Started on	Saturday, 3 May 2025, 3:17 PM
State	Finished
Completed on	Saturday, 3 May 2025, 3:56 PM
Time taken	39 mins 9 secs
Grade	**80.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

LONGEST COMMON SUBSTRING PROBLEM

The longest common substring problem is the problem of finding the longest string (or strings) that is a substring (or are substrings) of two strings.

Answer: (penalty regime: 0 %)

```
def lcss(s1,s2):
    m=len(s1)
    n=len(s2)
    dp=[[0] *(n+1) for _ in range(m+1)]
    maxl=0
    end=0
    for i in range(1,m+1):
        for j in range(1,n+1):
            if s1[i-1]==s2[j-1]:
                dp[i][j]=dp[i-1][j-1]+1
                if dp[i][j]>maxl:
                    maxl=dp[i][j]
                    end=i
    lcs=s1[end-maxl:end]
    return lcs
s1=input()
s2=input()
res=lcss(s1,s2)
print('The longest common substring is',res)

```

	Input	Expected	Got	
✓	ABC BABA	`The longest common substring is AB`	`The longest common substring is AB`	✓
✓	abcdxyz xyzabcd	`The longest common substring is abcd`	`The longest common substring is abcd`	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

To Write a Python Program to find longest common subsequence using Dynamic Programming

For example:

Input	Result
abcbdab bdcaba	bdab

Answer: (penalty regime: 0 %)

```
def lcs(s1,s2):
    m=len(s1)
    n=len(s2)
    dp=[[0] *(n+1) for _ in range(m+1)]

    for i in range(1,m+1):
        for j in range(1,n+1):
            if s1[i-1]==s2[j-1]:
                dp[i][j]=dp[i-1][j-1]+1
            else:
                dp[i][j]=max(dp[i-1][j],dp[i][j-1])

    lcs=[]
    i=m
    j=n
    while i>0 and j>0:
        if s1[i-1]==s2[j-1]:
            lcs.append(s1[i-1])
            i-=1
            j-=1
        elif dp[i-1][j]>dp[i][j-1]:
            i-=1
```

	Input	Expected	Got	
✓	abcbdab bdcaba	bdab	bdab	✓
✓	treehouse elephant	eeh	eeh	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Not answered

Mark 0.00 out of 20.00

Write a python program to implement merge sort without using recursive function on the given list of float values.

For example:

Input	Result

`5`
`6.2`
`4.1`
`3.2`
`5.6`
`7.4`
	`left:  [6.2]`
	`Right:  [4.1]`
	`left:  [3.2]`
	`Right:  [5.6]`
	`left:  [7.4]`
	`Right:  []`
	`left:  [4.1, 6.2]`
	`Right:  [3.2, 5.6]`
	`left:  [7.4]`
	`Right:  []`
	`left:  [3.2, 4.1, 5.6, 6.2]`
	`Right:  [7.4]`
	`[3.2, 4.1, 5.6, 6.2, 7.4]`

`6`
`3.2`
`8.9`
`4.5`
`6.2`
`1.5`
`8.0`
	`left:  [3.2]`
	`Right:  [8.9]`
	`left:  [4.5]`
	`Right:  [6.2]`
	`left:  [1.5]`
	`Right:  [8.0]`
	`left:  [3.2, 8.9]`
	`Right:  [4.5, 6.2]`
	`left:  [1.5, 8.0]`
	`Right:  []`
	`left:  [3.2, 4.5, 6.2, 8.9]`
	`Right:  [1.5, 8.0]`
	`[1.5, 3.2, 4.5, 6.2, 8.0, 8.9]`

Answer: (penalty regime: 0 %)

Question **4**

Correct

Mark 20.00 out of 20.00

Create a Naive recursive python program to find the minimum number of operations to convert str1 to str2

For example:

Input	Result
Python Peithen	Edit Distance 3

Answer: (penalty regime: 0 %)

Reset answer

```
def LD(s, t):
    #########  Add your code here ###########
    if s=='':
        return len(t)
    if t=='':
        return len(s)
    if s[-1]==t[-1]:
        cost=0
    else:
        cost=1
    res=min([LD(s[:-1],t)+1, LD(s,t[:-1])+1,  LD(s[:-1],t[:-1])+cost])
    return res

str1=input()
str2=input()
print('Edit Distance',LD(str1,str2))


```

	Input	Expected	Got	
✓	Python Peithen	Edit Distance 3	Edit Distance 3	✓
✓	food money	Edit Distance 4	Edit Distance 4	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

Given a string s, return *the longest palindromic substring* in s.

Example 1:

```
Input: s = "babad"
Output: "bab"
Explanation: "aba" is also a valid answer.
```

Example 2:

```
Input: s = "cbbd"
Output: "bb"
```

For example:

Test	Input	Result
ob1.longestPalindrome(str1)	ABCBCB	BCBCB

Answer: (penalty regime: 0 %)

Reset answer

```
class Solution(object):
    def longestPalindrome(self, s):
      ###########  Add your code here #############
        n=len(s)
        dp=[[False]*n for i in range(n)]
        for x in range(n):
            dp[x][x]=True
        start=0
        maxl=1
        for l in range(2,n+1):
            for i in range(n-l+1):
                j=i+l-1
                if l==2:
                    if s[i]==s[j]:
                        dp[i][j]=True
                        maxl=l
                        start=i
                else:
                    if s[i]==s[j] and dp[i+1][j-1]:
                        dp[i][j]=True
                        maxl=l
                        start=i
```

	Test	Input	Expected	Got	
✓	ob1.longestPalindrome(str1)	ABCBCB	BCBCB	BCBCB	✓
✓	ob1.longestPalindrome(str1)	BABAD	ABA	ABA	✓

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.