Date: 04 - 01 - 2026
Assignment No.: 1

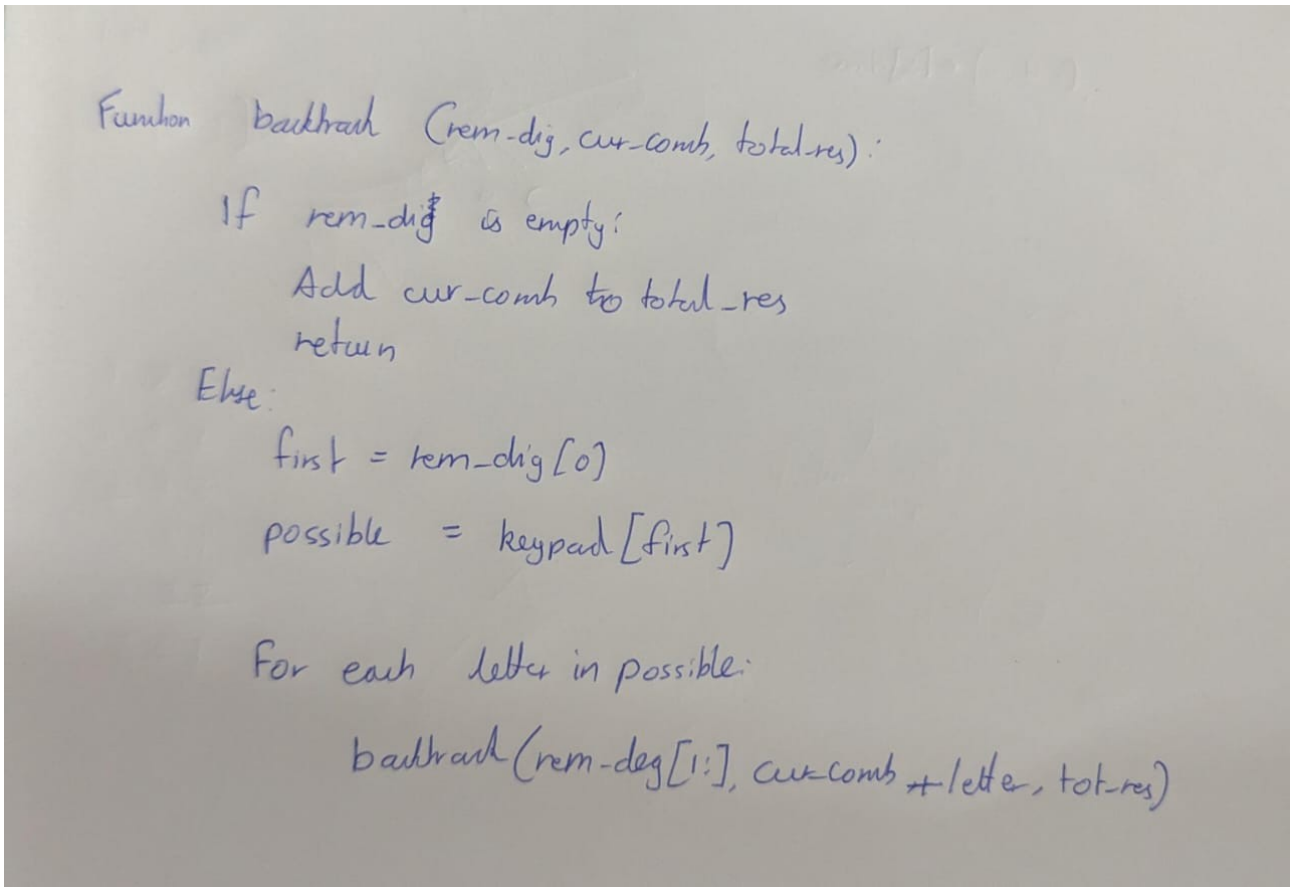
Name: Varghese K James
Reg No.: 3122245001192

