Report Term Project

Algorithm Design (CS4402)

Section 541 Semester 1/2016

Submitted to Aj. Thitipong Tanprasert

Submitted by

Mr. Kanin Kunapermsiri 5716712

Question ID: 1036

Question Name: Lucky Ticket

Question Difficulty: 299

Time Limit: 2.0 second

Memory Limit: 64 MB

Detail:

You are given a number $1 \le N \le 50$. Every ticket has its $2N$-digit number. We call a ticket lucky, if the sum of its first N digits is equal to the sum of its last N digits. You are also given the sum of ALL digits in the number. Your task is to count an amount of lucky numbers, having the specified sum of ALL digits.

Input:

Two space-separated numbers: N and S. Here S is the sum of all digits. Assume that $0 \le S \le 1000$.

Output:

The amount of lucky tickets.

Submissions:

Recent submissions

Author: Kanin

ID	Date	Author	Problem	Language	Judgement result	Test #	Execution time	Memory used
7151697	22:16:10 30 Nov 2016	Kanin	1036. Lucky Tickets	Python 2.7	Accepted		0.124	632 KB
7151676	22:04:54 30 Nov 2016	Kanin	1036. Lucky Tickets	Python 2.7	Accepted		0.14	632 KB
7151665	21:59:23 30 Nov 2016	Kanin	1036. Lucky Tickets	Python 2.7	Wrong answer	7	0.031	236 KB

Show 10 | 30 | 100 lines per page • Refresh every 15 | 60 | 240 seconds | no refresh

Code:

```
x = raw_input().split()
n = int(x[0])
s = int(x[1])

mm = [[0 for i in range(s / 2 + 1)] for j in range(n + 1)]

def LuckyTicket(n, s):
    global mm

    if n * 9 < s:
        return 0
    if s == 0:
        return 1

    mm[0][0] = 1

    for i in range(1, n + 1):

        for j in range(s + 1):

            for k in range(10):
                if j >= k:
                    mm[i][j] += mm[i-1][j-k]

    return mm[n][s]

if s&1 == 0:
    print LuckyTicket(n, s>>1) ** 2
else:
    print 0
```

The idea of solution is to generate every digit combination. Since first half of total digits must has the same value of second half, so I can divide total digits by half before do calculation.

The maximum of each digits is 9. If sum of every 9 is less than total sum, then there is no lucky ticket.

I use dynamic programming to reduce running time. I initiate table size of half of total digits and total sum.

As it is finding all combination, so big O is $O(n^2)$