Exercice 1

Ecrire une fonction

1. à partir du script de la division euclidienne ci-dessous, écrire celui d'une fonction `division` qui retourne le quotient entier de `a` par `b`. Les paramètres de la fonction seront `a` et `b`.

```
a = int(input('entrer la valeur de a :'))
b = int(input('entrer la valeur de b :'))
N = 0
...
...
...
print(N)
```

2. Ecrire l'instruction qui va *appeler* la fonction `division`, avec les arguments `a = 2024` et `b = 7`.

Exercice 2

Utiliser une fonction

La fonction `pourcentage` calcule le pourcentage d'une valeur X et retourne le résultat.

2.1 Compléter le script de la fonction

```
def pourcentage(P, X):
    """
    P: int, float: pourcentage de reduction
    par exemple: 30 signifie 30%
    X: int, float: valeur
    """
    return ... ...
```

Aide : Pour calculer 30% de 54, on pose l'opération : 30/100*54

2.2 Place des arguments

- Ecrire l'instruction qui appelle la fonction `pourcentage` avec les arguments 15 et 90.
- Comment appeler la fonction pour obtenir 50% de 2100 ?

Exercice 3

Valeur de retour d'une fonction

Pour la carte microbit utilisant le capteur de lunimosité :

```
def intensite(lumi):
    """retourne une valeur de 0 a 9
    9 <=> lumi = 0
    0 <=> lumi = 255
    """
    return int((255 - lumi)/255*9)
```

3.1 Les lignes entre les symboles """ ... """, sont-elles executées ? Quel est leur rôle ?

3.2 Qu'est ce qui est affiché lorsque l'on fait :

```
I = intensite(100)
print(I)
```

Exercice 4

Utiliser une structure conditionnelle dans une fonction

Pour les soldes, le prix d'un article est réduit de 30%.

4.1 Fonction `solde_30`

Ecrire le script d'une fonction appelée `solde_30`, qui retourne le prix d'un article mis en argument.

4.2 Structure conditionnelle

Le calcul du nouveau prix se fait maintenant de la manière suivante : si l'article fait strictement moins de 30 euros, il n'y a pas de redution. Si l'article fait entre 30 euros et 100 euros, il y a une reduction de 30%. Au délà, la reduction est fixe, d'un montant de 30 euros.

Ecrire le nouveau script pour la fonction `solde_30`, qui retourne le nouveau prix en fonction de la valeur mise en argument.

4.3 Appel de la fonction

Ecrire l'instruction qui calcule le prix pour un article de 120 euros.

Exercice 5

Volume d'une sphère

1. Écrire une fonction cube qui retourne le cube de son argument.
2. Écrire une fonction `volumeSphere` qui calcule le volume d'une sphère de rayon r fourni en argument et qui utilise la fonction cube.

Donnée :

$$V = \frac{4}{3} \times \pi \times R^3$$

3. Calculer le volume en cm^3 d'une sphere de rayon 10cm (à l'aide de la fonction `volumeSphere`)

Exercice 6

Paramètres d'une fonction

Une société est en train de créer un programme qui permet aux clients d'enregistrer le nombre de kilomètres parcourus. Le programme enverra des messages en fonction du nombre de miles consignés par le client. Vous créez le code Python suivant.

```
01 def ...
02     name = input("quel est votre nom ?")
03     return name
04
05 def ...
06     calories = mile * calories_per_mile
07     return calories
08
09 distance = int(input("Combien de miles avez vous fait cette
       semaine?")
10 burn_rate = 50
11 biker = get_name()
12 calories_burned = calc_calories(distance,burn_rate)
13 print(biker, " vous avez brulé ", str(calcories_burned), "
       calories")
```

Vous devez définir les deux fonctions requises.

Quels segments de code devez-vous utiliser pour les lignes 01 et 04?

A. `def get_name () :`

B. `def get_name (biker) :`

C. `def get_name (name) :`

D. `def calc_calories ():`

E. `def calc_calories (miles, burn_rate):`

F. `def calc_calories (miles, calories_per_mile):`

Exercice 7

Python turtle

La fonction suivante s'utilise avec le module *turtle* et permet de dessiner un rectangle.

```
def rectangle(a,b):
    t.forward(a)
    t.left(90)
    t.forward(b)
    t.left(90)
    t.forward(a)
    t.left(90)
    t.forward(b)
    t.left(90)
```

7.1 Boucle dans une fonction

Ré-écrire `rectangle` avec le moins de lignes possibles. Utiliser une boucle bornée.

7.2 Appel de fonction

Ecrire l'instruction qui appelle cette fonction pour dessiner un rectangle de côté 100 * 200 pixels.

7.3 Rectangles emboités

Ecrire avec le moins de lignes possibles une série d'instructions qui dessine des rectangles de plus en plus gros, en partant du même point initial, avec un des côtés qui augmente régulièrement de 10 pixels, et l'autre côté qui augmente de 20 pixels. Le premier rectangle fait 10 * 20 et le dernier 100 * 200. Utiliser une boucle bornée ainsi que la fonction `rectangle` de l'exercice précédent.

