

**OBJECTIFS** : L'objectif de ce TP est de créer le code d'une classe reprenant une structure de donnée en File.

Pour éviter à ses clients de faire la queue, une grande enseigne de distribution a mis en place sur son site web, une application. Elle permet à chaque client qui veut rentrer dans le magasin, de s'inscrire. Il est alors placé dans une file d'attente virtuelle. Le client sera prévenu lorsque ce sera à son tour de rentrer dans le magasin.

Le code de cette application utilise un paradigme de programmation objet. Chaque client est modélisé par un objet d'une classe *Clients*. La file d'attente est modélisée par un objet d'une classe *File\_Attente*. Les objets *Clients* qui la composent sont liés en utilisant le principe des **listes chaînées**.



**DOCUMENT A RENDRE** : Ce travail est évalué. Le code python réalisé sera rendu sur le site [nsibrantly.fr](http://nsibrantly.fr) en utilisant le code : **file**

## 1. Démarrage :

Le code de départ est le suivant. On y définit la classe *File\_Attente* avec son constructeur et une méthode.

Dans la partie programme principal, on crée un objet de la classe *File\_Attente*. Cet objet est nommé *Carrefour*.

On utilise la méthode de classe *enfiler()* pour insérer 4 objets dans la file.

```
# Classes
class File_Attente():
    def __init__(self):
        self.file = []

    def enfiler(self , client) :
        ...
        Ajoute client (string) en fin de liste self.file
        ...
        self.file.append(client)

# Main
Carrefour = File_Attente()
liste = ["JusDeCactus" , "Piguite" , "Tototra" , "BobLamateur" , "Loic"]
for noms in liste :
    Carrefour.enfiler(noms)
```

## 2. Travail à faire :

On vous demande de créer, dans l'ordre, les méthodes suivantes de la classe *File\_Attente* en utilisant les méthodes et fonctions natives de python et utilisables sur les listes : `append()` , `len()` , `pop()` , ...

- *estVide()* qui renvoie True si l'objet de la classe *File\_Attente* ne comporte aucun élément, False sinon
- *taille()* qui renvoie le nombre d'éléments de la File
- *affichage()* qui écrit dans le shell la valeur des éléments de la liste chaînée
- *defiler()* qui retire le premier élément de la file et qui renvoie son contenu
- 

Ainsi avec le programme principal suivant :

```
# Main
Carrefour = File_Attente()
liste = ["JusDeCactus" , "Piguite" , "Tototra" , "BobLamateur" , "Loic"]
for noms in liste :
    Carrefour.enfiler(noms)
print(Carrefour.taille())
Carrefour.affichage()
Carrefour.enfiler("Robin")
Carrefour.enfiler("Loric")
Carrefour.affichage()
print(Carrefour.defiler())
print(Carrefour.defiler())
print(Carrefour.defiler())
print(Carrefour.defiler())
Carrefour.affichage()
```

le résultat de l'exécution sera :

```
>>> (executing file "file_nsi.py")
5
Contenu de la file :  JusDeCactus Piguite Tototra BobLamateur Loic
Contenu de la file :  JusDeCactus Piguite Tototra BobLamateur Loic Robin Loric
JusDeCactus
Piguite
Tototra
BobLamateur
Contenu de la file :  Loic Robin Loric
```

**Note importante** : Ecrire le script de chaque méthode, l'une après l'autre. Effectuer toujours de multiples tests, ne pas hésiter à écrire avec des `print()` le contenu des variables dans le shell et à raisonner sur le papier avec un crayon.