Exercice 1 : Utilisation de *args

```python
# 1. Créez une fonction `addition` qui prend un nombre variable d'arguments et retourne leur somme.
def addition(*args):
    somme = sum(args)
    return somme

# Testez la fonction avec différentes séries d'arguments.
print(addition(1, 2, 3))           # Devrait afficher 6
print(addition(4, 5))              # Devrait afficher 9
print(addition(1, 2, 3, 4, 5, 6))  # Devrait afficher 21

# 2. Créez une fonction `liste_arguments` qui prend un nombre variable d'arguments et les affiche.
def liste_arguments(*args):
    for arg in args:
        print(arg)

# Testez la fonction avec différentes séries d'arguments.
liste_arguments('a', 'b', 'c')  # Devrait afficher chaque caractère sur une nouvelle ligne
liste_arguments(1, 2, 3, 4)     # Devrait afficher chaque nombre sur une nouvelle ligne
```

Exercice 2 : Utilisation de **kwargs

```python
# 1. Créez une fonction `infos_personnelles` qui prend un nombre variable
d'arguments par mot-clé et affiche ces informations.
def infos_personnelles(**kwargs):
    for cle, valeur in kwargs.items():
        print(f"{cle}: {valeur}")

# Testez la fonction avec différentes séries d'arguments.
infos_personnelles(nom="Alice", age=30, ville="Paris")
# Devrait afficher :
# nom: Alice
# age: 30
# ville: Paris

infos_personnelles(prenom="Bob", profession="Ingénieur")
# Devrait afficher :
# prenom: Bob
# profession: Ingénieur

# 2. Créez une fonction `multiplication_kwargs` qui multiplie les valeurs des
arguments par mot-clé qui sont des nombres.
def multiplication_kwargs(**kwargs):
    resultat = 1
    for cle, valeur in kwargs.items():
        if isinstance(valeur, (int, float)):
            resultat *= valeur
    return resultat

# Testez la fonction avec différentes séries d'arguments.
print(multiplication_kwargs(a=2, b=3, c=4))  # Devrait afficher 24
print(multiplication_kwargs(x=1, y=10, z=0.5))  # Devrait afficher 5.0
```

Exercice 3 : Combinaison de *args et **kwargs

```python
# 1. Créez une fonction `affiche_details` qui accepte un argument positionnel, un nombre variable d'arguments positionnels et un nombre variable d'arguments par mot-clé.
def affiche_details(arg1, *args, **kwargs):
    print(f"arg1: {arg1}")
    for argument in args:
        print(f"Argument supplémentaire: {argument}")
    for cle, valeur in kwargs.items():
        print(f"{cle}: {valeur}")

# Testez la fonction avec différentes combinaisons d'arguments.
affiche_details("Principal", 1, 2, 3, nom="Alice", age=30)
# Devrait afficher :
# arg1: Principal
# Argument supplémentaire: 1
# Argument supplémentaire: 2
# Argument supplémentaire: 3
# nom: Alice
# age: 30

affiche_details("Test", a=10, b=20)
# Devrait afficher :
# arg1: Test
# a: 10
# b: 20
```