

Chapitre 1

Premiers programmes et premières erreurs

Dans ce chapitre, l'élève découvre les premières notions de programmation Python : les expressions, les variables, les types de données, les opérations, la concaténation de chaînes de caractères, les conversions de types et les premières erreurs.

Exercice 1 – Priorité des opérations

1. Parenthésage explicite

Écrire les expressions suivantes avec un **parenthésage explicite** permettant d'indiquer l'ordre de priorité des opérations.

```
1  
1 >>> 8 - 3 - 2
```

```
2  
1 >>> 2 + 3 * 4
```

```
3  
1 >>> 24 / 3 ** 2
```

```
4  
1 >>> 20 / 5 / 2
```

```
5  
1 >>> 20 / 5 * 2
```

```
6  
1 >>> 3 * 2 ** 3
```

```
7  
1 >>> 2 ** 2 ** 3
```

```
8  
1 >>> 29 // 5 % 3
```

```
9  
1 >>> 7 % 2 ** 3
```

2. Réduire le nombre de parenthèses

Écrire les expressions suivantes avec **le moins de parenthèses possible** :

```
1  
1 >>> 4 + (7 * (6 - 2))
```

```
2  
1 >>> (8 - 3) - ((4 * 5) / (2 ** 3))
```

Exercice 2 – Prédire la valeur des variables

Avant d'exécuter un programme, il est important de savoir suivre l'évolution des variables instruction par instruction.

Consigne : Pour chaque séquence, déterminer sans exécuter le programme la valeur affichée.

Vérifier ensuite la réponse en créant un programme dans l'éditeur Python d'IDE PyBak.

Séquence 1

```
1 a = 7
2 a = a + 3
3 b = a
4 b = b ** 2 - a
5 print(b)
```

Question : Quelle valeur sera affichée ?

Séquence 2

```
1 a = 8
2 b = 5
3
4 a = a - b
5 b = b + a
6 a = b - a
7
8 print(a, b)
```

Question : Quelle valeur sera affichée ?

Séquence 3

On souhaite échanger les valeurs de deux variables.

```
1 a = 12
2 b = 7
3
4 a = a - b
5 b = b + a
6 a = b - a
7
8 print(a, b)
```

Question : Quelle valeur sera affichée ?

À retenir : expliquer comment les valeurs de **a** et **b** ont été modifiées à chaque étape.

Exercice 3 – Évolution d'un prix

Le prix d'une matière première est de **640 € la tonne** au début de l'année.

Au cours de l'année :

- au premier trimestre, le prix augmente de **120 €**;
- au deuxième trimestre, il augmente de **15 %**;
- au troisième trimestre, il diminue de **30 %**;
- au quatrième trimestre, il diminue de **75 €**.

Travail demandé

1. Dans votre espace personnel, créer un répertoire :

```
1 Chapitre1
```

2. Dans ce répertoire, créer un sous-répertoire :

```
1 TP1
```

3. Ouvrir **IDE PyBak** et créer un nouveau fichier Python dans l'éditeur.
4. Enregistrer le fichier sous le nom :

```
1 prix_matiere.py
```

5. Compléter le programme suivant dans l'éditeur :

```
1 prix = 640          # prix au début de l'année
2
3 prix = .....        # prix à la fin du premier trimestre
4
5 prix = .....        # prix à la fin du deuxième trimestre
6
7 prix = .....        # prix à la fin du troisième trimestre
8
9 prix = .....        # prix à la fin de l'année
10
11 print("Prix final :", prix)
```

6. Exécuter le programme avec **Run** → **Run Module**.

Questions

1. Quelle est la valeur du prix après le premier trimestre ?
2. Quelle est la valeur du prix après le deuxième trimestre ?
3. Quelle est la valeur du prix après le troisième trimestre ?
4. Quel est le prix à la fin de l'année ?
5. Modifier le programme afin d'afficher le prix après chaque trimestre.

