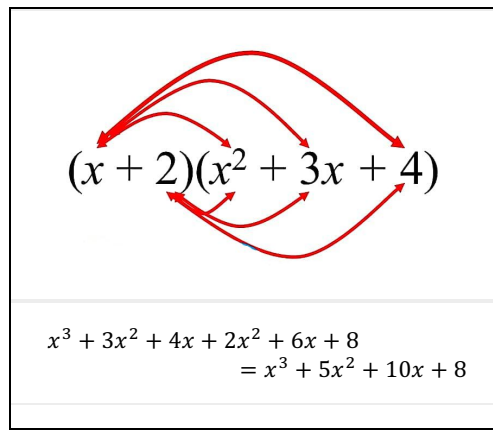


Exercice I: Question de TP (25 points)

Considérez le tp de polynômes. Ecrire la fonction **multiplier** qui, étant donné deux polynômes, renvoie un troisième polynôme représentant leur multiplication. Vous devez utiliser les types de données utilisés en tp. Tous les polynômes sont triés selon l'ordre décroissant des exposants. Consultez la figure 1 pour se rappeler de la multiplication polynomiale.



$$(x + 2)(x^2 + 3x + 4)$$

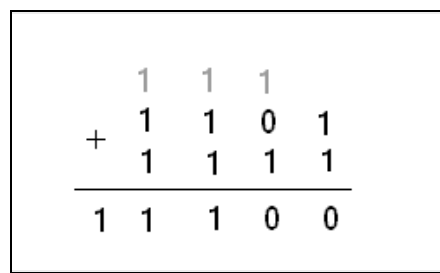
$$x^3 + 3x^2 + 4x + 2x^2 + 6x + 8$$

$$= x^3 + 5x^2 + 10x + 8$$

Figure 1

Exercice II: Ajout de deux grands nombres binaires (25 points)

Ecrire une fonction **add** qui, étant donné deux noms de fichiers texte, où chaque fichier contient un grand nombre binaire, représenté par une série de caractères '0' et '1', additionne les nombres binaires et écrit le résultat dans un troisième fichier texte dont le nom est la concaténation des deux noms de fichiers, séparés par le mot ADD. Par exemple, si le fichier "file1.txt" contient 1101 et le fichier "file2.txt" contient 1111, après avoir appelé la fonction **add** sur les deux noms de fichiers, nous devrions obtenir le fichier "file1ADDfile2.txt" contenant 11100.



$$\begin{array}{r} 1101 \\ + 1111 \\ \hline 11100 \end{array}$$

Figure 2

Exercice III: Aplatissement d'une liste chaînée (28 points)

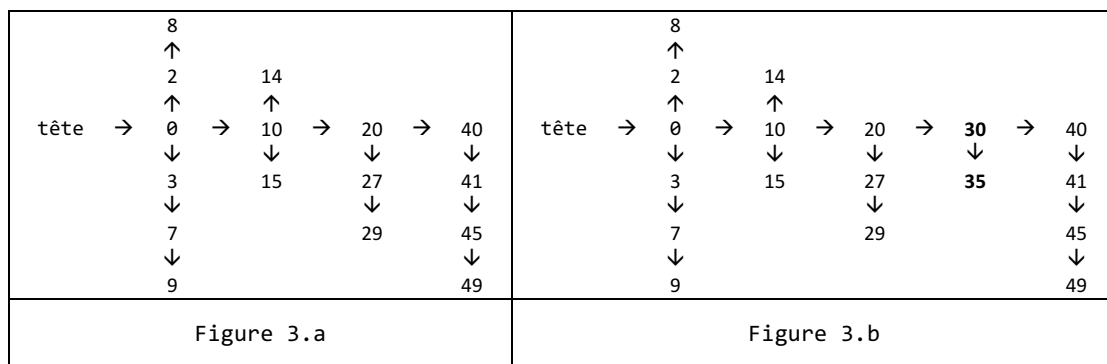
Étant donné une liste chaînée dont chaque nœud contient trois pointeurs de son type.

```
typedef struct noeud{
    int donnee;
    struct noeud *suivant, *hautPair, *basImpair;
} noeud;
```

Les nœuds principaux de la liste:

- leurs valeurs donnee sont des décimales arrondies à dixième,
- leur champ upEven pointe vers la liste des valeurs paires appartenant à la même plage de dixième, et
- le champ downOdd pointe vers la liste des valeurs impaires appartenant à la même plage de dixième.

Les nœuds dans les listes paires ont toujours leurs champs suivant et basImpair définis sur NULL. De même, les nœuds des listes impaires ont toujours leurs champs suivant et hautPair définis sur NULL. Toutes les listes liées sont triées. Voir l'exemple de la figure 3.a.



1. Écrire la fonction **insérer** qui prend un entier et le place à sa place dans cette structure de données. Notez que, si le nœud représentant la décimale arrondie au dixième de l'entier donné est manquant, la fonction doit d'abord ajouter la décimale arrondie à la structure, puis ajouter le nœud correspondant à l'entier donné. Exemple, si nous insérons 35 dans la liste de la figure 3.a, nous obtenons la liste de la figure 3.b.
2. Écrire la fonction **aplatir** qui aplatit la structure de données dans une liste chaînée triée. La solution fournie ne doit utiliser aucune allocation ou désallocation dynamique. De plus, tout en aplatissant, chaque nœud ne devrait être visité qu'une seule fois. Par exemple, pour la liste d'entrée ci-dessus dans la figure 3.a, la liste de sortie doit être :

0→2→3→7→8→9→10→14→15→20→27→29→40→41→45→49