }

Returns: 6 Selecting 1 as active yields 1 + 1 = 2, selecting 3 yields 3, selecting 4 yields 4 and selecting 6 yields 6, which is the maximum number of points.

2) { 5, 3, 5, 3, 3 }

Returns: 10

2 – Topcoder SRM 308DIV2 250 – Median of Numbers Problem Statement In a set of distinct numbers, the median is an element M such that the number of elements greater than M is equal to the number of elements smaller than M. For example, in a set {1, 4, 2, 5, 7} the median is 4 because two elements (5 and 7) are greater than 4 and 2 elements (1 and 2) smaller than 4. The set {1, 5, 8, 3} has no median because no element from it satisfies the definition above. You are given a int[] numbers. Return the median of numbers or -1 if numbers has no median. Definition Class: MedianOfNumbers Method: findMedian Parameters: int[] Returns: int Method signature: int findMedian(int[] numbers) (be sure your method is public)

Limits Time limit (s): 2.000 Memory limit (MB): 64

Constraints - numbers will contain between 1 and 50 elements, inclusive. - Each element of numbers will be between 1 and 100, inclusive. - All elements of numbers will be distinct. Examples 0) {1, 4, 2, 5, 7}

Returns: 4 The example from the statement.

1) {1, 5, 8, 3}

Returns: -1

2) {7}

Returns: 7 There are zero elements that are greater than 7 and zero elements that are smaller than 7.

3) {7, 12}

Returns: -1

4) {66, 53, 47, 86, 18, 21, 97, 92, 15}

Returns: 53

5) {32, 54, 27, 4, 69, 96, 73, 1, 100, 15, 21}

Returns: 32

3 – Topcoder SRM 184DIV2 250 RaceApproximator (Bonus) Problem Statement If a runner races a distance D in time T, and later races a distance 2D, that runner will likely take more than 2T time to finish it. An examination of how times change with distances for a given runner can lead to the following approximation for the time it will take that runner to finish a given distance. Given two races with distances D1 and D2 which a runner ran in times T1 and T2, respectively, the approximate time it will take a runner to run a distance D is given by: T1\*e^(ln(T2/T1)\*ln(D1/D)/ln(D1/D2)).

When you race it is nice to have a time in mind that you'd like to be able to finish your race in. You are somewhat new to running and have only run two races of different distances. You are running a third race soon, and you want to use this equation to give you an estimate of how fast you should run. Since your upcoming race is a distance that falls between your first and second races' distances, you know this approximation will probably be fairly accurate. Create a class RaceApproximator with a method timeToBeat that takes ints d1, t1, d2, t2, and raceDistance, and returns a String that is the time you should be able to run in your upcoming race. d1, t1, d2 and t2 represent your shorter race's distance, your time in that race, your longer race's distance, and your time in that race, respectively. raceDistance is the distance of your upcoming race. All distances are in meters and all times are in seconds. Your return value should be truncated to an integer value, and formatted as "h:mm:ss" (all quotes are for clarity only) with"h" being the number of hours, "mm" being the number of minutes, and "ss" being the number of seconds. Definition Class: RaceApproximator Method: timeToBeat Parameters: int, int, int, int, int Returns: String Method signature: String timeToBeat(int d1, int t1, int d2, int t2, int raceDistance) (be sure your method is public)

Limits Time limit (s): 2.000 Memory limit (MB): 64

Notes - In C++ e^x can be done with exp(x), and the natural log, ln(x), can be done with log(x), both functions are in math.h. - In C# e^x can be done with Math.Exp(x), and the natural log, ln(x), can be done with Math.Log(x). The Math class is in the System namespace. - In Java e^x can be done with Math.exp(x), and the natural log, ln(x), can be done with Math.log(x). - In Visual Basic e^x can be done with Exp(x), and the natural log, ln(x), can be done with Log(x), both functions are in the System.Math namespace. Constraints - d1, t1, d2, t2, and raceDistance will all be between 1 and 10000, inclusive. - d1 will be less than d2 - t1 will be less than t2 - raceDistance will be greater than d1 and less than d2 - To make the approximation reliable, all speeds (distance/time) will be between 1 meter/second and 10 meters/second, inclusive. - To avoid rounding errors, the return will never be within 1e-9 of an integer value. Examples 0) 800 118 5000 906 1500

Returns: "0:03:57" Suzy Favor Hamilton's times for 800 meters and 5000 meters indicate that she should run 1500 meters in 3:57, which, in fact, is her time for 1500 meters.

1) 400 65 1600 350 800

Returns: "0:02:30" You can run 400 meters in 65 seconds, and 1600 meters in 5 minutes and 50 seconds, so you can probably run 800 meters in about 2 minutes and 30 seconds.

2) 1600 299 10000 2360 5000

Returns: "0:18:00"

3) 100 17 10000 4500 9000

Returns: "1:06:00"

4) 156 117 3863 2754 1755

Returns: "0:21:06"

Assignment #1 SRM 227-div2-250 Problem Statement Your company is writing a spell-checker system, and you have been tasked with writing a function to determine how closely two words resemble each other. The algorithm you are to use, albeit not a very good one, is to compare the two words character by character, and count how many times the characters in a given position are the same. For instance, the words "TICK" and "TOCK" have a score of 3, since three characters (T, C, K) are the same. Similarly, "CAT" and "DOG" score 0, since no letters match. You are given Strings a and b and are to return an int indicating the score (as defined above) of how closely the two match. Definition Class: StringCompare Method: simpleDifference Parameters: String, String Returns: int Method signature: int simpleDifference(String a, String b) (be sure your method is public)

Notes - The two strings may have different lengths. In that case, your comparison should only process characters until it reaches the end of either string.

