Corrigé – Les dictionnaires

Exercice 1

voir feuille de cours

Exercice 2 – *(Les données EXIF d'une image)*

1. `>>> exif = {"largeur": 4592, "hauteur": 2584}`
2. `>>> exif["hauteur"]`

Exercice 3 – *(Un groupe de rock)*

1. ```
   rockband = {"chanteur": "Julien Casablancas", "guitariste rythmique":
   ↪  "Albert Hammond Jr", "guitariste principal": "Nick Valensi", "bassiste":
   ↪  "Nikolai Fraiture", "batteur": "Fabrizio Moretti"}
   ```

2. `rockband.values()`
3. Écrire une fonction `est_membre(rockband, nom)` renvoyant `True` si `nom` est présent dans le groupe, `False` sinon. On fera un parcours de dictionnaire.

   ```
   def est_membre(rockband, nom):
   for nom_membre in rockband.values():
       if nom_membre == nom:
          return True
   return False
   ```

4. Tester si `"Nick Valensi"` et `"Bjork"` appartiennent au groupe ou non.

   ```
   >>> est_membre(rockband, "Nick Valensi")\\
   True\\
   >>> est_membre(rockband, "Bjork")\\
   False\\
   ```

Exercice 4 – *(Manipulation de dictionnaires)*

1. `"charlotte"`
2. `21`
3. `print(panier[2]["nombre"])`
4. ```
   repertoire = [{'nom':'Dupont', 'tel':'5234'},
   {'nom':'Tournesol', 'tel':'5248'}, {'nom':'Dupond', 'tel':'3452'}]

   for i in range(len(repertoire)):
     if if repertoire[i]["nom"] == "Dupond":
       print(repertoire[i]['tel'])
   ```

Exercice 5 – *(Recherche de maximum)*

```
def recherche_max(dico):
  max = 0
  cle_max = None
  for cle in dico:
    if dico[cle] > max:
      max = dico[cle]
        cle_max = cle
  return cle_max
```

Exercice 6 – *(Comptage d'occurrences)*

1. ```
   fonction compte_occurrence(chaine):
   | dico = {}
   | pour chaque caractere c de chaine:
   || si c n'est pas une clé de dico:
   ||| créer la clé dans dico et l'initialiser à 0
   || ajouter 1 à la valeur de la clé c
   | renvoyer dico
   ```

2. ```
   # FONCTIONS
   def compte_occurrences(chaine):
       dico_occur = {}
       for carac in chaine:
           if carac not in dico_occur:
               dico_occur[carac] = 0
           dico_occur[carac] += 1
       return dico_occur

   # SCRIPT
   print(compte_occurrences("bonjour"))
   # affiche : {'b': 1, 'o': 2, 'n': 1, 'j': 1, 'u': 1, 'r': 1}
   ```

Exercice 7 – *(TP : résultats d'une élection)*

Une élection est organisée, pour laquelle les votes sont enregistrés par une machine électronique. L'objectif de ce TP est d'écrire les fonctions qui vont permettre de comptabiliser automatiquement les votes et déterminer le vainqueur de l'élection.

Représentation des données

Les votes sont enregistrés dans un tableau de chaînes de caractères. Les électeurs peuvent sélectionner parmi une liste de candidats, voter nul en entrant un nom qui ne fait pas partie des candidats ou bien voter blanc.

Par exemple, à la fin du vote, la machine a en mémoire le tableau suivant :

```
votes = ["Alan Turing", "Ada Lovelace", "Ada Lovelace", "",
         "George Boole", "Mark Zukerberg", ""]
```

Une chaîne de caractères vide correspond à un vote blanc.

1. Les candidats étant Alan Turing, Ada Lovelace et George Boole, quel est le vainqueur ? Combien y a-t-il de votes pour les autres candidats, de votes nuls et de votes blancs ?

h

Dans la suite, on stocke cette information dans une variable :

`candidats = ["Alan Turing", "Ada Lovelace", "George Boole"].`

Dépouillement

On veut écrire une fonction `depouillement` qui prend en paramètres :

- un tableau de chaînes de caractères `candidats` ;
- un tableau de chaînes de caractères `votes`.

et qui renvoie un dictionnaire dont :

- les **clés** sont les noms des candidats (stockés dans `candidats`) et les chaînes de caractères "Nul" et "Blanc" ;
- les **valeurs** sont le nombre de bulletins associé (stockés dans `votes`).

2. Il faut commencer par créer un dictionnaire nommé `occurrences`, dont les clés sont les noms des candidats, ainsi que "Nul" et "Blanc", et les valeurs sont initialisées à 0.

   Écrire le programme correspondant à cette initialisation.

→ On peut aussi utiliser la compréhension sur les dictionnaires, avec la syntaxe suivante :

```
d = {cle:valeur for cle in tableau}
```

3. Écrire une fonction `est_present(tab, v)` prenant en paramètres un tableau `tab` de type `list` Python et une valeur `v`, et renvoyant `True` si `v` est présente dans `tab`, `False` sinon.
4. Écrire la fonction `depouillement`, en se basant sur les questions précédentes et sur l'algorithme écrit ci-dessous en pseudo-code (langage naturel) :

```
def depouillement(candidats, votes):
  # initialisation du dictionnaire occurrences - question 1
  pour chaque element v de votes
    si v est présent dans candidats   # fonction écrite dans la question 2
      incrémenter la valeur associée à la clé v dans occurrences
    sinon si v vaut ""
      incrémenter la valeur associée à la clé "Blanc" dans occurrences
    sinon
      incrémenter la valeur associée à la clé "Nul" dans occurrences
  renvoyer occurrences
```

> **Note**
> La syntaxe `if v in tab` pourrait être utilisée ligne 4. Cela revient à écrire une fonction de recherche d'occurrence dans un tableau renvoyant un booléen. Vous pourrez l'utiliser par la suite, mais il faut **savoir écrire la fonction de recherche d'occurrence** !

5. Tester cette fonction avec les variables `candidats` et `votes` définies dans l'introduction. Vérifier qu'on obtient bien le dictionnaire suivant :

```
{'Alan Turing': 1, 'Ada Lovelace': 2, 'George Boole': 1, 'Blanc': 2, 'Nul': 1}
```

Trouver le vainqueur

On considère qu'il n'y a qu'un seul vainqueur à l'élection, c'est-à-dire qu'un candidat a obtenu plus de votes que tous les autres.

1. Réécrire ci-dessous un programme permettant de trouver le maximum d'un tableau de nombres t (vous n'avez pas besoin de l'écrire sur l'ordinateur):

   h

2. Adapter ce programme pour écrire une fonction `vainqueur` qui recherche le maximum des valeurs stockées cette fois-ci dans un dictionnaire : ce sera la fonction `vainqueur(d)`, avec `d` ayant pour clés les candidats (chaînes de caractères) et pour valeurs le nombre de votes qui leur est associé (obtenu avec la fonction `depouillement` de la partie A).

   Cette fonction renvoie la clé associée à la plus grande valeur. On prendra soin d'exclure les votes nuls et blancs.

3. Appliquer la fonction `vainqueur` au dictionnaire obtenu à la question 5, et stocker le résultat du vote dans une variable `resultat`.

   Vérifier qu'Ada Lovelace est bien la grande gagnante de ce scrutin.