Laboratoire 4 – Liste chainée simple

Objectifs ASD

Mettre en œuvre en C++ la structure de donnée de liste chainée simple, et en particulier

- Insertion, accès à et suppression d'un élément en tête
- Insertion, accès à et suppression d'un élément en position quelconque
- Recherche linéaire d'un élément dans la structure
- Copie et effacement de la structure.
- Tri par fusion de la structure, en ne modifiant que les liens entre nœuds et pas les données stockées.

Comprendre et documenter la complexité de toutes ces opérations.

Objectifs C++

Mettre en œuvre une classe C++ générique, et en particulier

- Allocation dynamique des nœuds avec new et delete
- Gestion dynamique des nœuds en n'y accédant uniquement via des pointeurs
- Ecriture des constructeurs et destructeurs
- Ecriture des constructeurs de copie et opérateurs d'affectation
- Exceptions à lever quand l'état de la structure ne permet pas l'exécution d'une méthode.
- Garantie forte à fournir en cas d'exception pour toutes les méthodes
- Fonctions itératives pour les méthodes parcourant les éléments
- Fonctions récursives pour le tri par fusion

Consignes

Ce laboratoire se déroule sur une période de 3 semaines. Des résultats intermédiaires doivent être fournis à la fin de chaque semaine sous la forme d'un codecheck testant votre classe C++.

- Codecheck 1 teste l'insertion, l'accès et la suppression d'éléments en tête ou en position quelconque, ainsi que les constructeurs et destructeurs
- Codecheck 2 teste les mêmes éléments que le 1, mais aussi la recherche linéaire, le constructeur de copie et l'opérateur d'affectation, les exceptions levées par votre code.

- Codecheck 3 teste les mêmes éléments que le 2, mais aussi le tri par fusion et la garantie forte en cas d'exception lancée par le type générique lors de copie/affectation

L'interface publique de la classe à mettre en œuvre, la structure de nœuds à utiliser ainsi que les attributs privés de la classe sont fournis et ne peuvent être modifiés. Ces attributs sont

- Un pointeur `head` vers le nœud de tête. `nullptr` si la liste est vide.
- Un entier non signé `nbElements` stockant le nombre d'éléments dans la liste.

Le contenu de ces deux attributs doit rester synchronisé en permanence.

Vous pouvez/devez ajouter des méthodes privées quand elles permettent d'éviter de dupliquer du code, ainsi que pour la mise en œuvre récursive du tri par fusion.

Pour le dernier codecheck uniquement, toutes les méthodes, publiques ou privées, doivent être commentées dans le style doxygen, y compris un champ @remark documentant la complexité algorithmique de la fonction.

Contacts

- Prof. Olivier Cuisenaire : olivier.cuisenaire@heig-vd.ch
- Prof. Mireille Goud : mireille.goud@heig-vd.ch
- Prof. Laura Elena Raileanu : laura.raileanu@heig-vd.ch

- Dominique Jollien : dominique.jollien@heig-vd.ch
- Raphaël Racine : raphael.racine@heig-vd.ch