Chapitre 5 : Piles 3

Exercice 1 :

Vous devez simuler le fonctionnement d'une crêperie (de façon très arbitraire).

Dans un premier temps, vous allez créer un dictionnaire qui contiendra les différentes crêpes et galettes proposées.

Exemple :

```
dicCrepes = {1:"crèpe sucre",2:"crèpe confiture",3:"galette
complète",4:"galette raclette"}
```

Puis vous allez créer une pile de 10 crêpes dans laquelle vous allez empiler des crêpes (aléatoirement).

Ensuite, vous irez rapidement distribuer ces crêpes chaudes aux clients.

Résultat attendu :

```
Etat de la pile:
|       4       |
|       1       |
|       3       |
|       4       |
|       1       |
|       2       |
|       3       |
|       4       |
|       3       |
|       1       |

La galette raclette est servie
La crèpe sucre est servie
La galette complète est servie
La galette raclette est servie
La crèpe sucre est servie
La crèpe confiture est servie
La galette complète est servie
La galette raclette est servie
La galette complète est servie
La crèpe sucre est servie
```

Exercice 2 :

Vous allez améliorer le fonctionnement de l'exercice précédent.

Vous allez gérer deux dictionnaires, un pour le sucré et l'autre pour le salé.

Exemple :

```
dicCrepes = {1:"crêpe sucre",2:"crêpe confiture",3:"crêpe nutella",
4:"crêpe chantilly"}
```

```
dicGalettes = {1:"galette jambon",2:"galette andouillette",3:"galette
complète",4:"galette raclette"}
```

Vous aurez deux piles de 5.

Maintenant, votre programme demandera à l'utilisateur ce qu'il veut (salé ou sucré).

Il affichera la liste des propositions pour que l'utilisateur choisisse.

Ensuite, vous irez rapidement distribuer les assiettes aux clients quand les piles seront pleines.

Vous proposerez aussi au client de passer à l'addition (ce qui équivaut à quitter le programme).

Vous devrez gérer les exceptions/erreurs de saisies possibles de l'utilisateur.

Résultat attendu :

Exemple différents choix OK :

```
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 1
1   galette jambon
2   galette andouillette
3   galette complète
4   galette raclette
quel est votre choix ?2
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 2
1   crêpe sucre
2   crêpe confiture
3   crêpe nutella
4   crêpe chantilly
quel est votre choix ?4
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 3

C'est cadeau !!

Au revoir et à bientôt !
```

Exemple remplissage pile galettes OK

```
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 1
1   galette jambon
2   galette andouillette
3   galette complète
4   galette raclette
quel est votre choix ?2
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 1
1   galette jambon
2   galette andouillette
3   galette complète
4   galette raclette
quel est votre choix ?4
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 1
1   galette jambon
2   galette andouillette
3   galette complète
4   galette raclette
quel est votre choix ?3
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 1
1   galette jambon
2   galette andouillette
3   galette complète
4   galette raclette
quel est votre choix ?2
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 1
1   galette jambon
2   galette andouillette
3   galette complète
4   galette raclette
quel est votre choix ?1

Il est temps de servir les galettes
        La galette jambon est servie
        La galette andouillette est servie
        La galette complète est servie
        La galette raclette est servie
        La galette andouillette est servie

Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition :
```

Exemple choix KO : gestion des exceptions et des erreurs

```
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition : 0
1 pour salé, 2 pour sucré, 3 pour l'addition : 5
1 pour salé, 2 pour sucré, 3 pour l'addition : d
Erreur de saisie, ce n'est pas un nombre
1 pour salé, 2 pour sucré, 3 pour l'addition : 2
1   crêpe sucre
2   crêpe confiture
3   crêpe nutella
4   crêpe chantilly
quel est votre choix ?0
quel est votre choix ?5
quel est votre choix ?d
Erreur de saisie, ce n'est pas un nombre
quel est votre choix ?2
Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition :
```

Exercice 3 :

Vous allez maintenant améliorer le fonctionnement et envisager d'utiliser une File plutôt qu'une Pile. Ainsi, les plats seront servis selon leur ordre de commande.

Dans un premier temps, on remplit chaque file de 5 plats.

Dans un second temps, on distribue dans l'ordre (du premier au dernier).

Pour cela, il va falloir créer un module File (en s'inspirant de ce qui a été fait pour la Pile) avec les méthodes suivantes :

A la place de la méthode `empiler`, il faudra une méthode `ajouter` (juste un changement de nom)

A la place de la méthode `depiler`, il faudra une méthode `enlever` (tant que la file n'est pas vide, on enlève le premier élément de la file et on décale toutes les suivantes, cf exemple suivant)

Avant :

```
1   3   2   3   1   5
```

Après :

```
3   2   3   1   5   None
```

Puis :

```
2   3   1   5   None None
```

A la place de la méthode `lireSommet`, il faudra une méthode `recupererPremier` (premier élément ajouter à la file)

Il faudra aussi modifier l'affichage de la méthode `__str__` , comme suit :

```
Etat de la file:
-1--4--3--2--4-
```

Pour la méthode `estVide`, il faudra tester si la taille est == 0 ou pas (taille = nombre d'éléments de la file).

Pour la taille, il faudra compter combien il y a d'éléments différents de None.

Pas de changement pour les méthodes `__init__` et `estPleine`

Pour rappel :

```
# Récupère la dernière valeur ajoutée à la pile
def lireSommet(self) :
    if self.estVide() == False :
        if self.nombreDelements() == 1 :
            return self.valeurs[0]
        else :
            return self.valeurs[-1]
```

Affichage de test pour visualiser la File :

```
Etat de la file:
-1--4--3--2--4-

Il est temps de servir les crêpes
        La crêpe sucre est servie

Etat de la file:
-4--3--2--4--None-
        La crêpe chantilly est servie

Etat de la file:
-3--2--4--None--None-
        La crêpe nutella est servie

Etat de la file:
-2--4--None--None--None-
        La crêpe confiture est servie

Etat de la file:
-4--None--None--None--None-
        La crêpe chantilly est servie

Etat de la file:
-None--None--None--None--None-

Bonjour, que souhaitez-vous commander ?
Du salé ou du sucré
1 pour salé, 2 pour sucré, 3 pour l'addition :
```

```
class File:
    def __init__(self, capacite) :
        self.valeurs = []
        self.capacite = capacite

    # Retourne le nombre d'éléments de la file
    def recupTaille(self) :
        # A faire
        return 0

    # Renvoie True si la pile et vide et False sinon
    def estVide(self) :
        # A faire
        return True

    def estPleine(self) :
        if (self.recupTaille() == self.capacite) :
            return True
        else :
            return False

    # Ajoute un élément à la file
    def ajouter(self, valeur) :
        # A faire (reprendre empiler de la Pile)

    # Supprime le dernier élément ajouté
    def enlever(self) :
        # A faire
        return element

    # Récupère le premier élément ajouté
    def recupererPremier(self) :
        # A faire
        return element

    # Affichage de la file
    def __str__(self) :
        ch = ''
        # A faire
        return ch
```