



ISITCOM

1LM

Examen DS PYTHON

A.U 2023-2024

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

← Cocher votre numéro CIN ici

☐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☐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☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

Nom , Prénom , Groupe

.....

.....

.....

*"In the face of ambiguity, refuse the temptation to guess." The zen of python***Question 1** Donner l'output du code suivant :

```
1 my_list = [1, 2, 3, 4, 5]
2 new_list = [x * 2 for x in my_list if x % 2 == 0]
3 print(new_list)
```

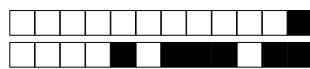
☐ [2, 4]☐ [2, 4, 6, 8, 10]☐ [4, 8]☐ [2, 4, 6, 8]**Question 2** Qu'affiche le code suivant :

```
1 my_string = "Bonjour {0}, comment allez-vous ?".format("John")
2 print(my_string)
```

☐ Bonjour John, comment allez-vous ?☐ Bonjour 0, comment allez-vous ?☐ Bonjour , comment allez-vous ?☐ Erreur de syntaxe**Question 3** Qu'affiche le code suivant :

```
1 my_string = "hello world"
2 count = 0
3 while count < len(my_string):
4     if my_string[count] == 'o':
5         count += 1
6         continue
7     print(my_string[count], end=" ")
8     count += 1
```

☐ hllowrld☐ hell wrld☐ hll wrld☐ hello world



Question 4 Donner l'output du code suivant :

```
1 x = "abcdef"
2 i = "i"
3 while i in x:
4     print(i, end=" ")
```

- ☐ a b c d e f
- ☐ n'affiche rien
- ☐ i i i i i
- ☐ Aucune réponse n'est correcte

Question 5 Donner l'output du code suivant :

```
1 a= [1, 2, 3, 4, 5]
2 for i in range(1, 5):
3     a[i-1] = a[i]
4 for i in range(0, 5):
5     print(a[i], end=" ")
```

- ☐ 1 2 3 4 5 ☐ 2 3 4 5 5 ☐ 2 3 4 5 1 ☐ 1 2 3 4 4

Question 6 Donner l'output du code suivant :

```
1 i = 2
2 while True:
3     if i%3 == 0:
4         break
5     print(i, end=' ')
6     i += 2
```

- ☐ 2 4 ☐ 1 2 3 4 ☐ 4 3 2 1 ☐ 4 2

Question 7 Donner l'output du code suivant :

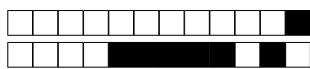
```
1 def display(b, n):
2     while n > 0:
3         print(b*n, end=" ")
4         n=n-1
5 display('z', 3)
```

- ☐ z z z ☐ 3z 3z 3z ☐ z3 z3 z3 ☐ zzz zz z

Question 8 ♣ On veut transformer la chaîne `ch = '2a3b1c1d1a2e2r1f'` en utilisant une liste de compréhension et la méthode `.join()` pour obtenir le résultat suivant `'aabbcbdaecrrf'`.

Pour cela, on peut utiliser l'instruction suivante :

- ☐ `".join([ch[i] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ `".join([ch[i+1] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ `".join([ch[i] * int(ch[i+1]) for i in range(0, len(ch), 2)])`
- ☐ `".join([ch[i+1] * int(ch[i]) for i in range(0, len(ch), 2)])`
- ☐ Aucune de ces réponses n'est correcte.



