Je pratique Python...

Exercice nº 1

Ecrire une fonction **multiplicationV1** prenant en paramètres deux nombres entiers POSITIFS **a** et **b**, et qui renvoie le produit de ces deux nombres.
L'usage de la touche * est interdite. Exemples :

```
>>> multiplicationV1(3,5)
15
>>> multiplicationV1(4,8)
32
>>> multiplicationV1(0,2)
0.0
```

Exercice nº 2

Écrire une fonction **moyenne** prenant en paramètres une liste d'entiers et qui renvoie la moyenne des valeurs de cette liste. Exemples :

```
>>> moyenne([10,20,30,40,60,110])
45.0
```

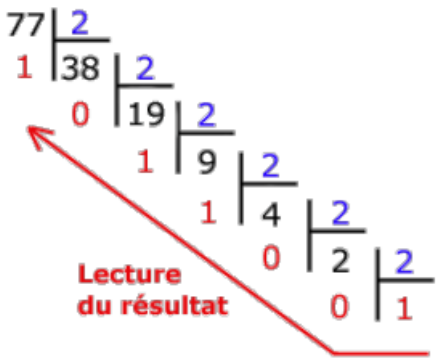
Exercice nº 3

Écrire une fonction python appelée **nb_repetitions** qui prend en paramètres un élément **elt** et une liste **tab** et renvoie le nombre de fois où l'élément apparaît dans la liste. Exemples :

```
>>> nb_repetitions(5,[2,5,3,5,6,9,5])
3
>>> nb_repetitions('A',[ 'B', 'A', 'B', 'A', 'R'])
2
```

Exercice nº 4

Pour rappel, la conversion d'un nombre entier positif en binaire peut s'effectuer à l'aide des divisions successives comme illustré ici :



En python, on obtient le reste de la division en utilisant : %, et le quotient en utilisant //. Exemple lors de la division de 78 par 2 :

```
>>> 77 % 2 #le reste
1
>>> 77 // 2 #le quotient
38
```

Voici une fonction python basée sur la méthode des divisions successives permettant de convertir un nombre entier positif en binaire :

```
def binaire(a):
    bin_a = str(...)
    a = a // 2
    while a ... :
        bin_a = ...(a%2) + ...
        a = ...
    return bin_a
```

Compléter la fonction binaire.

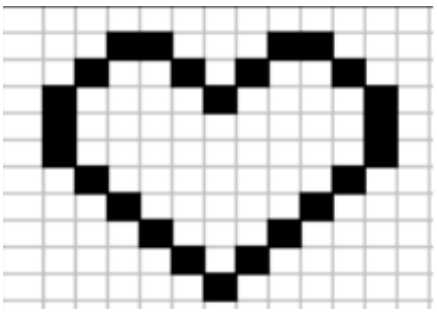
Exemples :

```
>>> binaire(0)
'0'
>>> binaire(77)
'1001101'
```

Exercice nº 5

On travaille sur des dessins en noir et blanc obtenu à partir de pixels noirs et blancs : La figure « cœur » ci-contre va servir d ?exemple. On la représente par une grille de nombres, c ?est-à-dire par une liste composée de sous-listes de mêmes longueurs.

Chaque sous-liste représentera donc une ligne du dessin.



Dans le code ci-dessous, la fonction affiche permet d'afficher le dessin. Les pixels noirs (1 dans la grille) seront représentés par le caractère "*" et les blancs (0 dans la grille) par deux espaces.