Assignment 1: Iterative and Recursive Algorithms

Question 1

Aim: Letter Combinations of a Phone Number

Algorithm



Code

```python
class Solution:
    comb = [
        [], [],
        ['a', 'b', 'c'],
        ['d', 'e', 'f'],
        ['g', 'h', 'i'],
        ['j', 'k', 'l'],
        ['m', 'n', 'o'],
        ['p', 'q', 'r', 's'],
        ['t', 'u', 'v'],
        ['w', 'x', 'y', 'z']
```

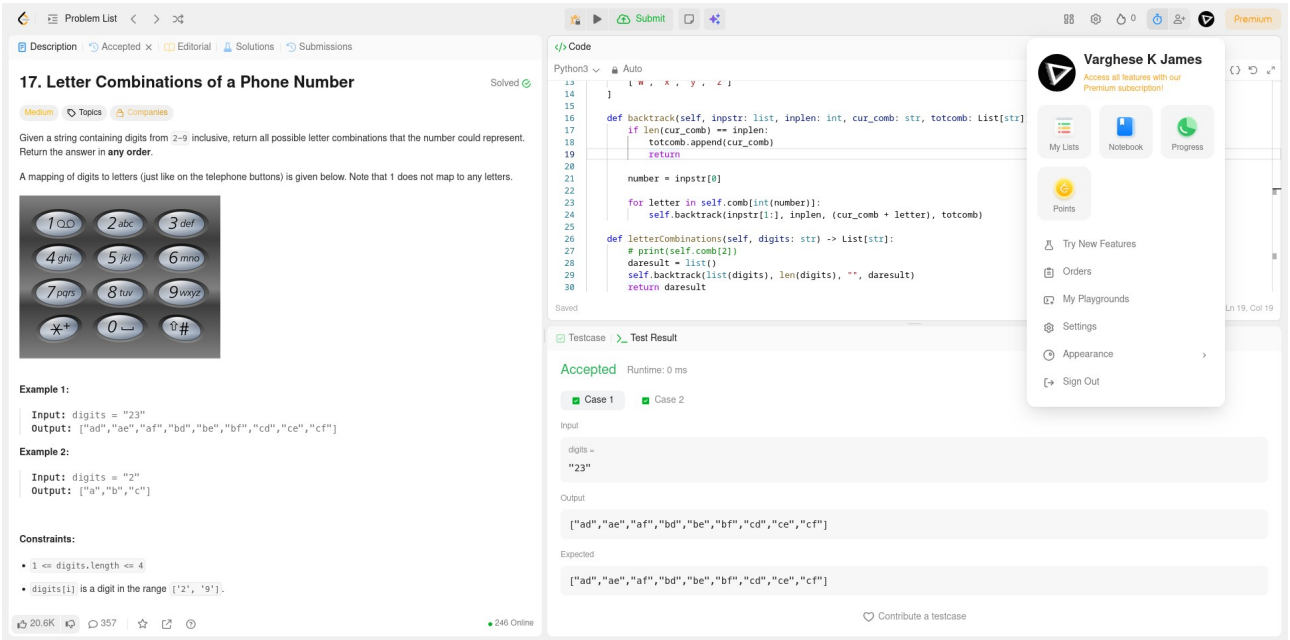
```
    ]

    def backtrack(self, inpstr: list, inplen: int, cur_comb: str, totcomb: List[str]):
        if len(cur_comb) == inplen:
            totcomb.append(cur_comb)
            return

        number = inpstr[0]

        for letter in self.comb[int(number)]:
            self.backtrack(inpstr[1:], inplen, (cur_comb + letter), totcomb)

    def letterCombinations(self, digits: str) -> List[str]:
        daresult = list()
        self.backtrack(list(digits), len(digits), "", daresult)
        return daresult
```

Time complexity

This is a recursive solution with a branching factor of worst case 4 each time, hence the time complexity is

O(4^N)