Question 9 Compléter l'exécution manuelle du code suivant :

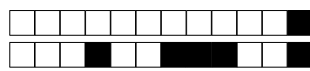
Remarques : dans la colonne $i == 0$ or $s[i] \neq s[i - 1]$ on complètera par True ou False

```
1 def mystere(s:str)-> str:
2     if not s:
3         return ""
4     result= [str(s.count(s[i])) + s[i] for i in range(len(s)) if i == 0 or s[i] != s[i-1]]
5     return ''.join(result)
6
7 input_string = 'aabbcddee'
8 print(mystere(input_string))
```

i	$s[i]$	$s.count(s[i])$	$i == 0$ or $s[i] \neq s[i - 1]$	result
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				

Donner ce que affiche print de la ligne 8 du code précédent:

☐ O ☐ P ☐ Espaces réservé au correcteur



Question 10 Le Motus est un jeu de mots où un joueur doit deviner un mot mystère en proposant des mots de même longueur. Pour chaque proposition, le jeu indique le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Ecrire une fonction en Python qui prendra en entrée le mot mystère et la proposition du joueur, puis retournera le nombre de lettres bien placées et le nombre de lettres présentes mais mal placées.

Instructions :

- La fonction doit s'appeler `verifier_proposition`.
- Elle prendra deux arguments : `mot_mystere` et `proposition`.
- Les deux arguments seront des chaînes de caractères de même longueur.
- La fonction devra retourner une liste contenant deux nombres : le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Exemple :

```
>>> verifier_proposition("PONEY", "MONEY")  
[1, 3]
```

Compléter le code suivant dans le rectangle ci dessous :

```
1 def verifier_proposition(mot_mystere: str, proposition: str) -> list:  
2     if len(mot_mystere) != len(proposition):  
3         return "Les mots n'ont pas la même longueur"  
4     # Compléter la fonction
```

☐ O ☐ P ☐ Espaces réservé au correcteur

Question 11 **Question Bonus** Proposer une autre solution de la question 10 en utilisant des listes de comprehensions.

☐ O ☐ P ☐ Espaces réservé au correcteur



ISITCOM

1LM

Examen DS PYTHON

A.U 2023-2024

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

← Cocher votre numéro CIN ici

☐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☐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☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

Nom , Prénom , Groupe

.....

.....

.....

*"In the face of ambiguity, refuse the temptation to guess." The zen of python***Question 1** Donner l'output du code suivant :

```
1 my_list = [1, 2, 3, 4, 5]
2 new_list = [x * 2 for x in my_list if x % 2 == 0]
3 print(new_list)
```

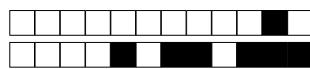
☐ [2, 4]☐ [2, 4, 6, 8]☐ [2, 4, 6, 8, 10]☐ [4, 8]**Question 2** Qu'affiche le code suivant :

```
1 my_string = "Bonjour {0}, comment allez-vous ?".format("John")
2 print(my_string)
```

☐ Bonjour John, comment allez-vous ?☐ Bonjour 0, comment allez-vous ?☐ Bonjour , comment allez-vous ?☐ Erreur de syntaxe**Question 3** Qu'affiche le code suivant :

```
1 my_string = "hello world"
2 count = 0
3 while count < len(my_string):
4     if my_string[count] == 'o':
5         count += 1
6         continue
7     print(my_string[count], end=" ")
8     count += 1
```

☐ hello world☐ hll wrld☐ hell wrld☐ hllowrld



Question 4 Donner l'output du code suivant :

```
1 x = "abcdef"
2 i = "i"
3 while i in x:
4     print(i, end=" ")
```

- ☐ n'affiche rien
- ☐ a b c d e f
- ☐ Aucune réponse n'est correcte
- ☐ i i i i i

Question 5 Donner l'output du code suivant :

```
1 a= [1, 2, 3, 4, 5]
2 for i in range(1, 5):
3     a[i-1] = a[i]
4 for i in range(0, 5):
5     print(a[i], end=" ")
```

- ☐ 1 2 3 4 4 ☐ 1 2 3 4 5 ☐ 2 3 4 5 5 ☐ 2 3 4 5 1

Question 6 Donner l'output du code suivant :

```
1 i = 2
2 while True:
3     if i%3 == 0:
4         break
5     print(i, end=' ')
6     i += 2
```

- ☐ 1 2 3 4 ☐ 4 2 ☐ 4 3 2 1 ☐ 2 4

Question 7 Donner l'output du code suivant :

