DAY 3

CODILITY

**1.BINARY GAP**

**CODING:**

using System;

// you can also use other imports, for example:

// using System.Collections.Generic;

// you can write to stdout for debugging purposes, e.g.

// Console.WriteLine("this is a debug message");

class Solution {

public int solution(int N) {

string binary = Convert.ToString(N, 2);

int left = 0;

int right = 0;

int maxGap = 0;

while (right < binary.Length) {

if (binary[right] == '1') {

maxGap = Math.Max(maxGap, right - left - 1);

left = right;

}

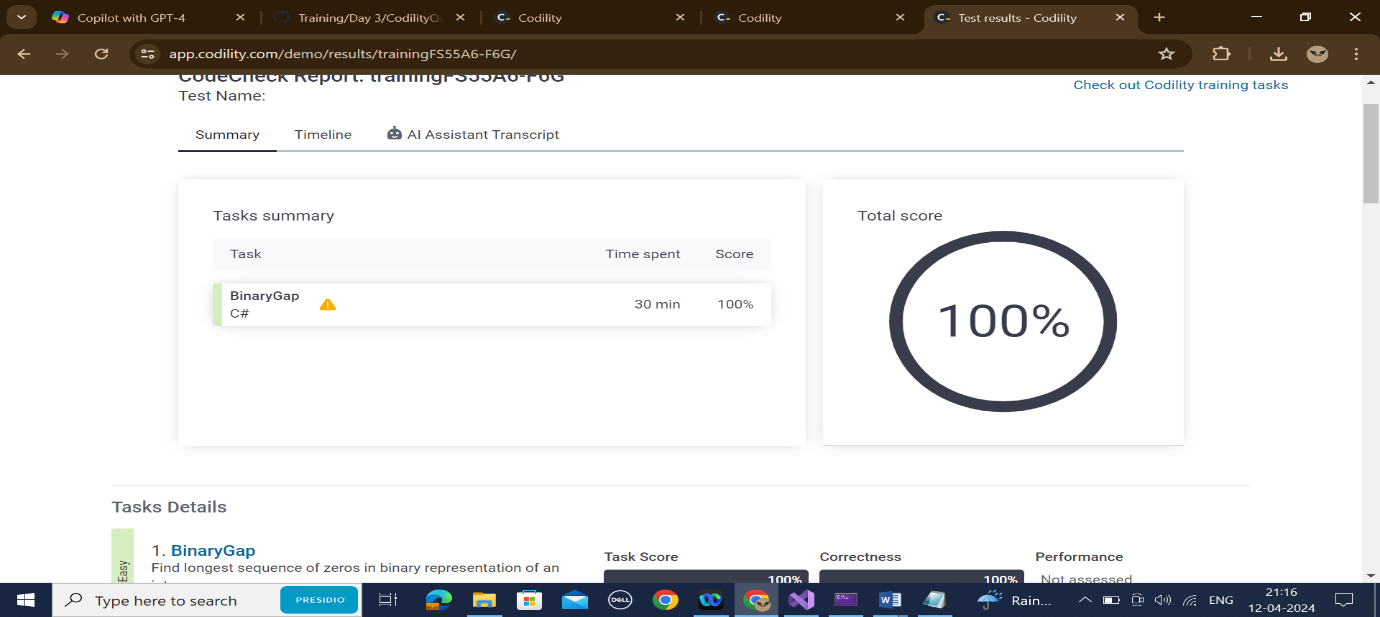
right++;

}

return maxGap;

}

}

****

**2.CYCLIC ROTATION**

**CODING:**

using System;

public class Solution {

public int[] solution(int[] A, int K) {

int N = A.Length;

int[] result = new int[N];

for (int i = 0; i < N; i++) {

result[(i + K) % N] = A[i];

