**1.** public class Solution

{

public int[] TwoSum(int[] nums, int target)

{

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

{

for (int j = i + 1; j < nums.Length; j++)

{

if ((nums[i] + nums[j]) == target)

{

return new int[] { i, j };

}

}

}

return null;

}

}

A screenshot of a computer

Description automatically generated

**2.** public class Solution {

public bool IsPalindrome(int x) {

if(x < 0){

return false;

}

int remainder = 0;

int reversed = 0;

int original = x;

while (x != 0) {

remainder = x % 10;

reversed = reversed \* 10 + remainder;

x /= 10;

}

if(reversed == original){

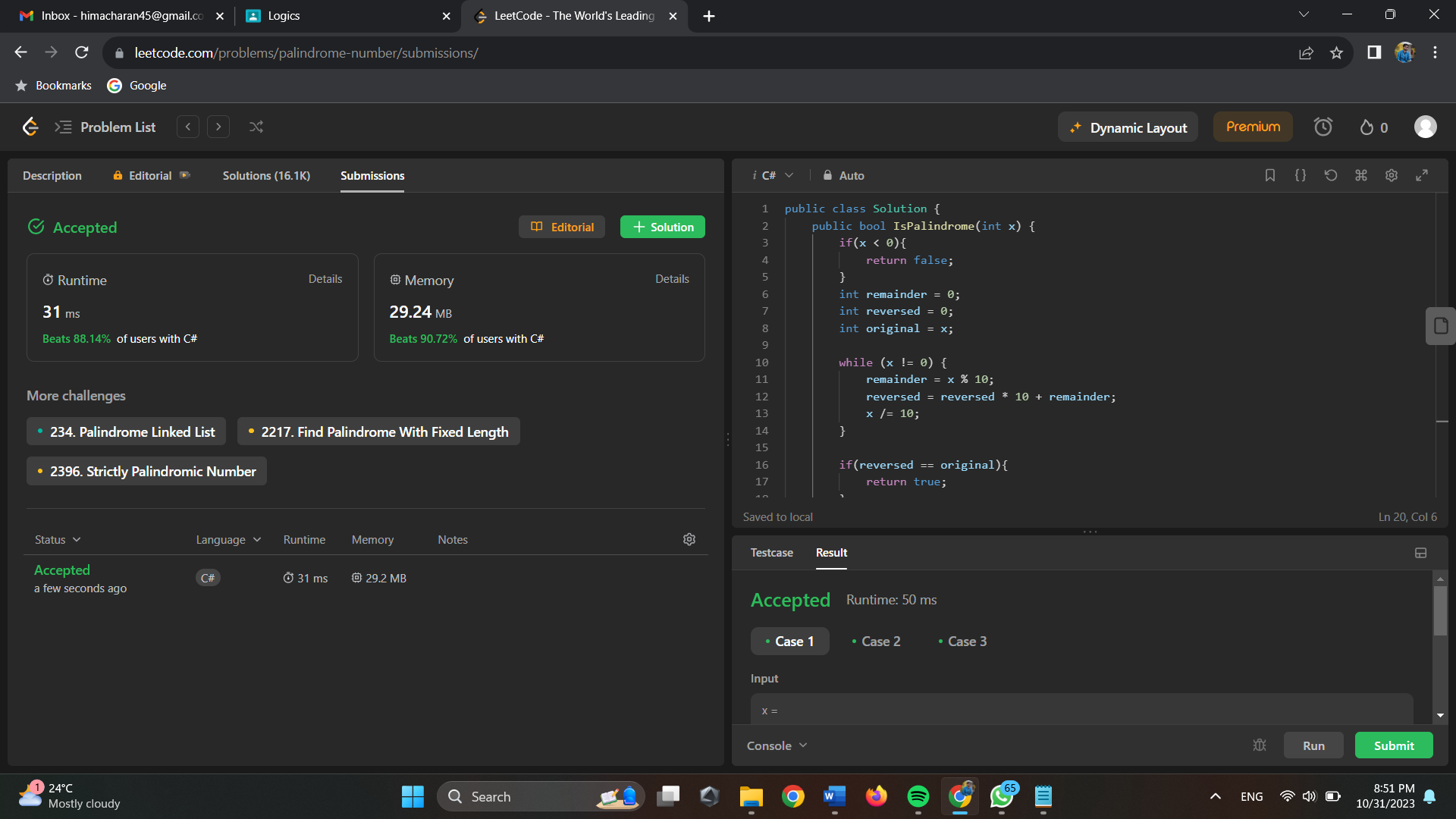
return true;

}

return false;

