**DIU EIL : Février 2021 : Projet complexité :**

**Difficultés et apports pédagogiques.**

# Difficultés que peuvent rencontrer les élèves :

* Selon nous, la première difficulté lorsque l’on souhaite démarrer un tel projet n’est pas de coder un algorithme de tri car de nombreux exemples sont disponibles sur Internet mais c’est de savoir comment structurer nos résultats. Les outils utilisés tels que les jeux de test aléatoires, le comptage du nombre d’actions, les temps de calcul ou les graphiques créés à partir de matplolib sont des éléments à part entière d’un cours de programmation. Nous avons donc préféré initier les élèves à tous ces outils dans un cours qui reprend l’exemple du tri par insertion, non demandé dans le projet.

Nous avons profité de ce cours pour introduire les notions de terminaison et de preuves d’algorithmes. Bien que ces notions ne soient pas demandées dans le projet, car trop complexes, elles font partie du programme de première et peuvent être abordées dès le premier algorithme de tri. Le cours est donc conséquent et un des challenge des élèves sera d’aller rechercher l’information au bon endroit.

* Nous avons trouvé plus judicieux que les élèves remplissent un Jupyter Notebook pré-structuré afin de regrouper les scripts python et les explications. C’est un outil qui doit être utilisé tout au long de l’année et il ne doit pas être une découverte pour ce projet. Cependant, certains élèves auront peut-être des difficultés avec des outils propres au projet comme la création d’un sommaire ou de tableaux. C’est pourquoi nous avons mis ces outils à disposition dans la fiche de cours afin qu’ils s’en inspirent.
* Afin de créer des jeux de test adaptés au meilleur des cas et au pire des cas, l’élève devra penser à utiliser la fonction native sort() de python et son option reverse. Pour cela, nous avons fait une introduction à la fin du cours sur la fonction sort() mais ce sera à l’élève de se documenter pour trouver l’option reverse. Nous avons également précisé dans les consignes que les tailles des tableaux utilisées en cours (de 1 à 100 avec un pas de 10) n’est certainement pas suffisante. Il faudra donc qu’il pense à augmenter la taille maximale du tableau et à régler le pas sans pour autant aller dans les extrêmes car les temps de calcul doivent rester raisonnables.
* Selon nous, une des plus grandes difficultés de ce projet est la compréhension de la complexité des algorithmes. C’est un domaine de l’informatique théorique qui nécessite des bases solides en mathématiques, que tous les élèves de première générale n’ont pas nécessairement. En conséquent, nous ne demandons pas de preuves formelles mais une idée globale : par exemple pour le tri à insertion si n est la taille de la liste, le contenu de la première boucle est exécuté environ n fois et la deuxième boucle n fois au maximum donc on dit que le nombre de calculs de cet algorithme est de l'ordre de n².
* Enfin, nous demandons aux élèves de conclure en comparant les temps de calcul et les nombres d’actions effectuées mais nous laissons beaucoup de liberté sur cette question. Les résultats peuvent être présentés sous forme de tableaux ou de graphiques. Ces outils sont tous les deux présentés dans le cours mais la difficulté sera alors de les exploiter à bon escient.

# Apports pédagogiques :

Dans le cours sur le tri insertion, nous avons donc souhaité mettre à disposition de l’élève les outils nécessaires à son projet à l’aide de petits exercices corrigés, voici la liste de ces nouveaux outils

* Calcul du nombre d’actions élémentaires (notion d’incrémentation)
* Détermination du temps d’exécution d’un algorithme (fonction timer de python)
* Jeux de test remplis aléatoirement et de taille variable (fonction random.sample)
* Visualisation graphique des résultats (module matplotlib) : nous avons créé des fonctions clé en main faciles de réutilisation.