SANI Abdou-Rahmane

STRUCTURES DE DONNÉES FONDAMENTALES SESSION 2

PRESENTATION DU PROJET :

Le projet consistait à créer en langage C, des listes qui soient capable de pouvoir implémenter des grands entiers positifs permettant de pouvoir ainsi dépasser la taille borné sur 32 bits ([0,2^32-1]), et sur 64 bits ([0,2^64-1]). Pour cela, je faire en sorte que chaque cellule de la liste soit en mesure de pouvoir implémenter une décimale de l'entier.Grâce à cette méthode, je devais être en mesure de pouvoir effectuer les opérations arithmétique de base à savoir l'addition, la soustraction, la multiplication et la division, et ainsi à partir de ces opérations pouvoir créer une calculatrice en « polonaise inversée ».

FICHIER SOURCE : main.o

CONSIGNES DE FONCTIONNEMENT :

Trois requêtes seront posé concernant le fonctionnement de cette algorithme. La première est « Premier entiers signés », vous devrez saisir un nombre entiers positifs et appuyer sur la touche entrée. Une deuxième question vous sera alors demandé, « Deuxième entiers signés », c'est le même fonctionnement que la première question. La troisième requête est « Opération (+ , - , * , /, p) », vous devrez saisir le signe représentant l'opération arithmétique correspondant ou saisir « p » représentant la calculatrice « polonaise inversée » et saisir entrée. Le résultat de l'opération vous sera alors affiché.

ALGORITHME MIS EN OEUVRE :

J'ai mis en œuvre plusieurs fonctions pour pouvoir effectuer les opérations. Tout d'abord ceux qui permettant d'implémenter la liste: créer ma liste (CreerListe) , extraire la tête de liste (tete), insérer un élément dans la liste (insertTete) , passer à l'élément suivant (queue),
vérifier si la liste est vide (estVide) , afficher la liste (AfficheListe), déterminer le plus grande de deux liste (PlusGrand), déterminer la taille d'une liste (TailleListe) et insérer un élément en fin de liste.
J'ai aussi mis les structures permettant d'implémenter une pile pour l'exercice 2, reprenant les même fonctions que la liste puisqu'ils ont des structures identiques. J'ai aussi deux fonctions qui permettent d'inverser les éléments d'une liste et d'une pile inverse et

inversePile respectivement.
J'ai créer plusieurs fonctions qui sont utiles pour pouvoir effectuer les opérations :

Liste CreerLL() **:** Cette fonction permet de pouvoir créer la liste, de manière à ce que chaque cellule de la liste puisse contenir une nombre décimal de l'entier non signés.La fonction comprend que si la saisie de nombre n'est pas décimal ou soit un caractère, il ne sera pas mis dans la liste. Le cast permet ainsi de pouvoir traduire la saisie de l'utilisateur sous format int. Ainsi grâce au code ASCII, si l'utilisateur appuie sur entrée/Line Feed (10 sous ASCII), il pourra mettre fin à la saisie car il utilisera le saut à la ligne.

Liste Addition (Liste L1,Liste L2) : Cette fonction gère l'addition de deux listes d'entiers non signées. L'algorithme parcourt l'intégralité des deux listes et additionne la décimal de la liste 1 et de la liste 2. Si le résultat est supérieur à 10, on ajoute la retenu puis on passe à la cellule suivante des deux listes. Si l'une des deux listes est vide, l'algorithme poursuit avec celle non vide, jusqu'à ce qu'elle aussi ne soit pas vide et s'assure qu'il ne reste plus aucune retenue.

Liste Soustraction (Liste L1,Liste L2) : Il reprend le même principe que l'addition. Il parcourt l'intégralité des deux listes, et soustrait la décimal de la liste 1 à celle de la 2.Si le résultat est inférieur à 10, on ajoute 1 à la retenu est on soustrait avec le résultat de la soustraction suivante. On s'assure par la même méthode que l'addition, que les deux listes soit vide, en même temps ou non. Et on ajoute la retenue si il en reste une.

Liste Multiplication (Liste L1,Liste L2) : Cette fonction prend un élément de la liste L2 et le multiplie avec la totalité des éléments de la liste L1, puis il passe à la cellule suivante de sa liste. Si le résultat est supérieur ou égal à 10, il ajoute 1 à la variable retenu et continue en prenant le soin de bien utiliser la retenu stocké même si les calculs sont terminés. (Dans mon projet, le résultat de la multiplication est au format 10*1 et ne peut pas être supérieur à 100).

Liste Division(Liste L1, Liste L2) : Cette fonction permet de diviser le premier élément de la liste 1 et le premier élément de la liste 2. (Dans mon projet, je n'ai pas réussi à pouvoir faire la division de manière automatique sur l'ensemble des deux

listes).

Liste PolonaiseInverse(Liste NPI) : Cette fonction reprend toutes celles énoncées précédemment. On utilise une pile, on empile tous les éléments et on dépile un opérateur dans une Liste a et l'autre dans une liste 2, si un opérateur est vérifié dans la pile.On détermine si cette opération est une addition,soustraction, division ou une multiplication. Ainsi on utilsera l'une des fonctions créer en réponse à cette vérification.

DIFFICULTES RENCONTREES

J'ai rencontré plusieurs difficultés durant ce mini-projet. La première, est la gestion des grands entiers à savoir comment l'implémenter dans une liste sans avoir de problème au niveau de l'occupation mémoire, la deuxième est la gestion de la retenu, le fait de devoir gérer toute les contraintes et la troisième est le temps qui n'était que d'une semaine avec les autres épreuves de session 2, que je doit passer.