740. Delete and Earn

- Difficulty:Medium
- Total Accepted:1.2K
- Total Submissions:3.3K
- Contributor: imsure

Given an array `nums` of integers, you can perform operations on the array.

In each operation, you pick any `nums[i]` and delete it to earn `nums[i]` points. After, you must delete **every** element equal to `nums[i] - 1` or `nums[i] + 1`.

You start with 0 points. Return the maximum number of points you can earn by applying such operations.

Example 1:

```
Input: nums = [3, 4, 2]
Output: 6
Explanation: 
Delete 4 to earn 4 points, consequently 3 is also deleted.
Then, delete 2 to earn 2 points. 6 total points are earned.
```

Example 2:

```
Input: nums = [2, 2, 3, 3, 3, 4]
Output: 9
Explanation: 
Delete 3 to earn 3 points, deleting both 2's and the 4.
Then, delete 3 again to earn 3 points, and 3 again to earn 3 points.
9 total points are earned.
```

Note:

- The length of `nums` is at most `20000`.
- Each element `nums[i]` is an integer in the range `[1, 10000]`.

Seen this question in a real interview before?

This question can be reduced to the House Robbers question also on LeetCode. Please have a look at it if you haven't seen it before.

Observations:

The order of nums does not matter.
Once we decide that we want a num, we can add all the occurrences of num into the total.

We first transform the nums array into a points array that sums up the total number of points for that particular value. A value of x will be assigned to index x in points.

nums: [2, 2, 3, 3, 3, 4] (2 appears 2 times, 3 appears 3 times, 4 appears once)
points: [0, 0, 0, 4, 9, 4] <- This is the gold in each house!

The condition that we cannot pick adjacent values is similar to the House Robber question that we cannot rob adjacent houses. Simply pass points into the rob function for a quick win

```
class Solution {
public:

    int rob(vector<int> num)
    {
        int rob = 0;
        int notrob = 0;
        for(int i=0;i<num.size();++i)
        {
            int currob = notrob+num[i];
            notrob=max(notrob,rob);
            rob=currob;
        }
        return max(notrob,rob);
    }

    int deleteAndEarn(vector<int>& nums) {
        vector<int> point(10001,0);
        for(auto elem:nums)
        {
            point[elem]+=elem;
        }
        return rob(point);
    }
};
```

198. House Robber

You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have security system connected and **it will automatically contact the police if two adjacent houses were broken into on the same night**.

Given a list of non-negative integers representing the amount of money of each house, determine the maximum amount of money you can rob tonight **without alerting the police**.

Credits:
Special thanks to @ifanchu for adding this problem and creating all test cases. Also thanks to @ts for adding additional test cases.

- Difficulty:Easy
- Total Accepted:168.1K
- Total Submissions:425.7K
- Contributor: LeetCode

```
public int rob(int[] num) {
    int rob = 0; //max monney can get if rob current house
    int notrob = 0; //max money can get if not rob current house
    for(int i=0; i<num.length; i++) {
        int currob = notrob + num[i]; //if rob current value, previous house
must not be robbed
        notrob = Math.max(notrob, rob); //if not rob ith house, take the max
value of robbed (i-1)th house and not rob (i-1)th house
        rob = currob;
    }
    return Math.max(rob, notrob);
}
```