1005-K次取反后最大化的数组和

题述

1005. K 次取反后最大化的数组和

难度 简单 211

给你一个整数数组 `nums` 和一个整数 `k` ，按以下方法修改该数组：

- 选择某个下标 `i` 并将 `nums[i]` 替换为 `-nums[i]` 。

重复这个过程恰好 `k` 次。可以多次选择同一个下标 `i` 。

以这种方式修改数组后，返回数组 **可能的最大和** 。

示例 1：

```
输入：nums = [4,2,3], k = 1
输出：5
解释：选择下标 1 ，nums 变为 [4,-2,3] 。
```

示例 2：

```
输入：nums = [3,-1,0,2], k = 3
输出：6
解释：选择下标 (1, 2, 2) ，nums 变为 [3,1,0,2] 。
```

示例 3：

```
输入：nums = [2,-3,-1,5,-4], k = 2
输出：13
解释：选择下标 (1, 4) ，nums 变为 [2,3,-1,5,4] 。
```

思路

这道题的贪心策略比较容易想到

贪心的思路，局部最优：让绝对值大的负数变为正数，当前数值达到最大，

整体最优：整个数组和达到最大

局部最优可以推出全局最优。

那么如果将负数都转变为正数了，K依然大于0，此时的问题是一个有序正整数序列，如何转变K次正负，让 数组和 达到最大。

那么又是一个贪心：局部最优：只找数值最小的正整数进行反转，当前数值可以达到最大（例如正整数数组{5, 3, 1}，反转1 得到-1 比 反转5得到的-5 大多了），全局最优：整个 数组和 达到最大。

那么本题的解题步骤为：

- 第一步：将数组按照绝对值大小从大到小排序，**注意要按照绝对值的大小**
- 第二步：从前向后遍历，遇到负数将其变为正数，同时K--
- 第三步：如果K还大于0，那么反复转变数值最小的元素，将K用完
- 第四步：求和

题解

C++

```
class Solution {
public:
    static bool cmp(int a,int b)
    {
        return abs(a) > abs(b);
    }

    int largestSumAfterKNegations(vector<int>& nums, int K)
    {
        //贪心算法  题目要求恰好翻转k次
        //局部最优：让绝对值大的负数变为正数，当前数值达到最大
        //若所有负数都变正了 k仍不等于0 仍然需要反转
        //这时，我们只能将绝对值小的整数进行翻转
        sort(nums.begin(),nums.end(),cmp);  //按照绝对值大小进行排序
        for(int i = 0;i< nums.size();i++)
        {
            //第二步
            if(nums[i]<0 && K > 0)
            {
                nums[i] *= -1;  //正负翻转
                K--;
            }
        }

        if(K % 2 == 1)
        {
            nums[nums.size() - 1] *= -1;     //如果K还大于0，那么反复转变数值最小的元
素，将K用完
        }
        int result = 0;
        for(int num:nums)
        {
            result += num;
        }
        return result;
    }
};
```

Python

```python
class Solution:
    def largestSumAfterKNegations(self, nums: List[int], k: int) -> int:
        nums=sorted(nums, key=abs, reverse=True)  #将nums数组按照绝对值大小进行排序
        for i in range(len(nums)):
            if k > 0 and nums[i] < 0:
                nums[i] *= -1
                k -= 1
        if k > 0:
            nums[-1] *= (-1)**k #取A最后一个数只需要写-1
        return sum(nums)
```

提交结果	执行用时	内存消耗	语言	提交时间	备注
通过	40 ms	15 MB	Python3	2022/02/12 17:46	
通过	4 ms	8.9 MB	C++	2022/02/12 17:32	

思考