

Result1.

随机测试情况:

##随机数试验

import random

a = random.randint(1, 100)

b = random.randint(1, 100)

c = random.randint(1, 100)

values(a, b, c)

input: a=32, b=48, c=16 无输出值

input: a=77, b=20, c=98

[x,y,z]: [98, 77, 20]

out: $98 + 77 - 10 \times 20 = -25$

-25

input: a=67, b=34, c=7

[x,y,z]: [67, 34, 7]

out: $67 + 34 - 10 \times 7 = 31$

31

给定情况:

#当 a=5, b=15, c=10 时

a,b,c=5,15,10

values(a, b, c)

input: a=5, b=15, c=10 无输出值

Result2.

test_list = [1,3,5,7,9]

$F(1) = F(\text{ceil}(1/3)) + 2 \times 1 = 1$

$F(3) = F(\text{ceil}(3/3)) + 2 \times 3 = 7$

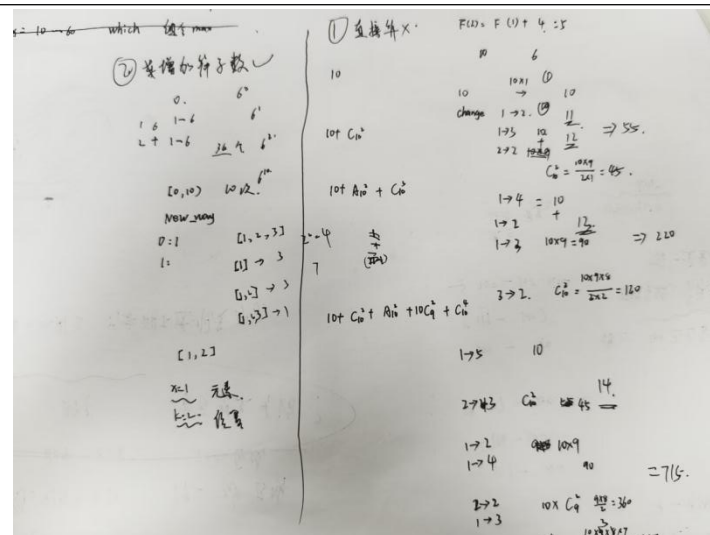
$F(5) = F(\text{ceil}(5/3)) + 2 \times 5 = 15$

$F(7) = F(\text{ceil}(7/3)) + 2 \times 7 = 21$

$F(9) = F(\text{ceil}(9/3)) + 2 \times 9 = 25$

Result3.

思路:



3.1:

```
def Find_number_of_ways(x):
    dice=10 #10个骰子
    faces=6 #每个骰子有6个面

    # 开始时，没有掷骰子，点数和为0的方法有 1 种
    ways={0: 1}

    # 加一个骰子，更新所有可能的和(骰子10)
    for i in range(dice):
        new_ways={}
        for current_sum, count in ways.items():
            # 当前已有的和加上一个新骰子(面6)
            for face in range(1, faces + 1):
                new_sum=current_sum+face
                new_ways[new_sum]=new_ways.get(new_sum, 0)+count
        ways=new_ways # 更新为新的结果

    # 返回点数和为x的方法数
    return ways.get(x, 0)
```

