

# 你的背包



包庆君

2020年3月22日

## 问题提出 Problem

给定一个列表 L 其中均为不重复的正整数，一个目标值 T，找出列表中，所有相加和为目标数的组合，输出其所在位置。

输入

L = [6, 3, 9, 12, 15]

T = 21

输出

[1, 2, 4]

[3, 4]

[1, 5]

## 问题提出 Problem

### 001. 两数之和

给定一个整数数组 `nums` 和一个目标值 `target`，请你在该数组中找出和为目标值的那两个整数，并返回他们的数组下标。

给定 `nums = [2, 7, 11, 15]` `target = 9`

因为 `nums[0] + nums[1] = 2 + 7 = 9`

所以返回 `[0, 1]`

<https://leetcode-cn.com/problems/two-sum/>

## 问题提出 Problem

背包问题：

有一个背包容量为  $T$ ，有  $n$  个不同大小的物品要装入背包，要求必须正好装满。请找出**所有**搭配方案。

输入

$L = [6, 3, 9, 12, 15, 2, 1]$

$T = 21$

输出

$[1, 2, 4]$

$[3, 4]$

$[1, 5]$



## 寻找思路 Solution

`L = [6, 3, 9, 12, 15]`      `T = 21`

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

$O(n^2)$

## 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

$j = 1:$



$i = 0:$



$L = [6, 3, 9, 12, 15]$

Sum = 9

Answer:

## 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

$j = 2:$

$i = 0:$

$L = [6, 3, 9, 12, 15]$

Sum = 15

Answer:



## 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

Answer:

$j = 3:$



$i = 0:$



$L = [6, 3, 9, 12, 15]$

Sum = 18

## 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

$j = 4:$



$i = 0:$



$L = [6, 3, 9, 12, 15]$

Sum = 21      OK

Answer:

[1, 5]

## 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

Answer:

[1, 5]

$j = 2:$

$i = 1:$

$L = [6, 3, 9, 12, 15]$

Sum = 12

## 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

Answer:

[1, 5]

$j = 3:$

$i = 1:$

$L = [6, 3, 9, 12, 15]$

Sum = 15



## 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

Answer:

[1, 5]

$j = 4:$    
 $i = 1:$    
 $L = [6, 3, 9, 12, 15]$   
  
 $\text{Sum} = 18$

# 寻找思路 Solution

$L = [6, 3, 9, 12, 15]$        $T = 21$

```
for i in range( len( L ) ) :  
    for j in range( i + 1, len( L ) ) :  
        if( L[i] + L[j] == T ) :  
            print( [i + 1, j + 1])
```

$j = 4:$

$i = 3:$

$L = [6, 3, 9, 12, 15]$

Sum = 24

Answer:

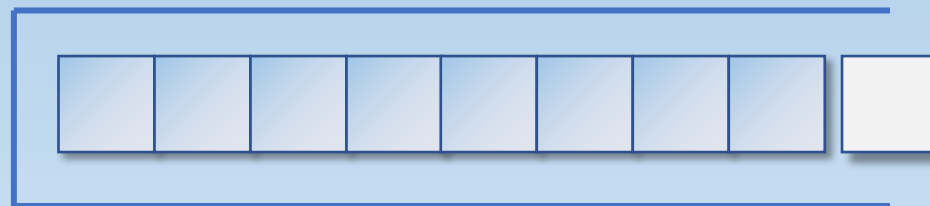
[1, 5]

[3, 4]

