

Initiation à l’algorithmique et à la programmation

TD/TP 1 – Premiers programmes en Python

Licence 3 – INALCO – 2026–2027

Objectifs de la séance

À la fin de la séance, vous saurez :

- créer, enregistrer et exécuter un fichier Python ;
- utiliser des variables contenant des nombres, du texte et des booléens ;
- afficher un résultat avec `print()` et lire une réponse avec `input()` ;
- convertir une saisie avec `int()` ou `float()` ;
- distinguer une erreur de syntaxe, une erreur d’exécution et une erreur de logique ;
- définir et appeler une première fonction simple.

Avant de commencer

Créez un dossier TP1. Chaque exercice de programmation doit être enregistré dans un fichier distinct : `ex05.py`, `ex06.py`, etc. Exécutez vos programmes depuis l’éditeur ou le terminal et conservez vos fichiers jusqu’à la fin du semestre.

Méthode de travail : pour chaque problème, suivez ce cycle :



Difficulté : * application directe ** plusieurs notions à combiner *** défi demandant davantage d’autonomie.

■ Prise en main – Mon premier fichier Python

Créez le fichier `bonjour.py`, saisissez le programme suivant, enregistrez-le puis exécutez-le.

```
1 # Mon premier programme Python
2 message = "Bienvenue au cours de programmation !"
3 print(message)
```

Modifiez ensuite le message pour afficher une salutation dans la langue de votre choix.

Test de validation

Votre programme s’exécute sans erreur, et le terminal affiche exactement le message contenu dans la variable `message`.

■ TD – Lire et comprendre avant de programmer

Exercice 1 – Variables, valeurs et types

[*]

Sans exécuter le code, indiquez la **valeur** et le **type** associés à chaque variable après l’exécution des instructions.

```
1 student_name = "Alice"
2 language = "japonais"
3 year = 3
```

```

4 weekly_hours = 4.5
5 is_beginner = True

```

Variable	Valeur	Type Python
student_name		
language		
year		
weekly_hours		
is_beginner		

Vérifiez ensuite vos réponses à l’aide de `print(type(...))`.

Exercice 2 – Une variable peut changer de valeur

[*]

Considérez le programme suivant.

```

1 word_count = 10
2 print(word_count, type(word_count))
3
4 word_count = "dix"
5 print(word_count, type(word_count))

```

- Prédisez les deux lignes affichées.
- Quel est le type de la valeur associée à `word_count` au début ? Et à la fin ?
- Complétez : en Python, une même variable peut

Exercice 3 – Prédire des expressions

[*]

Sans utiliser Python, donnez le résultat **et son type**. Vérifiez ensuite dans l’interpréteur.

- | | |
|----------------|-------------------------------------|
| a) $7 + 3 * 2$ | e) $2 ** 5$ |
| b) $10 / 4$ | f) <code>"bon" + "jour"</code> |
| c) $10 // 4$ | g) <code>"ha" * 3</code> |
| d) $10 \% 4$ | h) <code>"INALCO" + " Paris"</code> |

Que font les opérateurs `//`, `%` et `**` ?

.....

Exercice 4 – Trois familles d’erreurs

[**]

Pour chaque programme : (1) prédisez ce qui se passe ; (2) exécutez-le ; (3) classez le problème ; (4) proposez une correction.

Programme	Problème observé	Catégorie
A		syntaxe / type / logique
B		syntaxe / type / logique
C		syntaxe / type / logique

A	B	C
<pre>print("Bonjour")</pre>	<pre>age = "20" print(age + 1)</pre>	<pre>length = 5 width = 3 area = length + width print(area)</pre>

Point important : le programme C s’exécute. Pourquoi un programme sans message d’erreur peut-il tout de même être faux ?

.....

■ TP – Écrire et tester vos programmes

Exercice 5 – Profil linguistique

[*]

Écrivez `ex05.py`. Le programme demande le prénom, la langue étudiée et le nombre d’années d’étude, puis produit une phrase complète.

```
Votre prenom : Alice
Langue etudiee : japonais
Nombre d'annees d'etude : 3
Bonjour Alice ! Vous etudiez le japonais depuis 3 an(s).
```

Contraintes : utilisez trois variables, trois appels à `input()` et un seul appel à `print()`. Utilisez des virgules dans `print()`, comme dans le cours.

Test de validation

Testez le programme avec votre propre profil, puis avec un prénom contenant un trait d’union ou un espace. Vérifiez que la phrase reste lisible.

Exercice 6 – Budget de livres

[**]

Une étudiante achète trois livres. Demandez le prix de chaque livre, puis affichez le total et le prix moyen. Les prix peuvent contenir des décimales.

Avant de coder : quel type faut-il utiliser pour convertir les réponses de `input()` ?

Test de validation

Avec les prix 12.50, 20 et 9.50, votre programme doit obtenir un total de 42.0 et une moyenne de 14.0.

Exercice 7 – Durée d’un enregistrement

[**]

Un enregistrement audio dure un certain nombre entier de secondes. Affichez cette durée sous la forme `minutes + secondes restantes`.

Exemple : `Duration in seconds: 135` doit conduire à `2 minute(s) et 15 seconde(s)`.

Indice : utilisez `//` et `%`. Commencez par écrire sur papier les deux calculs nécessaires.

Test de validation

Testez au minimum 0, 59, 60, 61 et 135. Un seul exemple ne suffit pas à valider un programme.

Exercice 8 – Déboguer une fiche étudiante

[**]

Le programme doit calculer l’âge de l’étudiante l’année prochaine, mais il échoue.

```

1 name = input("Votre prenom : ")
2 age = input("Votre age : ")
3 next_age = age + 1
4 print("Bonjour " + name)
5 print("L'annee prochaine vous aurez " + next_age + " ans")

```

- Exécutez le programme et recopiez la dernière ligne du message d’erreur.
- Identifiez les types de `age`, de `1` et de `next_age` attendu.
- Corrigez le calcul, puis corrigez l’affichage final.
- Expliquez en une phrase pourquoi votre correction fonctionne.

Message observé :

Explication :

■ Bilan et remise

Avant de terminer, cochez chaque point.

- ☐ Mes fichiers portent les noms demandés et sont enregistrés dans le dossier TP1.
- ☐ Tous mes programmes s’exécutent sans erreur.
- ☐ J’ai testé chaque programme avec plusieurs valeurs, y compris une valeur limite.
- ☐ Je peux expliquer le rôle de chaque ligne que j’ai écrite.
- ☐ Je sais expliquer pourquoi `input()` renvoie une chaîne de caractères.
- ☐ Je sais distinguer un paramètre, un appel de fonction et une valeur de retour.

En cas de blocage

Procédez dans cet ordre : relisez le message d’erreur ; repérez la ligne concernée ; affichez les valeurs et leurs types ; comparez avec un exemple du cours ; demandez un indice à une autre personne ou à l’enseignant.

Si l’usage d’une IA générative est autorisé pour cet exercice, demandez un **indice** ou l’explication de l’erreur, pas le programme complet. Vous restez responsable du code rendu et devez pouvoir l’expliquer. Signalez brièvement tout usage autorisé.

Fichiers à conserver : `bonjour.py`, `ex05.py`, `ex06.py`, `ex07.py`, `ex08.py`.

FIN DU NIVEAU 1 – sauvegardez votre travail !