1ère NSI — Quiz 02 : dictionnaires

Question 1 – vérification de syntaxe Python

Quelles syntaxes parmi les suivantes sont correctes en Python ? (sachant que " ;" équivaut à un retour à la ligne) – *attention, il y a un (petit) piège...*

a. `dico = ["a": 1, "b": 2]`
b. `dico = {"a": 1, "b": 2}`
c. `dico = ("a"=1, "b"=2)`
d. `dico = dict(); dico['a'] = 1; dico['b'] = 2`
e. `dico = ["a", 1, "b", 2]`

*Réponses **a**, **d**... et **e** !*

- *Ce sont les accolades ({}) qu'on utilise pour créer des dictionnaires.*
- *Mais on peut également définir un dictionnaire vide au moyen de la fonction `dict()` puis le "remplir" clé par clé comme dans la réponse **d** ici.*
- *Enfin la syntaxe de la réponse **e** est tout à fait correcte – même si elle ne renvoie pas du tout un dictionnaire mais plutôt une liste ! (oui, c'est hors programme, mais on peut faire des listes de ce qu'on veut – y compris avec des types de données différents comme ici).*

Question 2 – boucle sur un dictionnaire

```
1 dico = {'a': 1, 'b': 2}
2 for cle in dico:
3     print(cle)
```

Si j'execute ce code, qu'est-ce qui va s'afficher à l'écran ?

L'affichage sera :

```
a
b
```

*Pour rappel quand on parcourt un dictionnaire au moyen d'une boucle comme ici, ce sont implicitement les **clés** du dictionnaire que l'on parcourt. Si l'on voulait, dans le code précédent, accéder aux valeurs à l'intérieur de la boucle, on devrait utiliser `dico[cle]` ; et si l'on voulait que la boucle parcourre non pas les clés mais les valeurs du dictionnaire on devrait utiliser :*

`for valeur in dico.values()`

Question 3 – liste de dictionnaires / code mystère

```
1 LstNoms = [
2     {'prenom': 'Fatima', 'nom': 'Diallo'},
3     {'prenom': 'Lucas', 'nom': 'Fernandez'}
4     ]
5
6 trouve = False
7 cherche = input("Quel nom? ")
8 for i in range(len(LstNoms)):
```

```
9       if LstNoms[i]['nom'] == cherche:
10          print("Son prénom est", LstNoms[i]['prenom'])
11          trouve = True
12
13  if trouve == False:
14      print("Absent de la liste!")
```

Que fait ce code ? (Si l'utilisateur entre "Fernandez" ? "Dupont" ?)

Exercice certes un peu plus compliqué mais qui ressemble comme deux gouttes d'eau à ce qu'on a fait en TP avec le fichier des langues vivantes dans les lycées français à la rentrée 2022 (je vous invite à revoir les deux en parallèle dans le cadre de vos révisions !).

`LstNoms` est une liste de dictionnaires, et chacun ne comporte que deux clés, `nom` et `prenom`.

`trouve` est une variable booléenne (donc qui ne peut prendre qu'une de deux valeurs – `True` ou `False`) que l'on a initialisée à la valeur `False`.

Lisons en détails ce code à partir de la ligne 7 :

- *On demande (au moyen de la fonction `input()`) à l'utilisateur de taper un nom et on le stocke dans la variable `cherche`.*
- *La boucle principale parcourt cette liste sur toute sa longueur. A chaque élément (donc à chaque dictionnaire identifié par `LstNoms[i]`) :*
  - *On regarde la valeur associée à la clé `nom` et on la compare au contenu de la variable `cherche` :*
    - *Si ils sont égaux on affiche un message dans lequel on incorpore la valeur associée à la clé `prenom` et on positionne la variable `trouve` à `True`.*
    - *Dans le cas contraire, on ne fait rien.*

Arrivés à la sortie de la boucle, si on a trouvé au moins une fois le nom dans la liste, alors la variable `trouve` aura été positionnée à `True` – et on ne fait rien. (note : si on a trouvé le nom plusieurs fois ça ne change rien – tout ce qu'on aura fait, c'est positionner plusieurs fois la variable à `True`).

Si en revanche on n'a pas trouvé le nom dans la liste, alors la variable `trouve` aura conservé sa valeur initiale (donc `False`) et on affichera donc le message `"Absent de la liste!"`.

Spécifiquement, il est aisé de se convaincre que :

- *Pour "Fernandez" on affichera `"Son prénom est Lucas"` ;*
- *Pour "Dupont" on affichera `"Absent de la lsite!"`.*

Question 4 – message d'erreur

Que veut dire ceci, sachant que "dico" est un dictionnaire qui a été correctement défini ?

```
Traceback (most recent call last):
File "C:\Users\Marc\PyProj\exemple.py", line 3, in <module>
dico[1]+=1
```

```
~~~~~~~^
KeyError: 1
```

Comme toujours, un message d'erreur se lit de bas en haut : à la fin on voit la nature de l'erreur ("`KeyError:1`"), au-dessus l'instruction qui a posé problème ("`dico[1] += 1`"), et encore au-dessus la localisation de cette instruction (dans ce cas la 3ème ligne du fichier exemple.py).

En l'occurence l'erreur vient du fait qu'on a cherché à accéder à une clé, 1, qui n'existe pas. Deux remarques :

- *Le plus probable est que le développeur a fait ici une erreur fréquente – essayer d'accéder à une valeur d'un dictionnaire par indice alors qu'on y accède uniquement par clé. Pour rappel si on a défini `dico = {'pommes': 3, 'bananes': 5}` alors pour accéder au nombre de bananes on doit utiliser `dico['bananes']`.*
- *On peut en revanche tout à fait avoir des clés qui sont des entiers. Par exemple si on avait défini notre dictionnaire ainsi : `dico = {0: 0, 1: 1, 2: 4}`, l'instruction n'aurait pas posé problème (et dico serait devenu `{0: 0, 1: 2, 2: 4}`).*