# 题目

给定一个整数数组，找出总和最大的连续数列，并返回总和。

**示例：**

输入：[-2,1,-3,4,-1,2,1,-5,4]

输出：6

解释：连续子数组[4,-1,2,1]的和最大，为6。

**进阶：**

如果你已经实现复杂度为O(n) 的解法，尝试使用更为精妙的分治法求解。

# 分析

## 方法一：动态规划

**思路：**

状态：dp[i]表示以i结尾的最大连续子序列

状态转移：

对于当前的nums[i]

如果nums[i-1] >= 0 则 dp[i] = dp[i-1] + nums[i];

否则dp[i] = nums[i];

其实我们可以把nums当做dp数组，直接在原数组上面操作，这样可以省掉O(n)的空间

**代码：**

// 动态规划

int maxSubArray(vector<int>& nums) {

if(nums.size() == 0) return INT\_MIN;

int maxSum = nums[0];

for(int i = 1; i < nums.size(); i++)

{

if(nums[i-1] >= 0)

nums[i] += nums[i-1];

maxSum = max(maxSum, nums[i]);

}

return maxSum;

}