Une entreprise fabrique des yaourts qui peuvent être soit nature (sans arôme), soit aromatisés (fraise, abricot ou vanille).
Pour pouvoir traiter informatiquement les spécificités de ce produit, on va donc créer une classe Yaourt qui possèdera un certain nombre d'attributs :

- Son genre : nature ou aromatisé
- Son arôme : fraise, abricot, vanille ou aucun
- Sa date de durabilité minimale (DDM) exprimée par un entier compris entre 1 et 365 (on ne gère pas les années bissextiles). Par exemple, si la DDM est égale à 15, la date de durabilité minimale est le 15 janvier.
  On va créer également des méthodes permettant d'interagir avec un objet de la classe pour attribuer un arôme ou récupérer un genre par exemple. La classe `Yaourt` est déclarée en Python de la manière suivante :

```
class Yaourt:
        """Classe définissant un yaourt caractérisé par :
        - son arome
        - son genre
        - sa durée de durabilité minimale"""

    def __init__(self,arome,duree):
        # Question 1 à compléter sur votre copie
        self.__arome = arome
        self.__duree = duree
        if arome == 'aucun':
            self.__genre = 'nature'
        else:
            self.__genre = 'aromatise'

    # Question 3 à compléter sur votre copie

    def GetDuree(self):
        return(self.__duree)

    def GetGenre(self):
        return ( self.__genre)

    # Question 4 à compléter sur votre copie

    def SetDuree(self,duree):
        if duree > 0 :
            self.__duree = duree

    def __SetGenre(self,arome):
        if arome == 'aucun':
            self.__genre = 'nature'
        else:
            self.__genre = 'aromatise'
```

1. Quelles sont les assertions à prévoir pour vérifier que l'arôme et la durée correspondent bien à des valeurs acceptables. Il faudra aussi expliciter les commentaires qui seront renvoyés.
Pour rappel :

- L'arôme doit prendre comme valeur `fraise` , `abricot` , `vanille` ou `aucun` .
- Sa date de durabilité minimale (DDM) est une valeur positive.

2. Pour créer un yaourt, on exécutera la commande `Mon_Yaourt = Yaourt('fraise',24)` . Quelle valeur sera affectée à l'attribut `genre` associé à `Mon_Yaourt` ?

3. Écrire en python l'accesseur `GetArome` , renvoyant l'arôme du yaourt créé.

4. Écrire en Python le mutateur `SetArome` permettant de modifier l'arôme du yaourt.
⚠️ **On vérifiera que l'arôme corresponde aux données acceptables et on veillera à garder une cohérence entre l'arôme et le genre.**

On veut créer une pile contenant le stock de yaourts. Pour cela il faut tout d'abord créer une pile vide :

```
def creer_pile():
    pile = [ ]
    return pile
```

5. Créer une fonction `empiler(p,y)` qui renvoie la pile `p` après avoir ajouté un objet `y` de type `Yaourt` à la fin.

6. Créer une fonction `depiler(p)` qui renvoie l'objet à dépiler.

7. Créer une fonction `estVide(p)` qui renvoie `True` si la pile est vide et `False` sinon.

8. Qu'affiche le bloc de commandes suivantes ci-dessous ?

```
mon_Yaourt1 = Yaourt('aucun',18)
mon_Yaourt2 = Yaourt('fraise',24)
mon_Yaourt3 = Yaourt('abricot',12)
ma_pile = creer_pile()
empiler(ma_pile, mon_Yaourt1)
empiler(ma_pile, mon_Yaourt2)
print(depiler(ma_pile).GetDuree())
print(depiler(ma_pile).GetGenre())
print(depiler(ma_pile).GetArome())
print(estVide(ma_pile))
```