Mariam Sarwat (1928777)
Anastasiya Basanets (1933929)

Algorithme Glouton

L'algorithme est composé des étapes suivantes :

- Déclarations de variables : $\Theta(1)$
- Itération de n fois une boucle while selon i et j (Pour chaque ville i, on cherche la ville j avec la distance la plus minimal) : $\Theta(n^2)$
- Conditions if vérifiant si on a visité toute les villes (calcul du chemin de retour) : $\Theta(1)$
- Conditions if vérifiant si un point a été visité : $\Theta(1)$
- Conditions if vérifiant si on a visité toute les ville j pour une ville i. Ajouter la distance minimale retrouvée à *totMinDist* et réinitialisé les variables : $\Theta(1)$

En utilisant la règle du maximum, nous avons donc une complexité de $\Theta(n^2)$.

Algorithme Programmation Dynamique

Avant d'utiliser l'algorithme, nous créons une matrice 2D qui sera. Pour chaque point, on viendra trouver la distance avec tous les autres points et l'insérer dans la matrice. Ainsi, nous avons n nombres de lignes pour n points et 2^n colonnes pour le nombre de sous-ensembles des n éléments. Nous avons ainsi, une complexité de $\Theta(n2^n)$.

L'algorithme est composé des étapes suivantes :

- Boucle for avec une itération de n fois sur la matrice 2D. Ainsi, chacun des sous-ensembles est soumis à n itération via la fonction *DPAlgo* : $\Theta(n^2 2^n)$
- Conditions if vérifiant si un point a été visité : $\Theta(1)$

En utilisant la règle du maximum, nous avons donc une complexité de $\Theta(n^2 2^n)$.

Algorithme Approximatif

L'algorithme est composé des étapes suivantes :

- Recherche du MST via *primMST*. Boucle for faisant n-1 itérations prenant n itérations pour faire la recherche de la distance minimale via la fonction *getMinKey* : $\Theta(mn)$ où m=n-1
- Création du préodre via *preOrdrer* en passant à travers le MST avec une boucle for de n itérations chacune. Cette boucle est englobée d'une boucle while qui empile et dépile le nombre de villes passées. Ce while, va donc faire 2n itérations. On aura alors un totale de : $\Theta(n^2)$.
- Boucle while dans *approxAlgo* de n itérations calculant la distance minimale : $\Theta(n)$.

En utilisant le maximum, nous avons donc une complexité de $\Theta(n^2)$.