Exercice 4 – Conversion de température

La température f , en degrés Fahrenheit, est obtenue à partir de la température c , en degrés Celsius, avec la formule :

$$f = 1,8 \times c + 32$$

On souhaite créer un programme permettant de convertir une température exprimée en degrés Celsius en degrés Fahrenheit.

1. Première version du programme

Dans **IDE PyBak**, créer un nouveau fichier dans le dossier **Chapitre1/TP1** et l'enregistrer sous :

```
1 temperature.py
```

Saisir dans l'éditeur :

```
1 c = input("Température en degrés Celsius ? ")
2 f = 1.8 * c + 32
3 print("La température en degrés Fahrenheit est de", f)
```

Exécuter le programme.

Questions

1. Une erreur est-elle produite ?
2. Quel type de problème rencontre-t-on ?
3. Ajouter dans le programme l'instruction suivante juste après l'instruction `input` :

```
1 print(type(c))
```

4. Exécuter à nouveau le programme.
5. Que représente la valeur affichée par `type(c)` ?

2. Conversion de la saisie

Modifier la première instruction :

```
1 c = input("Température en degrés Celsius ? ")
```

en :

```
1 c = float(input("Température en degrés Celsius ? "))
```

Exécuter de nouveau le programme.

Questions

1. Quel est maintenant le type de `c` ?
2. Quel est le rôle de la fonction `float()` ?
3. Consulter la documentation de cette fonction avec :

```
1 help(float)
```

3. Vérification du résultat

Tester le programme avec :

```
1 25
```

La température obtenue doit être :

```
1 La température en degrés Fahrenheit est de 77.0
```

Modifier ensuite le programme pour qu'il affiche :

```
1 25.0 °C correspond à 77.0 °F
```

Exercice 5 – Concaténation et conversion de types

Une chaîne de caractères est une suite de caractères entourée par des guillemets simples ou doubles.

En Python, l'opérateur `+` permet de **concaténer deux chaînes de caractères**, c'est-à-dire de les assembler.

1. Concaténer deux chaînes

Dans l'éditeur IDE PyBak, créer un fichier nommé :

```
1 concat.py
```

Saisir le programme suivant :

```
1 prenom = "Gilbert"
2 nom = "BANANKO"
3
4 nom_complet = prenom + " " + nom
5
6 print(nom_complet)
```

Question : Quel résultat obtient-on ?

2. Concaténer plusieurs chaînes

Compléter le programme suivant :

```
1 ville = "Lomé"
2 pays = "Togo"
3
4 message = ville + " est une ville du " + .....
5
6 print(message)
```

Question : Quel message doit être affiché ?

3. Que se passe-t-il avec un entier ?

Créer le programme suivant :

```
1 age = 25
2
3 message = "J'ai " + age + " ans."
4
5 print(message)
```

Exécuter le programme.

L'opérateur + ne permet pas directement de concaténer une chaîne de caractères (**str**) avec un entier (**int**) ou un nombre décimal (**float**).

Question : Quel type d'erreur obtient-on ?

4. Utiliser la fonction **str()**

Pour transformer une valeur numérique en chaîne de caractères, on peut utiliser la fonction **str()**.

Corriger le programme précédent :

```
1 age = 25
2
3 message = "J'ai " + str(age) + " ans."
4
5 print(message)
```

Question : Pourquoi le programme fonctionne-t-il maintenant ?

5. Concaténer avec un nombre décimal

On souhaite afficher le prix d'un produit.

```
1 prix = 149.90
2
3 message = "Le prix du produit est de " + ..... + " F CFA"
4
```

```
5 print(message)
```

Compléter le programme en utilisant la fonction `str()`.

6. Utiliser une f-string

Python permet également de construire une chaîne contenant directement la valeur d'une variable grâce aux **f-strings**.

Observer le programme suivant :

```
1 age = 25
2
3 message = f"J'ai {age} ans."
4
5 print(message)
```

Question : Quel résultat obtient-on ?

7. Comparaison entre `str()` et les f-strings

Les deux programmes suivants produisent le même résultat.

```
1 age = 25
2
3 message = "J'ai " + str(age) + " ans."
4
5 print(message)
```

et :

```
1 age = 25
2
3 message = f"J'ai {age} ans."
4
5 print(message)
```

Avec une f-string, on place le nom de la variable entre accolades `{ }` à l'intérieur d'une chaîne précédée de la lettre `f`.

