Started on	Saturday, 3 May 2025, 11:15 AM
State	Finished
Completed on	Saturday, 3 May 2025, 11:48 AM
Time taken	33 mins 32 secs
Grade	**80.00** out of 100.00

Question **1**

Not answered

Mark 0.00 out of 20.00

Write a python program to implement quick sort using the middle element as pivot on the list of given integer values.

For example:

Input	Result
8 6 3 5 1 2 9 8 7	[1, 2, 3, 5, 6, 7, 8, 9]

Answer: (penalty regime: 0 %)

```
1
```

Question **2**

Correct

Mark 20.00 out of 20.00

Create a python program to find the longest common subsequence using Memoization Implementation.

For example:

Input	Result
AGGTAB GXTXAYB	Length of LCS is 4

Answer: (penalty regime: 0 %)

```
def longest_common_subsequence(s1, s2):

    memo = [[-1] * (len(s2) + 1) for _ in range(len(s1) + 1)]


    def lcs_helper(i, j):
        if i == 0 or j == 0:
            return 0
        if memo[i][j] != -1:
            return memo[i][j]

        if s1[i - 1] == s2[j - 1]:
            memo[i][j] = 1 + lcs_helper(i - 1, j - 1)
        else:
            memo[i][j] = max(lcs_helper(i - 1, j), lcs_helper(i, j - 1))

        return memo[i][j]


    lcs_length = lcs_helper(len(s1), len(s2))


```

	Input	Expected	Got	
✔	AGGTAB GXTXAYB	Length of LCS is 4	Length of LCS is 4	✔
✔	SAMPLE SAEMSUNG	Length of LCS is 3	Length of LCS is 3	✔
✔	saveetha sabeetha	Length of LCS is 7	Length of LCS is 7	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Create a python program to find the Edit distance between two strings using dynamic programming.

For example:

Input	Result
Cats Rats	No. of Operations required : 1

Answer: (penalty regime: 0 %)

Reset answer

```
def edit_distance(str1, str2, m, n):

    dp = [[0] * (n + 1) for _ in range(m + 1)]


    for i in range(m + 1):
        dp[i][0] = i
    for j in range(n + 1):
        dp[0][j] = j


    for i in range(1, m + 1):
        for j in range(1, n + 1):
            if str1[i - 1] == str2[j - 1]:
                dp[i][j] = dp[i - 1][j - 1]
            else:
                dp[i][j] = 1 + min(dp[i - 1][j], dp[i][j - 1], dp[i - 1][j - 1])


    return dp[m][n]

if __name__ == '__main__':
```

	Input	Expected	Got	
✔	Cats Rats	No. of Operations required : 1	No. of Operations required : 1	✔
✔	Saturday Sunday	No. of Operations required : 3	No. of Operations required : 3	✔

Passed all tests! ✔

Marks for this submission: 20.00/20.00.

Question **4**

Correct

Mark 20.00 out of 20.00

Create a python program to find the longest palindromic substring using Brute force method in a given string.

For example:

Input	Result
mojologiccigolmojo	logiccigol

Answer: (penalty regime: 0 %)

Reset answer

```
def printSubStr(str, low, high):

    for i in range(low, high + 1):
        print(str[i], end = "")

def longestPalindrome(str):
    n=len(str)
    max_len=0
    start=0
    for i in range(n):
        for j in range(1,n):
            s=str[i:j+1]
            if s==s[::-1]:
                cur=j-i+1
                if cur>max_len:
                    max_len=cur
                    start=i
    printSubStr(str, start, start + max_len - 1)

if __name__ == '__main__':

    str = input()
```

	Input	Expected	Got	
✔	mojologiccigolmojo	logiccigol	logiccigol	✔
✔	sampleelpams	pleelp	pleelp	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

Create a Python program to find longest common substring or subword (LCW) of two strings using dynamic programming with top-down approach or memoization.

Problem Description

A string r is a substring or subword of a string s if r is contained within s. A string r is a common substring of s and t if r is a substring of both s and t. A string r is a longest common substring or subword (LCW) of s and t if there is no string that is longer than r and is a common substring of s and t. The problem is to find an LCW of two given strings.

For example:

Test	Input	Result
lcw(u, v)	potato tomato	Longest Common Subword: ato

Answer: (penalty regime: 0 %)

Reset answer

```
def lcw(u, v):
    m = len(u)
    n = len(v)


    memo = [[-1] * (n + 1) for _ in range(m + 1)]


    max_length = 0
    max_i = 0

    def lcw_helper(i, j):
        nonlocal max_length, max_i

        if i == 0 or j == 0:
            return 0

        if memo[i][j] != -1:
            return memo[i][j]

        if u[i - 1] == v[j - 1]:
            memo[i][j] = lcw_helper(i - 1, j - 1) + 1
```

	Test	Input	Expected	Got	
✔	lcw(u, v)	potato tomato	Longest Common Subword: ato	Longest Common Subword: ato	✔
✔	lcw(u, v)	snakegourd bottlegourd	Longest Common Subword: egourd	Longest Common Subword: egourd	✔

Passed all tests! ✔

Marks for this submission: 20.00/20.00.