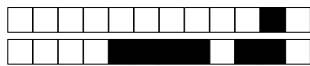
```
1 def display(b, n):
2     while n > 0:
3         print(b*n, end=" ")
4         n=n-1
5 display('z', 3)
```

- ☐ zzz zz z ☐ z z z ☐ 3z 3z 3z ☐ z3 z3 z3

Question 8 ♣ On veut transformer la chaîne `ch = '2a3b1c1d1a2e2r1f'` en utilisant une liste de compréhension et la méthode `.join()` pour obtenir le résultat suivant `'aabbcbdaecrrf'`.

Pour cela, on peut utiliser l'instruction suivante :

- ☐ `".join([ch[i] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ `".join([ch[i+1] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ `".join([ch[i] * int(ch[i+1]) for i in range(0, len(ch), 2)])`
- ☐ `".join([ch[i+1] * int(ch[i]) for i in range(0, len(ch), 2)])`
- ☐ Aucune de ces réponses n'est correcte.



Question 9 Compléter l'exécution manuelle du code suivant :

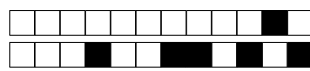
Remarques : dans la colonne $i == 0$ or $s[i] \neq s[i - 1]$ on complètera par True ou False

```
1 def mystere(s:str)-> str:
2     if not s:
3         return ""
4     result= [str(s.count(s[i])) + s[i] for i in range(len(s)) if i == 0 or s[i] != s[i-1]]
5     return ''.join(result)
6
7 input_string = 'aabbcddee'
8 print(mystere(input_string))
```

i	$s[i]$	$s.count(s[i])$	$i == 0$ or $s[i] \neq s[i - 1]$	result
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				

Donner ce que affiche print de la ligne 8 du code précédent:

☐ O ☐ P ☐ Espaces réservé au correcteur



Question 10 Le Motus est un jeu de mots où un joueur doit deviner un mot mystère en proposant des mots de même longueur. Pour chaque proposition, le jeu indique le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Ecrire une fonction en Python qui prendra en entrée le mot mystère et la proposition du joueur, puis retournera le nombre de lettres bien placées et le nombre de lettres présentes mais mal placées.

Instructions :

- La fonction doit s'appeler `verifier_proposition`.
- Elle prendra deux arguments : `mot_mystere` et `proposition`.
- Les deux arguments seront des chaînes de caractères de même longueur.
- La fonction devra retourner une liste contenant deux nombres : le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Exemple :

```
>>> verifier_proposition("PONEY", "MONEY")  
[1, 3]
```

Compléter le code suivant dans le rectangle ci dessous :

```
1 def verifier_proposition(mot_mystere: str, proposition: str) -> list:  
2     if len(mot_mystere) != len(proposition):  
3         return "Les mots n'ont pas la même longueur"  
4     # Compléter la fonction
```

☐ O ☐ P ☐ Espaces réservé au correcteur

Question 11 **Question Bonus** Proposer une autre solution de la question 10 en utilisant des listes de comprehensions.

☐ O ☐ P ☐ Espaces réservé au correcteur



ISITCOM

1LM

Examen DS PYTHON

A.U 2023-2024

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

← Cocher votre numéro CIN ici

☐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☐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☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

Nom , Prénom , Groupe

.....

.....

.....

"In the face of ambiguity, refuse the temptation to guess." The zen of python

Question 1 Donner l'output du code suivant :

```
1 my_list = [1, 2, 3, 4, 5]
2 new_list = [x * 2 for x in my_list if x % 2 == 0]
3 print(new_list)
```

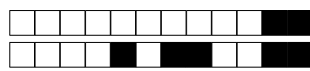
☐ [2, 4, 6, 8, 10]☐ [2, 4]☐ [2, 4, 6, 8]☐ [4, 8]**Question 2** Qu'affiche le code suivant :

```
1 my_string = "Bonjour {0}, comment allez-vous ?".format("John")
2 print(my_string)
```

