# Algorithmique et Programmation

## TD n° 2 : Sorting (English version)

École normale supérieure – Département d'informatique

2016-2017

**Exercise 1.** $k$TH ELEMENT OF AN ARRAY

We will use a variant of quicksort to determine the $k$th (smallest) element of a set of $n$ elements. We assume that the $n$ keys are distinct.

1. Describe a procedure SELECT using a pivot (as in the quicksort algorithm) which determines the $k$th element of an array.
2. Show that with no hypothesis on the pivot choice, the function SELECT may need $O(n^2)$ comparisons.
3. Show that if the pivot is chosen so that the size of the subarray of the recursive call do not exceed $\alpha n$, where $\alpha < 1$, number of comparisons SELECT makes is $O(n)$.

Consider the following selection algorithm :

- Split le array into $n/3$ blocs $\{B_1, \ldots, B_{n/3}\}$ of three Ã©lÃ©ments ;
- Determine the medians $m_k$ of $B_k$, $k \in \{1, \ldots, n/3\}$ ;
- Apply the algorithm recursively to determine the median $p$ of $m_1, \ldots, m_{n/3}$. Determine the rank of $p$ in the array, and then apply the algorithm recursively on part of the array.

4. Show that the chosen pivot is strictly greater than at least $n/3 - O(1)$ elements of the table and is less than or equal to at least $n/3 + O(1)$ of the array elements.
5. Compute the complexity of the SELECT function.

**Exercise 2.** The room INFO 4 is in high demand... All departments of ENS want to teach there and they send requests to the administration form $[d, f[$ to use the room from $d$ to $f$ (we can assume that $d$ and $f$ are always integers). The administration seeks to reject the fewest number of requests from the set $\mathcal{I} = \{[d_1, f_1[, [d_2, f_2[, [d_3, f_3[, \ldots, [d_n, f_n[\}$ it received.

We propose to use a greedy algorithm to solve this problem. The idea is to sort the intervals of $\mathcal{I}$ according to some order, to accept the first request, to remove requests that conflict and to repeat these three steps until no interval remains. For each of the following orders, determine if the corresponding greedy algorithm is optimal (by a proof or counter-example).

1. Sort the course in order of their starting time $d_i$.
2. Sort the course in order of their finishing time $f_i$.
3. Sort the course in order of their duration $f_i - d_i$.
4. Sort courses by ascending order of the number $c_i$ of other courses with which they conflict (*i.e.* $c_i = \#\{[d, f[ \in \mathcal{I} \mid [d_i, f_i[ \cap [d, f[ \neq \emptyset\}$).

**Exercise 3.** We have a set of identical glass balls. The problem is to determine from which floor of a building the glass balls will break when thrown out the window. You are in an $n$-story building (numbered 1 to $n$) and you have $k$ balls.

There is only one way to test if the height of a floor is fatal : throw a ball through the window. If it does not break, you can then re-use that ball for other tests, otherwise you cannot.

You must propose an algorithm for finding the height from which a throw is fatal (return an error if the ball thrown out the $n$-th floor remains intact). The complexity the algorithm is the number of throws needed to determine this height.

1. Propose an algorithm for $k = 1$ and give its complexity.
2. For $k \geq \lceil \log_2(n) \rceil$, propose an algorithm with complexity $O(\log(n))$.
3. For $k < \lceil \log_2(n) \rceil$, propose an algorithme with complexity $O\left(k + \frac{n}{2^{k-1}}\right)$. What is the complexity of this algorithm when $k = 2$ ?
4. Propose an algorithm with complexity $o(n)$ when $k = 2$.
5. Propose an algorithm which needs $\sqrt{2n}$ throws when $k = 2$.

**Exercise 4.** Maximum & Minimum

1. Give a simple algorithm that calculates the minimum and maximum on an $n$ element array and give the number of comparisons in the worst case.
2. Is it possible to reduce the number of comparisons made ? Describe an algorithm with fewer comparisons in the worst case.
3. Show a lower bound of $n$ comparisons.
4. Show a better lower bound on the number of comparisons needed.

   **Hint** : We can use an *adversarial* argument.

   Let $\mathcal{A}$ be an algorithm that finds the maximum and minimum. Describe a strategy for an adversary who chooses the answers to the comparison in order to force $\mathcal{A}$ to make more comparisons.

**Exercise 5.** Pivot selection

Analyze the running time of the randomized selection algorithm in the following case.

1. The pivot is chosen uniformly at random in the array.
2. The pivot is chosen as the median of three elements each chosen uniformly atrandom in the array.