Cette écriture est particulièrement pratique lorsque plusieurs variables doivent être intégrées dans une phrase.

8. Application

Créer un programme qui demande à l'utilisateur :

- son prénom ;
- son âge ;
- sa ville.

Le programme doit ensuite afficher une phrase de la forme :

```
1 Bonjour Gilbert ! Tu as 25 ans et tu habites à Lomé.
```

Utiliser obligatoirement une **f-string**.

9. Défi

Modifier le programme afin qu'il demande également :

- la température actuelle en degrés Celsius ;
- puis affiche cette température en degrés Fahrenheit.

Le résultat devra avoir une forme similaire à :

```
1 Bonjour Gilbert !  
2 Tu habites à Lomé.  
3 La température est de 28.0 °C, soit 82.4 °F.
```

Exercice 6 – Calcul du prix après remise

Un magasin propose un ordinateur dont le prix initial est de **480 000 F CFA**.

Pendant une promotion, le magasin applique une réduction de **15 %**.

Travail demandé

Dans IDE PyBak, créer le fichier :

```
1 remise.py
```

Écrire dans l'éditeur un programme qui :

1. stocke le prix initial dans une variable `prix` ;
2. calcule le montant de la réduction ;
3. calcule le prix après réduction ;
4. affiche le prix initial, la réduction et le prix final.

Le programme devra produire un résultat de la forme :

```
1 Prix initial : ...  
2 Réduction : ...  
3 Prix final : ...
```

Question supplémentaire

Modifier le programme pour appliquer ensuite une deuxième réduction de **5 %** au prix déjà réduit.

Exercice 7 – Conversion d’une durée

On souhaite convertir une durée exprimée en **secondes** en minutes et secondes.

Par exemple :

```
1 367 secondes
```

correspond à :

```
1 6 minutes et 7 secondes
```

Travail demandé

Créer dans Chapitre1/TP1 le fichier :

```
1 duree.py
```

Dans l’éditeur Python, écrire un programme contenant :

```
1 duree = 367
2
3 minutes = ...
4 secondes = ...
5
6 print("Durée :", minutes, "minutes et", secondes, "secondes")
```

Compléter le programme en utilisant les opérateurs :

```
1 //
```

et :

```
1 %
```

Tester avec

```
1 367
```

puis avec :

```
1 542
```

Exercice 8 – Corriger un programme

Le programme suivant doit calculer la moyenne de trois notes :

```
1 note1 = 14
2 note2 = 11
3 note3 = 16
4
5 moyenne = note1 + note2 + note3 / 3
6
7 print("La moyenne est :", moyenne)
```

Le programme s'exécute mais le résultat obtenu n'est pas celui attendu.

Travail demandé

1. Ouvrir un fichier `moyenne.py` dans l'éditeur IDE PyBak.
2. Recopier le programme.
3. Exécuter le programme.
4. Identifier l'erreur.
5. Corriger le programme en utilisant des parenthèses.
6. Exécuter à nouveau le programme.
7. Vérifier que la moyenne obtenue est correcte.

Bilan – Ce qu'il faut retenir

À la fin de cette série d'exercices, l'élève doit notamment maîtriser :

- les **priorités des opérations** ;
- les parenthèses en Python ;
- les opérateurs `+`, `-`, `*`, `/`, `//`, `%` et `**` ;
- l'affectation des variables ;
- l'évolution de la valeur d'une variable ;
- les types `str`, `int` et `float` ;
- la fonction `input()` ;
- la conversion avec `int()`, `float()` et `str()` ;
- la concaténation de chaînes de caractères ;
- l'utilisation des **f-strings** ;
- les erreurs de syntaxe et les exceptions ;
- la création et l'enregistrement d'un fichier Python dans **l'éditeur IDE PyBak** ;
- l'exécution d'un programme depuis l'éditeur.