动态规划法解决最大连续子序列和

问题描述 ：

数组 INT A[] = {-4 , 3 ,56 , -15 , 34 , 0 , -14 , 4} ; 某几个连续的子序列其和最大，比如A0+A1 = -1 。A1+A2+A3+A4 = 78 。则A1 A2 A3A4组成的数组即是所求。

解决方法：

此题尝试使用动态规划的方法进行解决，首先建立状态方程。

设B[J]表示第J处，以A[J] 结尾的子序列的最大和。

则B[J] = MAX(A[J] + B[J-1] , A[J]) ，而我们的所求的答案，就是从1- N对B数组求最大值。

代码如下：

/\*

最大连续字段和 时间复杂度为O(N)

定义b[j]为数组中包含a[j]的最大连续子序列和

注意一个误区，b[j] 并不是1-j中最大的连续子序列的和，只是包含a[j]的最大子序列的和

而我们所要求的是求出b[j]中最大的值，即为所求

状态方程为: b[j] = max(b[j-1] + a[j] , a[j])

\*/

#include <iostream>

using namespace std ;

const int N = 8 ;

int a[] = {-4 , 3 ,56 , -15 , 34 , 0 , -14 , 4} ;

int b[N] ; //b[i]表示包含a[i]的最大连续子序列之和

inline int max(int a , int b)

{

return (a > b) ? a : b ;

}

int main()

{

b[0] = a[0] ;

int i , mam = b[0];

for(i = 1; i < N ; i++)

{

b[i] = max(a[i] , b[i-1] + a[i]);

if(mam < b[i])

mam = b[i] ;

}

printf("max %d\n" , mam) ;

for(i = 0; i < N ; i++)

printf("%d\n" , b[i]) ;

system("pause") ;

return 0 ;

}