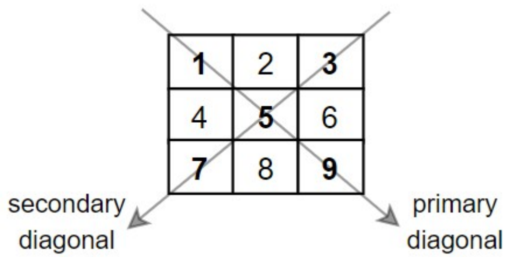
L2 Common Practice Questions

Arrays

1.Given a square matrix mat, return the sum of the matrix diagonals. Only include the sum of all the elements on the primary diagonal and all the elements on the secondary diagonal that are not part of the primary diagonal.



Example 1
Input: mat = [[1,2,3],
[4,5,6],
[7,8,9]]
Output: 25
Explanation: Diagonals sum: 1 + 5 + 9 + 3 + 7 = 25
Notice that element mat[1][1] = 5 is counted only once.

Example 2:
Input: mat = [[1,1,1,1],
[1,1,1,1],
[1,1,1,1],
[1,1,1,1]]
Output: 8

Example 3:
Input: mat = [[5]]
Output: 5
2.Given an array of positive integers arr, calculate the sum of all possible odd-length subarrays.A subarray is a contiguous sub sequence of the array. Return the sum of all odd-length sub arrays of arr.

Example 1:
Input: arr = [1,4,2,5,3]
Output: 58
Explanation: The odd-length sub arrays of arr and their sums are:
[1] = 1
[4] = 4
[2] = 2
[5] = 5
[3] = 3
[1,4,2] = 7
[4,2,5] = 11
[2,5,3] = 10
[1,4,2,5,3] = 15
If we add all these together we get 1 + 4 + 2 + 5 + 3 + 7 + 11 + 10 + 15 = 58

Example 2:
Input: arr = [1,2]
Output: 3
Explanation: There are only 2 sub arrays of odd length, [1] and [2]. Their sum is 3.

Example 3:
Input: arr = [10,11,12]
Output: 66

3.You are given an array of strings words and a string pref.Return the number of strings in words that contain pref as a prefix. A prefix of a string s is any leading contiguous substring of s.

Example 1:
Input: words = ["pay","attention","practice","attend"], pref = "at"
Output: 2
Explanation: The 2 strings that contain "at" as a prefix are: "attention" and "attend".

Example 2:
Input: words = ["leetcode","win","loops","success"], pref = "code"
Output: 0
Explanation: There are no strings that contain "code" as a prefix.

4.There are n kids with candies. You are given an integer array candies, where each candies[i] represents the number of candies the ith kid has, and an integer extraCandies, denoting the number of extra candies that you have. Return a Boolean array result of length n, where result[i] is true if, after giving the ith kid all the extraCandies, they will have the greatest number of candies among all the kids, or false otherwise.

Note that multiple kids can have the greatest number of candies.

Example 1:
Input: candies = [2,3,5,1,3], extraCandies = 3
Output: [true,true,true,false,true]
Explanation: If you give all extraCandies to:
- Kid 1, they will have 2 + 3 = 5 candies, which is the greatest among the kids.
- Kid 2, they will have 3 + 3 = 6 candies, which is the greatest among the kids.
- Kid 3, they will have 5 + 3 = 8 candies, which is the greatest among the kids.
- Kid 4, they will have 1 + 3 = 4 candies, which is not the greatest among the kids.
- Kid 5, they will have 3 + 3 = 6 candies, which is the greatest among the kids.

Example 2:
Input: candies = [4,2,1,1,2], extraCandies = 1
Output: [true,false,false,false,false]
Explanation: There is only 1 extra candy.
Kid 1 will always have the greatest number of candies, even if a different kid is given the extra candy.

Example 3:
Input: candies = [12,1,12], extraCandies = 10
Output: [true,false,true]

5.You are given a string s consisting of digits and an integer k.
A round can be completed if the length of s is greater than k. In one round, do the following:

Divide s into consecutive groups of size k such that the first k characters are in the first group, the next k characters are in the second group, and so on. Note that the size of the last group can be smaller than k.
Replace each group of s with a string representing the sum of all its digits. For example, "346" is replaced with "13" because 3 + 4 + 6 = 13.
Merge consecutive groups together to form a new string. If the length of the string is greater than k, repeat from step 1. Return s after all rounds have been completed.

Example 1:
Input: s = "11111222223", k = 3
Output: "135"
Explanation:

- For the first round, we divide s into groups of size 3: "111", "112", "222", and "23".

Then we calculate the digit sum of each group: 1 + 1 + 1 = 3, 1 + 1 + 2 = 4, 2 + 2 + 2 = 6, and 2 + 3 = 5.
So, s becomes "3" + "4" + "6" + "5" = "3465" after the first round.

- For the second round, we divide s into "346" and "5".