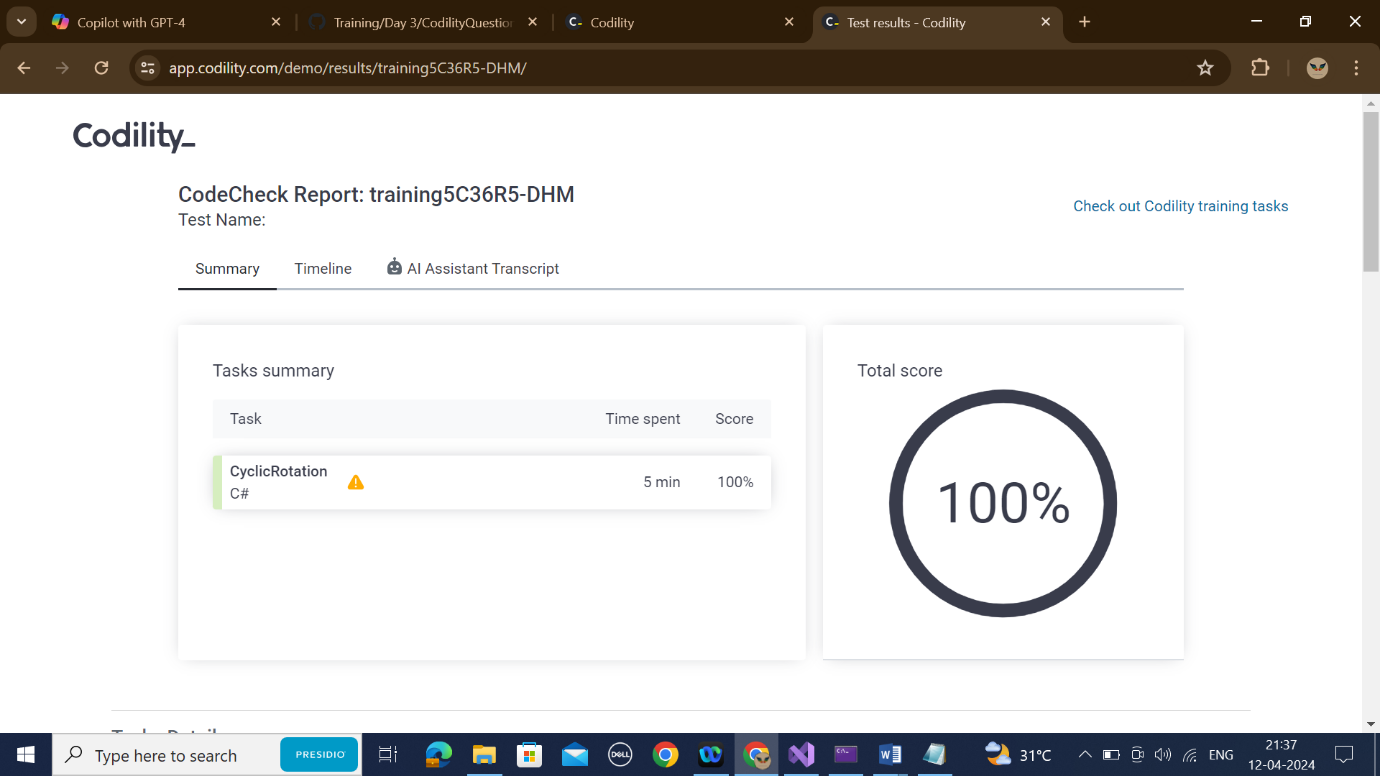
}

return result;

}

}

**OUTPUT:**



**3.FROG JUMP**

**CODING:**

public class Solution {

public int solution(int X, int Y, int D) {

int distance = Y - X;

if (distance % D == 0) {

return distance / D;