}

}



**3.** public class Solution {

public int RemoveDuplicates(int[] nums) {

int tmp = nums[0];

int cnt = 1;

for(int i = 1;i<nums.Length;i++)

{

if(nums[i]>tmp)

{

tmp = nums[i];

nums[cnt] = tmp;

cnt++;

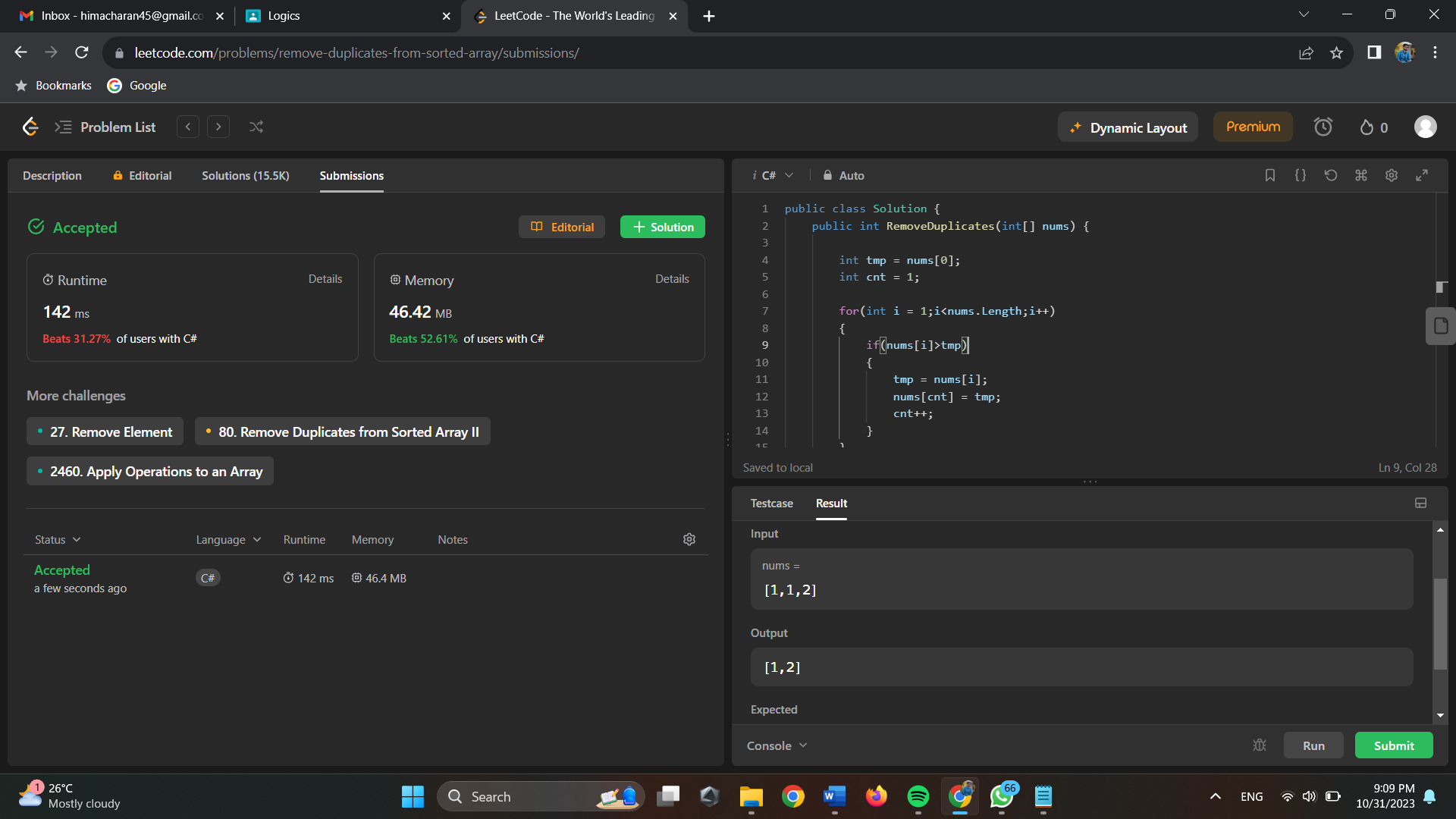
}

}

return cnt;

}

}



**4.**

public class Solution {

public string LongestCommonPrefix(string[] strs)

{

for (int i = 0; i < strs[0].Length; i++)

{

char tmpChar = strs[0][i];

for (int j = 0; j < strs.Length; j++)

{

if (strs[j].Length == i || strs[j][i] != tmpChar)

{

return strs[0].Substring(0, i);