Then we calculate the digit sum of each group: 3 + 4 + 6 = 13, 5 = 5.
So, s becomes "13" + "5" = "135" after second round.
Now, s.length <= k, so we return "135" as the answer.

Example 2:
Input: s = "00000000", k = 3
Output: "000"
Explanation:
We divide s into "000", "000", and "00".
Then we calculate the digit sum of each group: 0 + 0 + 0 = 0, 0 + 0 + 0 = 0, and 0 + 0 = 0.
s becomes "0" + "0" + "0" = "000", whose length is equal to k, so we return "000".

6.We are given a list nums of integers representing a list compressed with run-length encoding.
Consider each adjacent pair of elements [freq, val] = [nums[2*i], nums[2*i+1]] (with i >= 0). For each such pair, there are freq elements with value val concatenated in a sublist. Concatenate all the sublists from left to right to generate the decompressed list.
Return the decompressed list.

Example 1:
Input: nums = [1,2,3,4]
Output: [2,4,4,4]
Explanation: The first pair [1,2] means we have freq = 1 and val = 2 so we generate the array [2].
The second pair [3,4] means we have freq = 3 and val = 4 so we generate [4,4,4].
At the end the concatenation [2] + [4,4,4] is [2,4,4,4].

Example 2:
Input: nums = [1,1,2,3]
Output: [1,3,3]

7.You are given an array of n strings strs, all of the same length.
The strings can be arranged such that there is one on each line, making a grid. For example, strs = ["abc", "bce", "cae"] can be arranged as:

abc

bce
cae
You want to delete the columns that are not sorted lexicographical. In the above example (0-indexed), columns 0 ('a', 'b', 'c') and 2 ('c', 'e', 'e') are sorted while column 1 ('b', 'c', 'a') is not, so you would delete column 1.

Return the number of columns that you will delete.

Example 1:

Input: strs = ["cba","daf","ghi"]
Output: 1
Explanation: The grid looks as follows:
cba
daf
ghi
Columns 0 and 2 are sorted, but column 1 is not, so you only need to delete 1 column.

Example 2:
Input: strs = ["a","b"]
Output: 0
Explanation: The grid looks as follows:
a
b
Column 0 is the only column and is sorted, so you will not delete any columns.

Example 3:
Input: strs = ["zyx","wvu","tsr"]
Output: 3
Explanation: The grid looks as follows:
zyx
wvu
tsr
All 3 columns are not sorted, so you will delete all 3.

8.Given an integer array arr, return the mean of the remaining integers after removing the smallest 5% and the largest 5% of the elements.
Answers within 10-5 of the actual answer will be considered accepted.

Example 1:
Input: arr = [1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3]
Output: 2.00000
Explanation: After erasing the minimum and the maximum values of this array, all elements are equal to 2, so the mean is 2.

Example 2:
Input: arr = [6,2,7,5,1,2,0,3,10,2,5,0,5,5,0,8,7,6,8,0]
Output: 4.00000
Example 3:
Input: arr = [6,0,7,0,7,5,7,8,3,4,0,7,8,1,6,8,1,1,2,4,8,1,9,5,4,3,8,5,10,8,6,6,1,0,6,10,8,2,3,4]
Output: 4.77778

9.You are given an array of distinct integers arr and an array of integer arrays pieces, where the integers in pieces are distinct. Your goal is to form arr by concatenating the arrays in pieces in any order. However, you are not allowed to reorder the integers in each array pieces[i].
Return true if it is possible to form the array arr from pieces. Otherwise, return false.

Example 1:
Input: arr = [15,88], pieces = [[88],[15]]
Output: true
Explanation: Concatenate [15] then [88]

Example 2:
Input: arr = [49,18,16], pieces = [[16,18,49]]
Output: false
Explanation: Even though the numbers match, we cannot reorder pieces[0].

Example 3:
Input: arr = [91,4,64,78], pieces = [[78],[4,64],[91]]
Output: true
Explanation: Concatenate [91] then [4,64] then [78]

10.You are given an m x n integer grid accounts where accounts[i][j] is the amount of money the ith customer has in the jth bank. Return the wealth that the richest customer has.
A customer's wealth is the amount of money they have in all their bank accounts. The richest customer is the customer that has the maximum wealth.

Example 1:
Input: accounts = [[1,2,3],[3,2,1]]
Output: 6
Explanation:
1st customer has wealth = 1 + 2 + 3 = 6
2nd customer has wealth = 3 + 2 + 1 = 6
Both customers are considered the richest with a wealth of 6 each, so return 6.

Example 2:
Input: accounts = [[1,5],[7,3],[3,5]]
Output: 10
Explanation:
1st customer has wealth = 6
2nd customer has wealth = 10
3rd customer has wealth = 8
The 2nd customer is the richest with a wealth of 10.

Example 3:
Input: accounts = [[2,8,7],[7,1,3],[1,9,5]]
Output: 17

Strings

1.Write a function that reverses a string. The input string is given as an array of characters s. You must do this by modifying the input array in-place with O(1) extra memory.

