+1/1/60+

Programmation impérative en Python — SPUF21

Année 2022-2023 — Partiel

Nom : .....

Prénom : .....

Numéro d'étudiant : .....

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9
☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

Durée : 2 heures.

Aucun document n'est autorisé. L'usage de la calculatrice ou de tout autre appareil électronique est interdit.

Les exercices sont indépendants. Au sein d'un même exercice, vous pouvez utiliser les variables et fonctions des questions précédentes, même si vous n'avez pas su les faire ; chaque question est donc indépendante.

À part les méthodes et fonctions de base, vous n'avez pas le droit d'utiliser les fonctions et les méthodes « avancées », sauf si l'énoncé vous conseille l'utilisation de certaines d'entre elles.

```python
# Fonctions autorisées
len(…)
range(…)
print(…)

# Méthode autorisée
L.append(x)
```

```python
# Par exemple les méthodes et fonctions suivantes sont entre autres interdites
max(…) min(…) sum(…) abs(…)
s.split(…) s.index(…) L.extend(…)

# Vous n'avez pas le droit d'utiliser des compréhensions ou des slices
# ou de tester l'appartenance à une séquence avec le mot-clé « in ».
# À la place vous devez utiliser des boucles.
[ x for x in range(L) ]
chaine[début:fin:pas]
x in L
```

Exercice 1 Trois petits exercices . . . . . 4 points

☐0 ☐0,5 ☐1 ☐1,5 ☐2 ☐2,5 ☐3 ☐3,5 ☐4

1. Écrire la fonction `suite(n)` qui calcule et renvoie la valeur de u_n où $(u_n)_{n\in\mathbb{N}}$ est la suite définie par récurrence par $u_0 = 12$ et $u_{n+1} = 3 \times u_n + 5$.

```
def suite(n):
    if n==0:
        return 12
    else:
        return 3*suite(n-1) + 5
```

2. Écrire une fonction `affichage(n)` qui affiche les nombres de 0 à n (compris) avec un pas de 0,25.

```
>>> affichage(1.5)
0.0  0.25  0.5  0.75  1.0  1.25  1.5
```

```
def affichage(n):
    x=0.0       # Il faut utiliser une boucle while
    while x<n: # car le for n'autorise que des pas entiers.
        print(x,end=" ") # L'affichage sur une ligne n'est pas demandé
        x = x+0.25
    print(x)
```

3. Écrire une procédure `absolue(liste)` qui modifie une liste en remplaçant chacun de ses termes par sa valeur absolue. Il est bien évidemment interdit d'utiliser la fonction `abs`.

```
>>> L
[1, -3, -1.5, 7]
>>> absolue(L)
>>> L
[1, 3, 1.5, 7]
```

```
def absolue(L):
    for i in range(len(L)):
        if L[i]<0:
            L[i] = -L[i]
```

Exercice 2 Inflation et révolution . . . . . 3,5 points

☐0 ☐0,5 ☐1 ☐1,5 ☐2 ☐2,5 ☐3 ☐3,5

1. Écrire une fonction `inflation_futur(année,n)` qui prend en arguments deux entiers et affiche `n` lignes commençant par le numéro de l'année et se terminant alternativement par +3% ou +5% comme dans l'exemple ci-dessous.

```
>>> inflation_futur(2023,4)
2023 +3%
2024 +5%
2025 +3%
2026 +5%
```

```
def inflation_futur(année,n):
    for i in range(n):
        if i%2==0:
            print(année+i,'+3%')
        else:
            print(année+i,'+5%')
```

2. Dans cette question, l'inflation est de 15% par an. Des experts prévoient une nouvelle révolution quand le prix de la baguette dépassera les 2€. Écrire une fonction `révolution(année,pain)` qui prend en argument une année avec le prix du pain correspondant et affiche pour chaque année le nouveau prix (avec l'augmentation de 15% par an). On utilisera une boucle `while` et on s'arrêtera lorsque le prix dépassera les 2€ en affichant l'année du début de la révolution comme dans l'exemple ci-dessous.

```
>>> révolution(2023,1.2) # en 2023 la baguette coûte 1,2€
2024 prix du pain : 1.38 €
2025 prix du pain : 1.5869999999999997 €
2026 prix du pain : 1.8250499999999996 €
2027 prix du pain : 2.0988074999999995 €
2027 année de la révolution !
```

```
def révolution(année,pain):
    i=0
    prix=pain
    while prix<2:
        prix = prix * (1+15/100)
        i = i+1
        print(année+i, "prix du pain :", prix, "€")
    print(année+i, "année de la révolution !")
```

