Exercise 1

Approach for a solution: Firstly, wrote the rough scenario for given problem. After that checked for all possible procedures to write pseudocode for it and its corresponding code. Tried to code in memorization and dynamic programming way but not able to compile code as it was throwing error and wan unable to fix that error. So, after attempting multiple times, recursion method found as easy to understand and easy to write. In recursion, there are different ways to solve it but below solution is much easier to understand.

Pseudocode:

Step 1: Input S1 and S2

Step 2: If S1 and S2 both strings are empty then return no value

Step 3: If both the strings are not empty then check common characters between them

Step 4: Take 1st character of S1 and check if it exists in S2. If it exists, then consider that character for LCS and go for next character of S1 and search that character in remaining character list of S2(character start onwards last matched character) likewise check all character of S1 one by one.

Step 5: If 1st character of S1 is not present in S2 then cursor move to next position of character in S1 and check all the characters in S2 from start till get matching character.

Step 6: At the end program return longest common subsequence

GitHub Link:

https://github.com/shivangigujar23/Big-Data-Programming/blob/master/Exercise1.py

Exercise 2

Approach for a solution: While trying to write pseudocode for this exercise, was not able to get result for all the possible combinations of S1 and S2. So, tried to go for making of pseudocode for those combination which are easy to test like both the strings empty, both the strings contain same number of characters and are equal, one string is empty and other one contains some characters. After that tried to code for dot and star but not able to get 100% output as required. For some cases it gave correct output but for others it did not work. Earlier used functions like re.match, re.findall, re.search to get the values for dot and star combinations of string (as per recent class guidelines came to know that libraries cannot be used for this exercise) so changed code again and with the help of while loop try to solve the problem of dot and star combination.

Pseudocode:

Step 1: Input S1 and S2

Step 2: Define characters which are going to use in code

Step 3:

Case 1 – If len(S1) == 0 and len(S2) == 0, return true

Case 2 – If ((len(S2) > 1 and S2[0] == '*') or (len(S2) > 1 and S2[0] == '.')) and len(S1) == 0, return false

Case 3 – If S1 = [A-Z] and S2 = [A-Z] and S1 is substring of S2, return true

Case 4 - If (S1[i] != S2[j]), return false

Case 5 - If (S1[i] == S2[j]) and S2[j] contains '.' or '*' then match each character of S1 with S2 and return true on matching and false on mismatch.

GitHub Link:

https://github.com/shivangigujar23/Big-Data-Programming/blob/master/Exercise2.py

Referred links –

https://docs.python.org/2/library/re.html

https://www.w3schools.com/python/python_while_loops.asp

https://stackoverflow.com/questions/3437059/does-python-have-a-string-contains-substring-method

https://www.geeksforgeeks.org/printing-longest-common-subsequence/

https://www.geeksforgeeks.org/longest-common-subsequence-dp-using-memoization/

https://www.geeksforgeeks.org/regular-expression-python-examples-set-1/

https://www.tutorialspoint.com/python/python_reg_expressions.htm