Screenshot of Output



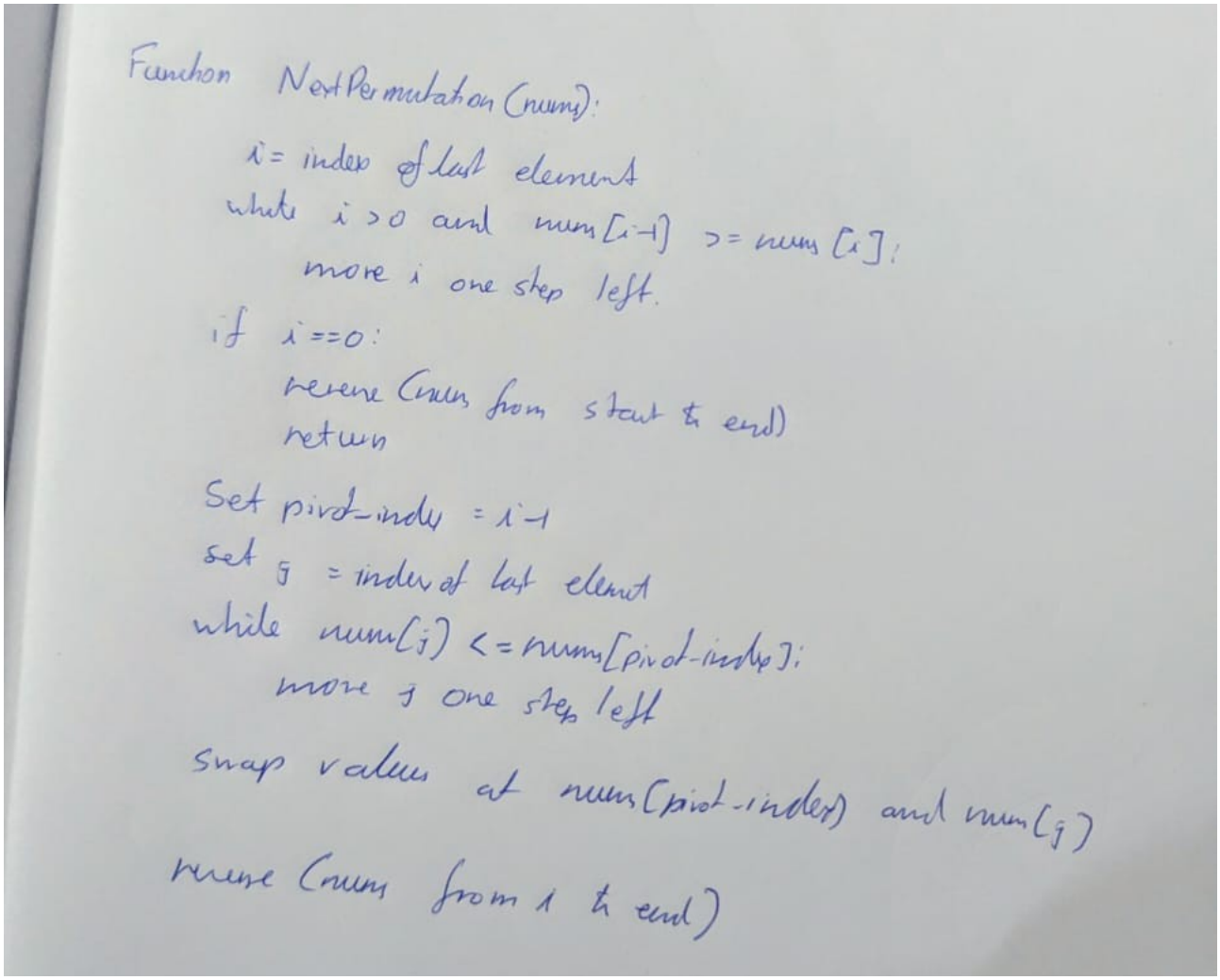
Date: 04 - 01 - 2026
Assignment No.: 1

Name: Varghese K James
Reg No.: 3122245001192

Question 2

Aim: Next Permutation

Algorithm



Code

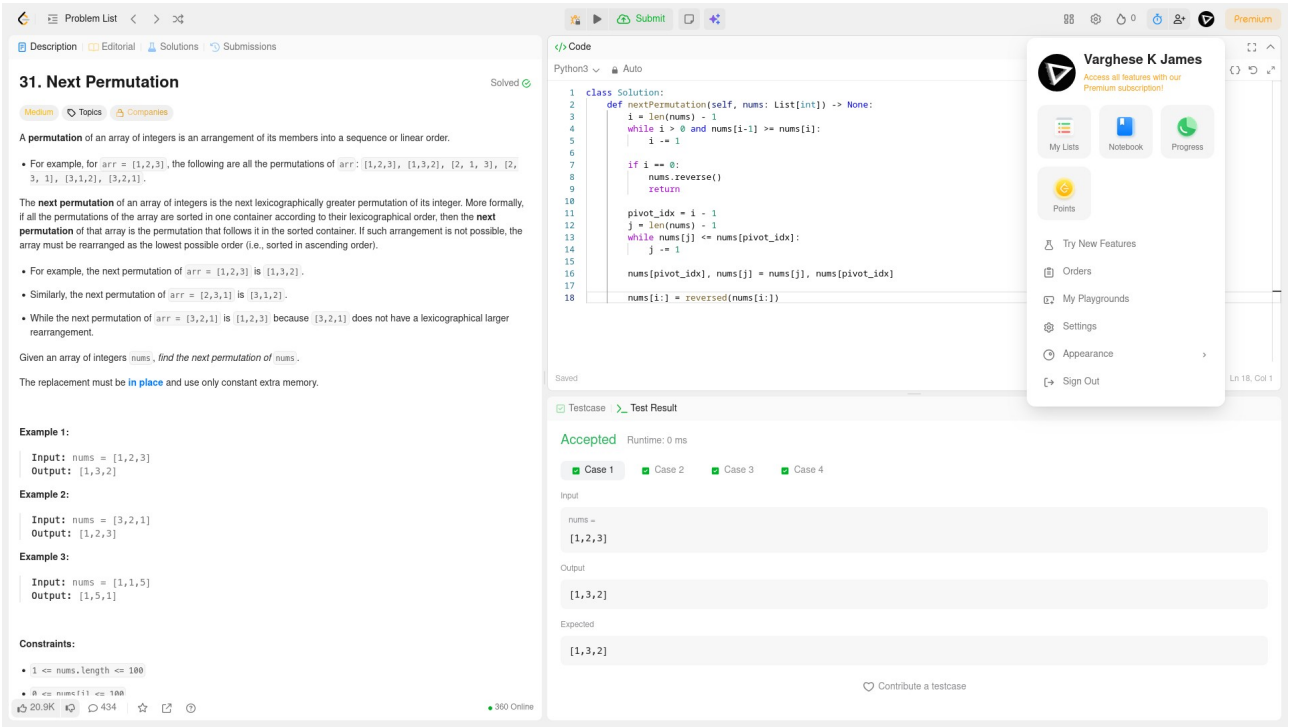
```
class Solution:
    def nextPermutation(self, nums: List[int]) -> None:
        i = len(nums) - 1
        while i > 0 and nums[i-1] >= nums[i]:
            i -= 1

        if i == 0:
            nums.reverse()
            return
```

```
        pivot_idx = i - 1
        j = len(nums) - 1
        while nums[j] <= nums[pivot_idx]:
            j -= 1

        nums[pivot_idx], nums[j] = nums[j], nums[pivot_idx]

        nums[i:] = reversed(nums[i:])
```

Time complexity

This is a linear algorithm as even in worst case it only passes through the same element twice, hence it is of O(N) time complexity.

