

Travaux Pratiques N 2

Variables et Opérateurs en Python

Module : Python

Programmation et Structures de Contrôle

Dr. ABOUABID Hamza

31 octobre 2025

Table des matières

1 Exercices Pratiques	3
1.1 Exercice 1 : Déterminer si un Nombre est Pair ou Impair	3
1.2 Exercice 2 : Calcul de l'Indice de Masse Corporelle (IMC)	3
1.3 Exercice 3 : Détermination de l'Année Bissextille	4
1.4 Exercice 4 : Boucle ‘for‘ et Conditions	4
1.5 Exercice 5 : Boucle ‘while‘ et Conditions	5
1.6 Exercice 6 : Affichage Infini avec Boucle ‘while‘	6
1.7 Exercice 7 : Calcul du Factoriel d'un Nombre	6
1.8 Projet Final - Jeu de Devine le Nombre	7
1.8.1 Description du Projet	7
1.8.2 Étapes à Suivre	7
1.8.3 Exemple de Résultat Attendu	8
2 Annexes	8
2.1 Références Utiles	8

1 Exercices Pratiques

1.1 Exercice 1 : Déterminer si un Nombre est Pair ou Impair

Objectif

Apprendre à utiliser les instructions conditionnelles pour déterminer si un nombre est pair ou impair.

Consignes :

1. Écrivez un programme qui :

- Demande à l'utilisateur de saisir un nombre entier.
- Utilise une condition pour déterminer si le nombre est pair ou impair.
- Affiche "Le nombre [nombre] est pair." ou "Le nombre [nombre] est impair." en conséquence.

Exemple de Résultat Attendu

Entrez un nombre entier : 4
Le nombre 4 est pair.

Entrez un nombre entier : 7
Le nombre 7 est impair.

1.2 Exercice 2 : Calcul de l'Indice de Masse Corporelle (IMC)

Objectif

Apprendre à utiliser les conditions pour catégoriser des données basées sur des calculs.

Consignes :

1. Écrivez un programme qui :

- Prend en entrée le poids (en kg) et la taille (en m) de l'utilisateur.
- Calcule l'IMC en utilisant la formule : $\text{IMC} = \text{poids} / \text{taille}^2$.
- Affiche une catégorie selon l'échelle suivante :
 - $\text{IMC} < 18.5$: "Maigre"
 - $18.5 \leq \text{IMC} < 25$: "Normal"
 - $25 \leq \text{IMC} < 30$: "Surpoids"
 - $\text{IMC} \geq 30$: "Obésité"

Exemple de Résultat Attendu

Entrez votre poids (kg) : 70
Entrez votre taille (m) : 1.75
Votre IMC est de 22.86. Catégorie : Normal

1.3 Exercice 3 : Détermination de l'Année Bissextille

Objectif

Utiliser des conditions imbriquées pour déterminer si une année est bissextile.

Consignes :

1. Écrivez un programme qui :

- Demande à l'utilisateur de saisir une année.
- Utilise des conditions pour déterminer si l'année est bissextile selon les règles suivantes :
 - Une année est bissextile si elle est divisible par 4.
 - Mais si elle est divisible par 100, elle n'est pas bissextile.
 - Cependant, si elle est aussi divisible par 400, elle est bissextile.
- Affiche "L'année [année] est bissextile." ou "L'année [année] n'est pas bissextile." en conséquence.

Exemple de Résultat Attendu

```
Entrez une année : 2020
L'année 2020 est bissextile.
```

```
Entrez une année : 1900
L'année 1900 n'est pas bissextile.
```

```
Entrez une année : 2000
L'année 2000 est bissextile.
```

1.4 Exercice 4 : Boucle 'for' et Conditions

Objectif

Apprendre à utiliser les boucles `for` combinées avec des conditions pour itérer et filtrer des données.

Consignes :

1. Déclarez une liste de nombres : `nombres = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]`.
2. Utilisez une boucle `for` pour parcourir la liste et afficher uniquement les nombres pairs.
3. Modifiez la boucle pour afficher un message personnalisé pour chaque nombre pair, par exemple "Nombre pair : [nombre]".

Exemple de Résultat Attendu

```
Nombre pair : 2
Nombre pair : 4
Nombre pair : 6
Nombre pair : 8
Nombre pair : 10
```

1.5 Exercice 5 : Boucle ‘while’ et Conditions

Objectif

Apprendre à utiliser les boucles `while` avec des conditions pour contrôler le flux d'exécution.

Consignes :

1. Écrivez un programme qui :

- Initialise une variable `compteur` à 1.
- Utilise une boucle `while` pour afficher les nombres de 1 à 10.
- À chaque itération, vérifiez si le nombre est divisible par 3. Si oui, affichez "Fizz" au lieu du nombre.

