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**

1. #include<iostream>
2. **using** **namespace** std;
3. **int** GetMax(**int** a, **int** b) { //得到两个数的最大值
4. **return** a >b ? a : b;
5. }
7. **int** GetMaxAddOfArray(**int**\* arr, **int** sz) {
8. **if** (arr == NULL || sz <= 0)
9. **return** 0;
11. **int** Sum = arr[0];   //临时最大值
12. **int** MAX = arr[0];   //比较之后的最大值
14. **for** (**int** i = 1; i < sz; i++) {
15. Sum = GetMax(Sum + arr[i], arr[i]);   //状态方程
16. **if** (Sum >= MAX)
17. MAX = Sum;
18. }
19. **return** MAX;
20. }
22. **int** main() {
23. **int** array[] = { 2, 3, -6, 4, 6, 2, -2, 5, -9 };
24. **int** sz = **sizeof**(array) / **sizeof**(array[0]);
25. **int** MAX = GetMaxAddOfArray(array, sz);
26. cout << MAX << endl;
27. **return** 0;
28. }