Screenshot of Output



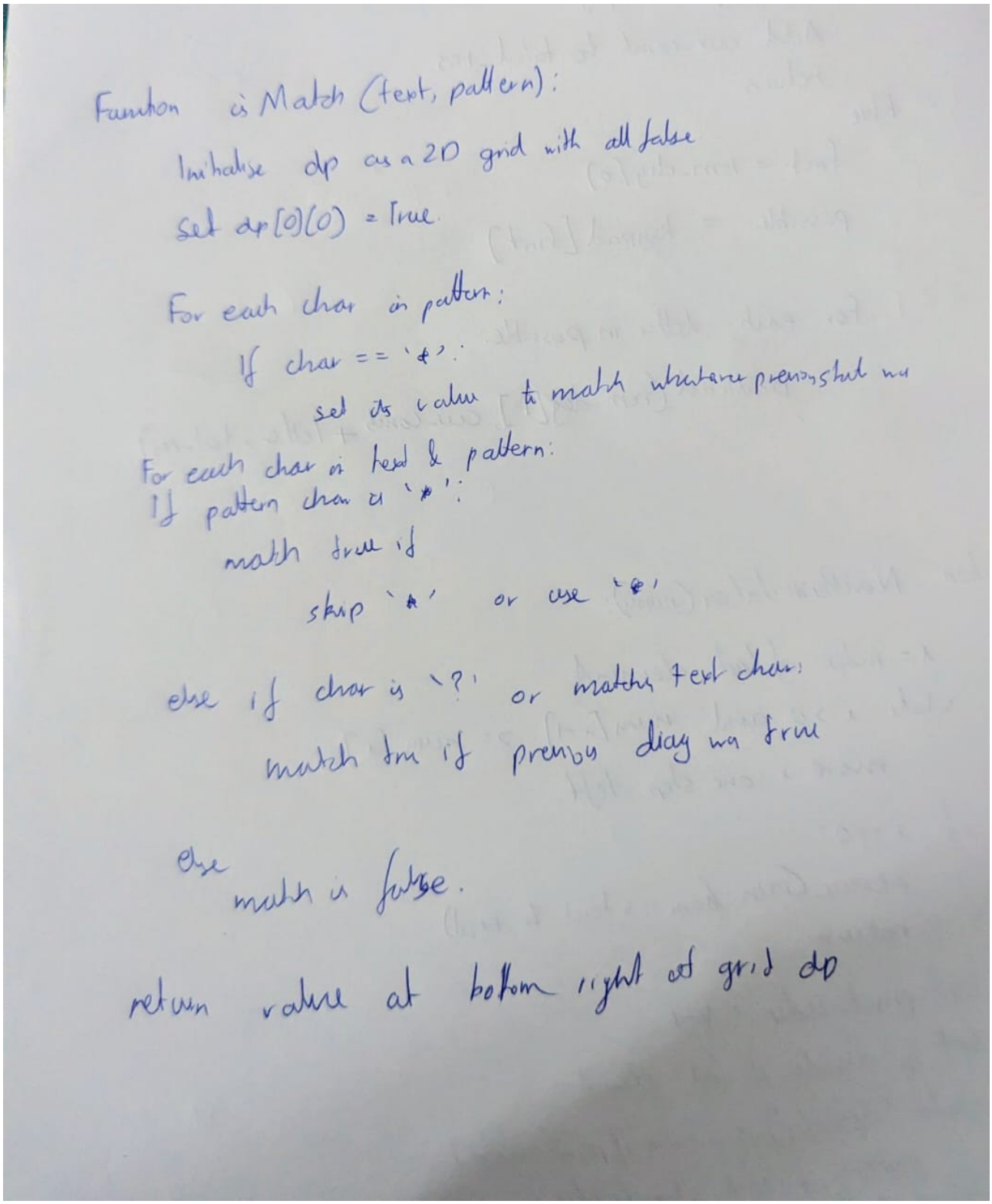
Question 3

Aim: Wildcard Matching

Date: 04 - 01 - 2026
Assignment No.: 1

Name: Varghese K James
Reg No.: 3122245001192

Algorithm



Date: 04 - 01 - 2026
Assignment No.: 1

Name: Varghese K James
Reg No.: 3122245001192

Code

```python
class Solution:
    def isMatch(self, text: str, pattern: str) -> bool:
        n = len(text)
        m = len(pattern)

        dp = [[False] * (m + 1) for _ in range(n + 1)]

        dp[0][0] = True

        for j in range(1, m + 1):
            if pattern[j-1] == '*':
                dp[0][j] = dp[0][j-1]

        for i in range(1, n + 1):
            for j in range(1, m + 1):

                char_text = text[i-1]
                char_pattern = pattern[j-1]

                if char_pattern == '*':
                    dp[i][j] = dp[i][j-1] or dp[i-1][j]

                elif char_pattern == '?' or char_pattern == char_text:
                    dp[i][j] = dp[i-1][j-1]

                else:
                    dp[i][j] = False

        return dp[n][m]
```