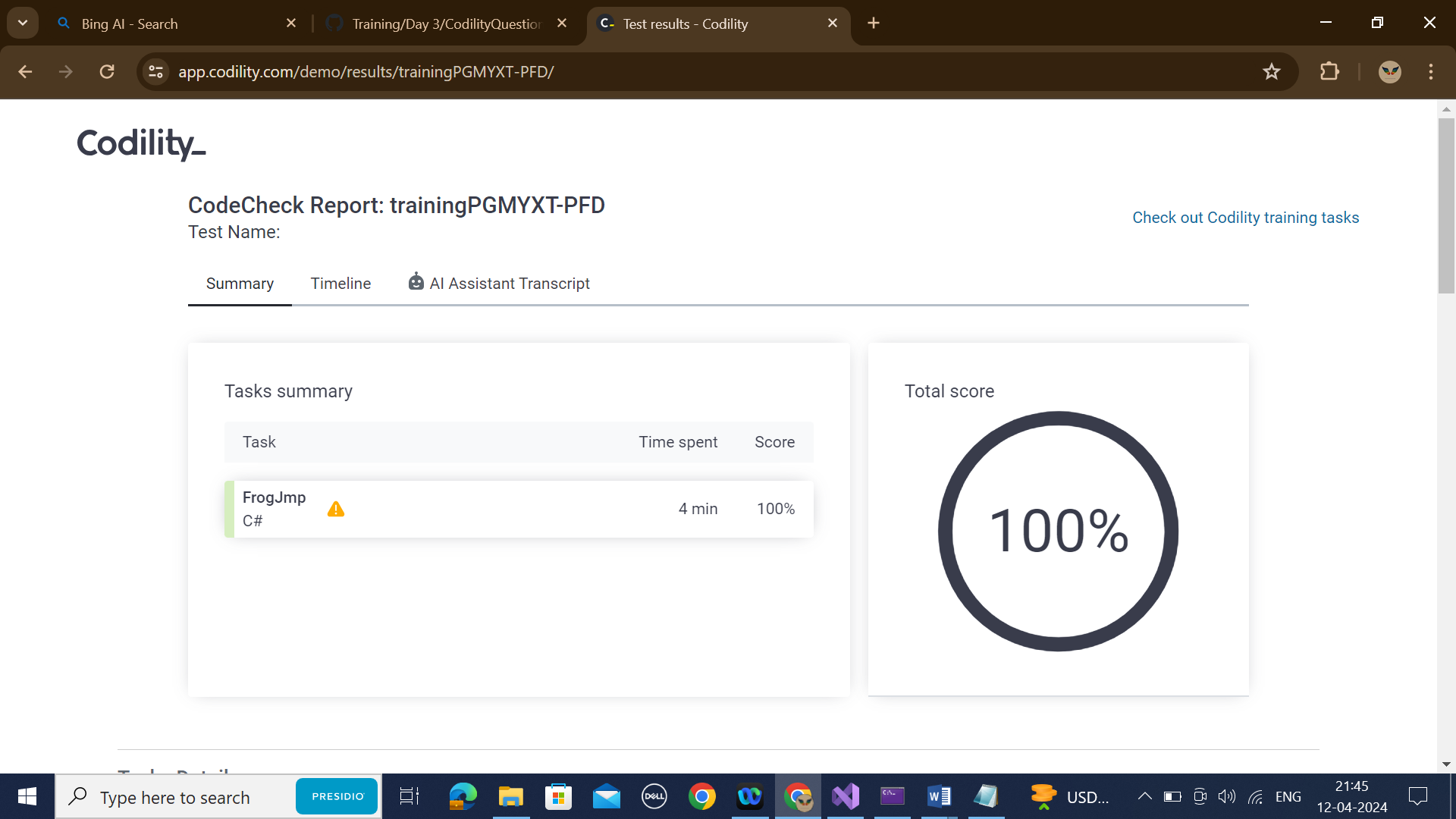
} else {

return distance / D + 1;

}

}

}



**4.ODD OCCURANCES IN AN ARRAY**

CODING:

public class Solution {

public int solution(int[] A) {

int res = 0;

for (int i = 0; i < A.Length; i++) {

res = res ^ A[i];

