**Generate All Parantheses**

**Understanding Problem:**

**How to know if this is a Recursive Problem:**

* At each point we have 2 choices: Use ‘(‘ or ‘)’
* At each point we need to make a decision, if the generated string is a balanced parentheses only then choose otherwise reject

**Recursion Tree**

Here we make use of input-output method for making the recursive tree:

Input : int

Output: str

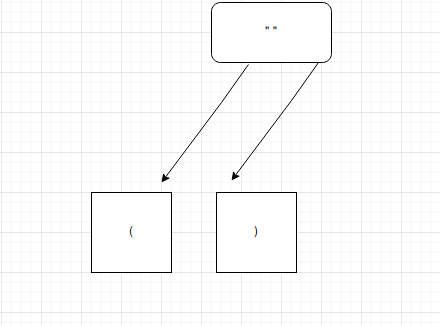
Init :

open = n

close = n

result = “”

‘ ‘



After making a dry run, we can see that from every state, open bracket choice is available provided open is not 0

But close bracket choice not always present.

Close bracket choice available only when close > open

**Solution:**

A recursive function with params: open, close, output

Base case:

if open==0 and close==0 add output to result

# Recursion

# open choice always available provided open != 0

If open!=0:

Add output + “(“

Call func( open-1, close, output + “(“ )

# close choice available only when close > open

If close > open:

Add output + “)”

Call func( open, close-1, output + “)”

**Code:**

op = n

close = n

res = []

def solve(op, close, r):

# Base case

if op == 0 and close == 0:

print(r)

res.append(r)

return

# open choice always exists provided open != 0

if op != 0:

r2 = r

r2 += "("

solve(op-1, close, r2)

# close choice only when close>op

if close > op:

r2 = r

r2 += ")"

solve(op, close-1, r2)

solve(op, close, "")

return res