

NOM Prénom :

1. Donner la valeur contenue dans la variable `val` après exécution et le type python du résultat.

instruction	valeur	type (python)
<code>val = 6//3</code>	2	int
<code>val = 6/3</code>	2.	float
<code>val = 7%2</code>	1 (reste de la division euclidienne de 7 par 2)	int
<code>val = 2*3.14</code>	6.28	float
<code>val = (4+3 > 5)</code>	True (7 > 5 est vrai)	bool
<code>val = 2*[1,3]</code>	[1,3,1,3]	list

2. Donner une expression booléenne testant si l'entier contenu dans la variable `n` est à la fois pair et non multiple de 51 : `(n%2==0) and not(n%51==0)`

3. On considère la liste `L = [ 2, 3, 5, 7, 9]`.

a) Écrire l'instruction permettant d'obtenir la longueur de la liste `L` : `len(L)`

b) Rappeler précisément la plage des indices admissibles pour la liste `L` :
les entiers de 0 inclus à la longueur de la liste exclu, ici les entiers 0, 1, 2, 3, 4

c) Donner la valeur de `L[1]` : valeur d'indice 1 ici 3

4. Donner l'instruction permettant d'initialiser la liste `C` à la liste des valeurs de k^2 pour k dans $\llbracket 2; 10 \rrbracket$.

On utilisera une définition par compréhension.

`C = [ k**2 for k in range(2,10+1) ]`

5. a) Importer le module `maths` de python sous l'alias `m` : `import maths as m`

b) Compléter la définition de la fonction

```
def cosinus(L) :  
    """ L : liste de flottants  
    renvoie la liste des cosinus des angles de la liste L """  
    Lcos=[ ]  
    for val in L : # parcours par valeur  
        res = m.cos(val)  
        # ajouter à la liste Lcos la valeur res  
        Lcos.append(res)  
    return Lcos
```

c) Écrire l'instruction permettant d'obtenir la documentation de la fonction `cosinus` : `help(cosinus)`

6. La fonction decrypt prend en argument une chaîne de caractères et renvoie la chaîne de caractères obtenue en ne conservant qu'une lettre sur trois à partir de la première lettre.

a) Que doit renvoyer l'appel decrypt("ovtkz") ? "ok" (type str)

b) Écrire la fonction decrypt (sans documentation)

```
def decrypt(message) :
    decode=""
    n=len(message) # nombre de caractères dans le message
    for k in range(n) : # parcours par indices
        if k % 3 == 0 :
            decode = decode + message[k]
    return decode
```

b) Écrire un tableau d'exécution pour l'appel decrypt("ovtky")

k	test k%3 == 0	decode
avant la boucle		""
0	vrai	"o"
1	faux	"o"
2	faux	"o"
3	vrai	"ok"
4	faux	"ok"

On renvoie bien "ok"

7. On définit une suite par récurrence par : $\forall n \in \mathbb{N}, u_{n+1} = \begin{cases} 2 \times u_n & \text{si } n \text{ multiple de } 3 \\ u_n - 10 & \text{si } n \text{ est du type } 3k + 1 \\ -\frac{1}{2} \times u_n & \text{sinon} \end{cases}$

Écrire une fonction U de paramètres p et a deux entiers, et qui renvoie la valeur de u_p de la suite définie précédemment avec $u_0 = a$.

```
def U(p,a) :
    """ p : int, entier naturel
    a : int, valeur initiale de la suite
    renvoie la valeur de u indice p """
    val = a
    for k in range(0,p) : # dernier passage k vat p-1
        # calcul de u_{k+1}
        if k %3 == 0 :
            val = val * 2
        elif k %3 == 1 :
            val = val -10
        else :
            val = -val/2
    return val
```