CSE 301 - Algorithms Lab Session 5

Deadline: 8.11.2023 - 23:59
Format: Two implementations are in a single code **text file**. Extension should be the programming language you used.
Rules: Individual work. Plagiarism is not accepted, and once notified, the grade for the assignment will be automatically zero.
Suggestion: You can start a conversation on the questions on the team's general page and expect us to help. You can also message me or Gökhan Hoca if you can't proceed with your answer. However, we recommend you keep the questions the day before the deadline.

1. Cutting the Logs

You are given a "n" inches long log(tree log) and a table of prices for each inch of a piece. You need to recursively determine the maximum revenue r_n obtained by cutting up the rod.

Length[i]	= 1	2	3	4	5	6	7	8	9	10
Prices[i]	= 1	5	8	9	10	17	17	20	24	30

n = 4 -> 1,3 = 1 + 8 = 9, 2,2 = 5 + 5 = 10, 3, 1 = 8 + 1 = 9
1 + 1 + 2 = 7, 1 + 2 + 1 = 7, 2 + 1 + 1 = 7, 1 + 1 + 1 + 1 = 4

Hint: Keep in mind how we discover this operation in a recursive way

- ★ Implement a running program in any programming language for this problem with a recursive (self-calling) method.
- ★ Generate random integer input arrays of 10, 100, 1000, and 10000. Using the system clock library of your programming language (you'll be able to find it through some googling, but if you cannot, message me on Teams), calculate the milliseconds while your program is executing. You can get the current value of milliseconds from your computer system before calling your recursive function in your "main" function. Then, get the current value of milliseconds after the execution ended in the same function. The difference will show you the milliseconds of your execution.
- ★ Print out the resulting times (in any datetime format) in a table for execution for each input (10, 100, 1000, 10000)
- ★ If you were to show which pieces you chose in the algorithm how would you do it? You can either explain in the comment section or can show a small implementation.

2. Longest Common Substrings

We are given two strings, X and Y, with lengths of m and n, respectively. And we wish to find the maximum length of a substring that occurs in both strings. Your main steps will be:

1. Base case: If either of the strings is empty, the result should be 0.
2. Matching character: Compare the last characters of both input strings. If they match:
    i. The LCS includes this common character. So, increment the length of the LCS by 1.
    ii. Recursively call the function with the last characters removed from both strings (m - 1 and n - 1).
    iii. This step effectively finds the LCS length of the remaining substrings after removing the matched character.
3. Character Mismatch: If the last characters of X and Y do not match:
    i. Either: Exclude the last character of string X and find the LCS of the remaining substring and Y. So, call lcs_recursive(X, Y, m, n - 1).
    ii. Or: Exclude the last character of string Y and find the LCS of X and the remaining substring of Y. So, call lcs_recursive(X, Y, m - 1, n).
4. Return:
    i. Return the maximum LCS length calculated from step 3. This value represents the length of the LCS of the original input strings X and Y.
5. Repeat:
    i. Recursively apply these steps for smaller subproblems until you reach the base cases and can calculate the LCS length for each substring.

★ Implement a running program in any programming language for this problem with a recursive (self-calling) method.
★ Generate random strings with sizes of 10, 100, 1000, and 10000. Using the system clock library of your programming language (you'll be able to find it through some googling, but if you cannot, message me on Teams), calculate the milliseconds while your program is executing. You can get the current value of milliseconds from your computer system before calling your recursive function in your "main" function. Then, get the current value of milliseconds after the execution ended in the same function. The difference will show you the milliseconds of your execution.
★ Print out the resulting times (in any datetime format) in a table for execution for each input (10, 100, 1000, 10000)
★ If you were to show the resulting substring (not just the strength), how would you solve this problem? You can either explain in the comment section or can show a small implementation.