Example 1:
Input: s = ["h","e","l","l","o"]
Output: ["o","l","l","e","h"]
Example 2:
Input: s = ["H","a","n","n","a","h"]
Output: ["h","a","n","n","a","H"]

2.Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

Symbol	Value
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

For example, 2 is written as II in Roman numeral, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II.
Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX.
There are six instances where subtraction is used:
I can be placed before V (5) and X (10) to make 4 and 9.
X can be placed before L (50) and C (100) to make 40 and 90.
C can be placed before D (500) and M (1000) to make 400 and 900.
Given a roman numeral, convert it to an integer.

Example 1:
Input: s = "III"
Output: 3
Explanation: III = 3.

Example 2:
Input: s = "LVIII"
Output: 58
Explanation: L = 50, V= 5, III = 3.

Example 3:
Input: s = "MCMXCIV"
Output: 1994
Explanation: M = 1000, CM = 900, XC = 90 and IV = 4.

3.Given two binary strings a and b, return their sum as a binary string.
Example 1:
Input: a = "11", b = "1"
Output: "100"

Example 2:
Input: a = "1010", b = "1011"
Output: "10101"

4.A phrase is a palindrome if, after converting all uppercase letters into lowercase letters and removing all non-alphanumeric characters, it reads the same forward and backward. Alphanumeric characters include letters and numbers.
Given a string s, return true if it is a palindrome, or false otherwise.

Example 1:
Input: s = "A man, a plan, a canal: Panama"
Output: true
Explanation: "amanaplanacanalpanama" is a palindrome.

Example 2:
Input: s = "race a car"
Output: false
Explanation: "raceacar" is not a palindrome.

Example 3:
Input: s = " "
Output: true
Explanation: s is an empty string "" after removing non-alphanumeric characters.
Since an empty string reads the same forward and backward, it is a palindrome.

5.Suppose Andy and Doris want to choose a restaurant for dinner, and they both have a list of favorite restaurants represented by strings.
You need to help them find out their common interest with the least list index sum. If there is a choice tie between answers, output all of them with no order requirement. You could assume there always exists an answer.

Example 1
Input: list1 = ["Shogun","Tapioca Express","Burger King","KFC"], list2 = ["Piatti","The Grill at Torrey Pines","Hungry Hunter Steakhouse","Shogun"]
Output: ["Shogun"]
Explanation: The only restaurant they both like is "Shogun".

Example 2:
Input: list1 = ["Shogun","Tapioca Express","Burger King","KFC"], list2 = ["KFC","Shogun","Burger King"]
Output: ["Shogun"]
Explanation: The restaurant they both like and have the least index sum is "Shogun" with index sum 1 (0+1).

6.Given two strings s and goal, return true if and only if s can become goal after some number of shifts on s.
A shift on s consists of moving the leftmost character of s to the rightmost position.
For example, if s = "abcde", then it will be "bcdea" after one shift.

Example 1:
Input: s = "abcde", goal = "cdeab"

Output: true

Example 2:
Input: s = "abcde", goal = "abced"
Output: false

7.Given a string array words, return an array of all characters that show up in all strings within the words (including duplicates). You may return the answer in any order.

Example 1:
Input: words = ["bella","label","roller"]
Output: ["e","l","l"]

Example 2:
Input: words = ["cool","lock","cook"]
Output: ["c","o"]

8.You are given an array of n strings strs, all of the same length.
The strings can be arranged such that there is one on each line, making a grid. For example, strs = ["abc", "bce", "cae"] can be arranged as:
abc
bce
cae
You want to delete the columns that are not sorted lexicographically. In the above example (0-indexed), columns 0 ('a', 'b', 'c') and 2 ('c', 'e', 'e') are sorted while column 1 ('b', 'c', 'a') is not, so you would delete column 1. Return the number of columns that you will delete.

Example 1:
Input: strs = ["cba","daf","ghi"]
Output: 1
Explanation: The grid looks as follows:
cba
daf
ghi
Columns 0 and 2 are sorted, but column 1 is not, so you only need to delete 1 column.

Example 2:
Input: strs = ["a","b"]
Output: 0
Explanation: The grid looks as follows:
a
b
Column 0 is the only column and is sorted, so you will not delete any columns.

Example 3:
Input: strs = ["zyx","wvu","tsr"]
Output: 3
Explanation: The grid looks as follows:
zyx
wvu
tsr

All 3 columns are not sorted, so you will delete all 3.

9.You are given an alphanumeric string s. (Alphanumeric string is a string consisting of lowercase English letters and digits).
You have to find a permutation of the string where no letter is followed by another letter and no digit is followed by another digit. That is, no two adjacent characters have the same type.
Return the reformatted string or return an empty string if it is impossible to reformat the string.

