Classe de Terminale NSI | **Devoir surveillé** | *Durée : 50 min + 30 min*

Première partie (12 points)

Exercice 1 (4 points)

On interroge l'API *Découpage administratif communes* du gouvernement français en envoyant une requête à l'adresse :

```
https://geo.api.gouv.fr/communes?nom=Lille&boost=population&fields=population,surface
```

pour récupérer les données géographiques (population et superficie) de la ville de Lille. La réponse de l'API est :

```
[{"population": 234475, "surface": 3474.9, "nom": "Lille"},
 {"population": 9810, "surface": 2658.93, "nom": "Lillers"},
 {"population": 367, "surface": 379.17, "nom": "Lillemer"},
 {"population": 13151, "surface": 315.78, "nom": "Saint-André-lez-Lille"},
 {"population": 8797, "surface": 1464.34, "nom": "Lillebonne"},
 {"population": 10798, "surface": 484.71, "nom": "Marquette-lez-Lille"}]
```

1. Comment s'appelle le format dans lequel les données sont reçues ?

On suppose que la réponse de l'API est stockée dans une variable `reponse`.

2. Quel est le type de la variable `reponse` ? de `reponse[0]` ? de `reponse[0]["surface"]` ?
3. Dans le contexte de l'API, à quoi correspond `len(reponse)` ?
4. Quelles expressions permettent :
   - d'accéder à la population de la ville de Lille ?
   - d'accéder à la superficie de la ville de Lille ?
   - de calculer la densité de la ville de Lille ?
5. À quelle adresse faut-il interroger l'API pour récupérer les données géographiques de la ville de Paris ?

Exercice 2 (4 points)

On donne la définition d'une fonction `api_openfoodfacts` :

```
def api_openfoodfacts(code_barres):
    url = f"https://world.openfoodfacts.org/api/v0/product/{code_barres}.json"
    reponse = requests.get(url)
    reponse = reponse.json()
    if reponse["status"] == 1:
        return {"nom": reponse["product"]["product_name"],
                "nutri_score": reponse["product"]["nutrition_grade_fr"]}
    else:
        return {"nom": "Produit inconnu",
                "nutri_score": None}
```

Si on interroge l'API Open Food Facts à partir d'un code-barres erroné, on obtient une réponse du type :

```
{"code": "8076800376998", "status": 0, "status_verbose": "product not found"}
```

1. Dire, en justifiant, ce que renvoie la fonction `api_openfoodfacts` dans le cas où on l'appelle avec un code-barres erroné en argument.

L'appel `api_openfoodfacts("3166352967037")` renvoie ceci :

```
{"nom": "Poisson à l'andalouse riz safrané et courgettes grillées",
 "nutri_score": "a"}
```

2. Écrire la spécification de la fonction `api_openfoodfacts`.

3. Définir une fonction `nutriscore` qui :
   - prend en paramètre d'entrée un tableau `tab_codes` contenant des numéros de code-barres, et qui
   - renvoie le dictionnaire qui associe à chacune des chaînes `"a"`, `"b"`, `"c"`, `"d"` et `"e"` la liste des noms des produits au nutriscore correspondant.

`nutriscore(["3497917000495", "3560070682683", "3245412511561", "3046920022651"])` doit par exemple renvoyer :

```
{"a": [],
 "b": [],
 "c": ["Chips saveur Indian curry", "Yaourt à la GRECQUE NATURE"],
 "d": [],
 "e": ["Biscuits chocolat au lait", "Lindt Chocolate Excellence"]}
```

Exercice 3 (4 points)

On donne la définition d'une fonction `mystere` :

```
def mystere(n):
    """
    Calcule [redacted].
    - Entrée : [redacted]
    - Sortie : [redacted]
    """
    if n <= 0:
        return 0
    else:
        return n**2 + mystere(n-1)
```

1. Justifier que `mystere` est une fonction récursive, et identifier quel est le cas de base et quel est le cas récursif.

2. Déterminer quelle valeur est renvoyée par l'appel `mystere(4)`, en dessinant l'arbre des appels récursifs provoqués par cet appel.

3. Compléter la spécification de la fonction `mystere`.

4. Expliquer pour quelle raison l'appel `mystere(5000)` provoque une erreur.

Seconde partie (8 points)

Copier sur le bureau le fichier `devoir1.zip` depuis le dossier `Devoir` du réseau.

Lorsque vous aurez terminé, vous renommerez votre fichier `nom_prenom.ipynb` et vous le déposerez dans le dossier `Rendu` du réseau. Attention, une fois déposé dans `Rendu`, votre travail n'est plus modifiable.