

## Junior C# .NET Developer

### Client Name

ValueMomentum

### Candidate Name

Komal Dhanraj Shimpi

### Date of Attempt

11-Mar-2025

### Candidate ID

11165442



# Index

## Score Analysis

Your scores, a quick overview of your performance and your overall percentage.

## Section Score Analysis

A quick overview of sectional performance along with percentages.

## Section Skill Analysis

An overview of your proficiency in specific skills.

## Individual Development Plan – IDP

Focus on your strengths and the areas of improvement, along with developmental tips to work on.

## Difficulty Level Analysis

A comprehensive insight into the candidate's performance at 3 difficulty levels.

## Proctoring Analysis

A quick overview of the proctoring-related aspects of the assessment.

## Test Log

A quick overview of the test status, timestamp, and recorded IP address.

## Question Details

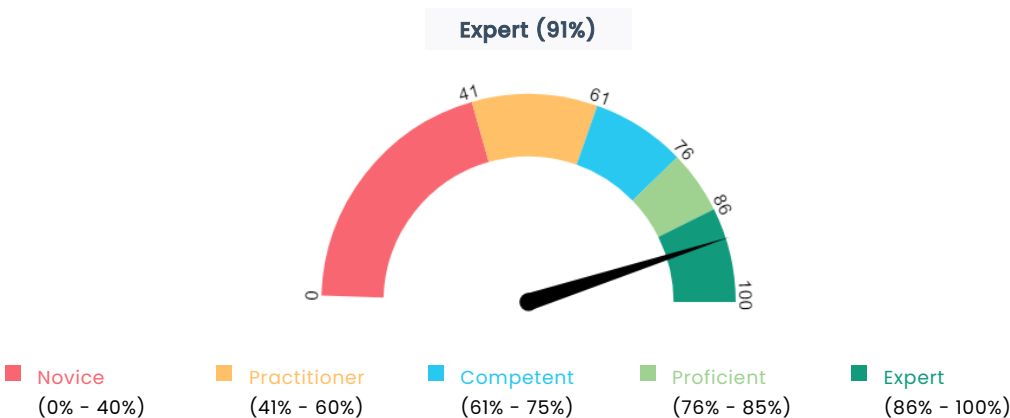
An overview of each question and the candidate's response, offering a thorough assessment of their performance.

## Disclaimer

Disclaimer on subjective customised assessments.

### Score Analysis

Score: 32/ 35    Time Taken: 30 min 50 sec / 47 min



Komal Dhanraj Shimpi scored **91%** and completed assessment in **66%** of the allotted time

### Section Score Analysis

#### Section Percentage



### Section Skill Analysis

#### Section 1: C# .Net and Frameworks

Total Score: **5/ 6**    Negative Points: **0**    Time Taken: **6 min 11 sec/8 min**

##### Question Analysis:

Total Question: **6**    Correct: **5**    Wrong: **1**    Skipped: **0**    Not Answered: **0**

Skills	#Questions	Skill Score
C# Basics	3	2/3
Dot Net OOPs	3	3/3

#### Section 2: Database

Total Score: **2/ 4**    Negative Points: **0**    Time Taken: **3 min 45 sec/4 min**

##### Question Analysis:

Total Question: **4**    Correct: **2**    Wrong: **2**    Skipped: **0**    Not Answered: **0**

Skills	#Questions	Skill Score
SQL Server	2	2/2
SQL	2	0/2

#### Section 3: C# Coding

Total Score: **25/ 25**    Negative Points: **0**    Time Taken: **20 min 54 sec/35 min**

##### Question Analysis:

Total Question: **2**    Correct: **2**    Wrong: **0**    Skipped: **0**    Not Answered: **0**

Skills	#Questions	Skill Score
Coding - Easy	1	10/10
Coding - Medium	1	15/15

## Identification of strengths and skill improvement needs

### ● Strength

Congratulations! We have identified **Object-Oriented Design and Programming, Programming** as your strengths.

### ● Improvement area

Based on your score, **SQL** are the identified areas of improvement.

## A guide to get started on your Individual Development Plan (IDP) :

### Difficulty Level Analysis

Level	Number of Questions	Correct Attempts	Correctness
Easy	6	5	83.33%
Medium	4	3	75%
Hard	2	1	50%

### Proctoring Analysis

Images Captured: 63

Image Violations: 0

🟢 Image violations detected, within tolerable limit.

Multiple Faces Detected 0  
No Face Detected 0  
Unrecognized Face Detected 0

Note: The total violations are based on the custom violation settings for this test. The number of consecutive images considered as one violation is configured for all the categories (Unrecognized Face, Multiple Faces, No Face) and may differ from the default settings.