Aide : dans la boucle, on fera :

```
a = a + 10
b = b + 20
```

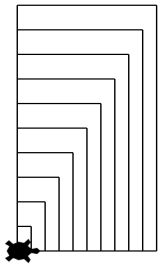

FIGURE 1 – rectangles emboités

Exercice 8

Fonction et séquences (Term NSI)

8.1 Rappels sur les séquences

Rappeler le rôle des méthodes de liste à partir des exemples suivants :

```
>>> L = ["A","B","C"]
>>> L.append("D")
>>> print(L)
# affiche ...
>>> x = L.pop()
>>> y = L.pop()
>>> print(x, y, L)
# affiche ...
>>> for i in range(len(L)):
>>> .... print(i)
# affiche ...
```

8.2 Retournement d'une liste

A partir du *docstring* de la fonction, compléter le script de la fonction `retourne_liste`.

```
def retourne_liste(L):
    """cree une liste dont les elements sont dans l'ordre
    inverse de la liste d'origine
    @param:
    L: list
    @return:
    L_inverse: list, contient les elements de L mis dans l'ordre
   inverse
    @exemple:
    >>> retourne_liste(["A","B","C"])
    ['C', 'B', 'A']
    """
    L_inverse = []
    for i in .. ...  ... ... ... :
        elem = ... ...
        L_inverse... ... ...
    return L_inverse
```

Exercice 9

TP : force d'un mot de passe (Term NSI)

prerequis : parcours d'une chaine de caractères avec une boucle bornée

- `c.islower()` : teste si le caractère c est en minuscules.
- `c.isupper()` : teste si le caractère c est en majuscules.
- `c.isalpha()` : teste si le caractère c est une lettre
- `c.isdigit()` : teste si le caractère c est un chiffre

Un administrateur d'un site web veut assurer un maximum de sécurité pour les utilisateurs du site. Pour ceci il décide de réaliser une application qui évalue la force des mots de passe des différents utilisateurs du site, sachant qu'un mot de passe est une chaîne de caractères qui ne comporte pas d'espaces et de lettres accentuées.

La force d'un mot de passe varie, selon la valeur d'un score calculé, de 'Très faible' jusqu'à 'Très fort' :

- Si le score <20, la force du mot de passe est 'Très faible'
- Sinon si le score<40, la force d'un mot de passe est 'Faible'
- Sinon si le score <80, la force du mot de passe est 'Fort'
- Sinon la force du mot de passe est 'Très fort'

Le score se calcule en additionnant des bonus et en retranchant d'eventuelles pénalités (niveau difficile).

Les bonus attribués sont :

- Nombre total de caractères * 4
- (Nombre total de caractères – nombre de lettres majuscules) * 2
- (Nombre total de caractères – nombre de lettres minuscules) * 3
- Nombre de caractères non alphabétiques * 5

Exemple

Pour le mot de passe 'P@cSI_promo2017', le score se calcule comme suit :

- La somme de bonus = `15*4 + (15-3) *2 + (15-6) *3+6*5=141`
- Le nombre total de caractères = 15
- Le nombre de lettres majuscules = 3
- Le nombre de lettres minuscules=6
- Le nombre de caractères non alphabétiques =6
- Le score final = 141 ; puisque 141>80 alors le mot de passe est considéré comme 'Très fort'

9.1 Travail demandé

1. Ecrire une fonction NbCMin(p) qui retourne le nombre de caractères minuscules.
2. Ecrire une fonction NbCMaj(p) qui retourne le nombre de caractères majuscules.
3. Ecrire une fonction NbCAlpha(p) qui retourne le nombre de caractères non alphabétiques.
4. Ecrire une fonction Score(p) qui affiche le score d'un mot de passe

9.2 Travail demandé : niveau difficile

Les pénalités imposées sont :

- La longueur de la plus longue séquence de lettres minuscules * 2
- La longueur de la plus longue séquence de lettres majuscules * 3

Pour le mot de passe 'P@cSI_promo2017', le score se calcule comme suit :

- La somme de bonus = `15*4 + (15-3) *2 + (15-6) *3+6*5=141`
- Le nombre total de caractères = 15
- Le nombre de lettres majuscules = 3
- Le nombre de lettres minuscules=6
- Le nombre de caractères non alphabétiques =6
- La longueur de la plus longue séquence de lettres minuscules('promo') =5
- La longueur de la plus longue séquence de lettres majuscules('SI') =2
- La somme des pénalités = 5*2+2*3=16
- Le score final = 141-16=125 ; puisque 125>80 alors le mot de passe est considéré comme 'Très fort'

1. Ecrire une fonction NbCMin(p) qui retourne le nombre de caractères minuscules.
2. Ecrire une fonction NbCMaj(p) qui retourne le nombre de caractères majuscules.
3. Ecrire une fonction NbCAlpha(p) qui retourne le nombre de caractères non alphabétiques.
4. Ecrire une fonction LongMaj(p) retourne la longueur de la plus longue séquence de lettres majuscules.
5. Ecrire une fonction LongMin(p) retourne la longueur de la plus longue séquence de lettres minuscules.
6. Ecrire une fonction Score(p) qui affiche le score d'un mot de passe

Source :https ://developpement-informatique.com