# QuickSort Randomizat

1. Alegerea pivotului se face aleator;
2. Interschimbam ultimul element din subvector (care era pe pozitia pivotului Intr-un caz nerandomizat) cu elementul aflat pe pozitia generata aleator.

```
RANDOMIZED-PARTITION(A, p, r)
  i = RANDOM(p, r)
  exchange A[r] with A[i]
  return PARTITION(A, p, r)
```

```
RANDOMIZED-QUICKSORT(A, p, r)
  if p < r
      q = RANDOMIZED-PARTITION(A, p, r)
      RANDOMIZED-QUICKSORT(A, p, q − 1)
      RANDOMIZED-QUICKSORT(A, q + 1, r)
```

**Cormen – II.7.3 – II.7.4**

# QuickSort Randomizat

Observatii

1. Alegerea aleatoare asigura **aceeasi probabilitate** pentru un element sa fie pivot.
2. Se asteapta ca impartirea pe subvectori sa fie rezonabila datorita alegerii aleatoare.

## Analiza complexitatii

**Cazul cel mai defavorabil**

**(Analiza aplicabila atat cazului nerandomizat cat si cazului randomizat)**

Split – cazul cel mai defavorabil: Θ (n^2).
Notam T(n) – timpul de executie pe cazul cel mai defavorabil.

Relatia de recurenta: $$T(n) = \max_{0 \le q \le n-1} (T(q) + T(n-q-1)) + \Theta(n)$$

Exista o constanta c, pentru care avem relatia: $T(n) \le cn^2$

Substituind in relatia de recurenta, obtinem:

Cormen – II.7.3 – II.7.4

# QuickSort Randomizat

**Cazul cel mai defavorabil**

**(Analiza aplicabila atat cazului nerandomizat cat si cazului randomizat)**

Substituind in relatia de recurenta, obtinem:

$$\begin{aligned} T(n) &\le \max_{0\le q\le n-1} (cq^2 + c(n-q-1)^2) + \Theta(n) \\ &= c \cdot \max_{0\le q\le n-1} (q^2 + (n-q-1)^2) + \Theta(n) \,. \end{aligned}$$

Maximul expresiei $q^2 + (n-q-1)^2$ **se atinge pentru capetele intervalului [0, n -1].**

(Derivata a doua in raport cu variabila q este pozitiva pe [0,1]).

$$\max_{0\le q\le n-1}(q^2 + (n-q-1)^2) \le (n-1)^2 = n^2 - 2n + 1$$

Rezulta:

$$\begin{aligned} T(n) &\le cn^2 - c(2n-1) + \Theta(n) \\ &\le cn^2 \,, \end{aligned}$$

Se considera constanta c suficient de mare pentru ca c(2n-1) sa domine Θ (n).

Concluzie: $\boxed{T(n) = O(n^2)}$

**Cormen – II.7.3 – II.7.4**

# QuickSort Randomizat

## Analiza complexitatii (cazul Randomizat)

Algoritmul de QUICKSORT, respectiv RANDOMIZED-QUICKSORT difera doar prin modul de alegere al pivotului, deci analiza se poate focaliza in special pe procedura de partitie aleatoare.

De fiecare data cand procedura de partitionare este apelata, se returneaza un pivot, iar acest element **nu mai apare** in apelurile recursive de QUICKSORT

→ maxim n apeluri ale procedurii **PARTITION**

Apelul PARTITION → O(1) + timp proportional cu numarul de iteratii din instructiunea repetitiva **for .** Fiecare iteratie compara pivotul cu un alt element din vector.

**Cormen – II.7.3 – II.7.4**

# QuickSort Randomizat

**Lema**

Fie X numarul de comparatii efectuate in linia 4 a procedurii de partitionare in cadrul executiei intregului algoritm de QuickSort asupra unui vector de n elemente. Atunci, timpul de executie a algoritmului de QUICKSORT este O(n+X).

**Demonstratie**

- maxim n apeluri ale procedurii **PARTITION**
- nu vom numara efectiv comparatiile la fiecare apel recursiv

**Vrem sa vedem cand 2 elemente ale vectorului se compara intre ele si cand nu.**

```
PARTITION(A, p, r)
1  x = A[r]
2  i = p - 1
3  for j = p to r - 1
4      if A[j] ≤ x
5          i = i + 1
6          exchange A[i] with A[j]
7  exchange A[i + 1] with A[r]
8  return i + 1
```

**Cormen – II.7.3 – II.7.4**

# QuickSort Randomizat

Renumerotam, fara a restrange generalitatea, elementele vectorului A:

$$z_1, z_2, \ldots, z_n$$

Notam: $Z_{ij} = \{z_i, z_{i+1}, \ldots, z_j\}$

Doua elemente se compara cel mult o data. – doar daca fac parte din aceeasi partitie si una dintre ele este pivot.

Fie $X_{ij} = \mathrm{I}\{z_i \text{ is compared to } z_j\}$

Numarul total de comparatii realizat de algoritm:

$$X = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}$$

Aplicam linearitatea mediei unei variabile aleatoare:

$$\mathrm{E}[X] = \mathrm{E}\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right]$$

**Cormen – II.7.3 – II.7.4**

# QuickSort Randomizat

Aplicam linearitatea mediei unei variabile aleatoare:

$$\begin{aligned} \mathrm{E}[X] &= \mathrm{E}\left[\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}\right] \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mathrm{E}[X_{ij}] \\ &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \Pr\{z_i \text{ is compared to } z_j\} \end{aligned}$$

Calculam $\Pr\{z_i \text{ is compared to } z_j\}$

Daca alegem pivotul x astfel incat $z_i < x < z_j$, cele 2 elemente nu se vor compara niciodata

Ele se compara **daca si numai daca** primul element ales drept pivot in Zij este fie zi fie zj.

Probabilitatea ca unul dintre ele sa fie ales pivot este de **1/(j-i+1)** – toate elementele au prioritate egala

**Cormen – II.7.3 – II.7.4**

# QuickSort Randomizat

$$\begin{aligned}
\Pr\{z_i \text{ is compared to } z_j\} &= \Pr\{z_i \text{ or } z_j \text{ is first pivot chosen from } Z_{ij}\} \\
&= \Pr\{z_i \text{ is first pivot chosen from } Z_{ij}\} \\
&\quad + \Pr\{z_j \text{ is first pivot chosen from } Z_{ij}\} \\
&= \frac{1}{j-i+1} + \frac{1}{j-i+1} \\
&= \frac{2}{j-i+1} .
\end{aligned}$$

Inlocuind, obtinem:

$$\mathrm{E}[X] = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$$

**Cormen – II.7.3 – II.7.4**

# QuickSort Randomizat

Efectuand o schimbare de variabila k = j – I si marginind seria armonica, obtinem:

$$\begin{aligned}
\mathrm{E}[X] &= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1} \\
&= \sum_{i=1}^{n-1} \sum_{k=1}^{n-i} \frac{2}{k+1} \\
&< \sum_{i=1}^{n-1} \sum_{k=1}^{n} \frac{2}{k} \\
&= \sum_{i=1}^{n-1} O(\lg n) \\
&= O(n \lg n) .
\end{aligned}$$

**Cormen – II.7.3 – II.7.4**