Window Violation: 0

Time Violation: 0 min

### Test Log

Test Status	Date & Time	Captured IP address
Appeared On	11 Mar 2025, 10:34 AM	103.239.86.58
Completed On	11 Mar 2025, 11:07 AM	103.239.86.58
Report Generated On	11 Mar 2025, 11:07 AM	103.239.86.58

### Question Details

Question: #1	Type: Coding	Skill: Coding - Easy	Status: Answered
Result: Correct	Level: Easy	Time Taken: 7 min 46 sec	Average Time: 15 min 18 sec
Score: 10 / 10	Window Violation: 0 times	Time Violation: 0 sec	

#### Question #1

## Cost of the String

A coded string is defined as a string in which each character ("a" to "z") is replaced by a binary digit according to the following rule.

- If the character is a vowel, replace the character with the binary digit "1."
- If the character is a consonant, replace the character with the binary digit "0."

A string **S** is given, consisting of the lowercase English alphabet.

Convert the string **S** to a coded string and then calculate the cost of the coded string **S**.

The cost of the coded string is the decimal value of the coded string.

**Print the cost of the binary string.**

### Function Description

In the provided code snippet, implement the provided `binaryString(...)` method to print the cost of the binary string. You can write your code in the space below the phrase "WRITE YOUR LOGIC HERE".

There will be multiple test cases running so the Input and Output should match exactly as provided.

The base Output variable `result` is set to a default value of `-404` which can be modified. Additionally, you can add or remove these output variables.

### Input Format

The input line contains a string, **S**.

### Sample Input

```
abcd -- denotes S
```

### Constraints

$1 \leq S.size() \leq 60$

String **S** consists of the lowercase English alphabet only.

### Output Format

The output contains a single integer denoting the cost of the binary string.

### Sample Output

```
8
```

### Explanation

**S** = abcd

The coded binary string = 1000

The decimal value of the coded binary string is 8.

Hence, the output is **8**.

**Answer:**

**Coding Language:** C#

**Candidate Code:**

```
using System;using System.Collections.Generic;public class Test{    public static int binaryString(string S)    {        //this is default OUTPUT. You can change it.        int decimalValue = 0;        //write your Logic here:        string vowels = "aeiou";        s        {            string line = "All the Best!";            List<string> temp = new List<string> { };            while ( ! string.IsNullOrEmpty (line))            {                line = Console.ReadLine();                if (line!=null)                {                    string[] elements = line.Split(' ');                    foreach (string element in elements)                    {                        temp.Add(element);                    }                }            }            // OUTPUT [uncomment & modify if required]            Console.WriteLine(binaryString(s));        }    }
```

**Compilation Summary:**

Compilation Status: **Compile Successfully**

No Of Compilations: **3**

Default Input:  
**wlohfdatjrikg**

Candidate Output:

1092

#### Test Case Summary:

Test Case: **1**    Status: **Pass**    Score: **0**

Test Case Input	Expected Output	Actual Output
Test Case Input abcd	Expected Output 8	Actual Output 8

Test Case: **2**    Status: **Pass**    Score: **2**

Test Case Input	Expected Output	Actual Output
adfrev	34	34

Test Case: **3**    Status: **Pass**    Score: **2**

Test Case Input	Expected Output	Actual Output
wlohfdatjrikg	1092	1092

Test Case: **4**    Status: **Pass**    Score: **2**

Test Case Input	Expected Output	Actual Output
qnzscihprbaoexwyfg	4320	4320

Test Case: **5**    Status: **Pass**    Score: **2**

Test Case Input	Expected Output	Actual Output
opywacvkhdejnzsgqmurbt	2230280	2230280

Test Case: **6**    Status: **Pass**    Score: **2**

Test Case Input	Expected Output	Actual Output
qproqqjgflvcwsocub	33802	33802

Question: <b>#2</b>	Type: <b>Coding</b>	Skill: <b>Coding - Medium</b>	Status: <b>Answered</b>
Result: <b>Correct</b>	Level: <b>Medium</b>	Time Taken: <b>13 min 8 sec</b>	Average Time: <b>11 min 37 sec</b>
Score: <b>15 / 15</b>	Window Violation: <b>0 times</b>	Time Violation: <b>0 sec</b>	

#### Question #2

##### **Racing Cars**

A car showroom has **N** racing cars. Each car has a price and bonus associated with it. The price of the car is denoted by an array **P[i]** and the bonus associated is denoted by **B[i]**. You can sell **K** cars at the most. The maximum earning of the showroom is the sum of the prices of **K** cars multiplied by the minimum bonus amongst the selected **K** cars.

**Find the maximum earnings of the showroom by selling at most K cars.**

### Note

1-based indexing is used.