3.2

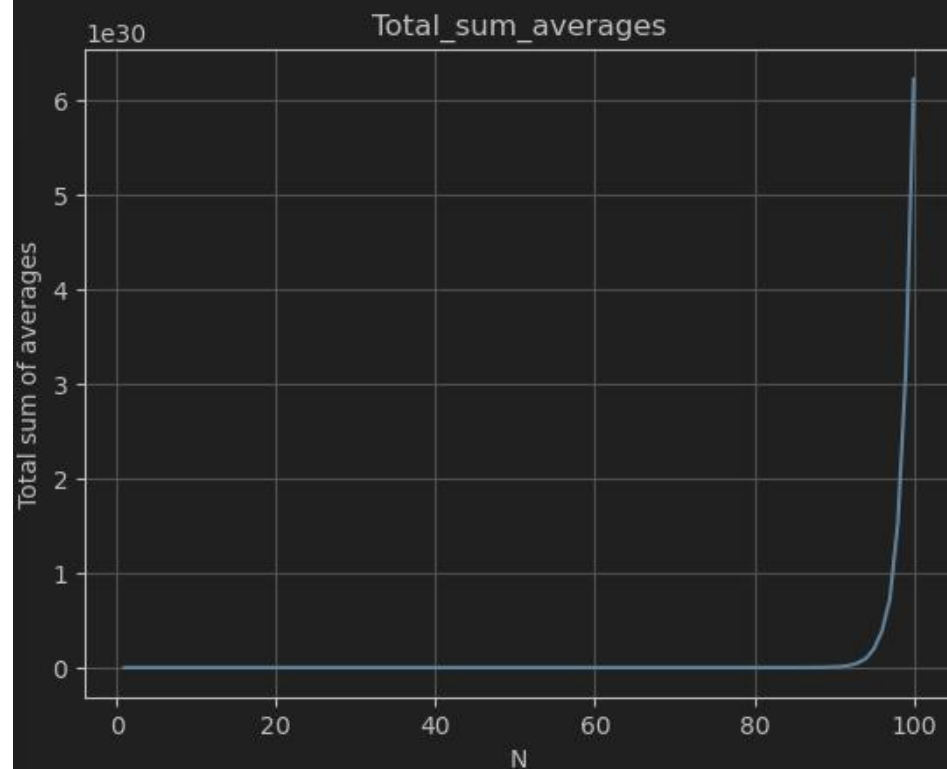
```
Number_of_ways = [1, 10, 55, 220, 715, 2002, 4995, 11340, 23760,
46420, 85228, 147940, 243925, 383470, 576565, 831204, 1151370,
1535040, 1972630, 2446300, 2930455, 3393610, 3801535, 4121260,
4325310, 4395456, 4325310, 4121260, 3801535, 3393610, 2930455,
2446300, 1972630, 1535040, 1151370, 831204, 576565, 383470, 243925,
147940, 85228, 46420, 23760, 11340, 4995, 2002, 715, 220, 55, 10,
1]

点数和为 35 的时候方法最多，共 4395456 种。
```

Result4.

结果:

```
已完成 N=20, 当前总和平均值: 6029306.25  
已完成 N=40, 当前总和平均值: 5525045929569.38  
已完成 N=60, 当前总和平均值: 5495592505292637184.00  
已完成 N=80, 当前总和平均值: 5742397643169487506112512.00  
已完成 N=100, 当前总和平均值: 6224164447120605415592890990592.00
```



应该是指数型增长，子集数量随 N 呈指数增长 (2^N)，虽然计算的是平均值之和，但子集数量的指数增长会影响最终结果。

参考链接: [Dynamic Programming or DP - GeeksforGeeks](https://www.geeksforgeeks.org/dynamic-programming-or-dp/)

Result5.

完整矩阵:

```
第1行: [1, 1, 0, 1, 0, 0, 0, 0]
第2行: [0, 0, 1, 1, 1, 0, 1, 1]
第3行: [0, 0, 1, 0, 0, 1, 1, 0]
第4行: [0, 1, 1, 1, 0, 1, 0, 0]
第5行: [0, 0, 0, 1, 0, 0, 1, 1]
第6行: [0, 1, 1, 1, 0, 0, 1, 1]
第7行: [0, 0, 0, 1, 0, 1, 0, 1]
第8行: [0, 0, 1, 0, 0, 1, 0, 1]
第9行: [1, 0, 0, 0, 1, 1, 1, 0]
第10行: [0, 0, 0, 0, 0, 1, 1, 1]
```

===1000次运行统计 ===

运行次数: 1000

总路径数: 445

平均路径数: 0.4450

参考链接: [Grid Unique Paths - Count Paths in matrix - GeeksforGeeks](#)
[彻底理解动态规划 哔哩哔哩 bilibili](#) (14 分钟往后涉及路线问题)