Example 1:
Input: s = "a0b1c2"
Output: "0a1b2c"
Explanation: No two adjacent characters have the same type in "0a1b2c". "a0b1c2", "0a1b2c", "0c2a1b" are also valid permutations.

Example 2:
Input: s = "leetcode"
Output: ""
Explanation: "leetcode" has only characters so we cannot separate them by digits.

Example 3:
Input: s = "1229857369"
Output: ""
Explanation: "1229857369" has only digits so we cannot separate them by characters.

10.Display the Diagonal pattern for the string of odd length

Sample Input 1:
Enter the String: racecar
Sample Output:

```
         e
      c     c
   a           a
r                 r
   a           a
      c     c
         e
```

Sample Input 2:
Enter the String: football
Sample Output 2:
Not Possible

Matrices

1.Given a matrix and a string, find whether a string is present in the matrix or not a search can be valid from top to bottom and left to right direction only.

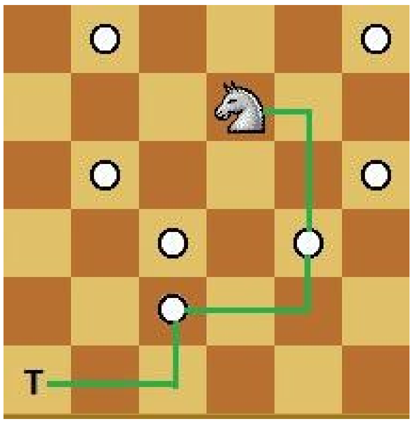
Sample Input 1
Input Matrix:
h o l z

a l o o
a b c h
o k j o

Input String:
zoho
Sample Output:
yes

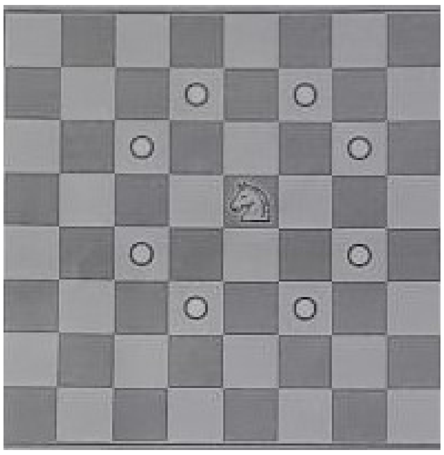
2.Given a square chessboard of N x N size, the position of the Knight and the position of a target are given. We need to find out the minimum steps a Knight will take to reach the target position.

Examples:
Input:



Output: 3
Explanation: In above diagram Knight takes 3 step to reach from (4, 5) to (1, 1)
(4, 5) -> (5, 3) -> (3, 2) -> (1, 1)

3.Here is a chessboard and a knight is placed in (4,3) position (0,0) is index of the top left position and (7,7) is the index of bottom right position, As you can see, a knight can move 8 possible positions (provided those positions are inside the 8x8 board)



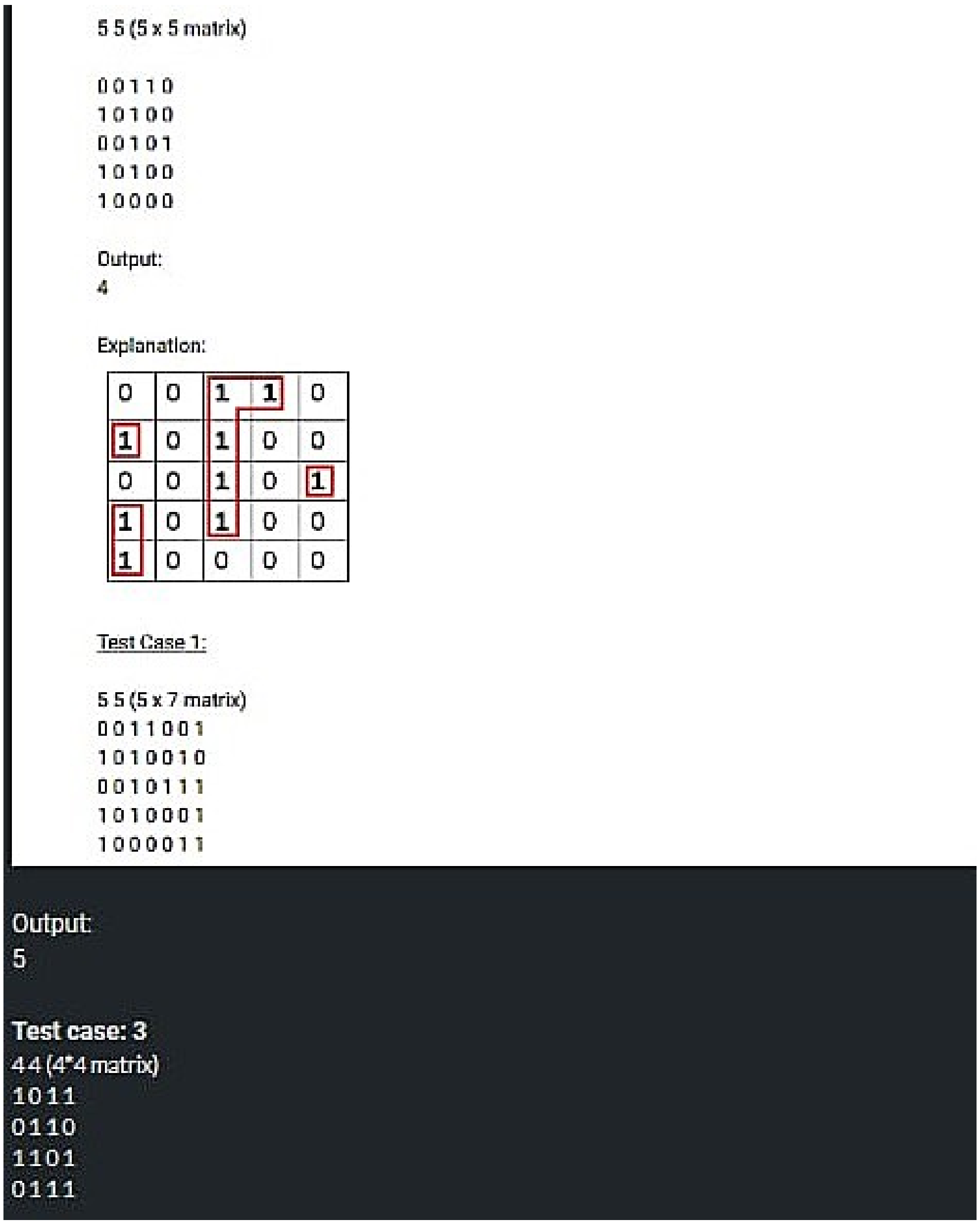
Find the minimum number of steps to be moved by the knight to reach given destination point (A,B)
(Chess rules followed)