☐ Bonjour 0, comment allez-vous ?☐ Bonjour , comment allez-vous ?☐ Erreur de syntaxe☐ Bonjour John, comment allez-vous ?**Question 3** Qu'affiche le code suivant :

```
1 my_string = "hello world"
2 count = 0
3 while count < len(my_string):
4     if my_string[count] == 'o':
5         count += 1
6         continue
7     print(my_string[count], end=" ")
8     count += 1
```

☐ hell wrld☐ hello world☐ hll wrld☐ hllowrld



Question 4 Donner l'output du code suivant :

```
1 x = "abcdef"
2 i = "i"
3 while i in x:
4     print(i, end=" ")
```

☐ i i i i i

☐ Aucune réponse n'est correcte

☐ n'affiche rien

☐ a b c d e f

Question 5 Donner l'output du code suivant :

```
1 a= [1, 2, 3, 4, 5]
2 for i in range(1, 5):
3     a[i-1] = a[i]
4 for i in range(0, 5):
5     print(a[i], end=" ")
```

☐ 2 3 4 5 1

☐ 2 3 4 5 5

☐ 1 2 3 4 5

☐ 1 2 3 4 4

Question 6 Donner l'output du code suivant :

```
1 i = 2
2 while True:
3     if i%3 == 0:
4         break
5     print(i, end=' ')
6     i += 2
```

☐ 4 3 2 1

☐ 1 2 3 4

☐ 2 4

☐ 4 2

Question 7 Donner l'output du code suivant :

```
1 def display(b, n):
2     while n > 0:
3         print(b*n, end=" ")
4         n=n-1
5 display('z', 3)
```

☐ z z z

☐ 3z 3z 3z

☐ zzz zz z

☐ z3 z3 z3

Question 8 ♣ On veut transformer la chaîne `ch = '2a3b1c1d1a2e2r1f'` en utilisant une liste de compréhension et la méthode `.join()` pour obtenir le résultat suivant `'aabbcbdaecrrf'`.

Pour cela, on peut utiliser l'instruction suivante :

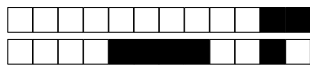
☐ `".join([ch[i] * int(ch[i+1]) for i in range(0, len(ch), 2)])`

☐ `".join([ch[i+1] * int(ch[i]) for i in range(0, len(ch), 2)])`

☐ `".join([ch[i] * int(ch[i-1]) for i in range(1, len(ch), 2)])`

☐ `".join([ch[i+1] * int(ch[i-1]) for i in range(1, len(ch), 2)])`

☐ Aucune de ces réponses n'est correcte.



Question 9 Compléter l'exécution manuelle du code suivant :

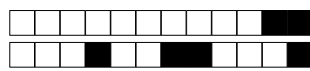
Remarques : dans la colonne $i == 0$ or $s[i] \neq s[i - 1]$ on complètera par True ou False

```
1 def mystere(s:str)-> str:
2     if not s:
3         return ""
4     result= [str(s.count(s[i])) + s[i] for i in range(len(s)) if i == 0 or s[i] != s[i-1]]
5     return ''.join(result)
6
7 input_string = 'aabbcddee'
8 print(mystere(input_string))
```

i	$s[i]$	$s.count(s[i])$	$i == 0$ or $s[i] \neq s[i - 1]$	result
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				

Donner ce que affiche print de la ligne 8 du code précédent:

☐ O ☐ P ☐ Espaces réservé au correcteur



Question 10 Le Motus est un jeu de mots où un joueur doit deviner un mot mystère en proposant des mots de même longueur. Pour chaque proposition, le jeu indique le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Ecrire une fonction en Python qui prendra en entrée le mot mystère et la proposition du joueur, puis retournera le nombre de lettres bien placées et le nombre de lettres présentes mais mal placées.

