

- Buscar un elemento en un arreglo arbitrario
- Ordenar un arreglo ordenado
 - enfoque 1
 - enfoque 2
- Obtener el subarreglo de suma máxima

Problema de buscar un elemento en un arreglo

Entrada: Un arreglo $A[0..N]$ de números enteros y un número entero v .

Salida: $\exists p \in [0..N] . A[p] = v$

Ponienteamiento Función Objetivo

Sea $0 \leq l \leq r \leq N$:

$\phi(l, r)$: Determina si v está en el arreglo $A[l..r]$

$\phi(l, r)$

Reformulación Especificación

Salida: $\phi(0, N)$

Planteamiento Recurrente

$$\emptyset(l, r) : \begin{cases} A[l] == v & \text{Si } r = l + 1 \\ A[l] == v \text{ or } \emptyset(l+1, r) & \text{Si } r > l + 1 \end{cases}$$

Problema de buscar un elemento en un arreglo ordenado

Entrada: Un arreglo $A[0..N]$ de números enteros ordenado ascendente y un número entero v .

Salida: $\exists p \in [0..N]. A[p] = v$

Planteamiento Función Objetivo

Sea $0 \leq l \leq r \leq N$:

$\emptyset(l, r)$: Determina si v está en el arreglo $A[l..r]$

Reformulación Especificación

Salida: $\emptyset(0, N)$

Planteamiento Recurrente

$$\emptyset(l, r) : \begin{cases} A[l] == v & \text{Si } r = l + 1 \\ \emptyset(l, mid) & \text{Si } r \neq l + 1 \wedge v < A[mid] \quad mid = l + (r-1)/2 \\ \emptyset(mid, r) & \text{Si } r \neq l + 1 \wedge v \geq A[mid] \end{cases}$$

Problema de ordenar un arreglo

Entrada: Un arreglo $A[0..N]$ de números

Salida: $A[0..N]$ ordenado ascendente

Planteamiento Función Objetivo

Sea $0 \leq l \leq N$:

$\emptyset(l)$: ordena el arreglo $A[l..N]$ ascendente

Reformulación Especificación

salida: $\emptyset(0)$

Planteamiento Recurrente (Versión 1)

$$\emptyset(l) : \left\{ \begin{array}{ll} \text{skip} & \text{if } l = N \\ A[], A[p] = A[p], A[l] & \text{if } l \neq N \\ \emptyset(l+1) & \\ \end{array} \right.$$