### Function Description

In the provided code snippet, implement the provided `racingCars(...)` method to find the maximum earnings of the showroom by selling at most **K** cars. You can write your code in the space below the phrase "WRITE YOUR LOGIC HERE".

There will be multiple test cases running, so the Input and Output should match exactly as provided. The base Output variable `result` is set to a default value of **-404**, which can be modified. Additionally, you can add or remove these output variables.

### Input Format

The first line contains two integers, **N** and **K**, denoting the number of racing cars and the number of cars you can sell, respectively.

The second line contains **N** space-separated integers of **array P[i]**, denoting the price of the **i<sup>th</sup>** car.

The third line contains **N** space-separated integers of **array B[i]**, denoting the bonus associated with the **i<sup>th</sup>** car.

### Sample Input

```
4 3          -- denotes N and K
200 400 350 100 -- denotes P[i]
10 10 5 9    -- denotes B[i]
```

### Constraints

$1 \leq K \leq N \leq 100000$

$1 \leq P[i] \leq 100000$

$1 \leq B[i] \leq 100000$

### Output Format

The output contains an integer denoting the maximum earnings of the showroom by selling at most **K** cars.

### Sample Output

```
6300
```

### Explanation

**N** = 4

**K** = 3

You can select at most 3 cars.

For maximum earnings, select 1<sup>st</sup>, 2<sup>nd</sup>, and 4<sup>th</sup> cars.

Prices of the 1<sup>st</sup>, 2<sup>nd</sup>, and 4<sup>th</sup> cars = 200 + 400 + 100 = 700.

The bonus associated with the 1<sup>st</sup>, 2<sup>nd</sup>, and 4<sup>th</sup> cars = 10 10 9 (9 is the minimum bonus among them).

Maximum earning = ( 200 + 400 + 100 ) \* 9 = 6300.

Hence, the output is **6300**.

### Answer:

#### Coding Language: C#

#### Candidate Code:

```
using System;using System.Collections.Generic;using System.Linq;public class Test{    public static int racingCars(int N,int K,int[] P,int[] B){        //this is default OUTPUT. You can change it.        //write your Logic here:        int maxEarnings = 0;        int priceSum=0;        int n = N;        for(int i=0; i< N; i++){            cars[i] = (P[i], B[i]);        }        cars = cars.OrderByDescending(car => car.bonus).ToArray();        string line = "All the Best!";        List<string> temp = new List<string> { };        while ( ! string.IsNullOrEmpty (line))        {            line = Console.ReadLine();            if (line!=null)            {                string[] elements = line.Split(' ');                foreach (string element in elements)                {                    temp.Add(element);                }            }        }        int K = Convert.ToInt32(temp[1]); int[] P = new int[N]; for(int i = 1+1; i < 1+1+N; i++) { P[i-
```

```
(1+i)] = Convert.ToInt32(temp[i]); } int[] B = new int[N]; for(int i = 1+1+N; i < 1+1+N+N; i++) { B[i-
(1+1+N)] = Convert.ToInt32(temp[i]); }
// OUTPUT [uncomment & modify if required] Console.WriteLine(racingCars(N,K,P,B)); }}
```

#### Compilation Summary:

Compilation Status: **Compile Successfully**  
 Default Input:  
**4 3<br>200 400 350 100<br>10 10 5 9**

No Of Compilations: **5**  
 Candidate Output:

6300

#### Test Case Summary:

Test Case: **1** Status: **Pass** Score:**0**

Test Case Input	Expected Output	Actual Output
4 3200 400 350 10010 10 5 9	6300	6300

Test Case: **2** Status: **Pass** Score:**3**

Test Case Input	Expected Output	Actual Output
2 2100 10020 40	4000	4000

Test Case: **3** Status: **Pass** Score:**3**

Test Case Input	Expected Output	Actual Output
1 110010	1000	1000

Test Case: **4** Status: **Pass** Score:**3**

Test Case Input	Expected Output	Actual Output
1 010010	0	0

Test Case: **5** Status: **Pass** Score:**3**

Test Case Input	Expected Output	Actual Output
5 23 2 4 3 15 2 4 3 2	28	28

Test Case: **6** Status: **Pass** Score:**3**

Test Case Input	Expected Output	Actual Output
7 546 90 409 29 94 100 5839 92 81 40 96 78 45	54054	54054

#### Disclaimer

*This report is generated electronically based on the information provided by the assessment participants. It also includes flags, especially when proctoring services are utilized. However, it is important to emphasize that this report should not be the sole basis for making any business, selection, entrance, or employment-related decisions. iMocha assumes no responsibility for any consequences arising from the use of this report or any actions taken or refrained from as a result of it. This includes all business decisions made in reliance on the information, advice, or AI flags contained in this report, as well as any sources of information mentioned or referred to in the report.*



