Exercice 1
On a défini la classe `Maillon` et la variable `lst` par le code suivant :

```
class Maillon:
    def __init__(self, val, suiv=None):
        self.valeur = val
        self.suivant = suiv


lst = Maillon(10,Maillon(20))
```

1) Quelles sont les instructions permettant d'insérer un nouveau maillon, de valeur 15, « entre » les deux maillons existant, pour que la variable `lst` représente la chaîne 10 – 15 – 20 ?

2) Quelles sont les instruction permettant d'ajouter un nouveau maillon en fin de chaîne, avec pour valeur 25?

3) Ecrire une fonction `ajoute_a_la_fin(lst, v)` qui ajoute à la fin d'une chaîne de maillons `lst` , un maillon supplémentaire de valeur `v`.
a) avec une boucle while
b) avec une fonction récursive.

Exercice 2
Écrire une fonction `occurrences(x, lst)` qui renvoie le nombre d'occurrences d'une valeur `x` dans une liste chaînée `lst`.
a) avec une fonction récursive
b) avec une boucle while

Exercice 3
Écrire une fonction `insere_trie(x, lst)` qui prend en arguments un entier `x` et une liste chaînée `lst`, supposée triée par ordre croissant, et qui renvoie une nouvelle liste chaînée triée dans laquelle `x` est inséré à sa place.
Par exemple, insérer 3 dans la liste 1,2,5,8 renvoie une nouvelle liste 1,2,3,5,8.
On suggère d'écrire cette fonction de manière récursive.

Exercice 4
Écrire une fonction `tri_par_insertion(lst)` qui prend en argument une liste chaînée `lst`, et qui renvoie une nouvelle liste chaînée contenant toutes les valeurs de `lst` triées dans l'ordre croissant
On suggère d'écrire cette fonction de manière récursive, en faisant appel à l'exercice précédent !

Exercice 5
Écrire une fonction `identiques(l1, l2)` qui prend en arguments deux listes chaînées et qui renvoie un booléen indiquant si ces deux listes contiennent exactement les mêmes éléments dans le même ordre. On suppose qu'on peut comparer les éléments entre eux avec l'égalité `==` de Python.