$p = \text{obtenerPosicionMenor}(l, N)$

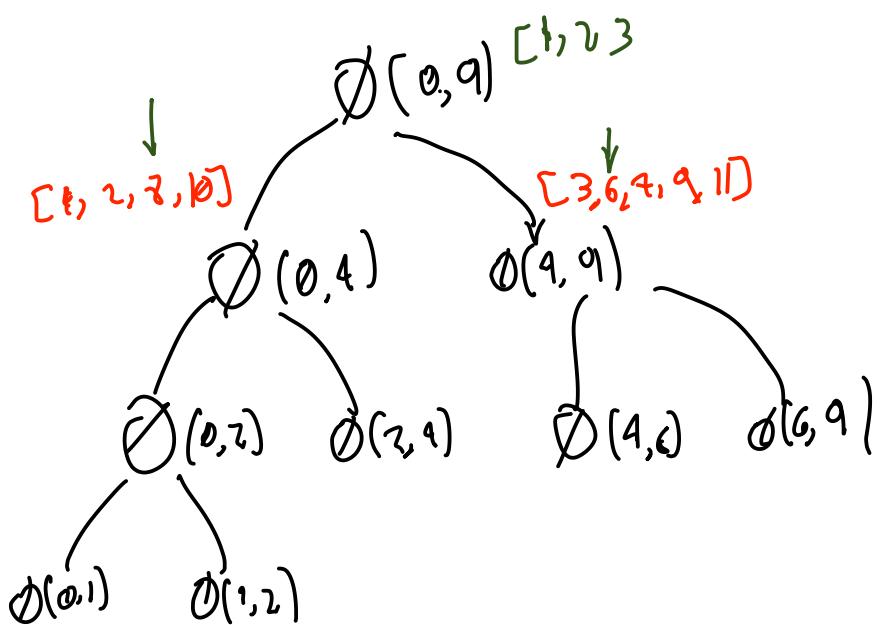
$T(N) \in O(N^2)$

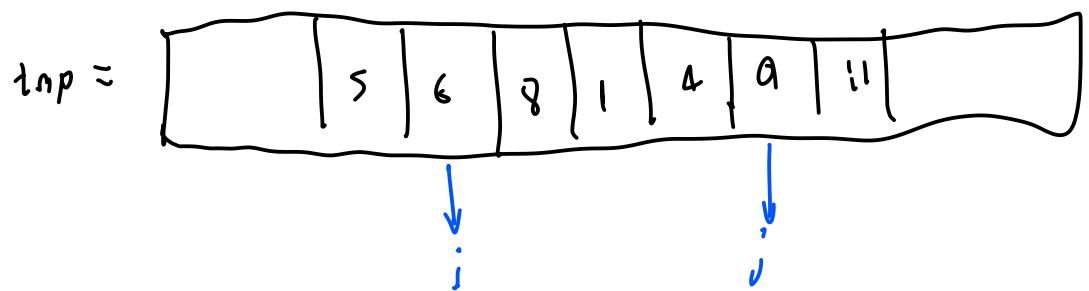
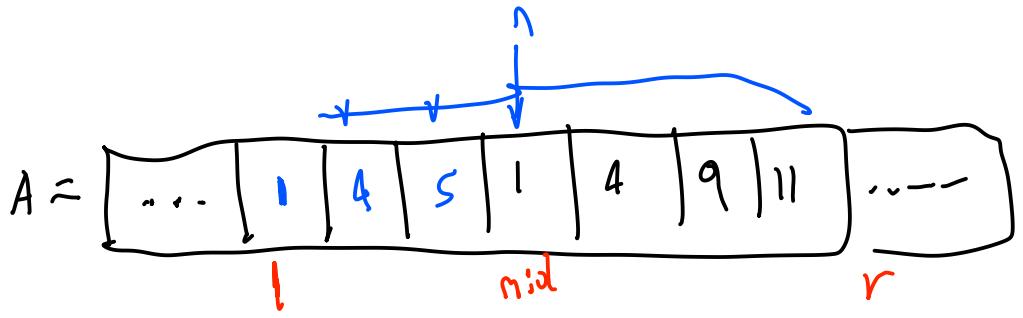
$$A = [x, 2, 3, \underbrace{8, 5, 9, 0}]$$



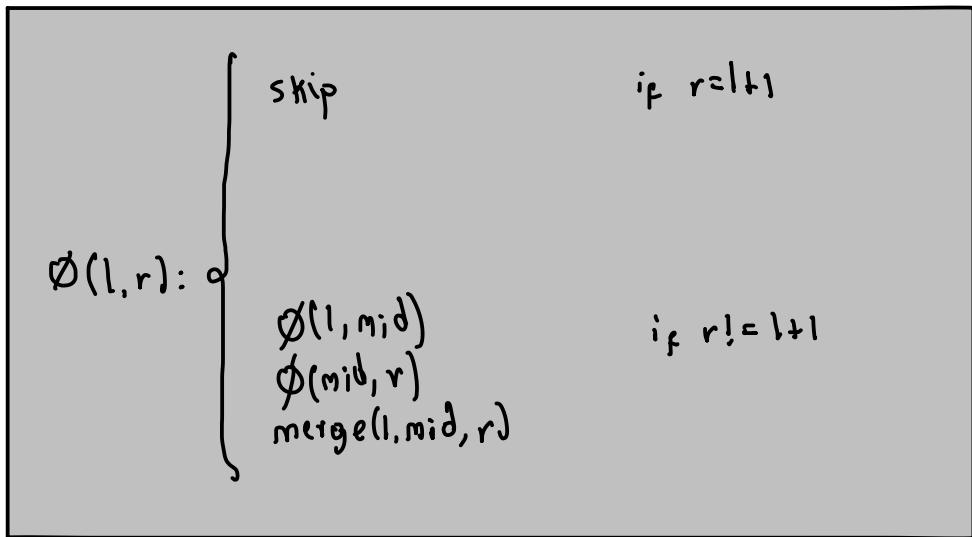
...

$$A = [10, 2, 1, 8, 11, 7, 3, 6, 9] \quad N = 9$$





Planteamiento Recurrente (Versión 2)



Problema del subarreglo de suma máxima

Entrada: Un arreglo $A[0..N]$ de números enteros

Salida: $(\text{start}, \text{end}, s) \mid s = \text{sumaElementos}(A, \text{start}, \text{end})$ y s es máximo

$$A = [10, 8, 4, 3, -4, 2] \quad (0, 3, 25)$$

$$A = [1, 4, 6, 7, -5, 4, 3, -8, 2, 5, -3, 2] \quad (0, 6, 21)$$

Planteamiento Función Objetivo

Sca $0 \leq l \leq r \leq N$:

$\emptyset(l, r)$: Determina la tripleta (st, en, sum) tal que $sum = \text{sumaElementos}(A, st, en)$ y sum es máximo en $A[l..r]$.

Reformulación Especificación

salida: $\emptyset(0, N)$

Planteamiento Recurrente

$$\emptyset(l, r) : \left\{ \begin{array}{ll} (l, r, A[l]) & \text{if } r = l + 1 \\ (st_1, e_1, sum_1) & \text{if } r = l + 1 \wedge sum_1 > sum_2 \wedge sum_1 \geq sum_3 \\ (st_2, e_2, sum_2) & \text{if } r = l + 1 \wedge sum_2 > sum_1 \wedge sum_2 \geq sum_3 \\ (st_3, e_3, sum_3) & \text{if } r = l + 1 \wedge sum_3 > sum_1 \wedge sum_3 \geq sum_2 \end{array} \right.$$

$(st_1, e_1, sum_1) = \emptyset(l, mid)$
 $(st_2, e_2, sum_2) = \emptyset(mid, r)$
 $(st_3, e_3, sum_3) = \text{obtenerSumaPasaMitad}(l, mid, r)$