Exercices Python Résumés

Voici tous les exercices extraits du document :

1. **Écrire une fonction** `maximum_2(a, b)` qui retourne le maximum entre `a` et `b`.
2. **Écrire une fonction** `maximum_3(a, b, c)` qui retourne le maximum entre `a`, `b` et `c` (vous pouvez utiliser `maximum_2`).
3. **Écrire une fonction qui retourne la factorielle d'un nombre passé en paramètre.**
4. **Écrire une fonction récursive** acceptant comme paramètre un entier positif `n` et retournant le `n`-ième terme de la suite de Fibonacci.
5. **Écrire une fonction** `compter_voyelles_consonnes(ch)` qui retourne le nombre de voyelles et de consonnes dans `ch` (le paramètre peut contenir des voyelles, consonnes ou espaces).
6. **Écrire une fonction** `char_in_string(char, string)` qui prend deux paramètres (`char` un caractère et `string` une chaîne de caractères) et retourne la position de la première occurrence de `char` dans `string`, ou `None` si absent.
7. **Écrire une fonction qui reçoit un nombre variable de paramètres et retourne un dictionnaire** contenant les types présents ainsi que leur nombre d'occurrences.
8. **Écrire une fonction** `operation(a, b, c, f1, f2)` qui accepte deux fonctions callback et retourne le résultat de la composition des deux (exemple : `operation(2, 3, 6, somme, produit)` retourne `30 = (2+3) * 6`).
9. **Écrire une fonction génératrice qui retourne les nombres premiers inférieurs à** `x`.
10. **Écrire une fonction génératrice qui retourne les** `n` **premiers termes de la suite de Fibonacci.**

Si tu veux que je te fournisse des solutions pour certains de ces exercices, dis-moi lesquels !

Voici les solutions aux exercices demandés :

1. Fonction `maximum_2(a, b)`

python

```python
def maximum_2(a, b):
    return a if a > b else b
```

```python
print(maximum_2(4, 7))  # Affiche 7
print(maximum_2(10, 3)) # Affiche 10
```

2. Fonction `maximum_3(a, b, c)`

python

```python
def maximum_2(a, b):
    return a if a > b else b

def maximum_3(a, b, c):
    return maximum_2(maximum_2(a, b), c)

print(maximum_3(4, 7, 2))  # Affiche 7
print(maximum_3(10, 3, 15)) # Affiche 15
```

3. Fonction Factorielle

python

```python
def factorielle(n):
    result = 1
    for i in range(2, n + 1):
        result *= i
    return result

print(factorielle(5))  # Affiche 120
print(factorielle(0))  # Affiche 1
```

4. Fonction Récursive (Suite de Fibonacci)

python

```python
def fibonacci(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    return fibonacci(n - 1) + fibonacci(n - 2)
```

```python
print(fibonacci(5))  # Affiche 5
print(fibonacci(10)) # Affiche 55
```

5. Fonction `compter_voyelles_consonnes(ch)`

```python
def compter_voyelles_consonnes(ch):
    voyelles = "aeiouyAEIOUY"
    nb_voyelles = sum(1 for char in ch if char in voyelles)
    nb_consonnes = sum(1 for char in ch if char.isalpha() and char not in voyelles)
    return nb_voyelles, nb_consonnes

print(compter_voyelles_consonnes("bonjour"))  # Affiche (3, 4)
print(compter_voyelles_consonnes("Python"))   # Affiche (1, 5)
```

6. Fonction `char_in_string`

```python
def char_in_string(char, string):
    return string.find(char) if char in string else None

print(char_in_string('o', 'bonjour'))  # Affiche 1
print(char_in_string('a', 'bonjour'))  # Affiche None
```

7. Fonction qui compte les types de paramètres

```python
def compute_types(*params):
    compteur = {}
    for e in params:
        type_str = type(e).__name__
        compteur[type_str] = compteur.get(type_str, 0) + 1
    return compteur

print(compute_types(2, 'bonjour', 'True', True, 3, '3', 9, 7, False))
# Affiche {'int': 4, 'str': 3, 'bool': 2}
```

8. Fonction `operation(a, b, c, f1, f2)`

python

```python
def somme(a, b):
    return a + b

def produit(a, b):
    return a * b

def operation(a, b, c, f1, f2):
    return f2(f1(a, b), c)

print(operation(2, 3, 6, somme, produit))  # Affiche 30
print(operation(2, 3, 6, produit, somme))  # Affiche 12
```

9. Fonction génératrice pour les nombres premiers inférieurs à `x`

python

```python
def est_premier(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True

def get_all_premier(x):
    for i in range(2, x):
        if est_premier(i):
            yield i

print(list(get_all_premier(20)))  # Affiche [2, 3, 5, 7, 11, 13, 17, 19]
```

10. Fonction génératrice pour la suite de Fibonacci

python

```python
def fibonacci_gen(n):
    a, b = 0, 1
    for _ in range(n):
        yield a
        a, b = b, a + b
```

```
print(list(fibonacci_gen(10)))  # Affiche [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

Si tu as besoin d’explications ou d’adaptations sur certaines fonctions, dis-moi !