Fichier requis	insertion_sort.py, selection_sort.py, bubble_sort.py, shell_sort.py, merge_sort.p quick_sort.py, comb_sort.py, README
Langage autorisé	Python
Correction	À distance
Taille de groupe	2/3

- Réaliser une série de tris.
- Évaluer leur efficacité.
- Analyser différents algorithmes de tri afin d'identifier l'algorithme le plus optimal à la situation à traiter.
- Distinguer les différences de fonctionnement entre les algorithmes de tri.

Vous devez penser et implémenter les algorithmes suivants dans des scripts respectifs nommés comme spécifié entre parenthèses :

- tri à bulle (*bubble_sort*)
- tri par sélection (*selection_sort*)
- tri par insertion (*insertion_sort*)
- tri fusion (*merge_sort*)
- tri rapide (*quick_sort*)
- tri shell (*shell_sort*)
- tri peigne (*comb_sort*)

Les scripts doivent se situer à la racine de votre dépôt.

Vos scripts seront lancés de la manière suivante :

```
python insertion_sort.py "1;-2;3.5;4"
```

Vos scripts doivent fonctionner en local sans aucune dépendance et être autosuffisants.
Vos scripts doivent écrire les informations suivantes sur la sortie standard :

- la série passée en paramètre
- le résultat obtenu
- le nombre de comparaisons
- le nombre d'itérations réalisées par le tri (excepté pour le tri fusion et le tri rapide)
- le temps en secondes de la réalisation du tri (deux nombres après la virgule)

Exemple du formatage exigé

```
$> py bubble_sort.php "1;-2;4;3.5;3"
Série : 1;-2;4;3.5;3
Résultat : -2;1;3;3.5;4
Nb de comparaison : 24
Nb d'itération : 25
Temps (sec) : 2.05
```

$>ort de connaissances

Pour chaque tri, vous devrez compléter votre `README` avec un paragraphe expliquant le fonctionnement de chaque algorithme. En plus de cela, réalisez une analyse comparée des différents algorithmes indiquant quel algorithme est plus intéressant que tel autre et pour quelles raisons.