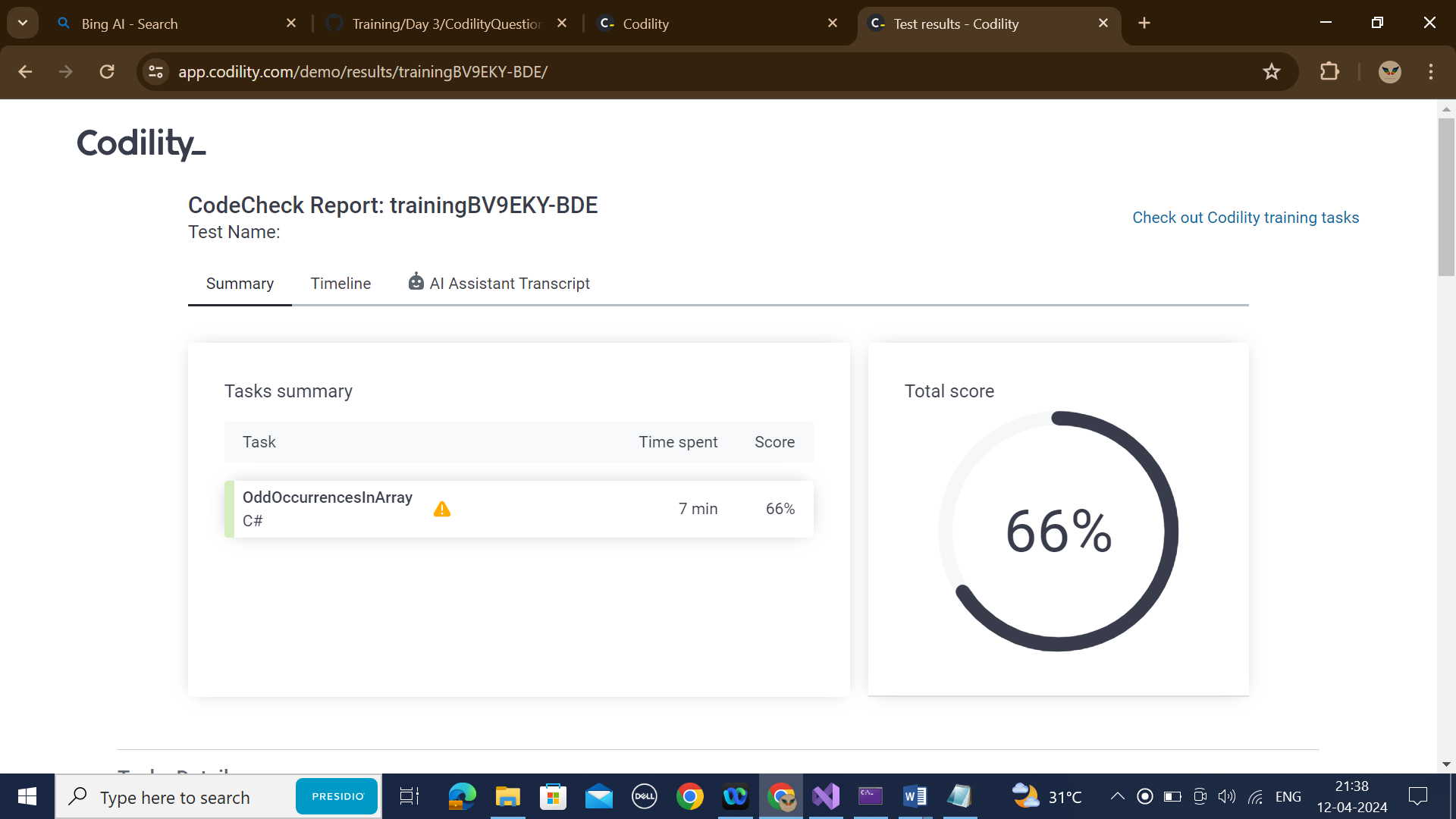
}

return res;

}

}

**OUTPUT:**



**5.PERMUTATION MISSING ELEMENT:**

CODING:

public class Solution {

public int solution(int[] A) {

long N = A.Length;

long totalSum = (N + 1) \* (N + 2) / 2;

long arraySum = 0;

for (int i = 0; i < N; i++) {

arraySum += A[i];