Instructions :

- La fonction doit s'appeler `verifier_proposition`.
- Elle prendra deux arguments : `mot_mystere` et `proposition`.
- Les deux arguments seront des chaînes de caractères de même longueur.
- La fonction devra retourner une liste contenant deux nombres : le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Exemple :

```
>>> verifier_proposition("PONEY", "MONEY")  
[1, 3]
```

Compléter le code suivant dans le rectangle ci dessous :

```
1 def verifier_proposition(mot_mystere: str, proposition: str) -> list:  
2     if len(mot_mystere) != len(proposition):  
3         return "Les mots n'ont pas la même longueur"  
4     # Compléter la fonction
```

☐ O ☐ P ☐ Espaces réservé au correcteur

Question 11 **Question Bonus** Proposer une autre solution de la question 10 en utilisant des listes de comprehensions.

☐ O ☐ P ☐ Espaces réservé au correcteur



ISITCOM

1LM

Examen DS PYTHON

A.U 2023-2024

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

← Cocher votre numéro CIN ici

☐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☐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☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

Nom , Prénom , Groupe

.....

.....

.....

*"In the face of ambiguity, refuse the temptation to guess." The zen of python***Question 1** Donner l'output du code suivant :

```
1 my_list = [1, 2, 3, 4, 5]
2 new_list = [x * 2 for x in my_list if x % 2 == 0]
3 print(new_list)
```

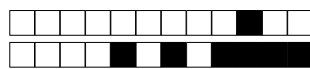
☐ [2, 4, 6, 8]☐ [4, 8]☐ [2, 4, 6, 8, 10]☐ [2, 4]**Question 2** Qu'affiche le code suivant :

```
1 my_string = "Bonjour {0}, comment allez-vous ?".format("John")
2 print(my_string)
```

☐ Erreur de syntaxe☐ Bonjour 0, comment allez-vous ?☐ Bonjour John, comment allez-vous ?☐ Bonjour , comment allez-vous ?**Question 3** Qu'affiche le code suivant :

```
1 my_string = "hello world"
2 count = 0
3 while count < len(my_string):
4     if my_string[count] == 'o':
5         count += 1
6         continue
7     print(my_string[count], end=" ")
8     count += 1
```

☐ hllowrld☐ hell wrld☐ hll wrld☐ hello world



Question 4 Donner l'output du code suivant :

```
1 x = "abcdef"
2 i = "i"
3 while i in x:
4     print(i, end=" ")
```

- ☐ a b c d e f
- ☐ Aucune réponse n'est correcte
- ☐ i i i i i
- ☐ n'affiche rien

Question 5 Donner l'output du code suivant :

```
1 a= [1, 2, 3, 4, 5]
2 for i in range(1, 5):
3     a[i-1] = a[i]
4 for i in range(0, 5):
5     print(a[i], end=" ")
```

- ☐ 2 3 4 5 5 ☐ 2 3 4 5 1 ☐ 1 2 3 4 5 ☐ 1 2 3 4 4

Question 6 Donner l'output du code suivant :

```
1 i = 2
2 while True:
3     if i%3 == 0:
4         break
5     print(i, end=' ')
6     i += 2
```

- ☐ 4 3 2 1 ☐ 4 2 ☐ 2 4 ☐ 1 2 3 4

Question 7 Donner l'output du code suivant :

```
1 def display(b, n):
2     while n > 0:
3         print(b*n, end=" ")
4         n=n-1
5 display('z', 3)
```

- ☐ z z z ☐ z3 z3 z3 ☐ 3z 3z 3z ☐ zzz zz z

Question 8 ♣ On veut transformer la chaîne `ch = '2a3b1c1d1a2e2r1f'` en utilisant une liste de compréhension et la méthode `join()` pour obtenir le résultat suivant `'aabbcbdaecrrf'`.

Pour cela, on peut utiliser l'instruction suivante :

- ☐ `".join([ch[i+1] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ `".join([ch[i+1] * int(ch[i]) for i in range(0, len(ch), 2)])`
- ☐ `".join([ch[i] * int(ch[i+1]) for i in range(0, len(ch), 2)])`
- ☐ `".join([ch[i] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ Aucune de ces réponses n'est correcte.



