**Partie 1**

Conception d’algorithmes de tri



*Pour ne pas avoir à trier tout ça à la main.*

*Image : Pierpaolo Rovero*

# Activité préliminaire

Par îlots de 2 ou 3 personnes et ordinateurs éteints, les îlots vont devoir concevoir deux algorithmes de tri différents, les formaliser sous forme de pseudo-code, et les implémenter.

|  |
| --- |
| **Note** A cette étape, l'idée principale est de concevoir vous-même deux algorithmes de tri. N’allez pas chercher de ressource sur internet pour vous aider, vous risqueriez de trouver la réponse ! |

Un algorithme de tri permet de prendre une liste de valeurs non triées en entrée, et de retourner les retourner triées, exemple :

**Liste en entrée :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **3** | **9** | **7** | **1** | **6** | **2** | **8** | **4** | **5** |

**Liste après tri :**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |

L’idée paraît relativement simple, et pourtant il existe de très (très) nombreux algorithmes différents permettant de résoudre ce même problème.

L’exercice qui vous est demandée est le suivant :

* Réfléchissez par îlot de 2 à 3 personnes à deux manières différentes de résoudre ce même problème.
* Pour chacune de ces deux méthodes, écrivez un pseudo-code permettant de formaliser de manière non ambiguë votre méthode de résolution. Sachez qu’il n’existe pas de convention universelle sur une syntaxe “pseudo-code” (<https://fr.wikipedia.org/wiki/Pseudo-code>). L’objectif global est de pouvoir communiquer, et transmettre une série d’instructions de manière indépendante d’un langage de programmation.
* Une fois le pseudo-code écrit, vous devez le traduire et l’implémenter en javascript.