Les files

Définition

En informatique, une file (en anglais : *queue*) est une structure de données pour lesquelles on dispose des méthodes suivantes.

- `enfile(el)` : ajoute l'élément `el` à la fin de la file (en anglais : *enqueue*);
- `defile()` : supprime et renvoie l'élément en tête de file (en anglais : *dequeue*).

Le principe de fonctionnement d'une file est FIFO (*First In First Out* : premier entré, premier sorti).
On fera l'analogie avec une file d'attente.
Selon les langages, on dispose en général des méthodes supplémentaires suivantes.

- `est_vide()` : renvoie `True` si la file est vide, `False` sinon (en anglais : *is_empty*);
- `taille()` : le nombre d'éléments dans la file (en anglais : *size*).

Exercice 1 – *(+)* Le langage Python ne proposant pas de type « file » au sens strict du terme, il est toutefois possible d'en définir un à l'aide de la programmation orientée objet.

1. Dans un fichier `file.py`, écrire la définition de la classe `File` dont on donne l'API ci-dessous.

```
class File(builtins.object)
 |  __init__(self)
 |      Construit une nouvelle file vide
 |
 |  enfile(self, el: Object) -> None
 |      Ajoute un élément à la fin de la file.
 |
 |  defile(self) -> Object
 |      Supprime et renvoie l'élément en tête de file.
 |      Déclenche une erreur si la file est vide.
 |
 |  taille(self) -> int
 |      Renvoie le nombre d éléments de la file.
 |
 |  est_vide(self) -> bool
 |      Renvoie True si la file est vide, False sinon.
 |
 |  __str__(self) -> str
 |      Renvoie une description du contenu de la file.
```

   On pourra s'inspirer de la classe `Pile` et consulter la documentation de la méthode `pop` sur le type `list`.

   /!\ Récupérer la méthode `__str__` proposée sur notre site.

2. Donner l'état de la file `f` à l'issue de ce script :

```
f =  File()
f.enfile("n")
f.enfile("s")
f.enfile("i")
x = f.defile()
f.defile()
f.enfile(x)
```

Dans la suite, on utilisera exclusivement uniquement des objets de la classe `File` définie dans l'exercice 1. (Interdit d'utiliser le type `list` de Python).

Exercice 2 – *(++)*

1. Écrire une fonction `passe_ton_tour(f)` qui place en queue de la file `f` l'élément en tête (la modification de la file se fait donc **en place**).
2. Écrire une fonction `nb_occurences(f, val)` qui renvoie le nombre d'éléments de la file `f` qui sont égaux à `val`.
3. Modifier la fonction précédente de sorte que, après l'appel de la fonction, la file passée en argument soit identique à ce qu'elle était initialement (utiliser pour cela une file auxiliaire).
4. Écrire une fonction `last_is_first(f)` qui place le dernier élément de la file `f` en premier.

Exercice 3 – *(+++)*

On demande dans cet exercice que les modifications de la file soit effectuées en place.

1. Écrire une fonction `permute_prochains` qui permute les deux prochains éléments d'une file.
2. Écrire une fonction `permute_derniers` qui permute les deux derniers éléments d'une file.

Exercice 4 – *(+++)*

1. Écrire une fonction `est_dans_la_file(f, val)` qui renvoie `True` si la valeur `val` apparaît au moins une fois dans la file `f`, et renvoie `False` sinon. Après l'appel, la file `f` doit être identique à ce qu'elle était initialement.
2. Écrire une fonction `supprime_doublons` qui prend en argument une file `f` et la modifie en place de telle sorte qu'elle ne contienne plus de doublons.

Exercice 5 – *(+++)*

On décide que les multiples de 12 sont des VIN (*Very Important Numbers*).

Écrire une fonction `vin_first` qui prend en argument une file de nombres entiers et la modifie de sorte que les VIN soient placés en premier dans la file.

Exercice 6 – *(++++)*

Écrire l'implémentation d'une classe `Pile` qui utilise deux files.

On rappelle que les méthodes à définir sont : `empile`, `depile`, `est_vide` et `taille`.

Indications :

- une des deux files est toujours vide ;
- la pile est vide lorsque les deux files sont vides ;
- pour empiler un élément, on l'enfile dans celle des deux files qui est non vide ;
- pour dépiler un élément, on transfère les éléments de la file non vide vers l'autre file, à l'exception du dernier élément, qui est renvoyé.