Constraints - a and b will each contain between 1 and 50 characters, inclusive. - Each character of a and b will be 'A'-'Z'. Examples 0) "TICK" "TOCK"

Returns: 3 The first example from the problem statement.

1) "CAT" "DOG"

Returns: 0 The second example from the problem statement.

2) "APPLE" "APPLES"

Returns: 5 Notice the lengths are different, so the most we can compare is 5 characters, which are all identical.

3) "FANTASTIC" "ANTASTIC"

Returns: 0

Assignment #2 ( SRM 234-div2-250) Problem Statement You are given a String composed of A's and B's. Each A and B denotes a move performed by players A and B, respectively. Return the length of the longest combo performed by either player. A combo is a string of moves executed by a player that is not interrupted by the opposing player (see the examples for further clarification). Definition Class: ComboLength Method: howLong Parameters: String Returns: int Method signature: int howLong(String moves) (be sure your method is public)

Constraints - moves will contain between 2 and 50 characters inclusive. - Each character of moves will be A or B. Examples 0) "AAA"

Returns: 3 Player A executes 3 moves in a row.

1) "AAABBBBAA"

Returns: 4 Player A executes 2 combos of length 3 and 2 respectively. Player B executes a single combo of length 4.

2) "BBAABBAABBAABBAA"

Returns: 2

3) "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"

Returns: 50

4) "AAAAAAAAAAAAAAAAAAAAAAAAABAAAAAAAAAAAAAAAAAAAAAAAA"

Returns: 25

Bonus #1 (SRM 270-div 2 - 250 Problem Statement Steve would like to buy a new car. He isn't wealthy, so he would prefer a reasonably cheap car. The only problem is that the quality of the cheapest cars is... let's say questionable. Thus Steve decided to make a list of car prices and to buy a car with the third lowest price. You will be given a int[] prices. The same price may occur multiple times in prices, but it should count only once in the ordering of available prices. See Example 1 for further clarification. Write a function that returns the third lowest price in this list. If there are less than three different car prices in prices, your method should return -1. Definition Class: BuyingCheap Method: thirdBestPrice Parameters: int[] Returns: Int Method signature: int thirdBestPrice(int[] prices) (be sure your method is public)

Constraints - prices contains between 1 and 50 elements, inclusive. - Each element in prices will be between 1 and 1000, inclusive. Examples 0) {10, 40, 50, 20, 70, 80, 30, 90, 60}

Returns: 30

1) {10, 10, 10, 10, 20, 20, 30, 30, 40, 40}

Returns: 30 The lowest price is 10, the second lowest is 20 and the third lowest is 30.

2) {10}

Returns: -1

3) {80, 90, 80, 90, 80}

Returns: -1

 [Eslam Taha](https://www.facebook.com/islam.taha)

https://fbstatic-a.akamaihd.net/rsrc.php/v2/y4/r/-PAXP-deijE.gif2014/02/14 11:34 مساءً

[**Eslam Taha**](https://www.facebook.com/islam.taha)

Bonus #2 (SRM 281-div 2 – 250) Problem Statement Run-length encoding is a simple compression technique which compresses strings of letters by replacing repeated consecutive letters (called runs) by the number of occurrences of the letter, followed by that letter. For example, AAAABBBCDDE compresses to 4A3BC2DE. The number 1 may be omitted in runs consisting of a single letter, as with letters 'C' and 'E' in the previous example. Any string consisting of uppercase letters where each letter is optionally preceded by a positive integer is called a properly encoded string. Given a properly encoded stringtext, return the decoded string. If the decoded string would be more than 50 characters long, return "TOO LONG" (without the quotes). Definition Class: RunLengthEncoding Method: decode Parameters: String Returns: String Method signature: String decode(String text) (be sure your method is public)

Constraints - text will contain between 0 and 50 characters ('0'-'9', 'A'-'Z'), inclusive. - text will be a properly encoded string: all numbers contained will be positive integers with no leading zeros and each number will precede a letter. Examples 0) "4A3BC2DE"

Returns: "AAAABBBCDDE" This is the example in the problem statement.

1) "1A1B1C1D1E"

Returns: "ABCDE" 1's can be omitted, but also may appear in the input. This input is valid, although we'd doubled the size of the string by "compressing" it.

2) "1A3A5A4BCCCC"

Returns: "AAAAAAAAABBBBCCCC" Although it isn't the best possible, this is also a properly encoded string.

3) "50A"

Returns: "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"

4) "21Z13S9A8M"

Returns: "TOO LONG"

 اليوم

 [Mahmoud Mohammed](https://www.facebook.com/engmahmoudaboahmed)

https://fbstatic-a.akamaihd.net/rsrc.php/v2/y4/r/-PAXP-deijE.gif06:43 مساءً

[**Mahmoud Mohammed**](https://www.facebook.com/engmahmoudaboahmed)

جزاك الله خيرا