Listes Chaînées

Dans ce TP, vous allez utiliser les listes chaînées dans différentes configurations.

Implémentation d'une Liste simplement chaînée

Une liste chaînée est une implémentation pour le stockage d'une collection de données qui a comme avantage principal la flexibilité/adaptabilité en fonction du nombre de maillons de la chaîne.

Soit la structure suivante :

```
typedef struct maillon {
  int x ;
   struct maillon *suiv ;
} maillon_t ;
```

Implémenter les fonctions suivantes :

- maillon_t* ajouter_début(maillon_t *pliste, int val)
- maillon_t* ajouter_fin(maillon_t *pliste, int val)
- maillon_t* rechecher(maillon_t *pliste, int val)
- maillon_t* supprimer_debut(maillon_t *pliste, int *pval)
- maillon_t* supprimer_fin(maillon_t *pliste, int *pval)
- maillon_t* supprimer_val(maillon_t *pliste, int val)
- void afficher(maillon_t *pliste)

Exercice supplémentaire : rendre toutes vos fonctions génériques (ne plus avoir un maillon contenant juste un entier mais n'importe quel type de donnée)

Gestion d'un Historique Applicatif avec une liste doublement chaînée circulaire

Ecrire les fonctions qui permettraient de gérer l'historique d'une application quelconque ; on aura à disposition les fonctionnalités suivantes décrites comme suit :

1. Ajouter un élément à l'historique
2. Retirer un élément ; ceci devrait retourner le dernier élément rentré et le supprimer de la liste
3. Naviguer dans l'historique dans les deux sens (avancer et reculer)

Définir la structure de votre Historique afin d'optimiser les fonctionnalités décrites ci-dessus.

Addition d'Entiers Très Larges

Les langages peuvent gérer des entiers mais jusqu'à un certain niveau ; pour s'en rendre compte, essayer le code suivant :

```
int main()
{
   int i;
   int res = 10;

   for (i = 0; i < 1000; i++ ) {
    res *= 2;
    printf("i = %d : %d\n", i, res);
    if (res < 0) {
      printf("negatif\n");
      break;
    }
   }
```

```
    return 0;
}
```

Pour cela, il va falloir écrire une librairie spécialisée pour la gestion des nombres très larges.

Un solution serait de placer les nombres dans une liste chaînée, où chaque nœud contiendrait un nombre maximal prédéterminé de chiffres (dans notre cas, on prendra 5). Il convient après de trouver l'algorithme qui permettrait d'additionner deux entiers larges ainsi codés.

Pour bien faire, ayez toujours la lucidité de trouver l'algorithme général en vous basant sur un exemple fait à la main (par exemple : 96789543216 + 55555555555), puis transposez cet algorithme en code C.