# Documentation Projet « externalsort »

Ce projet est composé de 2 documents. Le premier ‘data.txt’ est un exemple de dataset composé de milliers de lettres dans un ordre aléatoires. Le second, ‘externalsort.py’ est un fichier python qui réalisera le tri du dataset.

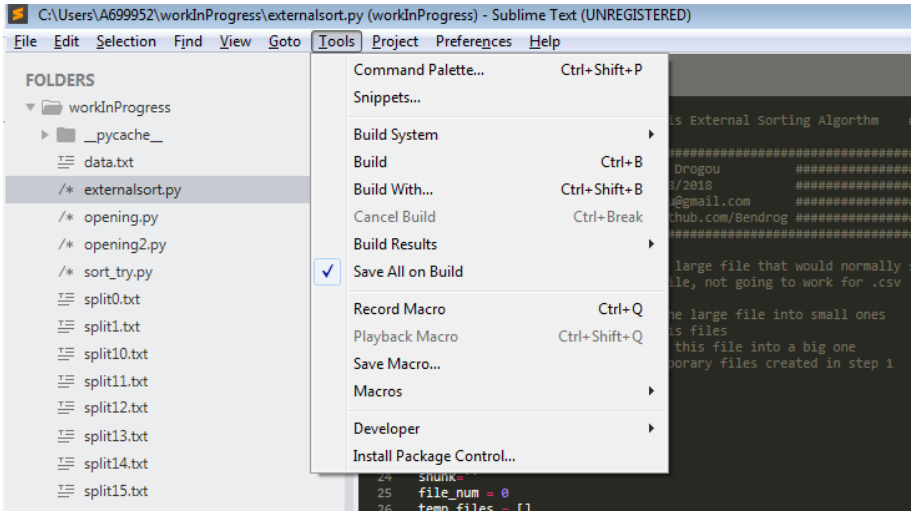
## 1- Comment faire tourner le projet ?

Pour que le projet compile bien il est important que les 2 fichiers soient au même niveau dans le même dossier.

Il y a 2 méthode pour compiler un projet.

### A- Compiler via l’IDE

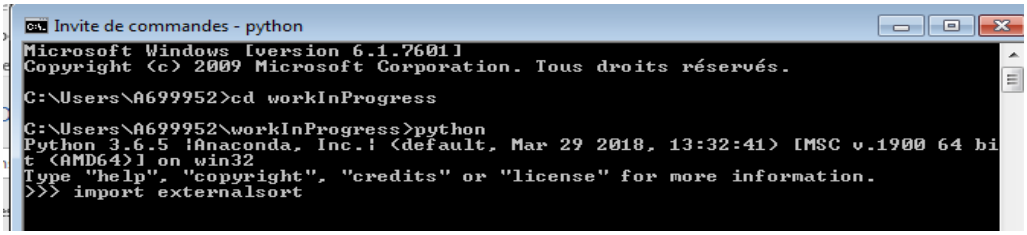
La première via l’IDE. Supposons que vous ouvriez le fichier via Sublime Text 3. Allez dans l’onglet ‘Tools’, puis sélectionnez ‘Build’. Ou bien directement via le raccourci clavier Ctrl + B.




### B- Compiler via l’invite de commande

La seconde méthode consiste à utiliser l’invite de commande.

Il faut au préalable avoir installé python sur l’ordinateur. Puis aller dans le dossier qui contient les deux fichiers du projet. Lancer le shell python grâce à la commande ‘python’, puis écrire ‘import externalsort’



Une fois que le programme a terminé de compiler (cela peut prendre 15 minutes), vous verrez que des fichiers « split*.txt » (avec * un nombre) se sont créés et que leur contenu est trié dans l'ordre alphabétique.

A noter, si le fichier 'data.txt' contient des majuscule et des minuscules, les majuscules seront traitées avant. L'ordre alphabétique est donc A…Za…z

## 2- Explications théorique

Le principe d'un algorithme d'external sorting est de pouvoir trier un fichier tellement lourd qu'il saturerait la RAM du PC en l'ouvrant.

Une façon de résoudre ce problème serait de travailler avec un Machine Virtuelle surpuissante. Une façon plus astucieuse serait de réaliser un algorithme de tri externe.

Pour ce faire il faut diviser le fichier en plusieurs fichiers de plus petite taille, trier ces fichiers, et les fusionner entre eux.

## 3- Pistes d'amélioration

- L'algorithme de tri peut être grandement amélioré
- Une amélioration serait que l'algorithme soit compatible avec d'autres extensions que « .txt ». Par exemple « .csv »
- Créer une class avec des fonctions à réutiliser plutôt qu'un fichier avec un seul bloc de code
- Le temps d'attente est trop long. Ceci en grande partie parce que la partie de fusion des fichiers copie les fichier ans leur globalité dans des listes. Le plus efficace serait :
    - Prendre les éléments de l'alphabet ayant le plus petit indice
    - Les mettre dans une liste
    - Ecrire dans un nouveau fichier
    - Vider la liste
    - Recommencer avec des éléments d'indice n+1 de l'alphabet