| | |
|---|---|
| **Started on** | Thursday, 31 July 2025, 3:13 PM |
| **State** | Finished |
| **Completed on** | Thursday, 31 July 2025, 3:55 PM |
| **Time taken** | 42 mins 6 secs |
| **Grade** | **100.00** out of 100.00 |

Question **1**

Correct

Mark 20.00 out of 20.00

## LONGEST COMMON SUBSTRING PROBLEM

Given two strings ‘X’ and ‘Y’, find the length of the longest common substring.

**Answer:** (penalty regime: 0 %)

```
def LongComSubS(st1, st2):
  ans = 0;
  for a in range(len(st1)):
         for b in range(len(st2)):
            k = 0;
            while ((a + k) < len(st1) and (b + k) < len(st2)
        and st1[a + k] == st2[b + k]):
                k = k + 1;
            ans = max(ans, k);
  return ans;

if __name__ == '__main__':

    A = input()
    B = input()
    i = len(A)
    j = len(B)
    print('Length of Longest Common Substring is', LongComSubS(A, B))
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | ABC<br>BABA | Length of Longest Common Substring is 2 | Length of Longest Common Substring is 2 | ✓ |
| ✓ | abcdxyz<br>xyzabcd | Length of Longest Common Substring is 4 | Length of Longest Common Substring is 4 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

Create a python program to compute the edit distance between two given strings using iterative method.

**For example:**

| Input | Result |
|---|---|
| kitten<br>sitting | 3 |

**Answer:** (penalty regime: 0 %)

```
def mind(x,y):
    m=len(x)
    n=len(y)
    dp = [[0] * (n + 1) for _ in range(m + 1)]
    for i in range(m+1):
        for j in range(n+1):
            if i==0:
                dp[i][j]=j
            elif j==0:
                dp[i][j]=i
            elif x[i-1]==y[j-1]:
                dp[i][j]=dp[i-1][j-1]
            else:
                dp[i][j]=min(dp[i-1][j-1],dp[i][j-1],dp[i-1][j])+1
    return dp[m][n]
x=input()
y=input()
print(mind(x,y))
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | kitten<br>sitting | 3 | 3 | ✓ |
| ✓ | medium<br>median | 2 | 2 | ✓ |

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 longest palindromic substring using optimal algorithm Expand around center.

**For example:**

| Test | Input | Result |
|---|---|---|
| findLongestPalindromicSubstring(s) | samsunggnusgnusam | sunggnus |

**Answer:** (penalty regime: 0 %)

Reset answer

```
def expand(s, low, high):
    length = len(s)
    while low >= 0 and high < length and s[low] == s[high]:
        low = low - 1
        high = high + 1

    return s[low + 1:high]


def findLongestPalindromicSubstring(s):

    if not s or not len(s):
        return ''
    start = 0
    end = 0

    for i in range(len(s)):
        len1 = expand(s, i, i)  # Odd-length palindrome
        len2 = expand(s, i, i + 1)  # Even-length palindrome

        # Take the longer of the two palindromes
        if len(len1) > len(len2):
```

| | Test | Input | Expected | Got | |
|---|---|---|---|---|---|
| ✓ | `findLongestPalindromicSubstring(s)` | `samsunggnusgnusam` | `sunggnus` | `sunggnus` | ✓ |
| ✓ | `findLongestPalindromicSubstring(s)` | `welcomeindiaaidni` | `indiaaidni` | `indiaaidni` | ✓ |

Passed all tests! ✓

Correct

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 solution of sudoku puzzle using Backtracking.

**For example:**

| Input | Result |
|---|---|
| `solve()` | 7 5 1 8 9 2 4 6 3<br>2 3 6 1 7 4 8 9 5<br>8 9 4 5 6 3 1 7 2<br>6 4 5 3 2 9 7 1 8<br>1 2 9 4 8 7 3 5 6<br>3 7 8 6 5 1 2 4 9<br>9 1 7 2 3 5 6 8 4<br>5 6 2 7 4 8 9 3 1<br>4 8 3 9 1 6 5 2 7 |

**Answer:** (penalty regime: 0 %)

Reset answer

```python
board = [
    [0, 0, 0, 8, 0, 0, 4, 0, 3],
    [2, 0, 0, 0, 0, 4, 8, 9, 0],
    [0, 9, 0, 0, 0, 0, 0, 0, 2],
    [0, 0, 0, 0, 2, 9, 0, 1, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 7, 0, 6, 5, 0, 0, 0, 0],
    [9, 0, 0, 0, 0, 0, 0, 8, 0],
    [0, 6, 2, 7, 0, 0, 0, 0, 1],
    [4, 0, 3, 0, 0, 6, 0, 0, 0]
]

def printBoard():
    for i in range(0, 9):
        for j in range(0, 9):
            print(board[i][j], end=" ")
        print()

```

```
def isPossible(row, col, val):
    for j in range(0, 9):
        if board[row][j] == val:
            return False
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | solve() | 7 5 1 8 9 2 4 6 3<br>2 3 6 1 7 4 8 9 5<br>8 9 4 5 6 3 1 7 2<br>6 4 5 3 2 9 7 1 8<br>1 2 9 4 8 7 3 5 6<br>3 7 8 6 5 1 2 4 9<br>9 1 7 2 3 5 6 8 4<br>5 6 2 7 4 8 9 3 1<br>4 8 3 9 1 6 5 2 7 | 7 5 1 8 9 2 4 6 3<br>2 3 6 1 7 4 8 9 5<br>8 9 4 5 6 3 1 7 2<br>6 4 5 3 2 9 7 1 8<br>1 2 9 4 8 7 3 5 6<br>3 7 8 6 5 1 2 4 9<br>9 1 7 2 3 5 6 8 4<br>5 6 2 7 4 8 9 3 1<br>4 8 3 9 1 6 5 2 7 | ✓ |

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 the length of longest common subsequence using naive recursive method

**For example:**

| Input | Result |
|---|---|
| AGGTAB<br>GXTXAYB | Length of LCS is  4 |

**Answer:** (penalty regime: 0 %)

```
def lcs(x,y,m,n):
    if m==0 or n==0:
        return 0
    elif x[m-1]==y[n-1]:
        return 1+lcs(x,y,m-1,n-1)
    else:
        return max(lcs(x,y,m,n-1),lcs(x,y,m-1,n))
X = input()
Y = input()
print ("Length of LCS is ", lcs(X , Y, len(X), len(Y)) )
```

| | Input | Expected | Got | |
|---|---|---|---|---|
| ✓ | AGGTAB<br>GXTXAYB | Length of LCS is  4 | Length of LCS is  4 | ✓ |
| ✓ | saveetha<br>engineering | Length of LCS is  2 | Length of LCS is  2 | ✓ |

Passed all tests! ✓

Correct

Marks for this submission: 20.00/20.00.