Exercice 3 L'art de l'espionnage . . . . . 4,5 points

☐0 ☐0,5 ☐1 ☐1,5 ☐2 ☐2,5 ☐3 ☐3,5 ☐4 ☐4,5

Vous venez d'être embauché comme espion. Votre mission, si vous l'acceptez, est d'écrire une fonction permettant automatiquement de déchiffrer un message TOP SECRET. Pour vous aider dans ce décryptage, on vous fournit sous forme de chaîne la liste de toutes les majuscules du français, définie de manière globale.

```
MAJ = "ABCDEFGHIJKLMNOPQRSTUVWXYZÀÂÄÉÈÊËÎÏÔÖÙÛÜŸÇŒÆ"
```

1. Écrire une fonction `est_majuscule(car)` qui prend en argument un caractère `car` et renvoie `True` si `car` est présent dans la chaîne `MAJ` ou si `car` correspond au symbole `"_"`. Sinon, la fonction renverra `False`. On utilisera la variable `MAJ` sans la redéfinir. Il est interdit d'utiliser le mot-clé `in` pour tester l'appartenance (`car in MAJ`) ce qui serait bien trop facile.

```
def est_majuscule(c):
    if c=='_':
        return True
    else:
        for i in range(len(MAJ)):
            if MAJ[i]==c:
                return True
    return False
```

2. Écrire une fonction `extraire(message)` qui prend une chaîne de caractères `message` et renvoie une nouvelle chaîne ne contenant que les majuscules (et le symbole `"_"`). On pourra utiliser la fonction de la question précédente.

```
>>> extraire("oN voIt lA MEDiocre_Ànesse !")
'NIAMED_À'
```

```
def extraire(message):
    réponse = ""
    for i in range(len(message)):
        if est_majuscule(message[i]):
            réponse = réponse + message[i]
    return réponse
```

3. On y est presque ! Écrivez maintenant une fonction `déchiffrer(message)` qui en utilisant la fonction précédente, affiche les lettres à l'envers et remplace les `"_"` par des espaces.

```
>>> déchiffrer("oN voIt lA MEDiocre_Ànesse !")
À DEMAIN
```

```
def déchiffrer(message):
    secret = extraire(message)
    n = len(secret)
    for i in range(n-1, -1, -1):
        if secret[i] == "_":
            print(" ", end='')
        else:
            print(secret[i], end='')
    print()
```

Exercice 4 Rugby : France — Angleterre (première partie) . . . . . 4,5 points

☐0 ☐0,5 ☐1 ☐1,5 ☐2 ☐2,5 ☐3 ☐3,5 ☐4 ☐4,5

Lors du Tournoi des 6 Nations au rugby, l'équipe de France a massacré les Anglais, chez eux, 53 à 10. Pour fêter cette victoire historique contre l'ennemi héréditaire, nous allons consacrer l'exercice au rugby.

On représente les résultats des matchs Angleterre-France par des quintuplets de la forme `(année, pays1, p1, p2, pays2)`. Par exemple, les trois derniers résultats sont :

- `(2023,"An",10,50,"Fr")` victoire 50 à 10 de la France en 2023
- `(2022,"Fr",25,13,"An")` victoire 25 à 13 de la France en 2022
- `(2021,"An",23,20,"Fr")` victoire 23 à 20 de l'Angleterre en 2021

1. Écrire une fonction `cocorico(résultats)` qui prend en argument une liste de matchs et renvoie la liste des années victorieuses de la France.

```
def cocorico(résultats):
    V = []
    for i in range(len(résultats)):
        (année, pays1, n1, n2, pays2) = résultats[i]
        if pays1 == "Fr" and n1>n2:
            V.append(année)
        elif pays2 == "Fr" and n2>n1:
            V.append(année)
    return V
```

— Voici les différentes actions qui au rugby permettent de marquer des points (chacune sera codée par un unique caractère, indiqué entre guillemets).

- `"E"` : essai (5 points) ;
- `"T"` : transformation (2 points) ;
- `"P"` : pénalité (3 points) ;
- `"D"` : *drop* (3 points).

