
Travaux Pratiques 3

Exercices Supplémentaires

Structures de Contrôle

Module : Python

Programmation et Structures de Contrôle

Dr. ABOUABID Hamza

7 novembre 2025

École Marocaine des Sciences de l'Ingénieur

10 Exercices pour approfondir vos compétences

Exercices Supplémentaires

Exercice 1 : FizzBuzz

Consignes

Écrivez un programme qui affiche les nombres de 1 à 100. Pour chaque nombre :

- Affichez "**Fizz**" si le nombre est divisible par 3.
- Affichez "**Buzz**" si le nombre est divisible par 5.
- Affichez "**FizzBuzz**" si le nombre est divisible par 3 et par 5.
- Sinon, affichez simplement le nombre.

Points clés :

- Utiliser une boucle `for` avec `range(1, 101)`
- Utiliser l'opérateur modulo (%) pour tester la divisibilité
- Tester d'abord la divisibilité par 15 (3 et 5)

Exercice 2 : Parcours de Labyrinthe

Consignes

Écrivez un programme qui simule un joueur dans un labyrinthe. Le labyrinthe est représenté par une série de cases numérotées de 1 à 50. Le joueur commence à la case 1 et peut avancer en lançant un dé (valeurs entre 1 et 6).

Règles du jeu :

1. Si le joueur tombe sur une case **multiple de 7**, il **recule de 3 cases**.
2. Si le joueur tombe sur une case **multiple de 13**, il **saute son prochain tour** (continue pour passer l'itération de la boucle).
3. Si le joueur **atteint exactement la case 50**, il **gagne** et le jeu se termine (`break` pour sortir de la boucle).
4. Si le joueur **dépasse la case 50**, il revient à la case précédente.

Affichage requis :

Votre programme doit afficher à chaque tour :

- ✓ La valeur obtenue au lancer du dé.
- ✓ La position actuelle du joueur.
- ✓ Les éventuelles actions spéciales (reculer, sauter un tour, etc.).

Exemple de Résultat

```
Tour 1 :  
Lancer du dé : 5  
Position actuelle : 6  
  
Tour 2 :  
Lancer du dé : 3  
Position actuelle : 9  
  
Tour 3 :  
Lancer du dé : 6  
Position actuelle : 15  
Case multiple de 7. Vous reculez de 3 cases.  
Position actuelle : 12  
  
Tour 4 :  
Lancer du dé : 4  
Position actuelle : 16  
  
Tour 5 :  
Lancer du dé : 6  
Position actuelle : 22  
Case multiple de 13. Vous passez votre prochain tour.  
  
Tour 6 :  
Vous passez votre tour.  
  
Tour 7 :  
Lancer du dé : 5  
Position actuelle : 27  
  
...  
  
Tour 14 :  
Lancer du dé : 2  
Position actuelle : 50  
Félicitations ! Vous avez gagné !
```

Note Importante

Astuce : Utilisez `import random` et `random.randint(1, 6)` pour simuler le lancer du dé.

Exercice 3 : Table de Multiplication

Consignes

Écrivez un programme qui demande un nombre à l'utilisateur et affiche la table de multiplication de ce nombre de 1 à 10.

Exemple de Résultat

```
Entrez un nombre : 3
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
3 x 10 = 30
```

Exercice 4 : Somme des N Premiers Entiers

Consignes

Écrivez un programme qui demande un nombre entier positif à l'utilisateur, puis calcule et affiche la somme des N premiers entiers.

Exemple de Résultat

```
Entrez un nombre : 5
La somme des 5 premiers entiers est : 15
(Calcul : 1 + 2 + 3 + 4 + 5 = 15)
```

Note Importante

Défi bonus : Comparez votre résultat avec la formule mathématique : $\text{somme} = \frac{n \times (n+1)}{2}$

Exercice 5 : Trouver les Nombres Premiers

Consignes

Écrivez un programme qui affiche tous les nombres premiers entre 1 et 100. Utilisez une boucle et des conditions pour vérifier si un nombre est premier.

Rappel : Un nombre premier est un nombre entier naturel supérieur à 1 qui n'est divisible que par 1 et par lui-même.

Algorithme suggéré :

- Pour chaque nombre n de 2 à 100
- Tester si n est divisible par un nombre entre 2 et \sqrt{n}
- Si aucun diviseur n'est trouvé, n est premier

Exercice 6 : Compter les Diviseurs

Consignes

Écrivez un programme qui demande un nombre à l'utilisateur, puis affiche tous ses diviseurs ainsi que leur nombre total.

Exemple de Résultat

```
Entrez un nombre : 12
Diviseurs de 12 : 1, 2, 3, 4, 6, 12
Nombre total de diviseurs : 6
```

Méthode : Un nombre a est un diviseur de n si $n \bmod a = 0$.

Exercice 7 : Vérifier si un Nombre est Parfait

Consignes

Écrivez un programme qui demande un nombre à l'utilisateur et vérifie s'il est un nombre parfait. Un nombre parfait est égal à la somme de ses diviseurs propres (diviseurs autres que le nombre lui-même).

Exemple de Résultat

```
Entrez un nombre : 6
Diviseurs propres de 6 : 1, 2, 3
Somme des diviseurs : 1 + 2 + 3 = 6
6 est un nombre parfait.

Entrez un nombre : 12
Diviseurs propres de 12 : 1, 2, 3, 4, 6
Somme des diviseurs : 1 + 2 + 3 + 4 + 6 = 16
12 n'est pas un nombre parfait.
```

Note Importante

Le saviez-vous ? Les premiers nombres parfaits sont : 6, 28, 496, 8128. Ils sont très rares!

Exercice 8 : Conversion d'Unités**Consignes**

Écrivez un programme qui demande une distance en kilomètres et la convertit en miles.

Note : 1 kilomètre = 0.621371 mile

Exemple de Résultat

```
Entrez une distance en kilomètres : 10
10 kilomètres = 6.21371 miles
```

Note Importante

Extension possible : Créez un convertisseur multi-unités qui propose plusieurs conversions (km → miles, °C → °F, kg → lbs, etc.)

Exercice 9 : Nombre de Chiffres**Consignes**

Écrivez un programme qui demande un nombre entier positif à l'utilisateur, puis affiche le nombre de chiffres dans ce nombre.

Exemple de Résultat

```
Entrez un nombre : 12345
Le nombre 12345 a 5 chiffres.

Entrez un nombre : 7
Le nombre 7 a 1 chiffre.
```

Deux approches possibles :

1. Utiliser `len(str(nombre))` pour convertir en chaîne et compter
2. Utiliser une boucle avec division par 10 jusqu'à obtenir 0

Exercice 10 : Palindrome Numérique

Consignes

Écrivez un programme qui demande un nombre à l'utilisateur et vérifie s'il s'agit d'un palindrome. Un palindrome est un nombre qui reste le même lorsqu'il est inversé.

Exemple de Résultat

```
Entrez un nombre : 121
121 est un palindrome.
```

```
Entrez un nombre : 123
123 n'est pas un palindrome.
```

```
Entrez un nombre : 12321
12321 est un palindrome.
```

Méthode suggérée :

- Convertir le nombre en chaîne de caractères
- Inverser la chaîne avec `chaine[::-1]`
- Comparer la chaîne originale avec la chaîne inversée

Fin des Exercices Supplémentaires

Bon courage dans vos travaux pratiques !
