|  |  |
| --- | --- |
| **Longest Common substring In C++** | |
| #include <iostream>  #include <string>  #include <vector>  using namespace std;  int LongestCommonSubstring(string s1, string s2) {  int m = s1.length();  int n = s2.length();  vector<vector<int>> dp(m + 1, vector<int>(n + 1, 0));  //int dp[m+1][n+1]={0};  int maxLen = 0;  for (int i = 1; i <= m; i++) {  for (int j = 1; j <= n; j++) {  if (s1[i - 1] == s2[j - 1]) {  dp[i][j] = dp[i - 1][j - 1] + 1;  maxLen = max(maxLen, dp[i][j]);  } else {  dp[i][j] = 0;  }  }  }  return maxLen;  }  int main() {  string s1 = "abcp";  string s2 = "abcy";  cout << LongestCommonSubstring(s1, s2) << endl;  return 0;  } | **Step-by-Step DP Table Construction**   | **i** | **j** | **s1[i-1]** | **s2[j-1]** | **Match?** | **dp[i][j] Calculation** | **Updated maxLen** | | --- | --- | --- | --- | --- | --- | --- | | 1 | 1 | a | a | ✅ | dp[0][0] + 1 = 1 | **1** | | 1 | 2 | a | b | ❌ | 0 | 1 | | 1 | 3 | a | c | ❌ | 0 | 1 | | 1 | 4 | a | y | ❌ | 0 | 1 | | 2 | 1 | b | a | ❌ | 0 | 1 | | 2 | 2 | b | b | ✅ | dp[1][1] + 1 = 2 | **2** | | 2 | 3 | b | c | ❌ | 0 | 2 | | 2 | 4 | b | y | ❌ | 0 | 2 | | 3 | 1 | c | a | ❌ | 0 | 2 | | 3 | 2 | c | b | ❌ | 0 | 2 | | 3 | 3 | c | c | ✅ | dp[2][2] + 1 = 3 | **3** | | 3 | 4 | c | y | ❌ | 0 | 3 | | 4 | 1 | p | a | ❌ | 0 | 3 | | 4 | 2 | p | b | ❌ | 0 | 3 | | 4 | 3 | p | c | ❌ | 0 | 3 | | 4 | 4 | p | y | ❌ | 0 | 3 |   **Final DP Table**   |  | **\_** | **a** | **b** | **c** | **y** | | --- | --- | --- | --- | --- | --- | | **\_** | 0 | 0 | 0 | 0 | 0 | | **a** | 0 | 1 | 0 | 0 | 0 | | **b** | 0 | 0 | 2 | 0 | 0 | | **c** | 0 | 0 | 0 | 3 | 0 | | **p** | 0 | 0 | 0 | 0 | 0 |   **Final Answer**   * **Longest Common Substring length** = **3** ("abc") * **Output**:   3 |
| Output:- 3 | |