**Finish!**

# 栈与队列 (Stack and Queue)

栈 Stack



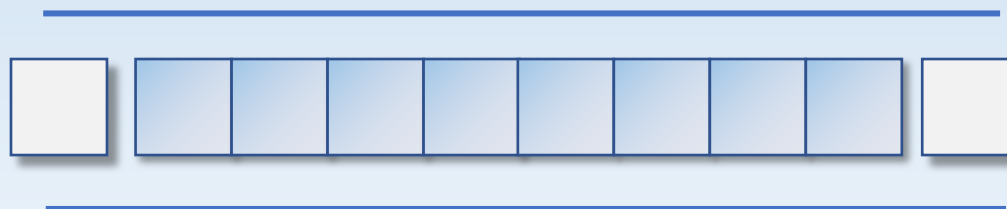
入栈 push



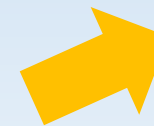
出栈 pop



队列 Queue



出队 dequeue

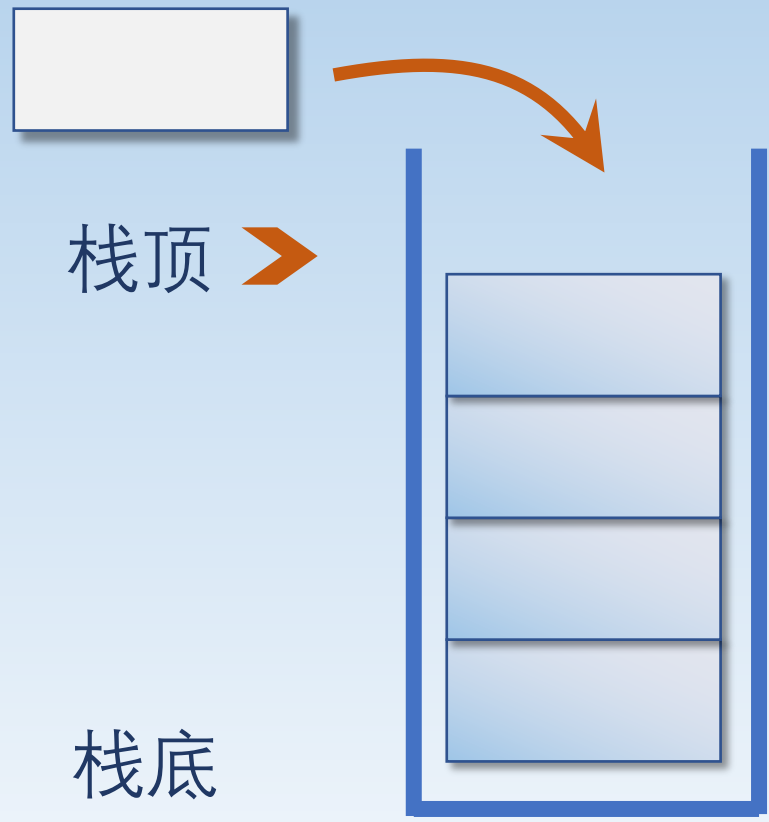


入队 enqueue



# 栈与队列 (Stack and Queue)

栈 Stack



入栈 add



栈顶 ➤

栈底

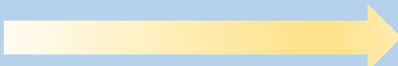




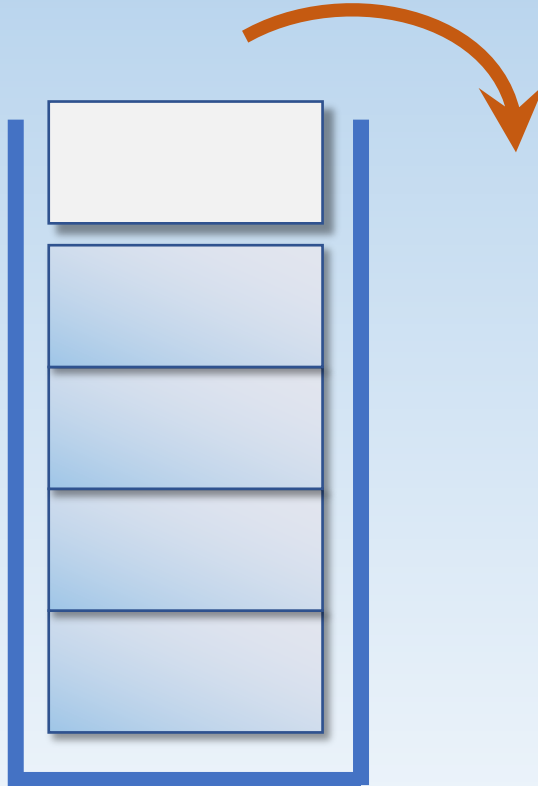
# 栈与队列 (Stack and Queue)

栈 Stack

出栈 pop



栈顶 ➤



栈底

栈顶 ➤



栈底

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

Sum = 48

L\_count

Sum = 15

1

[15, 12, 9, 6, 3, 2, 1]

[15]

[15, 12, 9, 6, 3, 2, 1]

2

[12, 9, 6, 3, 2, 1]

3

[9, 6, 3, 2, 1]

4

[6, 3, 2, 1]

5

L\_out :

[ ]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

ind

Sum = 27

1 [15, 12, 9, 6, 3, 2, 1] [15, 12] **.pop()** [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out :

[ ]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 24



1 [15, 12, 9, 6, 3, 2, 1] [15, 9] **.pop()** [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out :

[ ]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 21

OK

1 [15, 12, 9, 6, 3, 2, 1] [15, 6] **.pop()** [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out :

[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 18



1 [15, 12, 9, 6, 3, 2, 1] [15, 3] [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out :

[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 20



1	[15, 12, 9, 6, 3, 2, 1]	[15, 3, 2]	[15, 12, 9, 6, 3, 2, 1]
---	-------------------------	------------	-------------------------

2	[12, 9, 6, 3, 2, 1]
---	---------------------

3	[9, 6, 3, 2, 1]
---	-----------------

4	[6, 3, 2, 1]
---	--------------

5
---

L\_out :

