一、题目说明

题目是53. Maximum Subarray，求最长连续子序列最大和。难度是Easy!

二、我的解答

Easy的题目，居然没做出来。

后来看了用dp方法，其中dp[i]表示以第i个元素结尾的最大和。

`dp[i] = nums[i] > nums[i]+dp[i-1] ? nums[i] : nums[i]+dp[i-1];`

然后求出最大的dp即可。知道思路，实现非常简单，问题是没有往动态规划上面去想。

```
#include<iostream>
#include<vector>

using namespace std;
class Solution{
public:
    int maxSubArray(vector<int>& nums) {
        int len=nums.size();
        vector<int> dp;
        dp.resize(len);
        //以第i个元素结尾的最大和
        dp[0] = nums[0];
        int max = dp[0];
        for(int i=1;i<len;i++){
            dp[i] = nums[i] > nums[i]+dp[i-1] ? nums[i] : nums[i]+dp[i-1];
            max = max > dp[i] ? max : dp[i];
        }

        return max;
    }
};
int main(){
    Solution s;

    vector<int> m;
    m = {-2,1,-3,4,-1,2,1,-5,4};
    cout<<(6==s.maxSubArray(m))<<"\n";

    m = {-2,1,-3};
    cout<<(1==s.maxSubArray(m))<<"\n";

    m = {1};
    cout<<(1==s.maxSubArray(m))<<"\n";

    m = {-2,-1};
    cout<<(-1==s.maxSubArray(m))<<"\n";
    return 0;
}
```

性能居然还不错，空间复杂的可以优化，dp复用nums，还是算了，可读性不好：

```
Runtime: 4 ms, faster than 98.55% of C++ online submissions for Maximum
Subarray.
Memory Usage: 9.4 MB, less than 17.65% of C++ online submissions for Maximum
Subarray.
```

三、优化

题目说让用分治算法，分而治之。我想想，应该是类似“二分查找”。不会，看了大神的实现：

```
class Solution{
public:
    //divide & conquer approach
    int maxSubArray(vector<int>& nums) {
        return maxSubArrayPart(nums,0,nums.size()-1);
    }
private:
    int maxSubArrayPart(vector<int>& nums,int left,int right){
        if(left==right){
            return nums[left];
        }
        int mid = (left+right) / 2;
        return max(maxSubArrayPart(nums,left,mid),

max(maxSubArrayPart(nums,mid+1,right),maxSubArrayAll(nums,left,mid,right)));
    }

    //左右两边求和
    int maxSubArrayAll(vector<int>& nums,int left,int mid,int right){
        int leftSum = INT_MIN;
        int sum=0;
        for(int i=mid;i>=left;i--){
            sum += nums[i];
            if(sum>leftSum) leftSum=sum;
        }

        sum=0;
        int rightSum= INT_MIN;
        for(int i=mid+1;i<=right;i++){
            sum += nums[i];
            if(sum>rightSum) rightSum=sum;
        }

        return leftSum+rightSum;
    }
};
```

性能：

```
Runtime: 8 ms, faster than 74.25% of C++ online submissions for Maximum
Subarray.
Memory Usage: 9.4 MB, less than 33.33% of C++ online submissions for Maximum
Subarray.
```