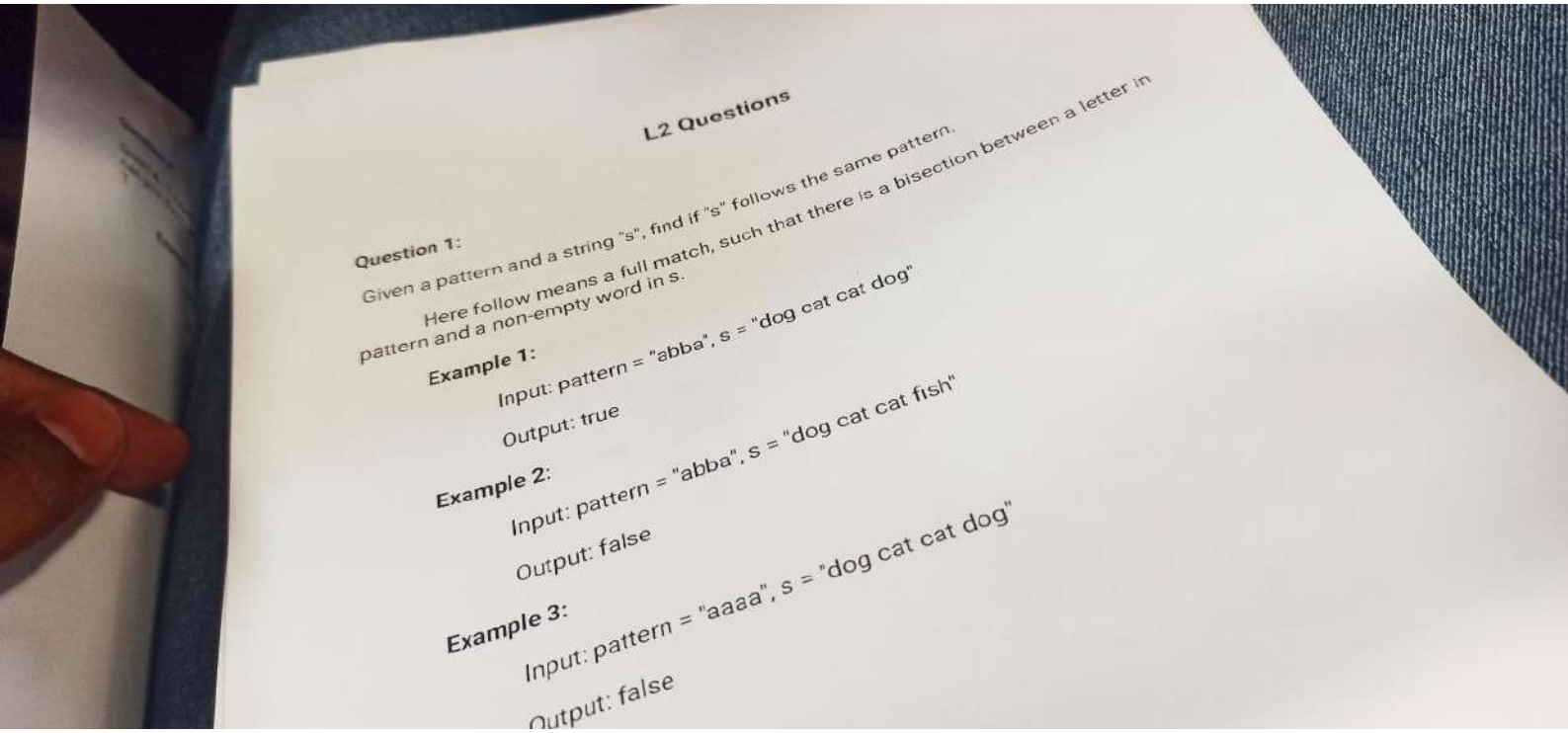


L2 Questions

Question 1:

Given a pattern and a string "s", find if "s" follows the same pattern.

Here follow means a full match, such that there is a bisection between a letter in pattern and a non-empty word in s.

Example 1:

Input: pattern = "abba", s = "dog cat cat dog"

Output: true

Example 2:

Input: pattern = "abba", s = "dog cat cat fish"

Output: false

Example 3:

Input: pattern = "aaaa", s = "dog cat cat dog"

Output: false

Question 2:

Given a string "s", reverse only all the vowels in the string and return it.

The vowels are 'a', 'e', 'i', 'o', and 'u', and they can appear in both lower and upper cases, more than once.

