Mastering Data Structures and Algorithms
Homework 6

Yiying Peng (Christine)

March 12, 2019

Problem

Create a recursive function that takes a string of digits and counts the possible ways in which the string can be partition into a sequence of even numbers:

input 10 ==> only one way to partition into even numbers 10

nput 22 ==> 2 2, 22. result is 2

input 333 => result is 0

clarifications:

For this HW what you have to do is test all possible partitions of the digits provided and count the ones that are even: i.e. if you get the digits abc you have the following possible partitions:

P1: $\{$a, b, c$\}$, P2: $\{$ab, c$\}$, P3: $\{$a, bc$\}$, P4: $\{$abc$\}$

you have to test whether the numbers in each partition are even, only then count that partition.

Sol. My Python algorithm implementation is as follows.

```
class Solution:
    def count_even_amount(self, idx, string):
        if len(string) == idx:
            return 0
        if int(string[idx]) % 2 == 0:
            return 1 + self.count_even_amount(idx+1, string)
        return self.count_even_amount(idx+1, string)

    def count_even_partition(self, string):
        n = len(string)
        if int(string[n-1]) % 2 != 0:
            return 0
        return 2 ** (self.count_even_amount(0, string) - 1)


sol = Solution()
test_cases = ['10', '22', '333']
for test_case in test_cases:
    print(sol.count_even_partition(test_case))
```

The time complexity is $O(n)$, and the space complexity is also $O(1)$. □