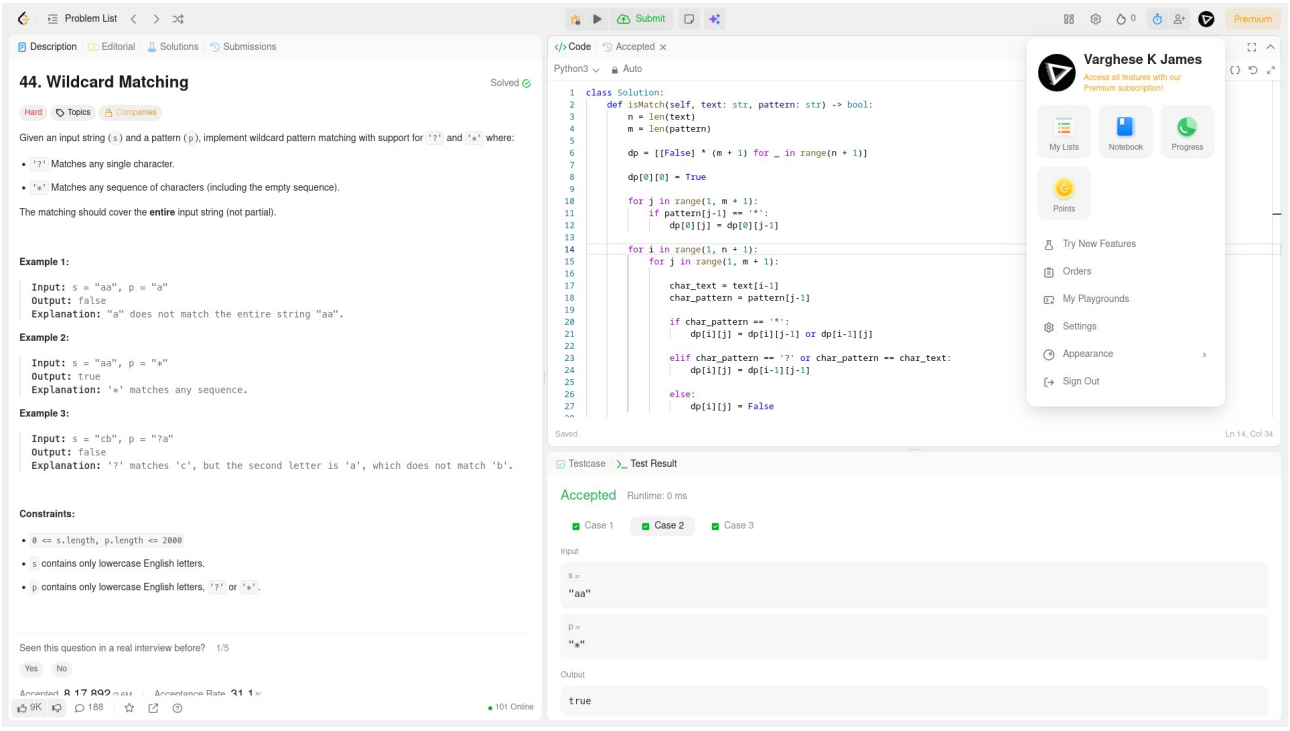
Time complexity

This is a nested loop with the outer loop running for each charecter in text and the inner loop running for each charecter in pattern. Hence time complexity is O(mxn) where m is length of text and n is length of pattern

Date: 04 - 01 - 2026
Assignment No.: 1

Name: Varghese K James
Reg No.: 3122245001192

Screenshot of Output



Learning Outcomes

- Learned about using backtracking algorithms to solve questions involving permutations and combination
- Learned about bespoke optimisations done to solve competitive programming questions.