Assignment 5 (Dynamic programming):

Cpp code

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;

int main(){
   // inputs
   string name1, name2;
   cout << "Enter string 1: ";
   cin >> name1;
   cout << "Enter string 2: ";
   cin >> name2;
   cout << "\n";
   // length and defining matrix
   int len_m = name1.length(), len_n = name2.length(), res=0;
   int matrix[len_m+1][len_n+1];
   // searching if the string is equal or not if equal increase the previous value by and 1 and
result is max of those
   for (int i=0; i<=len_m; i++)
   {
      for (int j=0; j<=len_n; j++)
      {
         if (i==0 || j==0)
         {
            matrix[i][j]=0;
         }
         else if (name1[i-1]==name2[j-1])
         {
            matrix[i][j] = matrix[i-1][j-1]+1;
            res = max(res, matrix[i][j]);
         }
         else
         {
          matrix[i][j]=0;
         }
      }
   }

   for (int i=0; i<=len_m+1; i++)
   {
      for (int j=0; j<=len_n+1; j++)
      {
         if (i==0 && j<=1 || i<=1 && j==0)
         {
            cout <<"   ";
         }
         else if (i==0 && j>1)
         {
            cout <<" " << name1[j-2] <<" ";
         }
         else if (i>1 && j==0)
```

```
            {
                cout << " " << name2[i-2] <<" ";
            }
            else
            {
              cout <<" " << matrix[i-1][j-1] <<" ";
            }
        }
        cout <<"\n";
    }
    cout << "\nlength of longest common sub string: "<<res;
    return 0;
}
```

- Test cases passed
- Completed on 19/10/22

Q/A:

1. How long did you spend on this assignment?
    a. 1hr
2. Based on your effort, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 10/10.
3. Based on your solution, what letter grade would you say you earned?
    a. On a scale of 1 to 10. I would grade this as 9/10.
4. Provide a summary of what doesn't work in your solution, along with an explanation of how you attempted to solve the problem and where you feel you struggled?
    a. First create a matrix of m+, n+1 size now imagine string 1 with m length and string 2 with n length. Now traverse over the matrix, for first row and first column set zeros and start checking the two strings with one and another with index position i and j if the two values are equal add a value 1 to previous diagonal value and increase the string value.