Partie 1 – Variables, types, listes et dictionnaires

A. Notions de base

Q1. Qu'est-ce qu'une variable en Python ? Donne un exemple concret (pas de calcul 1+1).

Q2. Associe chaque valeur Python à son type. Coche la bonne case.

Quel est le type de 930 ?	
☐ int	☐ float
☐ str	☐ bool

Quel est le type de 1.85 ?	
☐ int	☐ float
☐ str	☐ bool

Quel est le type de "Paris" ?	
☐ int	☐ float
☐ str	☐ bool

Quel est le type de True ?	
☐ int	☐ float
☐ str	☐ bool

Q3. Que va afficher le code suivant ? Écris le résultat dans la case.

```
distance_km = 930
vitesse = 110
duree = distance_km / vitesse
print(duree)
```

B. Opérations et f-strings

Q4. Complète le code pour calculer le coût du carburant d'un trajet de 400 km, avec une consommation de 7 L/100 km et un prix de 1.92 €/L.

```
distance = 400
consommation = 7
prix_litre = 1.92

litres = ...              # nombre de litres consommés
cout = ...               # coût total
print(f"Coût : {cout} €")
```

Résultat attendu : Coût : _____ €

Q5. Écris une instruction Python avec une f-string qui affiche :

Trajet de Nice à Paris : 930 km pour 3 passagers, soit 56.75 € par personne.

(Tu peux supposer que les variables sont déjà définies.)

C. Les listes

Q6. On a mesuré les temps de chute d'une bille (en secondes) depuis différentes hauteurs :

```
temps = [0.32, 0.45, 0.55, 0.64, 0.71, 0.78, 0.84, 0.90, 0.96, 1.01]
```

Que vaut temps[0] ? **Que vaut temps[4] ?** **Que vaut temps[-1] ?**	
Que renvoie len(temps) ? **Que renvoie max(temps) ?**	

Q7. On a fait une 11ème mesure : temps = 1.06 s. Écris la ligne de code qui l'ajoute à la liste.

Q8. Écris le code qui calcule et affiche le temps moyen de chute à partir de la liste ci-dessus. (Indice : utilise sum() et len())

D. Les dictionnaires

Q9. Le dictionnaire suivant représente la molécule d'eau. Complète les cases.

```
eau = {
    "nom": "Eau",
    "formule": "H2O",
    "H": 2,
    "O": 1
}
```

Que vaut eau["nom"] ?	
Que vaut eau["H"] ?	
Que vaut eau["O"] ?	

Q10. On veut ajouter la masse molaire de l'eau (18 g/mol) dans le dictionnaire. Écris la ligne de code correspondante.

Q11. On dispose des masses molaires atomiques : C = 12, H = 1, O = 16 g/mol. Écris le code complet qui calcule et stocke la masse molaire du dioxyde de carbone CO_2 (1 atome de C, 2 atomes de O) dans un dictionnaire co2.

Partie 2 – Les fonctions

A. Comprendre une fonction

Q1. Explique avec tes propres mots à quoi sert le mot-clé `return` dans une fonction Python.

Q2. Observe la fonction suivante, puis réponds aux questions.

```
def metres_en_km(distance_m):
    distance_km = distance_m / 1000
    return distance_km
```

Quel est le nom de la fonction ? **Quel est le nom du paramètre ?**	
Que renvoie `metres_en_km(2500)` ?	
Que renvoie `metres_en_km(42195)` ?	

Q3. Parmi les affirmations suivantes, coche toutes celles qui sont vraies.

☐	Une fonction doit être définie avant d'être appelée.
☐	On peut appeler une fonction autant de fois qu'on veut.
☐	Une fonction ne peut recevoir qu'un seul paramètre.
☐	Le mot-clé `def` sert à définir une nouvelle fonction.
☐	Une fonction qui contient `return` renvoie une valeur utilisable.

B. Écrire une fonction

Q4. Écris une fonction `secondes_en_heures(duree_s)` qui convertit une durée en secondes en heures. Puis appelle-la pour convertir 7200 secondes.

```
def secondes_en_heures(duree_s):
    ...                    # à compléter
    ...

resultat = ...
print(resultat)
```

Résultat attendu : 2.0 heures

Q5. On dispose des fonctions `metres_en_km` et `secondes_en_heures` déjà définies. Écris une fonction `vitesse_kmh(distance_m, temps_s)` qui calcule et renvoie une vitesse en km/h.

```
def vitesse_kmh(distance_m, temps_s):
    d_km = ...          # convertir les mètres
    t_h  = ...          # convertir les secondes
    v    = ...          # calculer la vitesse
    return v
```

C. Appliquer et composer des fonctions

Q6. En utilisant la fonction `vitesse_kmh`, calcule et affiche la vitesse d'un athlète ayant parcouru 400 m en 47 secondes.

Q7. Pourquoi est-il préférable d'utiliser des fonctions plutôt que de répéter le même calcul à chaque fois ? Donne deux raisons.

Q8. Mini-défi : Écris une fonction `temps_par_longueur(temps_total_s, nb_longueurs)` qui renvoie le temps moyen pour parcourir une longueur de 50 m. Puis appelle-la pour un nageur qui a fait 200 m en 128.5 secondes.

```
def temps_par_longueur(temps_total_s, nb_longueurs):
    ...                 # à compléter
    ...

# Appel pour 200 m en 128.5 s
...
print(f"Temps moyen par 50 m : {...} s")
```

Résultat attendu : 32.125 s

D. Documenter une fonction

Q9. La fonction ci-dessous est correcte mais non documentée. Ajoute une docstring (entre triple guillemets) qui explique ce qu'elle fait, ses paramètres et ce qu'elle renvoie.

```
def vitesse_kmh(distance_m, temps_s):
    """  ...  """
    d_km = metres_en_km(distance_m)
    t_h  = secondes_en_heures(temps_s)
    return d_km / t_h
```

Bon travail ! 🐍