动态规划

题目描述

给定一个数组 array[1, 4, -5, 9, 8, 3, -6]，在这个数字中有多个子数组，子数组和最大的应该是：[9, 8, 3]，输出 20，再比如数组为[1, -2, 3, 10, -4, 7, 2, -5]，和最大的子数组为[3, 10, -4, 7, 2]，输出 18。

思路分析

1、状态方程 ： max(dp[i]) = getMax(max(dp[i -1]) + arr[i] ,arr[i])

2、上面式子的意义是：我们从头开始遍历数组，遍历到数组元素 arr[i] 时，连续的最大的和 可能为 max(dp[i -1]) + arr[i] ， 也可能为 arr[i] ， 做比较即可得出哪个更大，取最大值。时间复杂度为 n。

代码实现

```
#include<iostream>
using namespace std;
int GetMax(int a, int b) { //得到两个数的最大值
    return a >b ? a : b;
}

int GetMaxAddOfArray(int* arr, int sz) {
    if (arr == NULL || sz <= 0)
        return 0;

    int Sum = arr[0];   //临时最大值
    int MAX = arr[0];   //比较之后的最大值

    for (int i = 1; i < sz; i++) {
        Sum = GetMax(Sum + arr[i], arr[i]);   //状态方程
        if (Sum >= MAX)
            MAX = Sum;
    }
    return MAX;
}

int main() {
    int array[] = { 2, 3, -6, 4, 6, 2, -2, 5, -9 };
    int sz = sizeof(array) / sizeof(array[0]);
    int MAX = GetMaxAddOfArray(array, sz);
    cout << MAX << endl;
    return 0;
}
```