PROGRAMMATION C
TD6

LICENCE MATHS-INFO
20 FÉVRIER 2012

Nous considérons ici des listes chaînées de nombres telles qu'introduites en cours :

```
typedef double OBJET;
typedef struct maillon {OBJET info;struct maillon *suiv;} MAILLON, *POINTEUR;

POINTEUR maillon(OBJET v, POINTEUR s){
    POINTEUR p;
    p = malloc(sizeof(MAILLON));
    if (!p) exit(EXIT_FAILURE);
    p->info = v;
    p->suiv = s;
    return p;
}
POINTEUR liste_vide(void){
    OBJET v;
    return maillon(v, NULL);
}
int est_vide(POINTEUR p){return !(p->suiv);}
void libere_maillon(POINTEUR p){free(p);}
```

Exercice 1 - recherche d'un élément

Écrivez une fonction qui prend en argument une valeur `x` de type `OBJET` et une liste, et rend l'adresse du premier maillon portant cette valeur (`NULL` s'il n'existe pas).
Prototype : `POINTEUR recherche_dans_liste(OBJET x, POINTEUR liste);`

Exercice 2 - concaténation de deux listes

Écrivez une fonction qui prend en argument deux variables `P1` et `P2` de type `POINTEUR` correspondant à deux listes et les concatène sans création de maillon. `P1` précèdera `P2`.
Prototype : `void concatene(POINTEUR P1, POINTEUR P2);`

Exercice 3 - suppression d'une valeur

Écrivez une fonction qui prend en argument une valeur `x` et une liste, supprime toutes les occurrences de cette valeur dans la liste puis renvoie le nombre de suppressions.
Prototype : `int supprime_valeur(OBJET x, POINTEUR liste);`

Exercice 4 - recopie d'une liste

Écrivez une fonction qui prend en argument une liste, la recopie et renvoie la copie.
Prototype : `POINTEUR recopie_liste(POINTEUR liste);`

Exercice 5 - inverse d'une liste par recopie

Écrivez une fonction qui prend en argument une liste, la recopie en inversant l'ordre des éléments et renvoie le résultat.
Prototype : `POINTEUR recopie_inverse_liste(POINTEUR liste);`

Exercice 6 - inverse d'une liste sans recopie

Écrivez une fonction qui prend en argument une liste, et la transforme en inversant l'ordre de ses éléments .
Prototype : `void inverse_liste(POINTEUR liste);`

Exercice 7 - tri d'une liste

Écrivez une fonction qui trie une liste par ordre croissant, sans création de nouveau maillon (sauf un maillon bidon). Utilisez le tri par insertion : parcourir la liste donnée et en insérer chaque maillon à sa place dans la liste triée.
Prototype : `void tri_liste(POINTEUR liste);`