

Initiation à l’algorithmique et à la programmation

TD/TP 3 – Fonctions et chaînes de caractères

Licence 3 – INALCO – 2026–2027

Objectifs de la séance

Définir et appeler une fonction, distinguer `print()` de `return`, manipuler du texte et tester les résultats.

Avant de commencer

Créez un dossier TP3. Prédisez avant d’exécuter. Enregistrez les exercices 3 et 4 dans `tp3.py`, avec les définitions avant les appels. Durée indicative : 45 à 60 minutes, hors défi.

Méthode de travail : comprendre → prédire → programmer → tester → expliquer.

Difficulté : * application directe ** plusieurs notions *** défi.

■ TD – Lire et corriger du code

Exercice 1 – Prévoir les sorties

[*]

Sans exécuter le code, notez chaque ligne affichée. Vérifiez ensuite avec Python.

```
1 def debut(texte):
2     return texte.strip()[:3]
3
4 mot = "  Bonjour  "
5 resultat = debut(mot)
6 print(resultat)
7 print(mot.lower())
8 print(len(resultat))
9 print(mot)
```

1. Identifiez le paramètre et l’argument de l’appel `debut(mot)`.
2. La variable `mot` a-t-elle changé ? Expliquez.
3. Que renvoie `debut("a")` ? Et `debut("")` ?

Exercice 2 – Afficher ou renvoyer ?

[**]

On veut obtenir une salutation en minuscules, puis afficher sa longueur.

```
1 def saluer(prenom):
2     print("Bonjour " + prenom.strip())
3
4 message = saluer("  Ada  ")
5 message.lower()
6 print(len(message))
```

1. Que vaut `message` après l’appel ? Quelle ligne provoque la première erreur ?
2. Corrigez la fonction pour qu’elle renvoie la salutation au lieu de l’afficher.
3. Corrigez aussi l’appel à `lower()`, puis affichez le message et sa longueur.

Résultats attendus : `bonjour ada`, puis `11`.

En cas de blocage

Vérifiez le message d’erreur, l’indentation et le `return`. Pour voir les espaces, affichez `"[" + texte + "]"`.

■ TP – Écrire des fonctions pour traiter du texte

Exercice 3 – Nettoyer et formater

[**]

Dans `tp3.py`, écrivez les deux fonctions suivantes. Ajoutez une docstring courte à chacune. Chaque fonction doit **renvoyer** son résultat.

1. `nettoyer_texte(texte)` : retirez les espaces aux extrémités et mettez le texte en minuscules. Les espaces internes restent présents.
2. `formater_nom(prenom, nom)` : retirez les espaces aux extrémités de chaque argument, mettez le nom de famille en majuscules et assemblez les deux éléments avec un espace. Gardez la casse du prénom.

Test de validation

Comparez vos résultats avec les valeurs attendues ci-dessous.

Appel	Valeur attendue
<code>nettoyer_texte(" Un CHAT ")</code>	"un chat"
<code>nettoyer_texte("")</code>	" "
<code>formater_nom(" Ada ", " lovelace ")</code>	"Ada LOVELACE"
<code>formater_nom("Alice", "Martin")</code>	"Alice MARTIN"

Exercice 4 – Une petite fiche de texte

[**]

1. Écrivez `compter_sans_espaces(texte)` : retirez les espaces ordinaires " " avec `replace()`, puis renvoyez la longueur obtenue.
2. Testez la fonction avec "un chat", " " et "". Les valeurs attendues sont respectivement 6, 0 et 0.
3. Après les définitions des fonctions, demandez une phrase avec `input()`. Utilisez `nettoyer_texte()` pour obtenir une version nettoyée.
4. Affichez le texte nettoyé, sa longueur, son nombre de caractères sans espaces et ses trois premiers caractères.

Exemple : pour la saisie " Un CHAT ", le programme doit afficher :

```
Texte : un chat
Longueur : 7
Sans espaces : 6
Debut : un
```

Attention : la valeur après `Debut` : contient aussi l’espace qui suit `un`. Testez également une saisie vide et un texte d’un seul caractère.

Exercice 5 – Pour aller plus loin : comparer des mots

[***]

Écrivez `meme_mot(mot1, mot2)` qui appelle `nettoyer_texte()` sur chaque argument et renvoie `True` si les deux résultats sont égaux, `False` sinon.

Tests : (" Chat ", "CHAT") donne `True`, ("chat", "chats") donne `False`. Essayez aussi ("été", "ete") : votre nettoyage conserve les accents.

■ Bilan

- ☐ Mon fichier `tp3.py` est enregistré dans TP3.
- ☐ Mes fonctions renvoient une valeur que je peux réutiliser.
- ☐ J’ai testé une entrée ordinaire, une entrée courte et une entrée vide.
- ☐ Je peux expliquer pourquoi `texte.lower()` seul ne modifie pas `texte`.

FIN DU TP 3 – sauvegardez votre travail !