Question 9 Compléter l'exécution manuelle du code suivant :

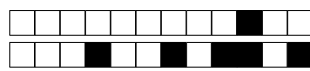
Remarques : dans la colonne $i == 0$ or $s[i] \neq s[i - 1]$ on complètera par True ou False

```
1 def mystere(s:str)-> str:
2     if not s:
3         return ""
4     result= [str(s.count(s[i])) + s[i] for i in range(len(s)) if i == 0 or s[i] != s[i-1]]
5     return ''.join(result)
6
7 input_string = 'aabbcddee'
8 print(mystere(input_string))
```

i	$s[i]$	$s.count(s[i])$	$i == 0$ or $s[i] \neq s[i - 1]$	result
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				

Donner ce que affiche print de la ligne 8 du code précédent:

☐ O ☐ P ☐ Espaces réservé au correcteur



Question 10 Le Motus est un jeu de mots où un joueur doit deviner un mot mystère en proposant des mots de même longueur. Pour chaque proposition, le jeu indique le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Ecrire une fonction en Python qui prendra en entrée le mot mystère et la proposition du joueur, puis retournera le nombre de lettres bien placées et le nombre de lettres présentes mais mal placées.

Instructions :

- La fonction doit s'appeler **verifier_proposition**.
- Elle prendra deux arguments : **mot_mystere** et **proposition**.
- Les deux arguments seront des chaînes de caractères de même longueur.
- La fonction devra retourner une liste contenant deux nombres : le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Exemple :

```
>>> verifier_proposition("PONEY", "MONEY")  
[1, 3]
```

Compléter le code suivant dans le rectangle ci dessous :

```
1 def verifier_proposition(mot_mystere: str, proposition: str) -> list:  
2     if len(mot_mystere) != len(proposition):  
3         return "Les mots n'ont pas la même longueur"  
4     # Compléter la fonction
```

☐ O ☐ P ☐ Espaces réservé au correcteur

Question 11 **Question Bonus** Proposer une autre solution de la question 10 en utilisant des listes de comprehensions.

☐ O ☐ P ☐ Espaces réservé au correcteur



ISITCOM

1LM

Examen DS PYTHON

A.U 2023-2024

☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

← Cocher votre numéro CIN ici

☐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☐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☐0 ☐1 ☐2 ☐3 ☐4 ☐5 ☐6 ☐7 ☐8 ☐9

Nom , Prénom , Groupe

.....

.....

.....

"In the face of ambiguity, refuse the temptation to guess." The zen of python

Question 1 Donner l'output du code suivant :

```
1 my_list = [1, 2, 3, 4, 5]
2 new_list = [x * 2 for x in my_list if x % 2 == 0]
3 print(new_list)
```

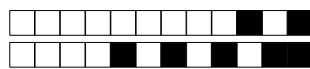
☐ [2, 4, 6, 8]☐ [2, 4]☐ [2, 4, 6, 8, 10]☐ [4, 8]**Question 2** Qu'affiche le code suivant :

```
1 my_string = "Bonjour {0}, comment allez-vous ?".format("John")
2 print(my_string)
```

☐ Bonjour John, comment allez-vous ?☐ Bonjour , comment allez-vous ?☐ Bonjour 0, comment allez-vous ?☐ Erreur de syntaxe**Question 3** Qu'affiche le code suivant :

```
1 my_string = "hello world"
2 count = 0
3 while count < len(my_string):
4     if my_string[count] == 'o':
5         count += 1
6         continue
7     print(my_string[count], end=" ")
8     count += 1
```

☐ hell wrld☐ hll wrld☐ hello world☐ hllowrld



Question 4 Donner l'output du code suivant :

```
1 x = "abcdef"
2 i = "i"
3 while i in x:
4     print(i, end=" ")
```