Input:Start (0,0) Destination (2,1)
Output: 1

Input: Start (0,0) Destination (6,3)
Output: 3

4.Write a program to count the number of consecutive 1's formed by connecting adjacent 1's horizontally and vertically in the 2D grid map of 0's and 1's

Case 1:



```
5 5 (5 x 5 matrix)

0 0 1 1 0
1 0 1 0 0
0 0 1 0 1
1 0 1 0 0
1 0 0 0 0

Output:
4
```

Explanation:

0	0	1	1	0
1	0	1	0	0
0	0	1	0	1
1	0	1	0	0
1	0	0	0	0

Test Case 1:

```
5 5 (5 x 7 matrix)
0 0 1 1 0 0 1
1 0 1 0 0 1 0
0 0 1 0 1 1 1
1 0 1 0 0 0 1
1 0 0 0 0 1 1

Output:
5
```

Test case: 3

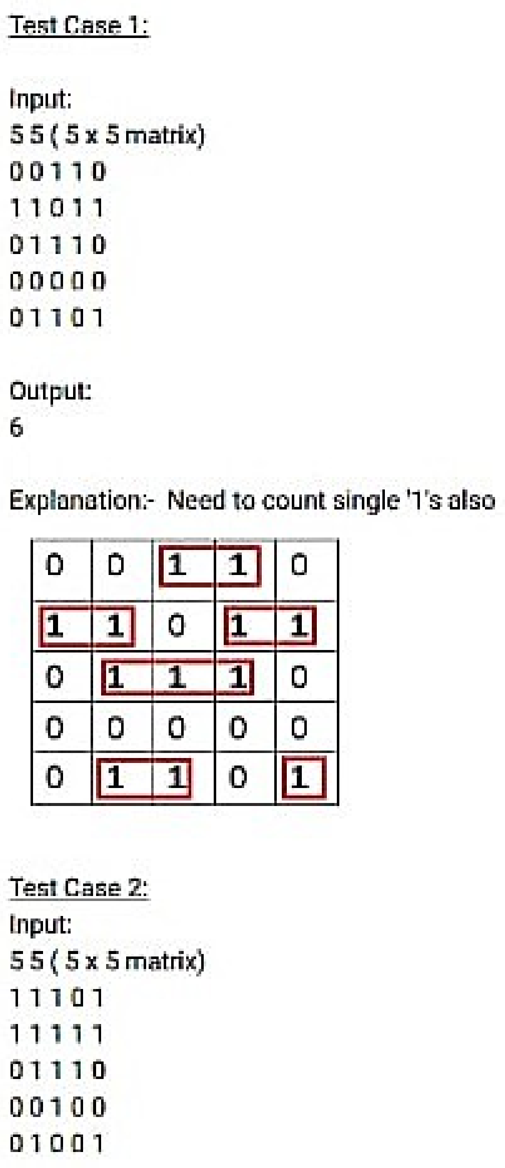
```
4 4 (4*4 matrix)
1 0 1 1
0 1 1 0
1 1 0 1
0 1 1 1
```

5.Write a program to count the number of 1's in the 2D grid map of 0's and 1's.
Case 1:
matrix: 3x3
1 0 1
0 1 1
0 0 0

Output: 4
Explanation: There are 4 number of 1's in the matrix

Case 2:
Matrix: 3x4
1 0 1 0
0 1 1 1
0 0 0 1
Output: 6

6.Write a program to count the number of consecutive 1's formed by connecting adjacent 1's horizontally in the 2D grid map of 0's and 1's.



