طراحی الگوریتم

(تقسیم و غلبه تصادفی)

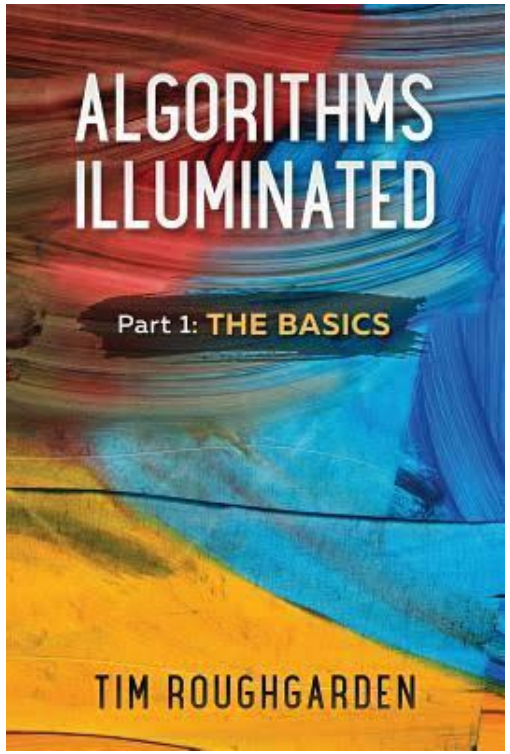
دانشکده مهندسی برق و کامپیوتر، دانشگاه صنعتی اصفهان

