Problem：給定一 array 為 $A[1,\dots,n]$

求此一 array 中具最大整數和之 subarray 的值

brute force：枚舉所有 subarray，並計算其和

① 利用 i,j 指標來得 subarray：$C^n_2 = O(n^2)$

② 計算 subarray 值：$O(n)$

$\Rightarrow O(n^3) \Rightarrow$ upperbound 在 $O(n^3)$

brute force 2：先做一些 preprocessing

① Let $B[1,\dots,n]$ be a new array

② $B[i] = \sum_{j=1}^{i} A[j] \Rightarrow O(n)$

之後，可簡化計算 subarray 值為：$A[i,\dots,j] = B[j] - B[i-1] \Rightarrow O(1)$

$\therefore T(n) = O(n) + O(n^2) = O(n^2)$

divide & conquer：

1. divide：切割 $A[1,\dots,n]$ 為 L 和 R 兩个 subarray
2. conquer：遞迴得到 L, R 之 max subarray 值 V_L, V_R
3. combine：從 L 之最右元素和 R 之最左元素開始找橫跨 LR 之 max subarray，令為 V_{mid}

   最終 return $\max\{V_L, V_R, V_{mid}\}$

$\Rightarrow T(n) = T(\frac{n}{2}) + O(n) \Rightarrow T(n) = O(n \lg n)$

Dynamic Programming：

① characterize subproblem：

令 r_i 為 $A[1,\dots,i]$ 中包含 $A[i]$ 之最大 subarray 和

求 $A[1,\dots,n]$ 之 max subarray 和為 $\max\{r_1, r_2, \dots, r_n\}$

令 r_i 对應之 optimal solution 為 OPT

(1). OPT 包含 $A[i-1]$：$r_i = r_{i-1} + A[i]$

(2). OPT 不包含 $A[i-1]$：$r_i = A[i]$

② derive recursive function：

$$r_i = \begin{cases} A[1] & \text{if } i=1 \\ \max\{r_{i-1} + A[i], A[i]\} \end{cases}$$

$\Rightarrow T(n) = O(n) \cdot O(1) = O(n)$