

Exercice 1

```

1 def convertir(T):
2     """
3     T est un tableau d'entiers, dont les éléments sont 0 ou 1 et
4     représentant un entier écrit en binaire. Renvoie l'écriture
5     décimale de l'entier positif dont la représentation binaire
6     est donnée par le tableau T
7     """
8     s = 0
9     for i in range(len(T)):
10         s += T[-i-1]*2**i
11     return s

```

Élément	Ligne	Explication
def	1	Permet de créer une fonction
convertir	1	Nom de la fonction
T	1	Paramètre T
s = 0	8	On crée une variable qui sera la somme des bits
for	9	Permet de créer une boucle for
i in range(len(T))	9	La variable i va prendre les valeurs de 0 à la longueur de T-1. Ce qui donne [0, 1, 2, 3, ..., len(T)-2, len(T)-1,]
s +=	10	Permet d'incrémenter la variable s
T[-i-1]*	10	Ici on prend la valeur d'indice i en parcourant la liste par la fin. Le -1 sert à ce qu'on commence à la fin de la liste. Cette valeur va soit être 1 ou 0. Dans le cas où elle est égale à 0, la variable s ne va pas être incrémentée grâce au *.
2**i	10	Correspond aux puissances de 2 en fonction de l'emplacement du 1 dans la liste.
return s	11	Renvoie la somme s, qui correspond à la valeur décimale du nombre binaire T qui est sous forme de liste.

Exercice 2

```

1 def tri_insertion(L):
2     n = len(L)
3     # cas du tableau vide
4     if n == 0:
5         return L
6     for j in range(1,n):
7         e = L[j]
8         i = j
9         # A l'étape j, le sous-tableau L[0,j-1] est trié
10        # et on insère L[j] dans ce sous-tableau en déterminant 3 / 3
11        # le plus petit i tel que 0 <= i <= j et L[i-1] > L[j].
12        while i > 0 and L[i-1] > L[j]:
13            i = i - 1
14            # si i != j, on décale le sous tableau L[i,j-1] d'un cran
15            # vers la droite et on place L[j] en position i
16        if i != j:
17            for k in range(j,i,-1):
18                L[k] = L[k-1]
19            L[i] = e
20    return L

```

Éléments	Ligne	Explication
def tri_insertion(L)	1	On crée la fonction tri_insertion, qui prend en paramètre la liste L que l'on doit trier.
n = len(L)	2	On crée la variable n à laquelle on attribue la longueur de la liste L.
if n == 0:	4	On vérifie le cas de la liste vide
return L	5	On renvoie L, qui est vide.
for j in range(1, n):	6	Pour chaque indice dans la liste L.
e = L[j]	7	On attribue à e la valeur d'indice j de la liste L.
i = j	8	On définit la valeur de i à j
while i > 0 and L[i-1] > L[j]:	12	On vérifie si la valeur de i est strictement supérieure à 0 et si la valeur d'indice i-1 de L est strictement supérieur à celle d'indice j.
i = i - 1	13	On diminue la valeur de i jusqu'à ce que la valeur d'indice i soit le minimum de la sous-liste de 0 à j.
if i != j:	16	Si i n'est pas égal à j, c'est-à-dire, si la valeur d'indice i n'est pas la même que la valeur d'indice j.
for k in range(j,i,-1) : L[k] = L[k-1]	17 & 18	On décale tous les éléments de la sous-liste de i à j-1 d'une place vers la droite. Cela permet de remettre la liste L dans un « état correct ».
L[i] = e	19	On place l'élément e qui était d'indice j (celui qu'on voulait déplacer), à l'indice i, ce qui permet d'insérer la valeur au bon endroit.
return L	20	On renvoie la liste L.

Schéma explicatif :

La main gauche est la partie de la liste qui est triée.

La main droite est la partie non triée de la liste.

