exo 1.3

Dans notre solution, l'algorithme de tri utilise la fonction de tri intégrée de Python (‘sorted()‘), qui a un temps d'exécution de $O(n \log n)$ dans le pire des cas. Ici, n représente le nombre de maisons. L'algorithme consiste en deux étapes principales : trier les positions des maisons et renvoyer la liste triée. Examinons pourquoi cela est polynomial :

1. Triage des positions des maisons : La première étape de notre algorithme consiste à trier les positions des maisons en utilisant la fonction de tri intégrée. Comme mentionné précédemment, le temps d'exécution de cet algorithme de tri est $O(n \log n)$ dans le pire des cas.

2. Retourner la liste triée : La deuxième étape implique simplement le retour de la liste triée. Cela se fait en temps constant ($O(n)$) car il suffit de renvoyer la liste triée résultante.

La complexité totale de l'algorithme est donc $O(n \log n)$, ce qui est polynomial par rapport à la taille de l'entrée (n). Cela signifie que la durée d'exécution de notre algorithme de tri croît de manière polynomiale avec la taille de l'entrée, ce qui est considéré comme efficace et pratique pour des valeurs raisonnables de n.

math

leonardo basbous

October 2023

1 Introduction