已知整数$A_1$, $A_2$, ..., $A_N$, 求max $\sum_{k=i}^{j} A_k$.

注意：如果这些数都是负数，则最大和值认为是0.

您能够想到那些求解算法？能不能用分治递归解决？
最好的时间复杂度是多少？

| $A_1$ | $A_2$ | | | | | | | | $A_n$ |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |

# 已知整数$A_1, A_2, \ldots, A_N$, 求max $\sum_{k=i}^{j} A_k$.

**Algorithm 1**

```
int  MaxSubsequenceSum ( const int A[ ],  int  N )
{
            int  ThisSum,  MaxSum,  i,  j,  k;
/* 1*/      MaxSum = 0;   /* initialize the maximum sum */
/* 2*/      for( i = 0; i < N; i++ )  /* start from A[ i ] */
/* 3*/          for( j = i; j < N; j++ ) {   /* end at A[ j ] */
/* 4*/                  ThisSum = 0;
/* 5*/                  for( k = i; k <= j; k++ )
/* 6*/                      ThisSum += A[ k ];  /* sum from A[ i ] to A[ j ] */
/* 7*/                  if ( ThisSum > MaxSum )
/* 8*/                      MaxSum = ThisSum;  /* update max sum */
              }  /* end for-j and for-i */
/* 9*/      return  MaxSum;
}
```

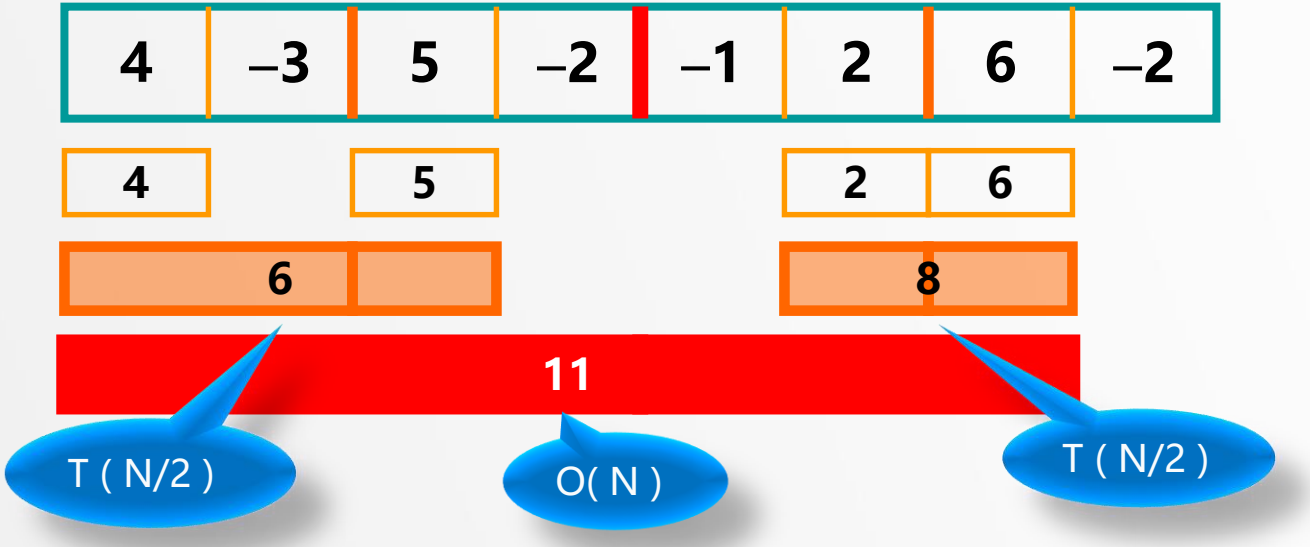
$$T(N) = O(N^3)$$

## Algorithm 2

```
int  MaxSubsequenceSum ( const int A[ ],  int  N )
{
            int  ThisSum,  MaxSum,  i,  j;
/* 1*/      MaxSum = 0;   /* initialize the maximum sum */
/* 2*/      for( i = 0; i < N; i++ )  {   /* start from A[ i ] */
/* 3*/          ThisSum = 0;
/* 4*/          for( j = i; j < N; j++ ) {   /* end at A[ j ] */
/* 5*/                  ThisSum += A[ j ];  /* sum from A[ i ] to A[ j ] */
/* 6*/                  if ( ThisSum > MaxSum )
/* 7*/                      MaxSum = ThisSum;  /* update max sum */
              }  /* end for-j */
            }  /* end for-i */
/* 8*/      return  MaxSum;
}
```

$$T(N) = O(N^2)$$

**Algorithm 3** Divide and Conquer




$$T(N) = 2\,T(N/2) + c\,N, \qquad T(1) = O(1)$$

$$= 2\,[2\,T(N/2^2) + c\,N/2] + c\,N$$

$$= 2^k\,O(1) + c\,k\,N \qquad \text{where } N/2^k = 1$$

$$= O(N \log N)$$

Also true for N ≠ 2k

| 4 | –3 | 5 | –2 | –1 | 2 | 6 | –2 |
|---|---|---|---|---|---|---|---|

| 4 | –3 | -5 | –2 | 1 | 5 | -6 | 2 |
|---|---|---|---|---|---|---|---|

## Algorithm 4 On-line Algorithm

```
int MaxSubsequenceSum( const int  A[ ],  int  N )
{
            int  ThisSum, MaxSum, j;
/* 1*/      ThisSum = MaxSum = 0;
/* 2*/      for ( j = 0; j < N; j++ ) {
/* 3*/           ThisSum += A[ j ];
/* 4*/           if  ( ThisSum > MaxSum )
/* 5*/                MaxSum = ThisSum;
/* 6*/           else if ( ThisSum < 0 )
/* 7*/                ThisSum = 0;
            }  /* end for-j */
/* 8*/      return MaxSum;
}
```

$T(N) = O(N)$

**A[ ] is scanned once only.**

# 4个算法的实际执行时间比较(单位：秒)

| 算法 | | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Time | | $O(N^3)$ | $O(N^2)$ | $O(N\log N)$ | $O(N)$ |
| 问题规模N | $N=10$ | 0.00103 | 0.00045 | 0.00066 | 0.00034 |
| | $N=100$ | 0.47015 | 0.01112 | 0.00486 | 0.00063 |
| | $N=1,000$ | 448.77 | 1.1233 | 0.05843 | 0.00333 |
| | $N=10,000$ | NA | 111.13 | 0.68631 | 0.03042 |
| | $N=100,000$ | NA | NA | 8.0113 | 0.29832 |

注意: 读入数据的时间没有包括在内