Spécialité NSI Terminale
Contrôle n° 1
Sujet A
Lundi 10 octobre 2022

Exercice 1 (10 points)

Thème : Exécution de programmes, recherche et corrections de bugs
Les questions proposées sont indépendantes les unes des autres.

1. On considère la fonction `somme(n)` qui reçoit en paramètre un entier `n` strictement positif et renvoie le résultat du calcul $1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}$.

```
def somme(n) :
    total = 0
    for i in range(n) :
        total = total + 1/i
    return total
```

Lors de l'exécution de `somme(10)`, le message d'erreur "`ZeroDivisionError: division by zero`" apparait. Identifier le problème et corriger la fonction pour qu'elle effectue le calcul demandé.

2. On considère la fonction `maxi(L)` qui prend comme paramètre une liste `L` de nombres et renvoie le plus grand nombre de cette liste :

```
def maxi(L) :
    indice = 0
    maximum = 0
    while indice <= len(L) :
        if L[indice] > maximum :
            maximum = L[indice]
        indice = indice + 1
    return maximum
```

a. Lors de l'exécution de `maxi([2, 4, 9, 1])` une erreur est déclenchée. Identifier et corriger le problème.

b. Le bug précèdent est maintenant corrigé. Que renvoie à présent l'exécution de `maxi([-2, -7, -3])` ? Modifier la fonction pour qu'elle renvoie le bon résultat.

3. On souhaite réaliser une fonction qui génère une liste de `n` joueurs identifiés par leur numéro. Par exemple on souhaite que l'appel `genere(3)` renvoie la liste `['Joueur 1', 'Joueur 2', 'Joueur 3']`.

```
def genere(n) :
    L = []
    for i in range(1, n+1) :
        L.append('Joueur '+i)
    return L
```

L'appel `genere(3)` déclenche l'erreur suivante : `TypeError: can only concatenate str (not "int") to str`.
Expliquer ce message d'erreur et corriger la fonction afin de régler le problème.

4. On considère la fonction `suite(n)` qui reçoit un entier positif et renvoie un entier.

```
def suite(n) :
    if n == 0 :
        return 0
    else :
        return 3+2*suite(n-2)
```

a. Quelle valeur renvoie l'appel de `suite(6)` ?

b. Que se passe-t-il si on exécute `suite(7)` ?

5. On considère le code Python ci-dessous :

```
x = 4
L = []
def modif(x, L) :
    x = x + 1
    L.append(2*x)
    return x, L

print(modif(x, L))
print(x, L)
```

a. Qu'affiche le premier `print` ?

b. Qu'affiche le second `print` ?

Exercice 2 (10 points)

On appelle palindrome un texte dont l'ordre des lettres reste le même, qu'on le lise de la droite vers la gauche ou de la gauche vers la droite.
Par exemple, un mot à une lettre est un palindrome, "BOB" est un palindrome, tout comme "LAVAL". Le mot "" est considéré comme un palindrome.
On souhaite programmer une fonction `palindrome` qui prend en paramètre une chaîne de caractères `txt`. Cette fonction renvoie `True` si la chaîne de caractères est un palindrome, `False` sinon.

On rappelle que :

- La fonction `len` prend une chaîne de caractères en paramètre et renvoie sa longueur, c'est-à-dire le nombre de lettres constituant la chaîne de caractères.
  Exemple : `len("bonjour")` vaut 7.
- Si `txt` est une chaîne de caractères `txt[i]` est la lettre de `txt` d'indice `i`. Attention, la première lettre a pour indice `0`.
  Exemple : si `txt = "bonjour"` alors `txt[2]` désigne `"n"`.

1. On donne ci-dessous une implémentation récursive incomplète pour la fonction `palindrome`. L'opération + à la ligne 8 permet de concaténer deux chaînes de caractères.
Exemple : Si `txt1 = "bon"` et `txt2 = "jour"`, l'instruction `txt1 + txt2` renvoie la chaîne de caractères `"bonjour"`.

```
def palindrome(txt) :
    """ Str -> Bool """
    ...
    ...
    taille = len(txt)
    interieur = ""
```

```
    for i in range(1, taille - 1) :
        interieur = interieur + txt[i]
    return (txt[0] == txt[taille - 1] and \
            (palindrome(interieur))
```

(a) Choisir parmi les propositions ci-dessous celle qui convient pour compléter la fonction `palindrome` (ligne 3 et 4).

Proposition 1 :

```
if len(txt) <= 2 :
    return True
```

Proposition 2 :

```
if len(txt) < 2 :
    return False
```

Proposition 3 :

```
if len(txt) < 2 :
    return True
```

Proposition 4 :

```
if len(txt) <= 2 :
    return False
```

(b) Lors de l'appel de `palindrome("bonjour")` indiquer les valeurs , aux lignes 9 et 10, de : `txt[0]`, `txt[taille - 1]` et `interieur`.

2. Proposer deux tests pour cette fonction qui permettent de tester deux cas de figure différents en justifiant ce choix

3. Écrire une version non récursive de la fonction `palindrome`.

4. On étudie dans cette question des chaînes de caractères utilisant uniquement les lettres `"A"`, `"T"`, `"C"` et `"G"`.

Exemple : `"AA"`, `"CAT"` et `"CCGATACG"`.

On associe à chacune de ces lettres une autre lettre appelée lettre complémentaire selon le tableau suivant :

Lettre	`"A"`	`"T"`	`"G"`	`"C"`
Lettre complémentaire	`"T"`	`"A"`	`"C"`	`"G"`

On obtient le complémentaire d'un mot en remplaçant chacune de ses lettres par sa lettre complémentaire.

Exemple : Le complémentaire à `"GAATTC"` est `"CTTAAG"`.

a. Écrire une fonction en Python, nommée `complementaire`, qui prend en paramètre une chaîne de caractère `txt` écrite uniquement avec les lettres `"A"`, `"C"`, `"G"` et `"T"`. Cette fonction renvoie la chaîne de caractères complémentaire de `txt`.

b. Une chaîne de caractères `txt` est dites palindromique si la concaténation de `txt` avec son complémentaire est un palindrome.

Exemples :

* `"GAATTC"` est palindromique car `"GAATTC" + "CTTAAG" = "GAATTCCTTAAG"` est un palindrome.
* `"GAAT"` n'est pas palindromique car `"GAAT" + "CTTA" = "GAATCTTA"` n'est pas un palindrome.

Déterminer si la chaîne de caractère `"GATCGT"` est palindromique.

c. Écrire une fonction `est_palindromique` prenant comme paramètre une chaîne de caractères `txt`. Cette fonction doit renvoyer `True` s'il s'agit d'une séquence palindromique, `False` sinon.

Remarque : Le code de la fonction `est_palindromique` pourra utiliser les fonctions `palindrome` et `complementaire` décrites dans les questions précédentes.