- ☐ Aucune réponse n'est correcte
- ☐ i i i i i
- ☐ a b c d e f
- ☐ n'affiche rien

Question 5 Donner l'output du code suivant :

```
1 a= [1, 2, 3, 4, 5]
2 for i in range(1, 5):
3     a[i-1] = a[i]
4 for i in range(0, 5):
5     print(a[i], end = " ")
```

- ☐ 2 3 4 5 5 ☐ 1 2 3 4 5 ☐ 2 3 4 5 1 ☐ 1 2 3 4 4

Question 6 Donner l'output du code suivant :

```
1 i = 2
2 while True:
3     if i%3 == 0:
4         break
5     print(i, end= ' ')
6     i += 2
```

- ☐ 4 2 ☐ 1 2 3 4 ☐ 4 3 2 1 ☐ 2 4

Question 7 Donner l'output du code suivant :

```
1 def display(b, n):
2     while n > 0:
3         print(b*n, end=" ")
4         n=n-1
5 display('z', 3)
```

- ☐ zzz zz z ☐ 3z 3z 3z ☐ z z z ☐ z3 z3 z3

Question 8 ♣ On veut transformer la chaîne `ch = '2a3b1c1d1a2e2r1f'` en utilisant une liste de compréhension et la méthode `join()` pour obtenir le résultat suivant `'aabbcbdaecerrf'`.

Pour cela, on peut utiliser l'instruction suivante :

- ☐ `".join([ch[i] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ `".join([ch[i] * int(ch[i+1]) for i in range(0, len(ch), 2)])`
- ☐ `".join([ch[i+1] * int(ch[i]) for i in range(0, len(ch), 2)])`
- ☐ `".join([ch[i+1] * int(ch[i-1]) for i in range(1, len(ch), 2)])`
- ☐ Aucune de ces réponses n'est correcte.



Question 9 Compléter l'exécution manuelle du code suivant :

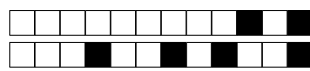
Remarques : dans la colonne $i == 0$ or $s[i] \neq s[i - 1]$ on complètera par True ou False

```
1 def mystere(s:str)-> str:
2     if not s:
3         return ""
4     result= [str(s.count(s[i])) + s[i] for i in range(len(s)) if i == 0 or s[i] != s[i-1]]
5     return ''.join(result)
6
7 input_string = 'aabbcddee'
8 print(mystere(input_string))
```

i	$s[i]$	$s.count(s[i])$	$i == 0$ or $s[i] \neq s[i - 1]$	result
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				

Donner ce que affiche print de la ligne 8 du code précédent:

☐ O ☐ P ☐ Espaces réservé au correcteur



Question 10 Le Motus est un jeu de mots où un joueur doit deviner un mot mystère en proposant des mots de même longueur. Pour chaque proposition, le jeu indique le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Ecrire une fonction en Python qui prendra en entrée le mot mystère et la proposition du joueur, puis retournera le nombre de lettres bien placées et le nombre de lettres présentes mais mal placées.

Instructions :

- La fonction doit s'appeler `verifier_proposition`.
- Elle prendra deux arguments : `mot_mystere` et `proposition`.
- Les deux arguments seront des chaînes de caractères de même longueur.
- La fonction devra retourner une liste contenant deux nombres : le nombre de lettres correctement placées et le nombre de lettres présentes mais mal placées.

Exemple :

```
>>> verifier_proposition("PONEY", "MONEY")  
[1, 3]
```

Compléter le code suivant dans le rectangle ci dessous :

```
1 def verifier_proposition(mot_mystere: str, proposition: str) -> list:  
2     if len(mot_mystere) != len(proposition):  
3         return "Les mots n'ont pas la même longueur"  
4     # Compléter la fonction
```

☐ O ☐ P ☐ Espaces réservé au correcteur

Question 11 **Question Bonus** Proposer une autre solution de la question 10 en utilisant des listes de comprehensions.

☐ O ☐ P ☐ Espaces réservé au correcteur