**Liste**

1. La fonction **Size()** prend un temps O(n) car elle parcours tous les noeuds pour les additionner le nombre d’élément et retourne le nombre de nœuds total à la fin.
2. La fonction push\_front : voici les opérations éffectués par la fonction push\_front :

* Creation d’un nouveau nœud : se fait en O(1)
* Vérification si la liste est vide : se fait en O(1)
* Ajout au début du SuperNoeud : se fait en O(1)
* Creation d’un nouveau SuperNoeud : se fait en temps O(1)

La fonction push\_front prend un temps O(1), car On insère un nouvel élément au début de la liste et cela la création de de nouveau noeud.

1. La fonction Push\_back : voici les opérations éffectués par la fonction :

* Creation d’un nouveau nœud : se fait en O(1)
* Vérification si la liste est vide : se fait en O(1)
* Ajout à la fin du SuperNoeud : se fait en O(1)
* Creation d’un nouveau SuperNoeud : se fait en temps O(1)

La fonction push\_back se fait en temps O(1) car, On insère un nouvel élément à la fin de la liste. Si le dernier super nœud a de la place, on peut l’ajouter directement ; sinon, on crée un nouveau super nœud.

1. La fonction pop\_front : voici les opérations éffectués par cette fonction :

* Vérifie si la liste est vide : O(1)
* Premiére possibilité : si le SuperNoeud n’a qu’un seul élément :

. supprimer un Nœud : O(1)

. supprimer le superNoeud : O(1)

. mettre à jour les pointeurs début et fin : O(1)

* Deuxième possibilité : si le SuperNoeud contient plus d’un élément :

. supprimer un Nœud et avancer le pointeur : O(1)

. mettre à jour le nombre d’éléments : O(1)

La fonction pop\_front prend un temps O(1) car, on retire l’élément en tête de la liste. Cela implique des opération qui modifie les pointeur

1. La fonction pop\_back : voici les opérations éffectués par cette fonction :

* Vérifie si la liste est vide : O(1)
* Premiére possibilité : si le dernier SuperNoeud n’a qu’un seul élément :

. supprimer un Nœud : O(1)

. supprimer le superNoeud : O(1)

. mettre à jour les pointeurs début et fin : O(1)

* Deuxième possibilité : si le dernier SuperNoeud contient plus d’un élément :
* . Parcourir les nœuds pour trouver l'avant-dernier nœud : O(n), où **n** est le nombre d'éléments dans le dernier "SuperNoeud".
* . Supprimer le dernier nœud et mettre à jour les pointeurs : O(1)

La fonction pop\_front prend un temps O(n) dans le pire cas, et O(1) si le super noeud à plus d’un élément. Car, On doit parcourir les nœuds du dernier super nœud pour trouver le dernier élément, ce qui nécessite une boucle, mais dans le cas où il reste plusieurs éléments, cela peut être fait en temps constant en modifiant les pointeurs.

1. La fonction Operator[] : voici les opérations éffectués par cette fonction ou **k** représente le nombre d’élément à parcourir :   
   - parcours des SuperNoeuds : O(n/k)

-Parcours des Nœuds dans le superNoeud trouvé : O(k)

La fonction opérator[] à une complexité de O(n/k + k),car on doit parcourir tous les super noeuds jusqu’à atteindre le i.