Sélection internationale Session 2016
École Normale Supérieure Paris
Épreuve de culture scientifique - **Informatique** Duration : 2 hours

For candidates who chose **computer science** as **secondary specialisation**

If you cannot answer a question, you may use it as hypothesis to later questions.
Calculators are not allowed.

Exercice 1.
A subsequence is anything obtained from a sequence by deleting a subset of elements; the elements of the subsequence need not be consecutive in the original sequence. For example, the strings `L`, `ELECTION`, `EEIOI`, `NTRNATNL`, and `SELECTIONINTERNATIONALE` are all subsequences of the string `SELECTIONINTERNATIONALE`

In all questions, first describe a recursive algorithm and only then transform it into an iterative algorithm.

1. Suppose we are given two arrays $A[1 \dots \ell]$ and $B[1 \dots m]$. A common subsequence of A and B is any subsequence of A that is also a subsequence of B. For example, `ETION` is a common subsequence of `SELECTION` and `INTERNATIONALE`. Describe and analyze an efficient algorithm to compute the length of the longest common subsequence of two given arrays $A[1 \dots \ell]$ and $B[1 \dots m]$.

2. Describe and analyze an efficient algorithm to compute the length of the longest common subsequence of three given arrays $A[1 \dots \ell]$, $B[1 \dots m]$ and $C[1 \dots n]$.

3. A shuffle of two strings X and Y is formed by interspersing the characters into a new string, such that the characters of X and Y remain in the same order. For example, given the strings `SELECTION` and `INTERNATIONALE`, the string `SELINETERNCTAITIONONALE` is indeed a shuffle of the two:

$$\texttt{SEL}^{\texttt{IN}}\texttt{E}^{\texttt{TERN}}\texttt{CT}^{\texttt{A}}\texttt{I}^{\texttt{TION}}\texttt{ON}^{\texttt{ALE}}$$

Given three strings $A[1 \dots \ell]$, $B[1 \dots m]$ and $C[1 \dots \ell + m]$, describe an algorithm to determine whether C is a shuffle of A and B. In all cases, first describe a recursive algorithm and only then transform it into an iterative algorithm.

Exercice 2. Consider the following algorithm where $\mathsf{Random}(1, n)$ returns a number picked uniformly at random between 1 and n (inclusive) in constant time.

```
SHUFFLE(A[1...n])
for i ← 1 to n
    B[i] ← null
for i ← 1 to n
    j ← Random(1, n)
    while B[j] ≠ null
        j ← Random(1, n)
    B[j] ← A[i]
for i ← 1 to n
    A[i] ← B[i]
```

1. Prove that SHUFFLE will shuffle the input array into a random order such that every permutation is equally likely.

2. What is the expected running time of the above algorithm? Justify your answer and give a tight asymptotic bound.

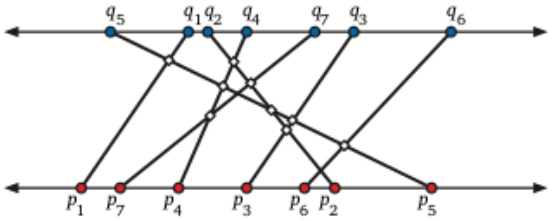
3. Describe an algorithm that randomly shuffles an n-element array, so that every permutation is equally likely, in $O(n)$ time.

Exercice 3.

1. An inversion in an array $A[1 \dots n]$ is a pair of indices (i, j) such that $i < j$ and $A[i] > A[j]$. The number of inversions in an n-element array is between 0 (if the array is sorted) and $\binom{n}{2}$ (if the array is sorted backward). Describe and analyze a divide-and-conquer algorithm to count the number of inversions in an n-element array in $O(n \log n)$ time. Assume all the elements of the input array are distinct.

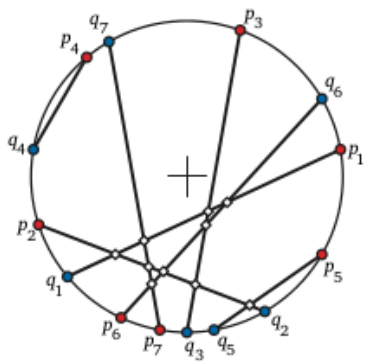
2. Suppose you are given two sets of n points, one set $\{p_1, p_2, \dots, p_n\}$ on the line $y = 0$ and the other set $\{q_1, q_2, \dots, q_n\}$ on the line $y = 1$. Create a set of n line segments by connect each point p_i to the corresponding point q_i . Describe and analyze a divide-and-conquer algorithm to determine how many pairs of these line segments intersect, in $O(n \log n)$ time.
   Hint: Use your solution to question 1.

   Assume a reasonable representation for the input points, and assume the x-coordinates of the input points are distinct. For example, for the input shown below, your algorithm should return the number 9.



3. Now suppose you are given two sets $\{p_1, p_2, \dots, p_n\}$ and $\{q_1, q_2, \dots, q_n\}$ of n points on the unit circle. Connect each point p_i to the corresponding point q_i . Describe and analyze a divide-and-conquer algorithm to determine how many pairs of these line segments intersect in $O(n \log^2 n)$ time.
   Hint: Use your solution to question 2.

   Assume a reasonable representation for the input points, and assume all input points are distinct. For example, for the input shown below, your algorithm should return the number 10.



4. ($\star$) Propose an improved algorithm that runs in $O(n \log n)$ time.