
Evaluation 1^{ère} NSI

Date : Vendredi 8 Novembre 2024

Durée de l'épreuve : 01h 00

Calculatrice non autorisée

Question 1 (1point)

On considère le script suivant :

```
t = [2, 8, 9, 2]
```

```
t[2] = t[2] + 5
```

Quelle est la valeur de t à la fin de son exécution ?

Réponses

A [2, 13, 9, 2]

B [2, 8, 14, 2]

C [7, 13, 14, 7]

D [7, 13, 9, 2]

Question 2 (1point)

On considère la liste de listes suivante :

```
tictactoe = [ ['X', 'O', 'O'],  
              ['O', 'O', 'O'],  
              ['O', 'O', 'X']]
```

Quelle instruction permet d'obtenir une diagonale de 'X' ?

Réponses

A tictactoe[3] = 'X'

B tictactoe[4] = 'X'

C tictactoe[1][1] = 'X'

D tictactoe[2][2] = 'X'

Question 3 (1point)

On a défini : T = [[1,2,3], [4,5,6], [7,8,9]].

Quelle expression parmi les suivantes a pour valeur le nombre 8 ?

Réponses

A T[1,2]

B T[1][2]

C T[2,1]

D T[2][1]

Question 4 (1point)

Que renverra l'exécution `f(4)` si la fonction `f` est définie par :

```
def f(x):  
    return [x, x**2]
```

Réponses

A [4, 16]

B [4, 8]

C [16, 4]

D [4, 4]

Question 5 (2 points)

Utilisation de `range()` dans une boucle `for`

La fonction `range()` est souvent utilisée pour contrôler le nombre d'itérations dans une boucle `for`.

Exemple 1 : Utilisation de `range(5)`

```
for i in range(5):  
    print(i)
```

Exemple 2 : Utilisation de `range(2, 8)`

```
for i in range(2, 8):  
    print(i)
```

Exemple 3 : Utilisation de `range(1, 10, 2)`

```
for i in range(1, 10, 2):  
    print(i)
```

Exemple 4 : Utilisation de `range(10, 2, -2)`

```
for i in range(10, 2, -2):  
    print(i)
```

Pour chaque exemple, écrivez les valeurs de `i` qui seront affichées à l'écran.

Question 6 (2 points)

On dispose dans le tableau `annee2019` les températures mensuelles moyennes d'une région française.

On exécute le script suivant :

```
annee2019 = [('janvier',6), ('février',6), ('mars',12),  
             ('avril',20), ('mai',23), ('juin',25),  
             ('juillet',29), ('août',25), ('septembre',22),  
             ('octobre',15), ('novembre',11), ('décembre',7)]
```

```
m = annee2019[0][1]  
for mois in annee2019:  
    if (m > mois[1]):  
        m = mois[1]
```

Que contient la variable `m` à la fin de cette exécution ?
(Proposition des réponses page suivante ...)

Réponses :

A le mois le plus froid

B le mois le plus chaud

C la température mensuelle moyenne la plus basse

D la température mensuelle moyenne la plus haute

Question 7 (2 points)

```
t = []  
for i in range(10):  
    t.append(0)
```

```
for i in range(10):  
    if i%2==0:  
        t[i] = i
```

Une et une seule des affirmations suivantes est vraie. Laquelle ?

Réponses

A La liste t contient tous entiers compris entre 0 et 10.

B La liste t contient tous les entiers pairs compris entre 0 et 20.

C La liste t contient tous les entiers impairs compris entre 0 et 10.

D La liste t contient tous les entiers pairs compris entre 0 et 8.

Question 8 (2 points)

On souhaite écrire une fonction Python appelée `compter_voyelles` qui prend en argument une chaîne de caractères texte (composée uniquement de lettres minuscules) et retourne le nombre de voyelles (a, e, i, o, u, y) qu'elle contient.

```
def compter_voyelles(texte):  
    voyelles = "aeiouy"  
    compteur = 0  
    for i in range(len(texte)):  
        if ... in voyelles:  
            compteur = ...  
    return compteur
```

Compléter le code de la fonction pour qu'elle retourne le bon résultat. (Réécrire le code sur la feuille).

Question 9 (3 points)

On veut écrire une fonction `mystere` qui prend deux arguments x et y (des nombres) et qui retourne ces deux valeurs en les intervertissant.

```
def mystere(x, y):  
    # Code à compléter pour intervertir x et y  
    temp = x  
    ...  
    ...  
    return x, y
```

Questions :

1. Compléter le code pour que la fonction `mystere` retourne les valeurs de x et y échangées.
2. Expliquer l'utilité de la variable `temp` dans cette fonction.

Question 10 (3 points)

On souhaite écrire une fonction `compter_jusqua` qui utilise une boucle `while` pour trouver combien d'itérations sont nécessaires pour que la somme des entiers de 1 jusqu'à un certain nombre dépasse une valeur cible donnée. La fonction doit prendre un seul argument `cible`, un nombre entier positif, et retourner le nombre d'itérations nécessaires.

Règles :

- Utilisez une boucle `while` pour effectuer la somme des entiers de 1 à n progressivement.
- Arrêtez la boucle dès que la somme dépasse ou atteint la valeur de cible.
- La fonction doit retourner le nombre d'itérations effectuées.

Voici le squelette de la fonction à compléter :

```
def compter_jusqua(cible):
    somme = 0
    compteur = 0
    while ..... : # A compléter
        compteur = ...          # A compléter
        somme = somme + compteur
    return compteur
```

Question : Compléter le code.

Exemples de résultats attendus :

- `compter_jusqua(10)` doit retourner 4 (car $1 + 2 + 3 + 4 = 10$).
- `compter_jusqua(15)` doit retourner 5 (car $1 + 2 + 3 + 4 + 5 = 15$).
- `compter_jusqua(21)` doit retourner 6 (car $1 + 2 + 3 + 4 + 5 + 6 = 21$).

Question 11 (2 points)

Vous trouverez ci-dessous une fonction mystere :

```
def mystere(liste):
    resultat = 0
    for i in range(len(liste)):
        if liste[i] % 2 == 0:
            resultat = resultat + liste[i]
            break
    else:
        resultat = resultat + 1
    return resultat
```

Devinez le résultat pour chacun des appels suivants :

Appel 1 : `mystere([1, 3, 5, 7])`

Appel 2 : `mystere([1, 2, 4, 7])`