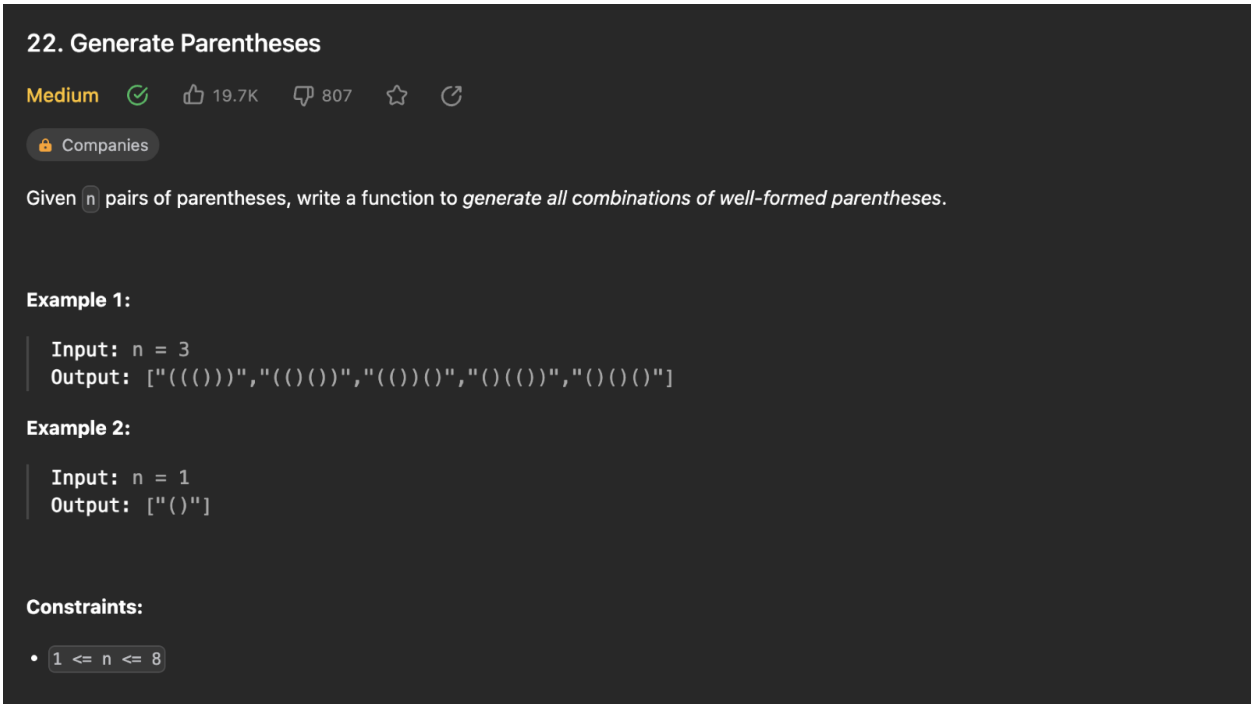
MCS – 253P ADVANCED PROGRAMMING AND PROBLEM SOLVING

LAB 1 Write Up (Generate Parenthesis)

Aswin Sampath
saswin@uci.edu
53844684

Date: 10/06/2023

Question:



22. Generate Parentheses

Medium 19.7K 807

Companies

Given `n` pairs of parentheses, write a function to *generate all combinations of well-formed parentheses*.

Example 1:

```
Input: n = 3
Output: ["((()))","(()())","(())()","()(())","()()()"]
```

Example 2:

```
Input: n = 1
Output: ["()"]
```

Constraints:

- `1 <= n <= 8`

Writeup:

Understanding the Problem:

- Problem: Generate valid combinations of parentheses for a given 'n'.
- Understanding the input range of ‘n.’
- Identifying what constitutes a valid combination?
- Observing how the output should be formatted?

Identifying Edge Cases:

- Edge Cases:
    - When 'n' is 0, the result should be an empty vector.
    - When 'n' is negative, the result should be an empty vector.
    - When 'n' is 1, the result should be a vector containing "()".

Effective Test Cases:

- Test Case 1:
    - Input: n = 3
    - Expected Output: ["((()))","(() ())","(()) ()","() (())","() () ()"]
- Test Case 2:
    - Input n=2
    - Expected Output: ["(())", "() ()"]

Algorithmic Solution:

- Use a recursive approach to generate combinations of parentheses.
- Initialize a function that takes 'n' as input.
- Within the function, initialize 'left' and 'right' counters to 'n'.
- Create a recursive function that appends '(' when 'left' is greater than 0 and ')' when 'right' is greater than 'left' and 'right' is greater than 0.
- When both 'left' and 'right' are 0, add the generated combination to the result vector.
- Return the result vector.

Time and Space Complexity Analysis:

- Time Complexity: O(2^n) - In the worst case, the algorithm explores 2^n possibilities.
- Space Complexity: O(2^n) - The space complexity is also proportional to the number of valid combinations.