— Le déroulé d'un match sera codé par une liste de couples de chaînes `(équipe, action)`. Par exemple :

```
[("Fr","E"), ("Fr","T"), ("An","P"), ("Fr","E"), ("Fr","T")]
```

Dans ce match, la France a commencé par marquer un essai, puis une transformation. L'Angleterre a marqué une pénalité, puis la France a de nouveau marqué un essai et une transformation.

2. Écrire une fonction `nombre_essai(match,équipe)` qui renvoie le nombre d'essais marqués par une équipe pendant un match.

```
>>> M=[("Fr","E"), ("Fr", "T"), ("An", "P"), ("Fr","E"), ("Fr", "T")]
>>> nombre_essai(M,'Fr')
2
>>> nombre_essai(M,'An')
0
```

```
def nombre_essai(match,équipe):
    n=0
    for i in range(len(match)):
        (éq,action) = match[i]
        if éq==équipe and action=="E":
            n=n+1
    return n
```

3. Écrire une fonction `score(match,équipe)` prenant en argument un match et un nom d'équipe et qui renvoie le nombre de points marqués par l'équipe.

```
>>> score(M,'Fr') # 2 essais + 2 transformations : 5 + 2 + 5 + 2 = 14 points
14
>>> score(M,'An') # 1 seule pénalité : 3 points
3
```

```
def score(match,équipe):
    points=0
    for i in range(len(match)):
        (éq, action) = match[i]:
        if éq == équipe:
            if action == "E":
               points = points + 5
            elif action == "T":
               points = points + 2
            else:
               points = points + 3
    return points
```

Exercice 5 Rugby : France — Angleterre (suite et fin) . . . . . 3,5 points

☐0 ☐0,5 ☐1 ☐1,5 ☐2 ☐2,5 ☐3 ☐3,5

Cet exercice est la suite du précédent. Toutes les fonctions de l'exercice précédent pourront être utilisées dans les questions qui suivent.

1. En rugby, une transformation ne peut être marquée que juste après un essai de la même équipe. Écrire une fonction `valide(match)` qui renvoie `True` si la règle est bien respectée et `False` sinon. Dans les trois exemples ci-dessous, le second renvoie Faux car la deuxième transformation `"T"` de la France n'est pas précédée d'un essai. Le dernier exemple est lui aussi faux car la transformation anglaise suit un essai français.

```
>>> valide([("Fr", "E"), ("An", "P"), ("Fr", "E"), ("Fr", "T")])
True
>>> valide([("Fr", "E"), ("Fr", "T"), ("An", "P"), ("Fr", "T")])
False
>>> valide([("An", "P"), ("Fr", "E"), ("An", "T"), ("An", "P")])
False
```

```
def valide(match):
    if len(match) == 0:
        return True
    if match[0][1] == "T":
        return False
    for i in range(1,len(match)):
        (équipe, action) = match[i]
        (équipe_précédente, action_précédente) = match[i-1]
        if action == "T" and action_précédente != "E":
            return False
        if action == "T" and équipe_précédente != équipe:
            return False
    return True
```

2. Dans le tournoi, à l'issue de chaque match, on attribue des points pour le classement des équipes :

   — en cas de victoire : 4 points,
   — en cas d'égalité : 2 points ;
   — en cas de défaite : 0 point.

À cela s'ajoutent deux éventuels bonus.

   — si l'équipe a marqué 4 essais ou plus : bonus offensif +1 point.
   — si l'équipe perdante perd de 7 points ou moins : bonus défensif + 1 point.

Écrire une fonction `nombre_points(match)` qui prend en argument un match France-Angleterre et renvoie le nombre de points gagnés par la France pour le classement des équipes à la suite de ce match.

```
def nombre_points(match):
    s1 = score(match,"Fr")
    s2 = score(match,"An")
    points=0
    if s1>s2: # victoire
        points = points+4
    elif s1==s2: # nul
        points = points+2
    elif s2-s1<=7: # bonus défensif
        points = points+1
    if nombre_essai(match,"Fr")>=4: # bonus offensif
        points = points+1
    reutrn points
```

3. Nous pourrions écrire une fonction qui à partir de tous les matchs du tournoi nous donnerait le gagnant, mais la première place de l'édition 2023 étant pour l'Irlande et non la France, il ne nous a pas semblé pertinent de vous faire écrire cette fonction…