```
# you can have more than 1 function calls in a recursive function

# 1, 1, 2, 3, 5 , 8, 13, 21
# 1  2  3  4   5  6  7   8

# f(n) = f(n-1)+f(n-2)

def getFibonacciNum(n):
  print("Function call n: -->", n)
  if n==1 or n==2:
    return 1
  return getFibonacciNum(n-1)+getFibonacciNum(n-2) # getFibonacciNum(7) + getFibonacciNum


getFibonacciNum(5)
```

```
Function call n: --> 5
Function call n: --> 4
Function call n: --> 3
Function call n: --> 2
Function call n: --> 1
Function call n: --> 2
Function call n: --> 3
Function call n: --> 2
Function call n: --> 1
5
```

```
# GATE DA

def count(child_dict, i):
 if i not in child_dict.keys():
  return 1
 ans = 1
 for j in child_dict[i]:
  ans += count(child_dict, j)
 return ans
child_dict = dict()
child_dict[0] = [1,2]
child_dict[1] = [3,4,5]
child_dict[2] = [6,7,8]
print(count(child_dict,0))

# Which ONE of the following is the output of this code?
```

```
9
```

```
# Consider the following Python function:
def fun(D, s1, s2):
 if s1 < s2:
  D[s1], D[s2] = D[s2], D[s1]
  fun(D, s1+1, s2-1)

d = [100,200,5,6,1000,1,10000,12,500]
print(d)

def myFun(n, sum):
  k = j = 0
  if n>0:
    k = n%10
    j = n//10
    sum = sum+k
    myFun(j, sum)
    print(k)

a = 2048
sum = 0
myFun(a, sum)
print(sum)
```

```
2
0
4
8
0
```

Start coding or generate with AI.