**（算法总结）寻找组合数，求出从整数1到n中和为m的所有组合**

2018年07月12日 12:44:15 [奋斗的小炎](https://me.csdn.net/Little_Fire) 阅读数 2203

采用背包问题原理，仅考虑具有最大的数字n是否存在与结果集合中，考虑以下两种情形：

（1）n在集合中，剩下的n-1个数字需要组成一个和为n-m的组合；

（2）n不在集合中，剩下的n-1个数字仍需要组成和为m的组合；

由于需要给出所有的组合可能，因此是一个回溯的过程。

算法设计思路：

由于是个回溯递归的过程，因此需要首先给出递归终止条件：当需要求和的数字小于等于0或所有数字都用完了的时候，就是程序终止的时候。

用一个列表数组item存储当前的候选集合，result列表存放最终的满足条件的结果，当满足条件时，将item加入result中。

因此，回溯的代码如下：

1. def find\_combine(item, n, m, result):
2. if n <= 0 or m <= 0:
3. return
4. if n == m:
5. result.append(item + [n])
6. item.append(n)
7. find\_combine(item, n-1, m-n, result)
8. item.pop()
9. find\_combine(item, n-1, m, result)

用一组数字，运行一下，这里假设n=9，m=10

1. if \_\_name\_\_ == '\_\_main\_\_':
2. item = []
3. result = []
4. find\_combine(item, 9, 10, result)
5. for r in result:
6. print(r)

运行结果如下：

1. [9, 1]
2. [8, 2]
3. [7, 3]
4. [7, 2, 1]
5. [6, 4]
6. [6, 3, 1]
7. [5, 4, 1]
8. [5, 3, 2]
9. [4, 3, 2, 1]