Example 1:

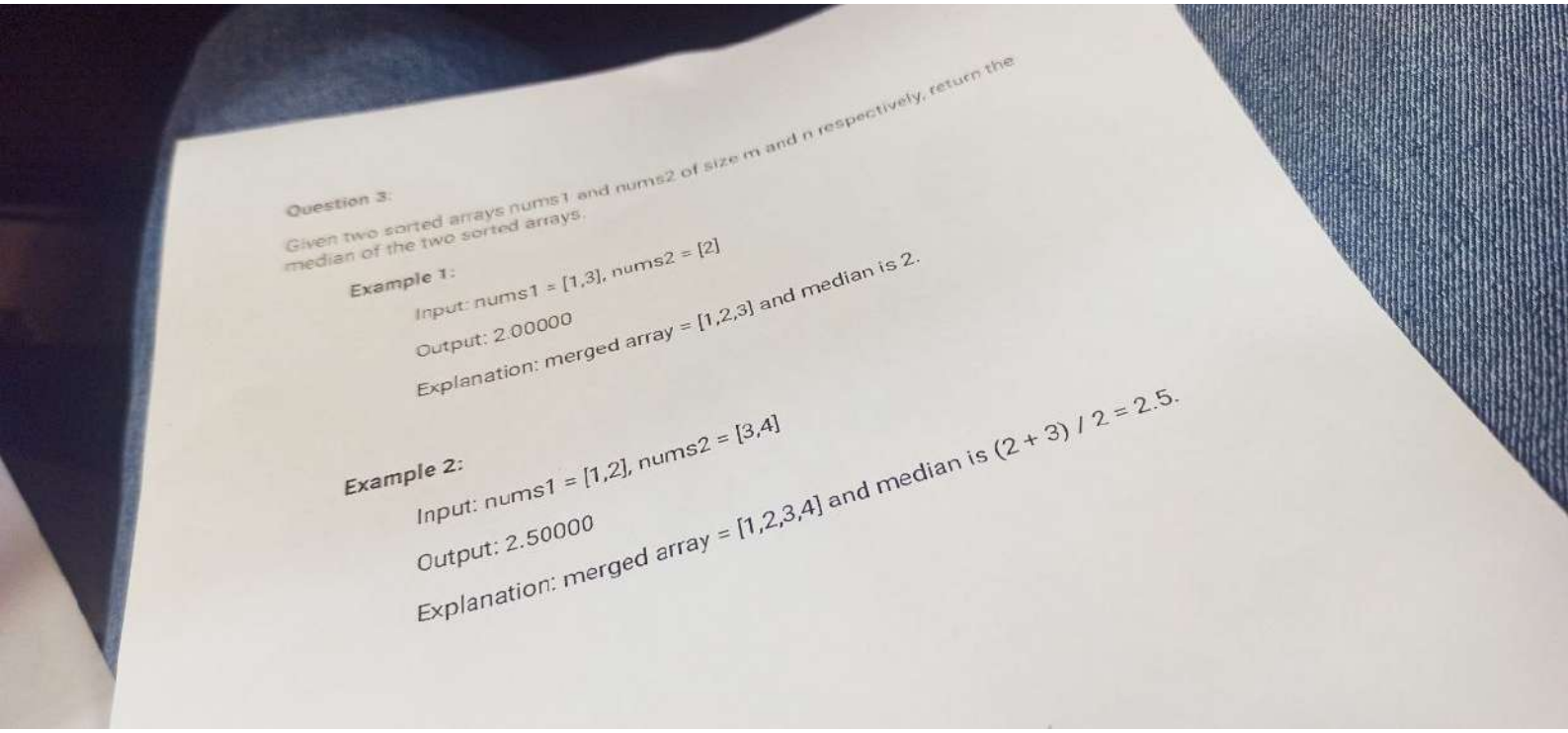
Input: s = "hello"

