链滴

两数之和 【每日一道算法题】

作者：141Mr-liu
原文链接：https://ld246.com/article/1592548539083
来源网站：链滴

前言

这段时间，由于临近实习，一直都在准备实习的内容。在这段时间，也是把 Java 的一些基础的内容习了一遍，也开始了 SpringCloud 微服务的学习，所以博客这段时间也就一直没更新。最近也是想起来博客这回事，刚好现在也在学习算法，因此决定开一个每日一道算法题的分类，每天一两道算法的学习，一个是对自己学习情况的一个记录，一个是对自己的监督。

好了，话不多少，开始吧

题目

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，并回他们的数组下标。
你可以假设每种输入只会对应一个答案。但是，数组中同一个元素不能使用两遍。

示例:
给定 nums = [2, 7, 11, 15], target = 9
因为 nums[0] + nums[1] = 2 + 7 = 9
所以返回 [0, 1]

题目分析

这是在某个大厂笔试的一道算法题，题目不是很难，算法入门级别，但是通过这段算法题，的确能够得出来一个人是否学习过算法。

这道题目前我找出了两种解题方法。

解题方法

暴力破解

因为当时没有深入学习算法，所以当时本人就使用了这个方法，结果呢也是可想而知。

话不多说，看代码：

```
public int[] demo1(int[] nums, int target){
    for (int i = 0; i < nums.length; i++) {
        for (int j = i; j < nums.length; j++) {
            if (nums[i] + nums[j] == target){
                return new int[] {i,j};
            }
        }
    }
    return null;
}
```

这个暴力破解的流程是这样的：

- 遍历数组，取出 num1
- 再次遍历数组，取出 num2
- 判断 num1 + num2 是否等于 target

所以这个破解方式的缺点还是很明显的，那就是时间复杂度比较大， O(n^2)

通过 Hash 表来破解

话不多少，先看代码

```
public int[] demo2(int[] nums, int target){
    Map<Integer , Integer> maps= new HashMap<>();
    for (int i = 0; i < nums.length; i++) {
        int com = target - nums[i];
        if (maps.containsKey(com)){
            return new int[] {i,maps.get(com)};
        }
        maps.put(nums[i] , i);
    }
    return null;
}
```

流程：

- 先遍历这个数组，取出 num1
- 计算 target - num1，获得我们要获得的目标值com
- 查看 HashMap 中是否有 com这个值，如果有就返回，没有就把num1放到 HashMap 中

这个方式和上个方式的区别很明显，在暴力破解中，我们使用了两次嵌套循环，所以时间复杂度是成数上升的，而在Hash表方法中，我们只使用了一次循环，所以时间复杂度只为 n。