Test Case 1:

Input:
5 5 (5 x 5 matrix)
0 0 1 1 0
1 1 0 1 1
0 1 1 1 0
0 0 0 0 0
0 1 1 0 1

Output:
6

Explanation:- Need to count single '1's also

0	0	1	1	0
1	1	0	1	1
0	1	1	1	0
0	0	0	0	0
0	1	1	0	1

Test Case 2:
Input:
5 5 (5 x 5 matrix)
1 1 1 0 1
1 1 1 1 1
0 1 1 1 0
0 0 1 0 0
0 1 0 0 1

7.A storekeeper is a game in which the player pushes boxes around in a warehouse trying to get them to target locations.
The game is represented by an m x n grid of characters grid where each element is a wall, floor, or box.
Your task is to move the box 'B' to the target position 'T' under the following rules:

The character 'S' represents the player. The player can move up, down, left, right in grid if it is a floor (empty cell).
The character '.' represents the floor which means a free cell to walk.
The character '#' represents the wall which means an obstacle (impossible to walk there).
There is only one box 'B' and one target cell 'T' in the grid.
The box can be moved to an adjacent free cell by standing next to the box and then moving in the direction of the box. This is a push.
The player cannot walk through the box.
Return the minimum number of pushes to move the box to the target. If there is no way to reach the target, return -1.
Example 1:
Input: grid =
[["#","#","#","#","#","#"],
["#","T","#","#","#","#"],
["#",".",".","B",".","#"],
["#",".","#","#",".","#"],
["#",".",".",".","S","#"],
["#","#","#","#","#","#"]]
Output: 3
Explanation: We return only the number of times the box is pushed.

Example 2:
Input: grid =
[["#","#","#","#","#","#"],
["#","T","#","#","#","#"],
["#",".",".","B",".","#"],
["#","#","#","#",".","#"],
["#",".",".",".","S","#"],
["#","#","#","#","#","#"]]
Output: -1

Example 3:
Input: grid =
[["#","#","#","#","#","#"],
["#","T",".",".","#","#"],
["#",".","#","B",".","#"],
["#",".",".",".",".","#"],
["#",".",".",".","S","#"],
["#","#","#","#","#","#"]]
Output: 5
Explanation: push the box down, left, left, up and up.

8.There is a ball in a maze with empty spaces and walls. The ball can go through empty spaces by rolling up,down,left or right, but it won't stop rolling until hitting a wall. When the ball stops, it could choose the next direction.
Given the ball's start position, the destination and the maze, find the shortest distance for the ball to stop at the destination. The distance is defined by the number of empty spaces traveled by the ball from the start position (excluded) to the destination (included). If the ball cannot stop at the destination, return
-1.

The maze is represented by a binary 2D array. 1 means the wall and 0 means the empty space. You may assume that the borders of the maze are all walls. The start and destination coordinates are represented by row and column indexes.

Example 1:
Input 1: a maze represented by a 2D array
```
0 0 1 0 0
0 0 0 0 0
0 0 0 1 0
1 1 0 1 1
0 0 0 0 0
```
Input 2: start coordinate (rowStart, colStart) = (0, 4)
Input 3: destination coordinate (rowDest, colDest) = (4, 4)
Output: 12
Explanation: One shortest way is : left -> down -> left -> down -> right -> down -> right.
The total distance is 1 + 1 + 3 + 1 + 2 + 2 + 2 = 12.

Example 2:
Input 1: a maze represented by a 2D array

```
0 0 1 0 0
0 0 0 0 0
0 0 0 1 0
1 1 0 1 1
0 0 0 0 0
```
Input 2: start coordinate (rowStart, colStart) = (0, 4)
Input 3: destination coordinate (rowDest, colDest) = (3, 2)
Output: -1
Explanation: There is no way for the ball to stop at the destination.
Note:
There is only one ball and one destination in the maze.
Both the ball and the destination exist on an empty space, and they will not be at the same position initially.
The given maze does not contain border (like the red rectangle in the example pictures), but you could assume the border of the maze are all walls.
The maze contains at least 2 empty spaces, and both the width and height of the maze won't exceed 100.

Recursion

1.Given an integer n, return true if it is a power of four. Otherwise, return false.
An integer n is a power of four, if there exists an integer x such that n == 4x.

