SAE R101-C – rapport

a) Objectif (types d’implémentations) et matériel employé

Durant cette Saé, l’objectif a été de comparer empiriquement l’efficacité de trois implémentations différentes de listes triées de chaînes de caractères. A savoir :

-Liste contiguë

-Liste chaînée

-Liste chaînée avec gestion de l’espace libre

Pour cette Saé nous avons utilisé un ordinateur personnel avec un processeur AMD Ryzen dont la vitesse oscille entre 1.8 Ghz et 4.3 Ghz , la mémoire est cadencée à 3200Mhz, le système d’exploitation est Windows 10 64-bit.

b) Algorithmes logiques

fonction adjlisT(l InOut : Liste(chaîne), nom : chaîne)

Début

P🡨tete(l)

si val(l,p) > nom

alors adjtlis(l,nom)

sinon

precedent🡨p

p🡨suc(l,p)

ajout🡨faux

tant que non finliste(l,p) et non ajout faire

si val(l,p) > nom

alors adjlis(l,précedent,nom)

ajout🡨vrai

fsi

precedent🡨p

p🡨suc(l,p)

ftant

si non ajout

alors adjqlis(l,nom)

fsi

fsi

fsi

fin

fonction suplisT (l InOut : Liste(chaîne), c : chaîne )

début

p🡨tete(l)

depassement🡨faux

tant que non finliste(l,p) et non depassement faire

chCourante🡨val(l,p)

si chcourante >= c

alors depassement🡨vrai

si chCourante = c

alors suplis(l,p)

fsi

sinon p🡨suc(l,p)

fsi

ftant

fin

c)Expériences et résultats

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Nous avons mis en place un compteur. Son rôle est de mesurer le temps écoulé pendant l’exécution de la boucle for des méthodes de remplissage ou de suppression ; pour ce faire, on récupère le temps et on l’affecte à une variable de type long aux deux extrémités de l’itération. Finalement, on affecte la valeur de la différence entre la valeur finale et initiale à une variable dureeD de type long et on la retourne.

Dans le cas de l’ajout au début de 10 éléments de type String :

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

(Temps en nanosecondes)

On s’aperçoit que la liste contiguë est plus rapide avec la méthodologie de mesure présentée ci-dessus. L’ordre de grandeur pour les différentes listes chaînées est le même et est une fois supérieur à celui de la liste contiguë.

Dans le cas de l’ajout en fin de liste de 10 éléments de type String :

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

(Temps en nanosecondes)

On voit que la liste chaînée avec gestion de places libres est légèrement plus rapide que la liste contiguë cependant les trois implémentations partagent le même ordre de grandeur. La liste chaînée de base est environ deux fois plus lente que les autres.

Dans le cas de la suppression au début de 10 éléments de type String :

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

(Temps en nanosecondes)

On remarque que le temps d’exécution est en moyenne plus rapide pour la suppression que pour l’ajout. La liste contiguë est en tête et les deux implémentations chaînées sont au même niveau.

Dans le cadre de la suppression à la fin de 10 éléments de type String :

Une image contenant texte, Police, capture d’écran, ligne

Description générée automatiquement

(Temps en nanosecondes)

Dernièrement, on voit que le liste contigüe et chaînée sont au même niveau et la chaînée avec gestion de places libres un peu plus lente en comparaison.

Question 9 :

La méthode suplisT est conçue pour éliminer un élément spécifique d'une liste. En cas d'insertion d'une chaîne qui ne fait pas partie de la liste, la fonction effectuera d'abord une recherche de cette chaîne. Si la méthode ne parvient pas à la localiser, la liste restera inchangée. Il pourrait être pertinent de répéter cette opération plusieurs fois afin de mettre à l'épreuve la fiabilité de l'algorithme logique et ainsi évaluer l'efficacité de la fonction suplisT.

Amélioration des tests :

Une image contenant texte, capture d’écran, logiciel

Description générée automatiquement

Nous avons modifié les tests simplement en faisant boucler 100 fois les méthodes précédemment utilisées, une par une.

Ajout-début

Une image contenant texte, Police, capture d’écran, typographie

Description générée automatiquement

La liste chaînée de base et contiguë plus ou moins équivalentes. L’implémentation chaînée avec places libres se démarque par un temps d’exécution plus lent.

Ajout-fin

Une image contenant texte, Police, capture d’écran, typographie

Description générée automatiquement

Ici, la première chose observée, relève du temps d’exécution qui est bien plus long que les autres.

10e7 ns revient à dire 10e(7-9) = 10e-2 s.

Donc dans ce cas de figure, c’est la liste chaînée de base qui est plus lente que les autres avec ~0.02 s de temps d’exécution.

Sup-début

Une image contenant texte, Police, capture d’écran, typographie

Description générée automatiquement

La liste contiguë et chaînéePL sont nettement plus rapides que la liste chaînée de base.

Sup-fin

Une image contenant texte, Police, capture d’écran, nombre

Description générée automatiquement

Finalement, la liste contiguë reste devant ici et la liste chaînéePL est nettement plus lente.

d)Conclusion

Pour conclure cette SAE, on peut dire que dans la grande majorité des cas, la liste contiguë reste la plus efficace des trois implémentations de listes triées testées. La liste chaînée avec gestion de places libres est dans certains cas plus rapide que la liste chaînée de base.