1 Introduction

Récupérez sur l'*Espace pédagogique* les fichiers `progListeSC.h`, `progListeSC.cpp` et `fichierTP2.cpp`.
Le fichier entête `progListeSC.h` et le fichier C++ `progListeSC.cpp` fournissent respectivement une définition et une implantation du Type `Liste Simplement Chaînée (ListeSC)`. Les opérations du type `ListeSC` sont celles vues en cours : creerLSC, insererDebutLSC, insererApresLSC, insererFinLSC, predecesseurLSC, supprimerLSC, supprimerFinLSC, supprimerDebutLSC, auxquelles ont été ajoutées des opérations d'entrée-sortie lireLSC, afficherLSC. Leurs spécifications sont rappelées dans le fichier entête `progListeSC.h`.
Les fichiers `fichierTP2.cpp` et `fichierTP3.cpp` contiennent des définitions de fonctions sur les listes simplement chaînées que vous aurez à compléter. Nous commencerons pour cette séance par le fichier `fichierTP2.cpp`.
Nous n'utiliserons pas la partie objet de C++ : le type liste est défini à l'aide de structure et non de classe.
Pour traduire les notions de paramètre *résultat* et *donnée-résultat*, nous utiliserons le mode de passage de paramètres par référence de C++, plus facile d'utilisation que le passage de paramètre par pointeur à la C.

2 Fichier fichierTP2.cpp

Ouvrez le fichier `fichierTP2.cpp` :

Question 1 Complétez le `main` en ajoutant dans le bloc `case 1` les instructions permettant de :
- ajouter un élément de valeur 11 en fin de la liste `l1` (attention à l'appel de `insererFinLSC`)
- insérer un élément de valeur 22 en deuxième position de la liste `l1` ; donnez 2 versions :
  - l'une utilisant l'opération `insererApresLSC`
  - la seconde transformant directement le chaînage de la liste `l1` en utilisant `creerLSC`
- supprimer le deuxième élément de `l1` ; donnez 2 versions :
  - l'une utilisant l'opération `supprimerLSC`,
  - l'autre modifiant directement le chaînage de `l1`.
- inverser les 2 premiers éléments de `l1` ; donnez 2 versions :
  - l'une inversant les valeurs (en modifiant les champs `info`),
  - l'autre inversant l'ordre des cellules (en modifiant les champs `succ`).

Pour vérifier vos réponses, compilez (`g++ -o tp2 fichierTP2.cpp progListeSC.cpp`) et lancez l'exécution du programme avec l'option 1.

Question 2 Complétez la définition des deux fonctions `dernierLSC` et `estTrieeLSC` dont les alogithmes ont été étudiés en TD. Testez ces fonctions : compilez et exécutez le programme avec l'option 2.

Question 3 Vous trouverez ensuite l'entête de la fonction `oterRepetitionLSC`. Cette fonction supprime les répétitions d'éléments consécutifs égaux en modifiant le chaînage de la liste : lorsque 2 éléments consécutifs sont égaux on supprime la deuxième cellule en modifiant le champ `succ` du premier. On vous demande d'écrire une version itérative et une version récursive (`oterRepetitionLSCR`).
Complétez les corps des 2 fonctions, compilez et testez (avec l'option 3).

Question 4 Le fichier contient ensuite les entêtes de 2 fonctions pour la concaténation de 2 listes
La première fonction (`concatLSC`) correspond aux spécifications :

Algorithme : concatLSC(**dr** L1 :Liste, **d** L2 : ListeSC)
Données : $L1, L2$ deux Listes chaînées
Résultat : Calcule dans $L1$ la concaténation de L1 et L2, en modifiant le chaînage de $L1$. L'algorithme ne crée aucune nouvelle cellule et ne renvoie rien.

En utilisant la fonction `dernierLSC`, complétez le corps de la fonction `concatLSC`.
Compilez et testez en exécutant la quatrième partie du `main` (avec l'option 4).
Le programme demande la saisie de 2 listes `l1` et `l2`, calcule et affiche le résultat de leur concaténation. Il affiche ensuite l'adresse des dernières cellules des 2 listes. Que constatez-vous ? Vérifions en ajoutant 44 en fin de la liste `l1` et en affichant les valeurs des 2 listes `l1` et `l2`. La liste `l2` a-t-elle été modifiée ?

Question 5 Donnez une deuxième fonction pour la concaténation de 2 listes. Cette fonction (`concatLSCCopie`)

Algorithme : concatLSCCopie(**d** L1 :Liste, **d** L2 : ListeSC) : ListeSC
Données : $L1, L2$ deux Listes chaînées
Résultat : Renvoie la concaténation de L1 et L2 en dupliquant les éléments de $L1$ et $L2$.

doit opérer en recopiant les 2 listes $L1$ et $L2$ passées en paramètre. Utilisez pour cela la fonction `insererFinLSC` pour insérer en fin de la liste résultat tous les éléments de la liste $L1$ puis tous ceux de la liste $L2$. Compilez et testez (avec l'option 5).
Le programme demande la saisie de 2 listes `l1` et `l2`, calcule `l3` leur concaténation et affiche cette dernière. Il affiche ensuite l'adresse des dernières cellules des 3 listes. Les 3 listes partagent-elles des cellules communes ?
Vérifions en ajoutant 55 en fin de la liste `l1` et en affichant les valeurs des 3 listes `l1`, `l2` et `l3`. Les listes `l2` et `l3` ont-elles été modifiées ?