}

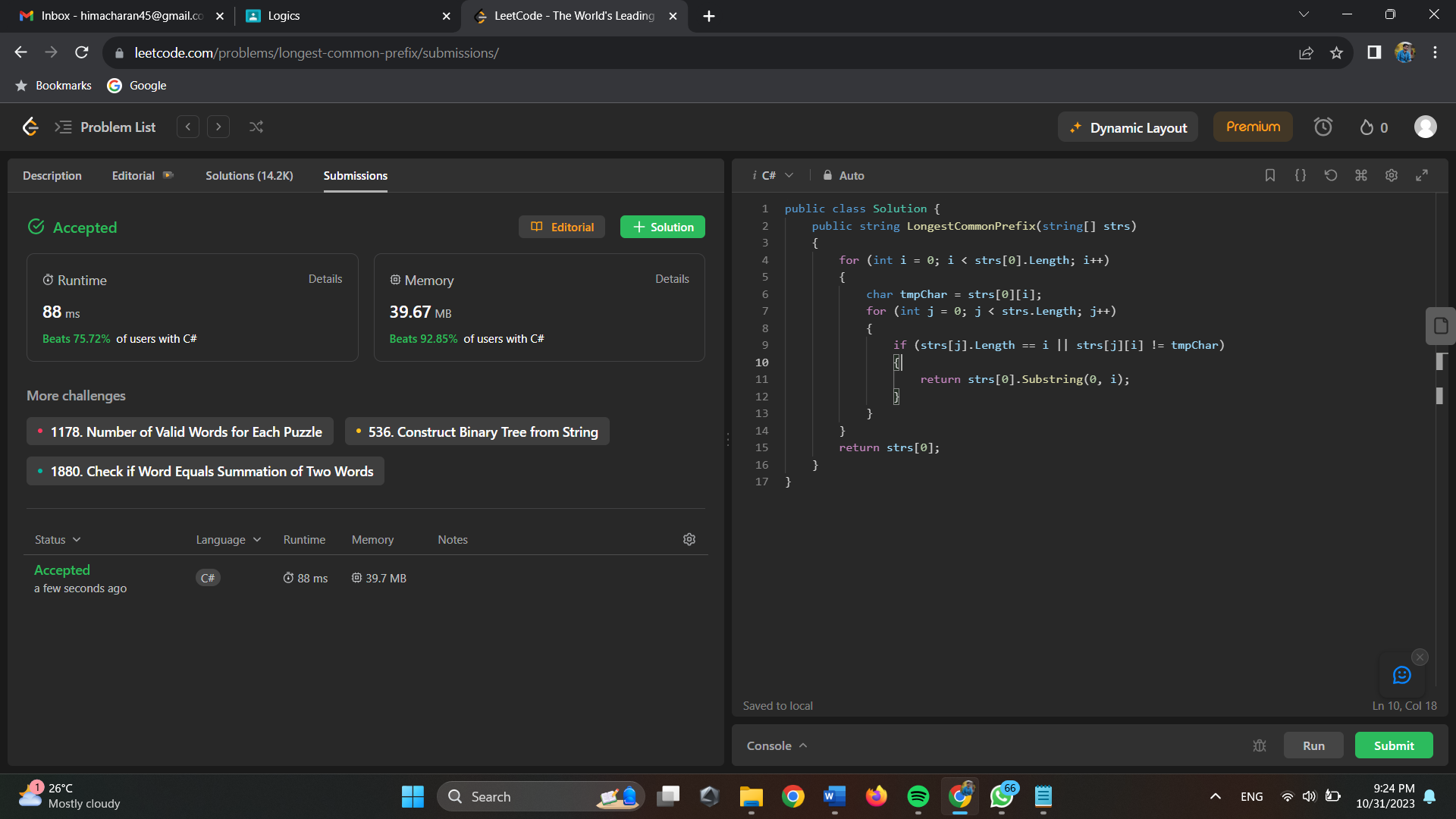
}

}

return strs[0];

}

}



**5.**

public class Solution {

public bool IsMatch(string s, string p) {

if (p.Length == 0)

{

return s.Length == 0;

}

if (p.Length == 1 || p[1] != '\*')

{

if (s.Length == 0 || (p[0] != '.' && p[0] != s[0]))

{

return false;

}

return IsMatch(s.Substring(1), p.Substring(1));

}

int i = -1;

while (i < s.Length && (i < 0 || p[0] == '.' || p[0] == s[i]))

{

if (IsMatch(s.Substring(i + 1), p.Substring(2)))

{

return true;

}

i++;

}

return false;

}

}

