Exercice 1

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.

1. (a) Que renvoie l'instruction `flotte[26]` ?
   (b) Que renvoie l'instruction `flotte[80]["etat"]` ?
   (c) Que renvoie l'instruction `flotte[22]["station"]` ?
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"]
```

   (a) Quelles sont les valeurs possibles de la variable choix ?
   (b) Expliquer ce que renvoie la fonction lorsque l'on choisit comme paramètre l'une des valeurs possibles de la variable choix.
3. (a) Écrire un script en langage Python qui **affiche** les identifiants (`id_velo`) de tous les vélos disponibles à la station `"Citadelle"`.
   (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, nommée `a_proximite` qui prend en paramètre les coordonnées GPS de l'utilisateur sous forme d'un tuple et qui renvoie, la liste des `id_velo` des vélos disponible dans une station situées à moins (strictement) de 800 mètres de l'utilisateur.

Exercice 2 2022 Centres Etrangers - Sujet 1 - Exercice 1

Cet exercice porte sur les structures de données (listes, p-uplets et dictionnaires).

On dispose de la liste jours suivante et du dictionnaire mois suivant :

```
jours=["dimanche", "lundi", "mardi", "mercredi", "jeudi", "vendredi", "samedi"]

mois={1 :("janvier",31) , 2 :("février",28) , 3 :("mars",31),
    4 :("avril",30) , 5 :("mai",31) , 6 :("juin",30) ,
    7 :("juillet",31) , 8 :("aout",31) , 9 :("septembre",30) ,
    10 :("octobre",31) , 11 :("novembre",30) ,
    12 :("décembre",31)}
```

1. (a) A partir de la liste jours, comment obtenir l'élément "lundi" ?
   (b) On rappelle que l'opérateur % (« modulo ») renvoie le reste de la division entière (division euclidienne).

   Exemple : `7%3` renvoie 1 qui est le reste de la division de 7 par 3.
   Que renvoie l'instruction `jours[18%7]` ?

2. On rappelle que `jours.index[element]` renvoie l'indice de `element` dans la liste jours par exemple `jours.index["mercredi"]` renvoie 3.
   Le nom du jour actuel est stocké dans une variable `j` (par exemple : `j = "mardi"`).
   Recopier et compléter l'instruction suivante permettant d'obtenir le numéro du jour de la semaine n jours plus tard :

```
numero_jour =(jours.index[ ........ ] + ....... )% ......
```

3. (a) A partir du dictionnaire mois, comment obtenir le nombre de jours du mois de mars ?

   (b) Le numéro du mois actuel est stocké dans une variable `numero_mois`, écrire le code permettant d'obtenir le nom du mois qu'il sera x mois plus tard à partir du dictionnaire mois.

   Par exemple :

   — si `numero_mois = 4` et `x = 5`, on doit obtenir `"septembre"`

   — si `numero_mois = 10` et `x = 3`, on doit obtenir `"janvier"`

4. On définit une date comme un tuple :

   `(nom_jour,numero_jour, numero_mois,annee)` .

   (a) Sachant que `date = ("samedi", 21, 10, 1995)`, que renvoie `mois[date[2]][1]` ?

   (b) Écrire une fonction `jour_suivant(date)` qui prend en paramètre une date sous forme de tuple et qui renvoie un tuple désignant la date du lendemain.

   Par exemple :

```
jour_suivant( ("samedi", 21, 10, 1995) ) renvoie
("dimanche", 22, 10, 1995)
jour_suivant( ("mardi", 31, 10, 1995) ) renvoie
("mercredi", 1, 11, 1995)
```

   On ne tient pas compte des années bissextiles et on considère que le mois de février comporte toujours 28 jours.

Exercice 3 2022 - Centres étrangers - Sujet 2 - Exercice 2

Cet exercice porte sur les structures de données (dictionnaires).

La cryptographie est un ensemble de techniques permettant de chiffrer un message. Une technique de cryptographie consiste à mélanger les lettres d'un alphabet et à réécrire le message avec ces permutations. En Python, on peut créer un dictionnaire dans lequel les clés sont les lettres de l'alphabet et les valeurs sont celles de l'alphabet mélangé.

Par exemple, si l'alphabet contient les 4 lettres `A, B, C et D`, et si le dictionnaire de l'alphabet mélangé est `alpha ={"A": "B", "B": "D", "C": "A", "D":"C"}`, la chaîne de caractères `"BAC"` sera chiffrée `"DBA"`.

Un tel dictionnaire sera appelé `dictionnaire de chiffrement`.

1. On souhaite chiffrer un message écrit avec l'alphabet `A, B, C, D, E, F, G` à l'aide du dictionnaire `alpha ={"A":"B", "B":"D", "C":"A", "D":"C", "E":"F", "F":"G", "G":"E"}`

   (a) Quelle est la valeur associée à la clé `"D"` ? En Python, comment l'obtenir ?

   (b) Chiffrer la chaîne de caractères `"BAGAGE"` avec le dictionnaire `alpha`.

2. On considère qu'un mot est une chaîne de caractères (un objet de type `str`) écrite uniquement avec les 26 lettres de l'alphabet en majuscule.

   Par exemple, `"ARBRE"` est un mot et `"L'ARBRE !"` n'est pas un mot à cause des caractères :

   `"'"`, `" "`(espace) et `"!"`.

   Écrire une fonction `chiffrer(mot, alpha)` qui prend en paramètres mot un mot et alpha un dictionnaire de chiffrement, et qui renvoie une chaîne de caractères chiffrée avec le dictionnaire de chiffrement `alpha`.

3. On souhaite déchiffrer un mot chiffré avec cette méthode.

   (a) Si un mot est chiffré avec le dictionnaire de chiffrement `alpha = {"A":"B", "B":"D", "C":"A", "D":"C", "E":"F", "F":"G", "G":"E"}`, donner un dictionnaire permettant de le déchiffrer.

   (b) Écrire une fonction en Python appelée `dico_dechiffrement(dico)` qui prend en paramètre dico un dictionnaire de chiffrement et qui renvoie un dictionnaire permettant le déchiffrement. On pourra s'inspirer du code incomplet ci-dessous ou proposer une autre solution :

```
def dico_dechiffrement(dico):
    nouveau = {}
    for lettre in dico :
        code = dico[.........................]
        nouveau[.........................] = .........................
    return nouveau
```

   (c) Écrire une fonction `dechiffre(mot, dico)` qui reçoit un mot chiffré et un dictionnaire de chiffrement et renvoie le mot décodé. On utilisera les fonctions écrites dans les questions précédentes.

4. On souhaite à présent créer un dictionnaire de chiffrement.

   Écrire une fonction `dico_chiffrement(alphabet)` qui prend en paramètre alphabet un tableau de lettres et qui renvoie un dictionnaire de chiffrement dont les clés sont les lettres du tableau alphabet et les valeurs sont les lettres du tableau alphabet mélangées. skip

   On pourra utiliser la fonction `shuffle` du module `random` qui mélange en place un tableau. Par exemple, on a :

```
>>> tab = ["A", "B", "C", "D"]
>>> shuffle(tab)
["B", "A", "D", "C"]
```