什么是前缀和

对给定的一个数组，额外的开辟一个前缀和数组进行预处理。

```
std::vector<int> prenums(nums.size()+1, 0);
for(int i = 0;i<nums.size();i++) {
  prenums[i+1] = prenums[i] + nums[i];
}
```

对前缀和数组`prenums`:

1. `prenums[i]`表示`nums[0..i-1]`的和；
2. 对`nums[i...j]`的和，只需要计算`prenums[j+1] - prenums[i]`即可。

求给定数组中连续子数组和为`k`的个数

利用前缀和，

```
class Solution {
public:
  int subarraySum(std::vector<int>& nums, int k) {
    int              len = nums.size();
    std::vector<int> prenums(len + 1, 0);

    // 计算前缀和
    for (int i = 0; i < len; i++) {
      prenums[i + 1] = prenums[i] + nums[i];
    }

    // 遍历求解
    int ans = 0;
    for (int i = 0; i <= len; i++) {
      for (int j = 0; j < i; j++) {
        if (k == prenums[i] - prenums[j]) {
          ans++;
        }
      }
    }

    return ans;
  }
};
```

双重循环，导致超时。

优化方案

```cpp
class Solution {
public:
  int subarraySum(std::vector<int>& nums, int k) {
    int                 len = nums.size();
    std::map<int, int> prenums;
    prenums[0] = 1;
    int ans    = 0;
    int sum_i  = 0;
    int sum_j  = 0;
    for (int i = 0; i < len; i++) {
      sum_i += nums[i];
      sum_j = sum_i - k;
      if (prenums.count(sum_j)) {
        ans += prenums[sum_j];
      }
      prenums[sum_i]++;
    }

    return ans;
  }
};
```