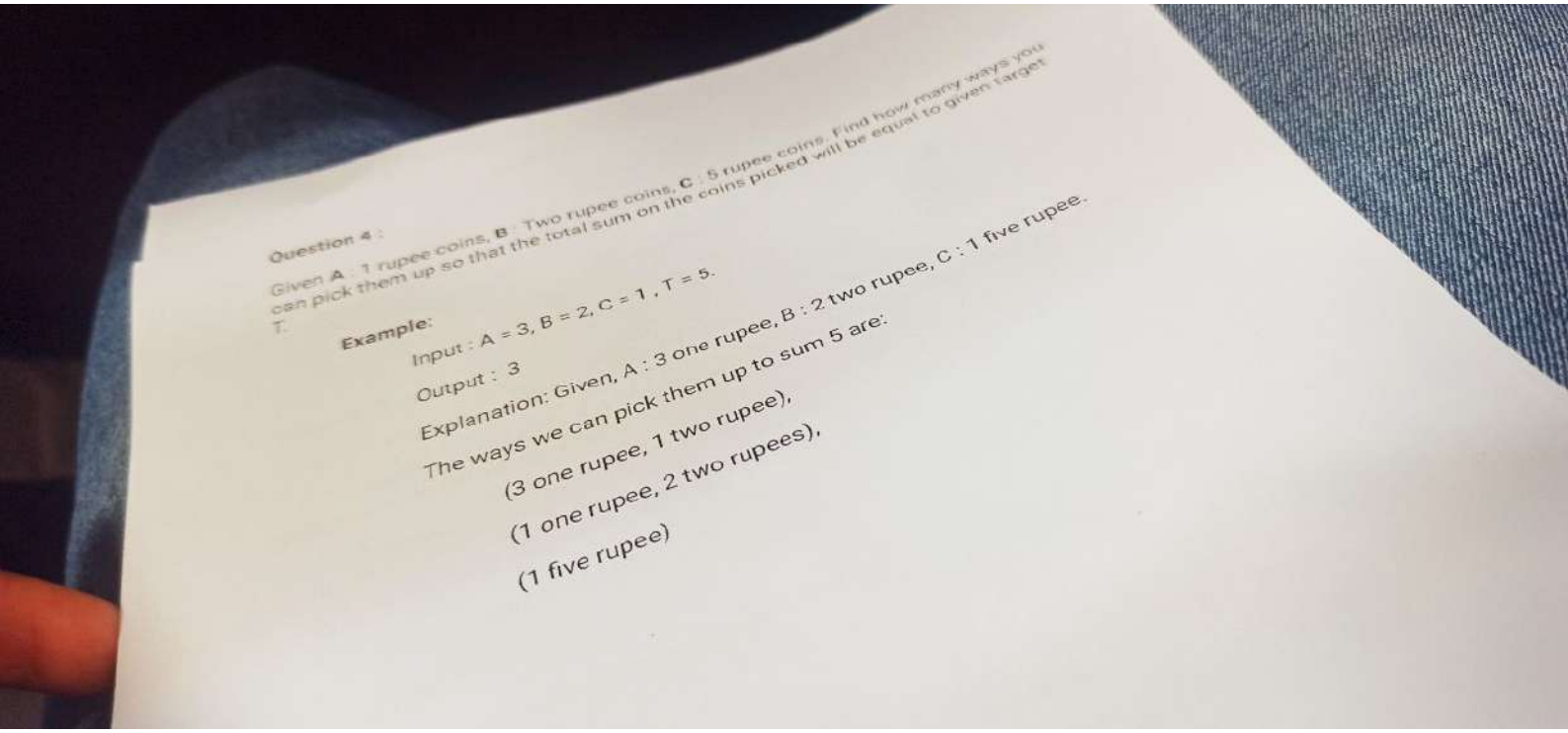
Output: "holle"

Example 2:

Input: s = "zoho corporation"

Output: "zohi carporotoon"


Question 3:
Given two sorted arrays nums1 and nums2 of size m and n respectively, return the median of the two sorted arrays.
Example 1:
Input: nums1 = [1,3], nums2 = [2]
Output: 2.00000
Explanation: merged array = [1,2,3] and median is 2.
Example 2:
Input: nums1 = [1,2], nums2 = [3,4]
Output: 2.50000
Explanation: merged array = [1,2,3,4] and median is (2 + 3) / 2 = 2.5.


Question 4 :

Given **A** : 1 rupee coins, **B** : Two rupee coins, **C** : 5 rupee coins. Find how many ways you can pick them up so that the total sum on the coins picked will be equal to given target T.

Example:

Input : A = 3, B = 2, C = 1 , T = 5.

Output : 3

Explanation: Given, A : 3 one rupee, B : 2 two rupee, C : 1 five rupee.

The ways we can pick them up to sum 5 are:

(3 one rupee, 1 two rupee),

(1 one rupee, 2 two rupees),

(1 five rupee)

Question 5:

You are given an n x n 2D matrix representing an image. Your task is to rotate the image by 90 degrees clockwise. The rotation should be done in place, meaning you have to modify the input 2D matrix directly. DO NOT allocate another 2D matrix.

Input: A square 2D matrix of size n x n, where n is the number of rows and columns.

Output: The input matrix rotated by 90 degrees clockwise.

Example 1:

Input Matrix: [[1, 2, 3],
[4, 5, 6],
[7, 8, 9]]

Output Matrix: [[7, 4, 1],
[8, 5, 2],
[9, 6, 3]]

Example 2:

Input Matrix: [[5, 1, 9, 11],
[2, 4, 8, 10],
[13, 3, 6, 7],
[15, 14, 12, 16]]

Output Matrix: [[15, 13, 2, 5],
[14, 3, 4, 1],
[12, 6, 8, 9],
[16, 7, 10, 11]]

Question 6:

Generate Parenthesis:

Given an integer n, representing the number of pairs of parentheses, write a function to generate all possible combinations of well-formed parentheses.

Example 1:

Input: n = 3

Output: ["((()))","(()())","(())()","()(())","()()()"]

Explanation: For n = 3, the function should generate all possible combinations of well-formed parentheses.

The valid combinations are: "((()))", "(()())", "(())()", "()(())", and "()()()".

Example 2:

Input: n = 1

Output: ["()"]

Explanation: For n = 1, the function should generate all possible combinations of well-formed parentheses.

The valid combination is only "()".

Constraints: 1 <= n <= 8

Note: The order of the output does not matter.

Each pair of parentheses should be well-formed, meaning that for every open parenthesis there is a corresponding closing parenthesis, and the parentheses are properly nested.