EXERCICE 1 (4 points)

Cet exercice porte sur les langages et la programmation (récursivité).

1. Voici une fonction codée en Python :

```
def f(n):
    if n == 0:
        print("Partez!")
    else:
        print(n)
        f(n-1)
```

a. Qu'affiche la commande `f(5)` ?

b. Pourquoi dit-on de cette fonction qu'elle est récursive ?

2. On rappelle qu'en python l'opérateur + a le comportement suivant sur les chaînes de caractères :

```
>>> S = 'a'+'bc'
>>> S
'abc'
```

Et le comportement suivant sur les listes :

```
>>> L = ['a'] + ['b', 'c']
>>> L
['a', 'b', 'c']
```

On a besoin pour les questions suivantes de pouvoir ajouter une chaîne de caractères `s` en préfixe à chaque chaîne de caractères de la liste `liste`. On appellera cette fonction `ajouter`.

Par exemple, `ajouter("a",["b","c"])` doit retourner `["ab","ac"]`.

a. Recopiez le code suivant et complétez `.....` sur votre copie :

```
def ajouter(s, liste):
    res = []
    for m in liste:
        res.....
    return res
```

b. Que renvoie la commande `ajouter("b", ["a","b","c"])` ?

c. Que renvoie la commande `ajouter("a", [""])` ?

3. On s'intéresse ici à la fonction suivante écrite en Python où `s` est une chaîne de caractères et `n` un entier naturel.

```
def produit(s, n):
    if n == 0:
        return [""]
    else:
        res = []
        for i in range(len(s)):
            res = res + ajouter(s[i], produit(s, n-1))
        return res
```

a. Que renvoie la commande `produit("ab", 0)` ? Le résultat est-il une liste vide ?

b. Que renvoie la commande `produit("ab", 1)` ?

c. Que renvoie la commande `produit("ab", 2)` ?