Example 1:
Input: n = 16
Output: true

Example 2:
Input: n = 5
Output: false

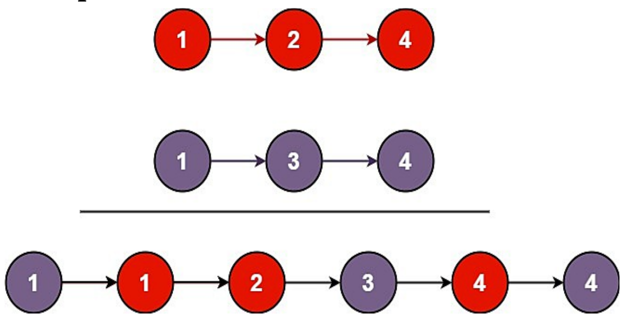
Example 3:
Input: n = 1
Output: true

2.Write a function that reverses a string. The input string is given as an array of characters s.
You must do this by modifying the input array in-place with O(1) extra memory.

Example 1:
Input: s = ["h","e","l","l","o"]
Output: ["o","l","l","e","h"]
Example 2:
Input: s = ["H","a","n","n","a","h"]
Output: ["h","a","n","n","a","H"]

3.You are given the heads of two sorted linked lists list1 and list2.
Merge the two lists in a one sorted list. The list should be made by splicing together the nodes of the first two lists.
Return the head of the merged linked list.

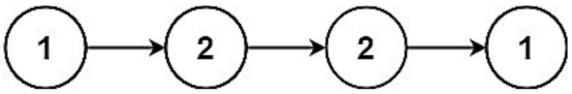
Example 1:



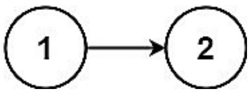
Input: list1 = [1,2,4], list2 = [1,3,4]
Output: [1,1,2,3,4,4]

Example 2:
Input: list1 = [], list2 = []
Output: []

Example 3:
Input: list1 = [], list2 = [0]
Output: [0]

4.Given the head of a singly linked list, return true if it is a palindrome or false otherwise.
Example 1:


Input: head = [1,2,2,1]
Output: true

Example 2:


Input: head = [1,2]
Output: false

5.Given an integer n, return true if it is a power of four. Otherwise, return false.
An integer n is a power of four, if there exists an integer x such that n == 4x.

Example 1:
Input: n = 16
Output: true

Example 2:
Input: n = 5
Output: false

Example 3:
Input: n = 1
Output: true

6.Given an integer n, return true if it is a power of three. Otherwise, return false.
An integer n is a power of three, if there exists an integer x such that n == 3x.

Example 1:
Input: n = 27
Output: true
Explanation: 27 = 33

Example 2:
Input: n = 0
Output: false
Explanation: There is no x where 3x = 0.

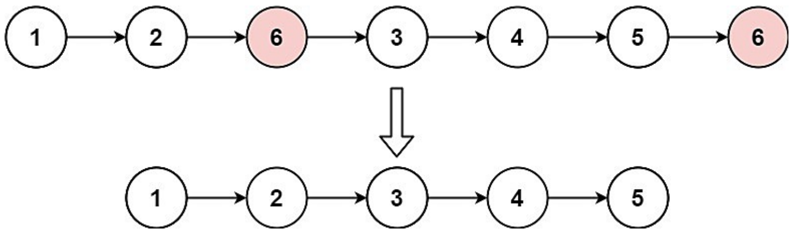
Example 3:
Input: n = -1
Output: false
Explanation: There is no x where 3x = (-1).

7.Given an integer n, return true if it is a power of two. Otherwise, return false.
An integer n is a power of two, if there exists an integer x such that n == 2x.

Example 1:
Input: n = 1
Output: true
Explanation: 20 = 1

Example 2:
Input: n = 16
Output: true
Explanation: 24 = 16

Example 3:
Input: n = 3
Output: false

8.Given the head of a linked list and an integer val, remove all the nodes of the linked list that has Node.val == val, and return the new head.
Example 1:


Input: head = [1,2,6,3,4,5,6], val = 6
Output: [1,2,3,4,5]

Example 2:
Input: head = [], val = 1
Output: []

Example 3:
Input: head = [7,7,7,7], val = 7
Output: []

Dynamic Programing

1.Given two strings s and t, return true if s is a sub sequence of t, or false otherwise.

A sub sequence of a string is a new string that is formed from the original string by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters. (i.e., "ace" is a sub sequence of "abcde" while "aec" is not).

Example 1:

Input: s = "abc", t = "ahbgdc"

Output: true

Example 2:

Input: s = "axc", t = "ahbgdc"

Output: false

2.You are given an integer array cost where cost[i] is the cost of ith step on a staircase. Once you pay the cost, you can either climb one or two steps.

You can either start from the step with index 0, or the step with index 1.

Return the minimum cost to reach the top of the floor.

Example 1:

Input: cost = [10,15,20]

Output: 15

Explanation: You will start at index 1.