Exemple de Résultat Attendu

```
1
2
Fizz
4
5
Fizz
7
8
Fizz
10
```

1.6 Exercice 6 : Affichage Infini avec Boucle ‘while’

Objectif

Comprendre l'utilisation des boucles infinies et la gestion des entrées utilisateur.

Consignes :

1. Écrivez un programme qui :

- Utilise une boucle `while` infinie pour répéter indéfiniment.
- À chaque itération, demande à l'utilisateur de saisir deux valeurs entières.
- Calcule la somme, la différence et le produit des deux valeurs.
- Si le deuxième nombre n'est pas zéro, calcule également la division entière, l'exponentiation et le modulo.
- Affiche les résultats des opérations.
- Gère les cas où le deuxième nombre est zéro en affichant un message d'erreur approprié.

Note Importante

Pour sortir d'une boucle infinie, utilisez `Ctrl + C` ou implémentez une condition de sortie avec `break`.

1.7 Exercice 7 : Calcul du Factoriel d'un Nombre

Objectif

Apprendre à utiliser les boucles `for` et les conditions pour calculer le factoriel d'un nombre.

Consignes :

1. Écrivez un programme qui :

- Demande à l'utilisateur de saisir un nombre entier positif.
- Utilise une boucle `for` pour calculer le factoriel du nombre saisi.
- Gère les cas où le nombre saisi est négatif en affichant un message d'erreur.
- Affiche le résultat du factoriel.

Exemple de Résultat Attendu

```
Entrez la valeur de "n" : 5
5! = 120
```

```
Entrez la valeur de "n" : -3
Erreur, le factoriel d'un nombre négatif n'existe pas.
```

1.8 Projet Final - Jeu de Devine le Nombre

Objectif

Appliquer les connaissances sur les conditions et les boucles pour développer un petit jeu interactif en Python.

1.8.1 Description du Projet

Créez un jeu où l'utilisateur doit deviner un nombre généré aléatoirement par le programme. Le jeu doit inclure les fonctionnalités suivantes :

- ✓ Le programme génère un nombre secret entre 1 et 100.
- ✓ L'utilisateur a un nombre limité de tentatives (par exemple, 10).
- ✓ Après chaque tentative, le programme indique si le nombre secret est plus grand ou plus petit que la tentative de l'utilisateur.
- ✓ Si l'utilisateur trouve le nombre avant d'épuiser les tentatives, il gagne. Sinon, il perd et le programme révèle le nombre secret.
- ✓ À la fin du jeu, proposez à l'utilisateur de rejouer.

1.8.2 Étapes à Suivre

1. Importation des Modules :

- Utilisez le module `random` pour générer le nombre secret.

2. Génération du Nombre Secret :

- Utilisez `random.randint(1, 100)` pour générer un nombre entre 1 et 100.

3. Saisie des Tentatives :

- Utilisez une boucle `while` pour permettre à l'utilisateur de saisir des tentatives jusqu'à ce qu'il gagne ou épuise les tentatives.

4. Vérification des Tentatives :

- Comparez la tentative de l'utilisateur avec le nombre secret.
- Indiquez si le nombre secret est plus grand ou plus petit.

5. Gestion des Tentatives :

- Comptez le nombre de tentatives et arrêtez la boucle après le nombre maximum.

6. Rejouer :

- Après la fin du jeu, demandez à l'utilisateur s'il souhaite rejouer.

1.8.3 Exemple de Résultat Attendu

Exemple de Résultat Attendu

Bienvenue dans le jeu de Devine le Nombre!
Je pense à un nombre entre 1 et 100.
Vous avez 10 tentatives pour le deviner.

Entrez votre tentative : 50
Le nombre secret est plus grand.

Il vous reste 9 tentatives.

Entrez votre tentative : 75
Le nombre secret est plus petit.

Il vous reste 8 tentatives.

Entrez votre tentative : 63
Félicitations! Vous avez trouvé le nombre 63.

Voulez-vous rejouer? (oui/non) : non
Merci d'avoir joué! À bientôt.

Note Importante

Conseils pour réussir le projet :

- ★ Testez votre code régulièrement après chaque étape.
- ★ Gérez les erreurs de saisie avec des blocs `try-except`.
- ★ Ajoutez des messages encourageants pour améliorer l'expérience utilisateur.
- ★ Pensez à valider que l'entrée de l'utilisateur est bien dans l'intervalle [1, 100].

2 Annexes

2.1 Références Utiles

- ▶ Documentation Officielle Python - Contrôle de Flux
- ▶ Real Python - Conditions en Python
- ▶ Real Python - Boucles `for` en Python
- ▶ Real Python - Boucles `while` en Python
- ▶ W3Schools - Python Tutorial

Bonne chance dans vos travaux pratiques!
