

DS — Python pratique (BTS SIO 1re année, Bloc B3 Cybersécurité)

La rigueur, la rédaction, le soin et le typage des fonctions sont évalués sur 2 points.

Si un de ces points n'est pas satisfait, 2 points seront retirés de la note.

Toutes les fonctions doivent être testées. Si une fonction n'est pas testée, l'intégralité des points ne sera pas comptabilisée.

Rappels autorisés

- `int(...)`, `str(...)`, `input(...)`, `print(...)`
- Concaténation de chaînes: `"bonjour" + " monde"` → `"bonjour monde"`, `"A" + "B"` → `"AB"`, `"B" + "A"` → `"BA"`
- Pour tester qu'un caractère appartient à un ensemble (chiffres, lettres, spéciaux), parcourir la chaîne de référence avec une boucle et comparer les caractères.
- Pour vérifier qu'un caractère est dans une chaîne de caractères (mot clef `in`): `caractere in chaine`
- On peut accéder au caractère à l'indice `i` de la chaîne `chaine` avec `chaine[i]`
- Pour obtenir la taille d'une chaîne de caractères, on peut utiliser la fonction `len(chaine)`

Partie A — Code à compléter

Exercice A1

Compléter la fonction `nb_chiffres(chaine)` qui prend en paramètre une chaîne et renvoie le nombre de chiffres présents.

```
def nb_chiffres(chaine: str) -> int:
    compteur = 0
    digits = "0123456789"
    for c in chaine:
        if ...:
            compteur = ...
    return ...
```

Exercice A2

Compléter la fonction `nb_lettres(chaine)` qui prend une chaîne et renvoie le nombre de lettres (a-z, A-Z).

```
def nb_lettres(chaine: str) -> int:
    compteur = 0
    alphabet = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
    for c in chaine:
        if ...:
            compteur = ...
    return ...
```

Exercice A3

Compléter `contient_majuscule(chaine)` qui renvoie `True` si la chaîne contient au moins une majuscule, `False` sinon.

```
def contient_majuscule(chaine: str) -> bool:
    maj = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
    for c in chaine:
        if ...:
            return True
    return ...
```

Exercice A4

Compléter `contient_special(chaine)` qui renvoie `True` si la chaîne contient au moins un caractère spécial parmi `!@#$$%^&*?`, sinon `False`.

```
def contient_special(chaine: str) -> bool:
    specials = "!@#$$%^&*?"
    for c in chaine:
        if ...:
            return True
    return ...
```

Exercice A5

Compléter `contient_deux_types(chaine)` qui renvoie `True` si la chaîne contient au moins deux catégories différentes parmi: lettre, chiffre, spécial; sinon `False`, en utilisant les fonctions précédentes (`nb_lettres`, `nb_chiffres`, `contient_special`).

```
def contient_deux_types(chaine: str) -> bool:
    if nb_lettres(chaine) > ... :
        a_lettre = ...
    else:
        a_lettre = ...
    if nb_chiffres(chaine) > ... :
        a_chiffre = ...
    else:
        a_chiffre = ...
    a_special = contient_special(...)
    if (a_lettre and a_chiffre) or (a_lettre and a_special) or (a_chiffre and a_special):
        return True
    return ...
```

Exercice A6

Compléter `score_motdepasse(chaine)` qui calcule un score simple:

- +2 si longueur ≥ 8
- +2 si `contient_deux_types` est vrai
- +2 si `contient_majuscule` est vrai
- +2 si `contient_special` est vrai
- +2 si `nb_chiffres(chaine) ≥ 2`

```
def score_motdepasse(chaine: str) -> int:
    score = 0
    # Compléter avec les fonctions précédentes
    if len(chaine) >= ...:
        score = score + ...
    if contient_deux_types(...):
        score = score + ...
    if contient_majuscule(...):
        score = score + ...
    if contient_special(...):
        score = score + ...
    if nb_chiffres(...) >= ...:
        score = score + ...
    return ...
```

Partie B — Fonctions à écrire

Exercice B1

Écrire une fonction `repetier_caractere(c, n)` qui prend en paramètre un caractère `c` et un entier `n`, et qui renvoie une chaîne contenant `c` répété `n` fois.

Exemple: `repetier_caractere('X', 5) → 'XXXXX'`

Exercice B2

Écrire une fonction `masquer_motdepasse(chaine)` qui renvoie une nouvelle chaîne où seules la première et la dernière lettre restent visibles et où toutes les autres positions sont remplacées par `*`.

Si la longueur est ≤ 2 , renvoyer la chaîne telle quelle.

Exemple: `masquer_motdepasse("Sécurité") → S*****e`

Exercice B3

Écrire une fonction `contient_trois_identiques(chaine)` qui prend en paramètre une chaîne `chaine` et renvoie `True` si la chaîne contient au moins 3 fois le même caractère d'affilée (ex: `aaa`, `111`, `@@@`), sinon `False`.

Indice: utiliser une boucle, comparer chaque caractère au précédent, compter les répétitions consécutives, réinitialiser quand le caractère change, retourner `True` si le compteur vaut 3.

Exercice B4

Écrire une fonction `est_motdepasse_valide(chaine)` qui renvoie `True` si le mot de passe est considéré « valide » selon ces règles:

- Longueur ≥ 8
- Au moins 2 types parmi lettre/chiffre/spécial
- Pas de suite consécutive de 3 caractères
- Pas d'espaces
- Au moins 1 majuscule
- Au moins 1 chiffre

Utiliser les fonctions écrites ou complétées précédemment.

Exercice B5

Écrire une fonction `conseil_motdepasse(chaine)` qui renvoie une chaîne explicative avec 2 à 3 conseils personnalisés (simples phrases concaténées) si `est_motdepasse_valide(chaine)` renvoie `False`.

Exemples de conseils: « ajouter une majuscule », « allonger à 8+ », « mettre un chiffre », « éviter suites ABC/123 », « ajouter un caractère spécial ».