[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Next



Sum = 21

OK

1 [15, 12, 9, 6, 3, 2, 1] [15, 3, 2, 1] [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out :

[15, 3, 2, 1]

[15, 6]



L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

Sum = 33

L\_count

Sum = 12

1 [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1] [12]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5



[12, 9, 6, 3, 2, 1]

L\_out :

[15, 3, 2, 1]

[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 21

1 [15, 12, 9, 6, 3, 2, 1] OK

2 [12, 9, 6, 3, 2, 1] [12, 9] .pop() [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5



L\_out :

[12, 9]  
[15, 3, 2, 1]  
[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 18

1 [15, 12, 9, 6, 3, 2, 1]



2 [12, 9, 6, 3, 2, 1] [12, 6] [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out :

[12, 6, 3]

[12, 9]

[15, 3, 2, 1]

[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 21

1 [15, 12, 9, 6, 3, 2, 1] OK

2 [12, 9, 6, 3, 2, 1] [12, 6, 3] .pop() [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out :

[12, 6, 3]

[12, 9]

[15, 3, 2, 1]

[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 20

1 [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

[12, 6, 2]

[12, 9, 6, 3, 2, 1]



L\_out :

[12, 6, 3]

[12, 9]

[15, 3, 2, 1]

[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 21

1 [15, 12, 9, 6, 3, 2, 1] OK

Next



2 [12, 9, 6, 3, 2, 1] [12, 6, 2, 1]

[12, 9, 6, 3, 2, 1]

3 [9, 6, 3, 2, 1]

4 [6, 3, 2, 1]

5

L\_out : [12, 6, 2, 1]  
[12, 6, 3]  
[12, 9]  
[15, 3, 2, 1]  
[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 21

Sum = 21

1 [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 **OK** [9, 6, 3, 2, 1] [9]

4 [6, 3, 2, 1]

5



[9, 6, 3, 2, 1]

L\_out : [12, 6, 2, 1]  
[12, 6, 3]  
[12, 9]  
[15, 3, 2, 1]  
[15, 6]

L\_sort : [15, 12, 9, 6, 3, 2, 1]

T : 21

ListCount

L\_list

L\_count

Sum = 21

Sum = 21

1 [15, 12, 9, 6, 3, 2, 1]

2 [12, 9, 6, 3, 2, 1]

3 **OK** [9, 6, 3, 2, 1] [9]

4 [6, 3, 2, 1]

5 **Finish!**



[9, 6, 3, 2, 1]

L\_out : [12, 6, 2, 1]  
[12, 6, 3]  
[12, 9]  
[15, 3, 2, 1]  
[15, 6]



## 问题升级 Problem Upgrade

0/1 背包问题：

有一个背包容量为  $T$ ，有  $n$  个大小不同，价值不同的物品要装入背包。第  $i$  件物体的体积是  $L[i]$ ，价值  $val[i]$ 。

求解：使装入背包的体积不超过容量  $T$ ，价值总和最大。

输入：  $L = [6, 3, 9, 12, 15]$

$val = [3, 5, 12, 25, 8]$

$T = 21$

## 问题升级 Problem Upgrade

0/1 背包问题：

有一个背包容量为  $T$ ，有  $n$  个大小不同，价值不同的物品要装入背包。第  $i$  件物体的体积是  $L[i]$ ，价值  $val[i]$ 。

求解：使装入背包的体积不超过容量  $T$ ，价值总和最大。

List[ max ]

$$f = \arg \max \{ \mathbf{f[i-1][v]}, f[i-1][v-w[i]]+val[i] \}$$

$$f[i][v] = \max\{ f[i-1][v], f[i-1][v-w[i]]+val[i] \}$$



# 问题升级 Problem Upgrade

## 1. 0/1背包问题

体积是 $w[i]$ ，价值是 $val[i]$

## 2. 完全背包问题

体积是 $w[i]$ ，价值是 $val[i]$ ，每种物品都有无限件可用。

## 3. 多重背包问题

物品不是无限的，第  $i$  种物品最多有  $n[i]$  件可用

## 4. 混合三种背包问题

- 有的物品只可以取一次（01背包）
- 有的物品可以取无限次（完全背包）
- 有的物品可以取的次数有一个上限（多重背包）

## 5. 二维费用背包问题

- 有两个不同容量的背包  $T1$  和  $T2$
- 对于每件物品，具有两种不同的体积  $L1[i]$   $L2[i]$

## 6. 分组背包问题

物品被划分为若干组，每组中的物品互相冲突，最多选一件

## 7. 有依赖的背包问题

物品间存在某种“依赖”的关系：

$a$  依赖于  $b$ ，若选物品  $a$ ，则必须选物品  $b$

## 8. 求背包问题的方案总数

除了求可得到的最大价值外，还得到装满背包或将背包装至某一指定容量的方案总数

## 更多思考 More...

- 时间安排
- 代码量
- 学习资料