La fonction **zoomListe** prend en argument une liste **liste_depart** et un entier **k**. Elle renvoie une liste où chaque élément de **liste_depart** est dupliqué **k** fois. La fonction **zoomDessin** prend en argument la grille dessin et renvoie une grille où toutes les lignes de dessin sont zoomées **k** fois et répétées **k** fois. Soit le code ci-dessous :

```
coeur = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0],
         [0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0],
         [0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0],
         [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
         [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0],
         [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
         [0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0],
         [0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0],
         [0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0],
         [0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0],
         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
```

```python
def affiche(dessin):
    ''' affichage d'une grille : les 1 sont représentés par
        des "*" , les 0 par deux espaces "  " '''
    for ligne in dessin:
        for col in ligne:
            if col == 1:
                print(" *",end="")
            else:
                print("  ",end="")
        print()
```

```python
def zoomListe(liste_depart,k):
    '''renvoie une liste contenant k fois chaque
       élément de liste_depart
       exemple :
       >>> zoomListe([1,0],3)
       [1,1,1,0,0,0]'''
    liste_zoom = ...
    for elt in ... :
        for i in range(k):
            ...
    return liste_zoom



def zoomDessin(grille,k):
    '''renvoie une grille où les lignes sont zoomées k fois
       ET répétées k fois'''
    grille_zoom=[]
    for elt in grille:
        liste_zoom = ...
        for i in range(k):
            ... .append(...)
    return grille_zoom
```

```
>>> affiche(coeur)

       * *       * *
     *     *   *     *
   *         *         *
   *                   *
   *                   *
     *               *
       *           *
         *       *
           *   *
             *
```

```
>>> affiche(zoomDessin(coeur,3))

                    * * * * * *                         * * * * * *
                    * * * * * *                         * * * * * *
                    * * * * * *                         * * * * * *
              * * *             * * *         * * *               * * *
              * * *             * * *         * * *               * * *
              * * *             * * *         * * *               * * *
        * * *                             * * *                           * * *
        * * *                             * * *                           * * *
        * * *                             * * *                           * * *
        * * *                                                             * * *
        * * *                                                             * * *
        * * *                                                             * * *
        * * *                                                             * * *
        * * *                                                             * * *
        * * *                                                             * * *
              * * *                                                 * * *
              * * *                                                 * * *
              * * *                                                 * * *
                    * * *                                   * * *
                    * * *                                   * * *
                    * * *                                   * * *
                          * * *                     * * *
                          * * *                     * * *
                          * * *                     * * *
                                * * *         * * *
                                * * *         * * *
                                * * *         * * *
                                      * * *
                                      * * *
                                      * * *
```

Exercice n° 6

En urilisant l'exercice1, écrire une fonction **multiplicationV2** prenant en paramètres deux nombres entiers positifs ou négatifs. **a** et **b**, et qui renvoie le produit de ces deux nombres.
L'usage de la touche * est toujours interdite.

```
>>> multiplicationV2(-4,-8)
32
>>> multiplicationV2(-2,6)
-12
>>> multiplicationV2(-2,0)
0
```

Exercice n° 7

Le codage de César transforme un message en changeant chaque lettre en la décalant dans l'alphabet. Par exemple, avec un décalage de 3, le A se transforme en D, le B en E, ..., le X en A, le Y en B et le Z en C. Les autres caractères (?!?, ???...) ne sont pas codés. La fonction **position_alphabet** ci-dessous prend en paramètre un caractère **lettre** et renvoie la position de lettre dans la chaîne de caractères ALPHABET s'il s'y trouve et -1 sinon.
La fonction **cesar** prend en paramètre une chaîne de caractères **message** et un nombre entier **deca-**

lage et renvoie le nouveau message codé avec le codage de César utilisant le décalage **decalage**.

```
ALPHABET='ABCDEFGHIJKLMNOPQRSTUVWXYZ'

def position_alphabet(lettre):
        return ALPHABET.find(lettre)

def cesar(message, decalage):
        resultat = ''
        for ... in message :
                if lettre in ALPHABET :
                        indice = ( ... )%26
                        resultat = resultat + ALPHABET[...]
                else:
                        resultat = ...
        return resultat
```

1. Compléter la fonction **cesar** puis tester avec les exemples :

```
>>> cesar('BONJOUR A TOUS. VIVE LA MATIERE NSI !',4)
'FSRNSYV E XSYW. ZMZI PE QEXMIVI RWM !'
>>> cesar('WJIEJPM V OJPN. QDQZ GV HVODZMZ IND !',-21)
'BONJOUR A TOUS. VIVE LA MATIERE NSI !'
```

2. A l'aide du codage de César on a obtenu le message chiffré ci-dessous :

```
'IMWDDW WKL DS VAXXWJWFUW WFLJW VWUZAXXJWJ WL VWUJQHLWJ ?'
```

Retrouver le message d'origine.