بهار ۹۹

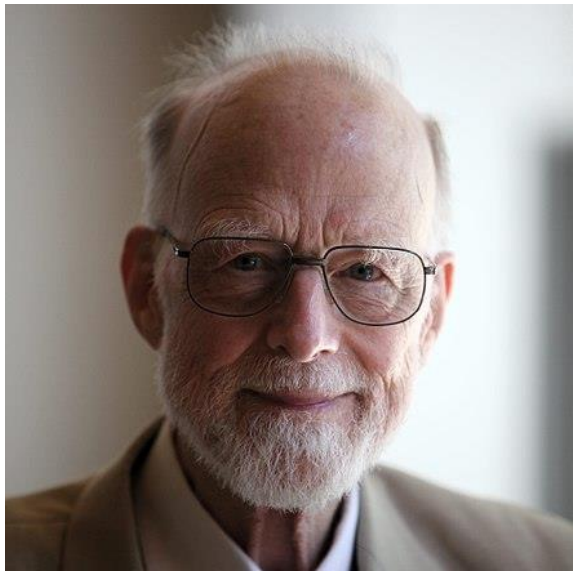
مرتب‌سازی سریع

ورودی: یک دنباله از اعداد متمایز با یک ترتیب دلخواه

هدف: مرتب‌سازی دنباله از کوچک به بزرگ



فصل پنجم، صفحه ۱۱۷

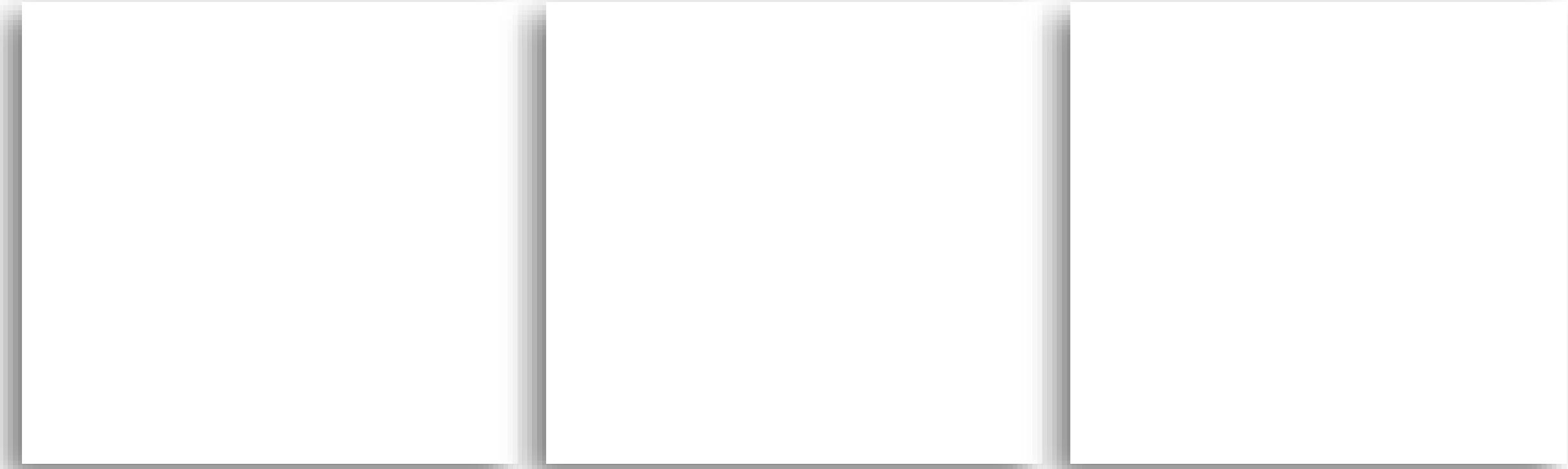


Tony Hoare, 1959

مرتب‌سازی سریع

ورودی: یک دنباله از اعداد متمایز با یک ترتیب دلخواه

هدف: مرتب‌سازی دنباله از کوچک به بزرگ



جستجوی سریع (توصیف سطح بالا)

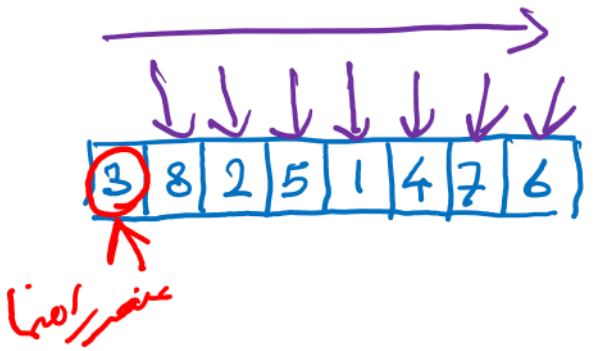
QuickSort (High-Level Description)

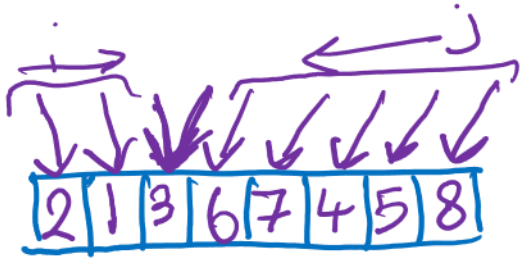
Input: array A of n distinct integers.
Postcondition: elements of A are sorted from smallest to largest.

```
if n ≤ 1 then                    // base case-already sorted
    return
choose a pivot element p         // to-be-implemented
partition A around p             // to-be-implemented
recursively sort first part of A
recursively sort second part of A
```

تقسیم به زیرمساله‌های کوچکتر برای جستجوی سریع

تقسیم به زیرمساله‌های کوچکتر برای جستجوی سریع





اگر حافظه اضافه از اندازه $\Theta(n)$

تخصیص دهیم.

تقسیم به زیرمساله‌های کوچکتر برای جستجوی سریع

۳	۸	۲	۵	۱	۴	۷	۶

۳	۸	۲	۵	۱	۴	۷	۶

۳	۲	۸	۵	۱	۴	۷	۶

۳	۲	۸	۵	۱	۴	۷	۶

۳	۲	۱	۵	۸	۴	۷	۶

۳	۲	۱	۵	۸	۴	۷	۶

۳	۲	۱	۵	۸	۴	۷	۶

۳	۲	۱	۵	۸	۴	۷	۶

تقسیم به زیرمساله‌های کوچکتر برای جستجوی سریع

Partition

Input: array A of n distinct integers, left and right endpoints $\ell, r \in \{1, 2, \ldots, n\}$ with $l \leq r$.
Postcondition: elements of the subarray $A[\ell], A[\ell+1], \ldots, A[r]$ are partitioned around $A[\ell]$.
Output: final position of pivot element.

```
p := A[ℓ]
i := ℓ + 1
for j := ℓ + 1 to r do
    if A[j] < p then          // if A[j] > p do nothing
        swap A[j] and A[i]
        i := i + 1                  // restores invariant
swap A[ℓ] and A[i − 1]      // place pivot correctly
return i − 1              // report final pivot position
```

جستجوی سریع

QuickSort

Input: array A of n distinct integers, left and right endpoints $\ell, r \in \{1, 2, \ldots, n\}$.

Postcondition: elements of the subarray $A[\ell], A[\ell+1], \ldots, A[r]$ are sorted from smallest to largest.

```
if ℓ ≥ r then                    // 0- or 1-element subarray
    return
i := ChoosePivot(A, ℓ, r)        // to-be-implemented
swap A[ℓ] and A[i]               // make pivot first
j := Partition(A, ℓ, r)          // j =new pivot position
QuickSort(A, ℓ, j − 1)           // recurse on first part
QuickSort(A, j + 1, r)           // recurse on second part
```


تاثیرانتخاب عضو
راهنما در زمان اجرا
چگونه است؟