


Institut Supérieur d'Informatique et de Mathématiques 	Année Universitaire : 2022-2023
	<p style="text-align: center;">Examen</p> Matière : Atelier de programmation 2 Filières : L1 Info Enseignant : Sakka Rouis Taoufik

Exercice 1: (2.5 p*8)

Soit la déclaration de la liste chaînée suivante :

```

struct cellule {
    int cle;
    struct cellule *suivant;

};
typedef struct {
    struct cellule * premier ;
    struct cellule * dernier ;
} liste ;
```

Implémenter les sous-programmes C suivants :

Indication : Vous pouvez utiliser les fonctions vues en cours sans les réimplémenter.

1- **struct cellule * AdresseElement (liste * L, int x)**

Cette fonction rend l'adresse de la première cellule portant la valeur x (NULL si $x \notin L$).

2- **struct cellule * AdresseSousListe (liste *L1, liste *L2)**

Cette fonction rend l'adresse du premier élément de la liste L2, si L2 est une sous liste de L1 (c'est-à-dire que L2 se trouve dans L1) ; NULL sinon.

3- **void SuppElement (liste * L, int x)**

Cette fonction supprime la première cellule contenant l'élément x dans la liste L.

4- **void SuppTElement (liste * L, int x)**

Cette fonction supprime toutes les cellules de la liste dont la clé contient l'entier x

5- **void ConcatListe (liste * L1, liste * L2, liste * L3)**

Cette fonction concatène les deux listes L1 et L2 dans une troisième liste L3.

6- **liste * FusionListe (liste * L1, liste * L2)**

Cette fonction permet de fusionner les deux listes **L1** et **L2** (supposées triées par ordre croissant) en une troisième liste triée.

7- **void InseretElement (liste * L, int x)**

Cette fonction insère (au bon endroit) l'élément x dans la liste **L** supposée déjà ordonnée par ordre croissant.

8- **void DecouperListe (liste * L, liste * L1, liste * L2, int (*Oper) (int))**

Cette fonction partitionne la liste **L** en deux listes : **L1** contient les éléments de L qui satisfont l'opération Oper (**Oper (du cle) > 0**), et **L2** contient les autres éléments.