

## EXERCICE 2 (4 points)

*Cet exercice porte sur les structures de données (files et la programmation objet en langage python)*

Un supermarché met en place un système de passage automatique en caisse. Un client scanne les articles à l'aide d'un scanner de code-barres au fur et à mesure qu'il les ajoute dans son panier. Les articles s'enregistrent alors dans une structure de données.

La structure de données utilisée est une file définie par la classe `Panier`, avec les primitives habituelles sur la structure de file. Pour faciliter la lecture, le code de la classe `Panier` n'est pas écrit.

```
class Panier():
    def __init__(self):
        """Initialise la file comme une file vide."""

    def est_vide(self):
        """Renvoie True si la file est vide, False sinon."""

    def enfiler(self, e):
        """Ajoute l'élément e en dernière position de la file,
        ne renvoie rien."""

    def defiler(self):
        """Retire le premier élément de la file et le renvoie."""
```

Le panier d'un client sera représenté par une file contenant les articles scannés. Les articles sont représentés par des tuples (`code_barre`, `designation`, `prix`, `horaire_scan`) où

- `code_barre` est un nombre entier identifiant l'article ;
- `designation` est une chaîne de caractères qui pourra être affichée sur le ticket de caisse ;
- `prix` est un nombre décimal donnant le prix d'une unité de cet article ;
- `horaire_scan` est un nombre entier de secondes permettant de connaître l'heure où l'article a été scanné.

1. On souhaite ajouter un article dont le tuple est le suivant (31002, "café noir", 1.50, 50525).  
Ecrire le code utilisant une des quatre méthodes ci-dessus permettant d'ajouter l'article à l'objet de classe `Panier` appelé `panier1`.
2. On souhaite définir une **méthode** `remplir(panier_temp)` dans la classe `Panier` permettant de remplir la file avec tout le contenu d'un autre panier `panier_temp` qui est un objet de type `Panier`.

Recopier et compléter le code de la méthode `remplir` en remplaçant chaque `.....` par la primitive de file qui convient.

```
def remplir(self, panier_temp):  
    while not panier_temp. .... :  
        article = panier_temp. ....  
        self. ....(article)
```

3. Pour que le client puisse connaître à tout moment le montant de son panier, on souhaite ajouter une **méthode** `prix_total()` à la classe `Panier` qui renvoie la somme des prix de tous les articles présents dans le panier. Ecrire le code de la méthode `prix_total`. **Attention, après l'appel de cette méthode, le panier devra toujours contenir ses articles.**
4. Le magasin souhaite connaître pour chaque client la durée des achats. Cette durée sera obtenue en faisant la différence entre le champ `horaire_scan` du dernier article scanné et le champ `horaire_scan` du premier article scanné dans le panier du client. Un panier vide renverra une durée égale à zéro. On pourra accepter que le panier soit vide après l'appel de cette méthode. Ecrire une **méthode** `duree_courses` de la classe `Panier` qui renvoie cette durée.