- Pay 15 and climb two steps to reach the top.

The total cost is 15.

Example 2:

Input: cost = [1,100,1,1,1,100,1,1,100,1]

Output: 6

Explanation: You will start at index 0.

- Pay 1 and climb two steps to reach index 2.

- Pay 1 and climb two steps to reach index 4.

- Pay 1 and climb two steps to reach index 6.

- Pay 1 and climb one step to reach index 7.

- Pay 1 and climb two steps to reach index 9.

- Pay 1 and climb one step to reach the top.

The total cost is 6.

3.Alice and Bob take turns playing a game, with Alice starting first.

Initially, there is a number n on the chalkboard. On each player's turn, that player makes a move consisting of:

Choosing any x with 0 < x < n and n % x == 0.

Replacing the number n on the chalkboard with n - x.

Also, if a player cannot make a move, they lose the game.

Return true if and only if Alice wins the game, assuming both players play optimally.

Example 1:

Input: n = 2

Output: true

Explanation: Alice chooses 1, and Bob has no more moves.

Example 2:

Input: n = 3

Output: false

Explanation: Alice chooses 1, Bob chooses 1, and Alice has no more moves.

4.The Tribonacci sequence Tn is defined as follows:

T0 = 0, T1 = 1, T2 = 1, and Tn+3 = Tn + Tn+1 + Tn+2 for n >= 0.

Given n, return the value of Tn.

Example 1:

Input: n = 4

Output: 4

Explanation:

T_3 = 0 + 1 + 1 = 2

T_4 = 1 + 1 + 2 = 4

Example 2:

Input: n = 25

Output: 1389537

Breadth First Search:

https://www.techiedelight.com/bfs-interview-questions/

1. Write an algorithm to perform BFS on a Tree.
2. Write an algorithm to perform BFS on a Graph.
3. Transitive closure of a graph
4. Check if a graph is strongly connected or not
5. Find root vertex of a graph
6. Efficiently print all nodes between two given levels in a binary tree
7. Chess Knight Problem | Find the shortest path from source to destination
8. Shortest path in a maze – Lee Algorithm
9. Find the shortest safe route in a field with sensors present
10. Flood Fill Algorithm
11. Count number of islands
12. Find the shortest path from source to destination in a matrix that satisfies given constraints
13. Calculate the height of a binary tree
14. Delete a binary tree
15. Find minimum passes required to convert all negative values in a matrix
16. Snake and Ladder Problem
17. Find the shortest distance of every cell from a landmine inside a maze
18. Check if an undirected graph contains a cycle or not
19. Find maximum cost path in a graph from a given source to a given destination
20. Total paths in a digraph from a given source to a destination having exactly m edges

Applications of BFS:

- Copying garbage collection, Cheney's algorithm.
- Finding the shortest path between two nodes u and v, with path length measured by the number of edges (an advantage over depth–first search).
- Testing a graph for bipartiteness.
- Minimum Spanning Tree for unweighted graph.
- Web crawler.
- Finding nodes in any connected component of a graph.
- Ford–Fulkerson method for computing the maximum flow in a flow network.
- Serialization/Deserialization of a binary tree.

Depth First Search:

https://www.techiedelight.com/dfs-interview-questions/

1. Arrival and departure time of vertices in DFS
2. Find the cost of the shortest path in DAG using one pass of Bellman-Ford
3. Find the longest path in a Directed Acyclic Graph (DAG)
4. Check if a set of words can be rearranged to form a circle
5. Replace all occurrences of 0 that are not surrounded by 1 in a binary matrix
6. Find the path from source to destination in a matrix that satisfies given constraints
7. Find all occurrences of the given string in a character matrix
8. Find the length of the longest path in a matrix with consecutive characters
9. Flood Fill Algorithm
10. Generate a list of possible words from a character matrix
11. Lexicographic sorting of a given set of keys
12. Find the maximum occurring word in a given set of strings
13. Find first k maximum occurring words in a given set of strings
14. Find the shortest unique prefix for every word in an array

Applications of DFS:

- Finding connected components in a graph.
- Topological sorting in a DAG(Directed Acyclic Graph).
- Finding 2/3-(edge or vertex)-connected components.
- Finding the bridges of a graph.
- Finding strongly connected components.
- Solving puzzles with only one solution, such as mazes.
- Finding biconnectivity in graphs and many more…

Backtracking:

https://www.techiedelight.com/backtracking-interview-questions/

1. Print all possible solutions to N-Queens problem
2. Print all possible Knight's tours on a chessboard
3. Find the shortest path in a maze
4. Find the longest possible route in a matrix
5. Find the path from source to destination in a matrix that satisfies given constraints
6. Find the total number of unique paths in a maze from source to destination
7. Find all combinations of elements satisfying given constraints
8. K-Partition Problem | Printing all partitions
9. Magnet Puzzle
10. Print all k-colorable configurations of a graph (Vertex coloring of a graph)