

Partie QCM

Question 1 (1 point)

Donnez la valeur de `15 - 3 * 7 // 4 % 5 + 3 + 2 * 4 - 1 + 43 % 11 // 7`

En respectant les priorités, on trouve **26**

Question 2 (1 point)

Donnez la valeur de `(42 // 4) / (42 % 4)`

Attention, la division `/` fournit un résultat de type float ! La réponse est **5.0**

Question 3 (1 point)

que vaut `not(True and not True) or 27%7 != 3*2 and not (3 != 7//2)`

En respectant les priorités booléennes, on trouve **True**

Question 4 (1 point)

Donnez la valeur de `'4' < '17'`

Ce sont des chaînes de caractères : comparaison par ordre alphabétique, donc **False**

Question 5 (1 point)

Que donne ce code ?

```
1 ch = "algorithmesDeTriBasiques"
2 print(ch[4:15:3] + ch[-11:-7])
```

En appliquant correctement les règles d'extraction on trouve **'rhsTTriB'**

Partie Analyse de code

Question 6 (1,5 points)

Ce code produit des affichages.
Donnez ces affichages dans le
tableau suivant :

```
1 l = [1, 2, 3]
2 t = (4, 5, 6)
3 for i in range(len(t)) :
4     l.insert(t[i] % 3, t[i])
5     print(f"i = {i} : l = {l}")
```

affichages
i = 0 : l = [1, 4, 2, 3]
i = 1 : l = [1, 4, 5, 2, 3]
i = 2 : l = [6, 1, 4, 5, 2, 3]

Question 7 (1,5 points)

a) Quel affichage produit ce code ?

'2103'

b) Quel nombre faudrait-il à la place de **147** pour afficher le résultat **3120** ?

216

```
1 def f(n) :  
2     c = ""  
3     x = n  
4     while x >= 1 :  
5         c = str(x % 4) + c  
6         x = x // 4  
7     return c  
8  
9 print(f(147))
```

Pour le b il faut reconnaître que la fonction donne l'écriture, en base 4, du nombre de départ. Ainsi, $147 = 2 * 4^3 + 1 * 4^2 + 0 * 4^1 + 3 * 4^0$.

Si on veut obtenir 3120, il faut donc partir de $3 * 4^3 + 1 * 4^2 + 2 * 4^1 + 0 * 4^0$ c'est-à-dire **216**

Question 8 (1,5 points)

a) Quel affichage produit ce code ?

[3, 5, 4, 0, 2, 1]

b) Quelle liste faudrait-il à la place de [3, 1, 5, 2, 4, 0] pour afficher [0, 1, 2, 3, 4, 5] ?

[0, 5, 1, 4, 2, 3]

```
1 a = []  
2 b = [3, 1, 5, 2, 4, 0]  
3  
4 for i in range(6) :  
5     j = min(i, len(b) - 1)  
6     a.append(b.pop(j))  
7  
8 print(a)
```

Pour le b on peut observer dans quel ordre sont repris les éléments de la liste a pour construire la liste b : le premier, puis le troisième, puis le cinquième, puis le sixième, le quatrième et enfin le deuxième.

Si pour ce procédé on veut trouver [0, 1, 2, 3, 4, 5], alors il faut bien partir de **[0, 5, 1, 4, 2, 3]**

Question 9 (sur 2 points)

Après exécution de ce programme, indiquez la valeur de a, b, c et d ainsi que leur type. La cohérence des réponses sera prise en compte, en cas d'erreur intermédiaire.

```
1 a = str(40 // 3 + 5 % 2 / 4)  
2 b = float(a[1 : -1] + a[len(a) - 1] + a[0])  
3 c = int((1000 * b) % 100)  
4 d = c % 3 == 0
```

variable	type	valeur
a	str	'13.25'
b	float	3.251
c	int	51
d	bool	True

Question 10 (sur 1,5 points)

Donnez les 4 valeurs affichées
par le programme suivant

```
1 def f(a, b) :  
2     if a == 0 or b < 0 :  
3         return 0  
4     else :  
5         return b + f(a//2, b-3)  
6  
7 print(f(9,14))  
8 print(f(10,6))  
9 print(f(19,11))  
10 print(f(8,17))
```

Réponses : $f(9, 14) = 38$

$f(10, 6) = 9$

$f(19, 11) = 26$

$f(8, 17) = 50$

Partie Ecriture de code

Question 11 (sur 1,5 point)

Complétez le code de la fonction **montant**, qui prend en paramètres **deux dictionnaires** (un pour le client, un pour les prix) et qui **retourne le montant total** dépensé par le client dans la boulangerie. Par exemple on pourra exécuter l'instruction **print(montant(client1, lesPrix))**. Dans cette version, vous **ne donnerez pas** un code en une ligne.

```
def montant(client, prix) :  
    m = 0  
    for article, quantite in client.items() :  
        m += prix[article] * quantite  
    return m
```

Question 12 (sur 1,5 point)

Même question que la précédente, mais cette fois votre code doit tenir en une seule ligne.

```
def montant(client, prix) :  
    return sum(prix[article] * quantite for article, quantite in client.items())
```

Question 13 (sur 2 points)

Complétez le code de la fonction **chiffreJournée**, qui prend en paramètres une liste de clients et un dictionnaire de prix, et qui retourne le chiffre total réalisé par la boulangerie au cours de la journée où sont passés les clients de la liste. Vous pourrez donner une version en une ligne ou en plusieurs lignes, comme vous voulez.

```
def chiffreJournée(clients, prix) :  
    return sum(montant(client, prix) for client in clients)
```

Question 14 (sur 2 points)

Complétez le code de la fonction **ajouterClient**, qui prend en paramètres une liste de clients et un dictionnaire de prix, et qui crée un client (dictionnaire) en demandant, pour chaque article de la boulangerie, la quantité achetée par le client. A la fin, ce client est ajouté à la liste des clients.

```
def ajouterClient(clients, prix) :  
    nouveau_client = {}  
    for article in prix :  
        quantite_article = int(input("combien d'articles " + article + " ? "))  
        nouveau_client[article] = quantite_article  
    clients.append(nouveau_client)
```