LCP 30.魔塔游戏

题目大意

一个长度为 n 的数组，其中有正整数也有负整数，问至少把多少个负数房间扔到数组末尾，才能使所有前缀和始终大于等于 0？

题目分析

观察数据范围，$n \leq 1e5$。考虑其中一种最简单的情况，先求出所有房间数值之和，如果结果小于0，则在加到最后一个房间时一定不满足条件，怎么交换都无力回天，输出 `-1` 。

当结果大于等于 0 时，只要把所有负数房间丢到最后，就能让玩家活下来，即保证有解。我们先尝试一种贪心思路：仅当受了当前怪物伤害会死的时候，才把当前房间的怪物丢到最后。一眼盯真，但仔细一想其实感觉不太正确，可以构造出如下反例：

`100 -100 -1 -1 -1 -1 100`

如果按照我们的贪心策略，在抗下100点伤害后，玩家不得不把后面的四个 `-1` 都丢到最后，最终结果为 4。但其实应当把前面那个影响更加致命的 `-100` 丢到最后，这样玩家可以顺利抗下 `-1 * 4` 的伤害，结果为 `1` 。

正解呼之欲出：每当玩家受到当前怪物伤害会死的时候，把此前伤害最高的怪物丢到最后，显然这样的操作是不劣于我们先前的贪心策略的，我们之前计算的和又保证了解的存在性，把最强的怪物丢到最后仅对随后的战斗有正面影响，且影响最大，因此我们的贪心算法可以求得最优解。

最后还有一些小小的细节：在最坏情况下，我们求出的和会爆 int，采用 long long 进行存储，我们用 `priority_queue<long long>` 来维护此前最强的怪物，时间复杂度为 $O(n \log n)$ 完整代码如下：

```
priority_queue<long  long> monster;
int magicTower(vector<int> &nums){
    long long sum = 1, hp = 1;
    int ret = 0;
    for (auto &ele : nums)
        sum += ele;
    if (sum <= 0)
        return -1;
    for (auto &ele : nums){
        hp += ele;
        if (ele < 0)
            monster.push(-ele);
        if (hp <= 0){
```

```
            hp += monster.top();
            monster.pop();
            ret++;
        }
    }
    return ret;
}
```