

Exercice 2

Notion abordée : structures de données (dictionnaires)

Une ville souhaite gérer son parc de vélos en location partagée. L'ensemble de la flotte de vélos est stocké dans une table de données représentée en langage Python par un dictionnaire contenant des associations de type `id_velo : dict_velo` où `id_velo` est un nombre entier compris entre 1 et 199 qui correspond à l'identifiant unique du vélo et `dict_velo` est un dictionnaire dont les clés sont : "type", "etat", "station".

Les valeurs associées aux clés "type", "etat", "station" de `dict_velo` sont de type chaînes de caractères ou nombre entier :

- "type" : chaîne de caractères qui peut prendre la valeur "electrique" ou "classique"
- "état" : nombre entier qui peut prendre la valeur 1 si le vélo est disponible, 0 si le vélo est en déplacement, -1 si le vélo est en panne
- "station" : chaînes de caractères qui identifie la station où est garé le vélo.

Dans le cas où le vélo est en déplacement ou en panne, "station" correspond à celle où il a été dernièrement stationné.

Voici un extrait de la table de données :

```
flotte = {
    12 : {"type" : "electrique", "etat" : 1, "station" : "Prefecture"},
    80 : {"type" : "classique", "etat" : 0, "station" : "Saint-Leu"},
    45 : {"type" : "classique", "etat" : 1, "station" : "Baraban"},
    41 : {"type" : "classique", "etat" : -1, "station" : "Citadelle"},
    26 : {"type" : "classique", "etat" : 1, "station" : "Coliseum"},
    28 : {"type" : "electrique", "etat" : 0, "station" : "Coliseum"},
    74 : {"type" : "electrique", "etat" : 1, "station" : "Jacobins"},
    13 : {"type" : "classique", "etat" : 0, "station" : "Citadelle"},
    83 : {"type" : "classique", "etat" : -1, "station" : "Saint-Leu"},
    22 : {"type" : "electrique", "etat" : -1, "station" : "Joffre"}
}
```

`flotte` étant une variable globale du programme.

Toutes les questions de cet exercice se réfèrent à l'extrait de la table `flotte` fourni ci-dessus. **L'annexe 1 présente** un rappel sur les dictionnaires en langage Python.

1.

- 1.a. Que renvoie l'instruction `flotte[26]` ?
- 1.b. Que renvoie l'instruction `flotte[80]["etat"]` ?
- 1.c. Que renvoie l'instruction `flotte[99]["etat"]` ?

2. Voici le script d'une fonction :

```
def proposition(choix):  
    for v in flotte:  
        if flotte[v]["type"] == choix and flotte[v]["etat"] == 1:  
            return flotte[v]["station"]
```

- 2.a. **Quelles sont** les valeurs possibles de la variable `choix` ?
- 2.b. **Expliquer** ce que renvoie la fonction lorsque l'on choisit comme paramètre l'une des valeurs possibles de la variable `choix`.

3.

- 3.a. Écrire un script en langage Python qui affiche les identifiants (`id_velo`) de tous les vélos disponibles à la station "Citadelle".
- 3.b. Écrire un script en langage Python qui permet d'afficher l'identifiant (`id_velo`) et la station de tous les vélos électriques qui ne sont pas en panne.

4. On dispose d'une table de données des positions GPS de toutes les stations, dont un extrait est donné ci-dessous. Cette table est stockée sous forme d'un dictionnaire.

Chaque élément du dictionnaire est du type:

`'nom de la station' : (latitude, longitude)`

```
stations = {  
    'Prefecture' : (49.8905, 2.2967) ,  
    'Saint-Leu' : (49.8982, 2.3017),  
    'Coliseum' : (49.8942, 2.2874),  
    'Jacobins' : (49.8912, 2.3016)  
}
```

On **admet** que l'on dispose d'une fonction `distance(p1, p2)` permettant de renvoyer la distance en mètres entre deux positions données par leurs coordonnées GPS (latitude et longitude).

Cette fonction prend en paramètre deux tuples représentant les coordonnées des deux positions GPS et renvoie un nombre entier représentant cette distance en mètres.

Par exemple, `distance((49.8905, 2.2967), (49.8912, 2.3016))` renvoie 9591

Écrire une fonction qui prend en paramètre les coordonnées GPS de l'utilisateur sous forme d'un tuple et qui renvoie, pour chaque station située à moins de 800 mètres de l'utilisateur :

- le nom de la station ;
- la distance entre l'utilisateur et la station ;
- les identifiants des vélos disponibles dans cette station.

Une station où aucun vélo n'est disponible ne doit pas être affichée.

Annexe 1 (exercice 2)
(à ne pas rendre avec la copie)

Action	Instruction et syntaxe
Créer un dictionnaire vide	<code>dico={}</code>
Obtenir un élément d'un dictionnaire existant à partir de sa clé renvoie une erreur si <code>cle</code> n'existe pas dans le dictionnaire	<code>dico[cle]</code>
Modifier la valeur d'un élément d'un dictionnaire à partir de sa clé	<code>dico[cle]=nouvelle_valeur</code>
Ajouter un élément dans un dictionnaire existant	<code>dico[nouvelle_cle]=valeur</code>
Supprimer et obtenir un élément d'un dictionnaire à partir de sa clé	<code>dico.pop(cle)</code>
Tester l'appartenance d'un élément à un dictionnaire (renvoie un booléen)	<code>cle in dico</code>
Objet itérable contenant les clés. 🐾 ce n'est pas un objet de type list	<code>dico.keys()</code>
Objet itérable contenant les valeurs. 🐾 ce n'est pas un objet de type list	<code>dico.values()</code>
Objet itérable contenant les couples (clé,valeur)	<code>dico.items()</code>
Afficher les associations <code>cle:valeur</code> du dictionnaire <code>dico</code>	<pre>for cle in dico: print(cle, dico[cle])</pre>