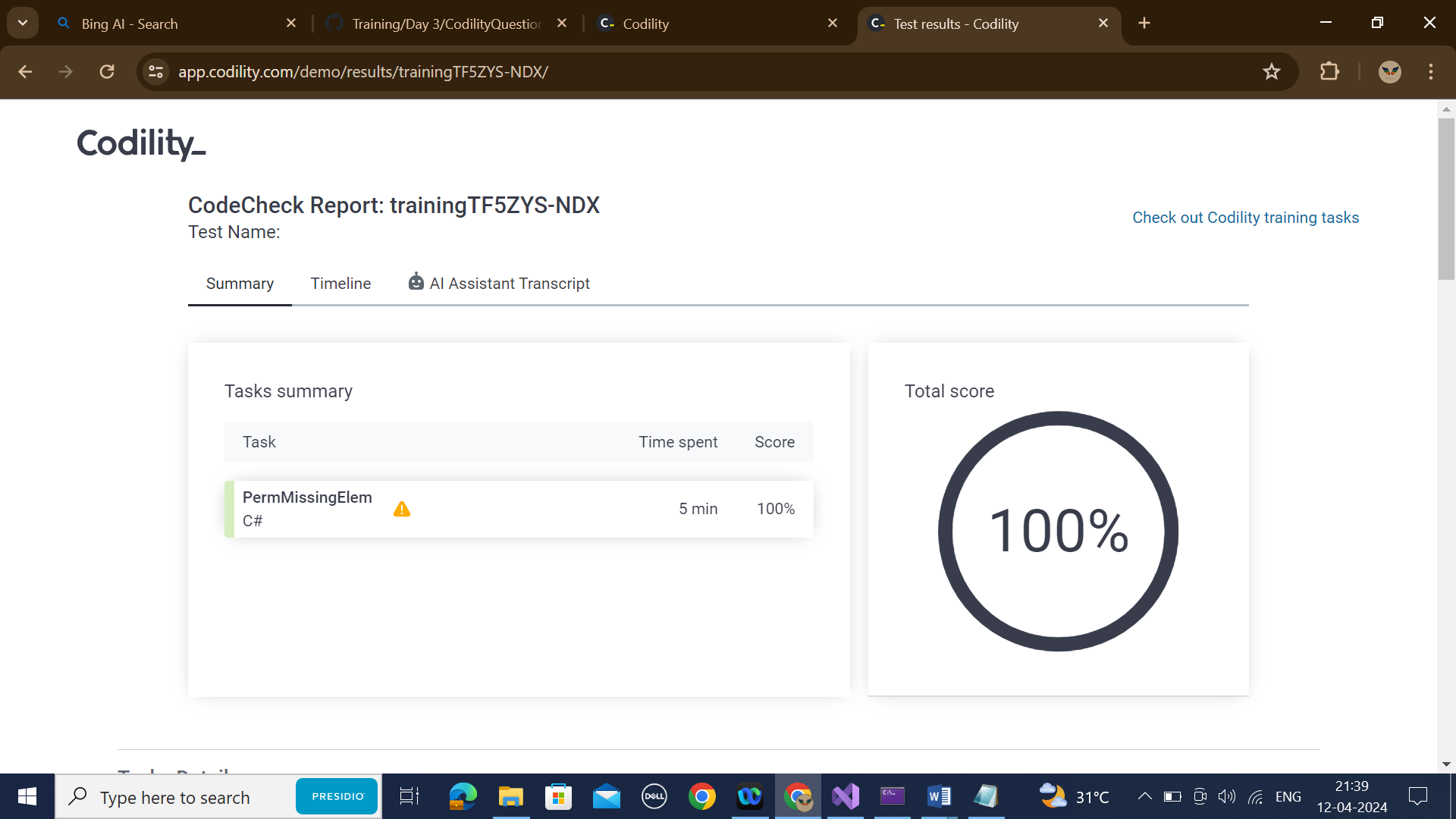
}

return (int)(totalSum - arraySum);

}

}

**OUTPUT:**



**6.TAPE EQUILIBRIUM:**

public class Solution {

public int solution(int[] A) {

long sumOfAll = 0;

for(int i=0; i<A.Length; i++) {

sumOfAll += A[i];

}

long sumOfFirstPart = 0;

long minDifference = long.MaxValue;

for(int p=0; p<A.Length-1; p++) {

sumOfFirstPart += A[p];

long sumOfSecondPart = sumOfAll - sumOfFirstPart;

long difference = Math.Abs(sumOfFirstPart - sumOfSecondPart);

if(difference < minDifference) {

minDifference = difference;

}

}

return (int)minDifference;}}

