Exercice 1. — Dans certains langages de programmation, comme Pascal ou Ada, les tableaux ne sont pas nécessairement indexés à partir de 0. C'est le programmeur qui choisit sa plage d'indices. Par exemple, on peut déclarer un tableau dont les indices vont de −10 à 9 si on le souhaite, Dans cet exercice, on se propose de construire une classe `Tableau` pour réaliser de tels tableaux. Un objet de cette classe aura deux attributs, un attribut `premier` qui est la valeur du premier indice et un attribut `contenu` qui est un « tableau » Python contenant les éléments. Ce dernier est un « vrai tableau » Python, indexé à partir de 0.

- Écrire un constructeur `__init__(self, imin, imax, v)` où `imin` est le premier indice, `imax` le dernier indice et `v` la valeur utilisée pour initialiser toutes les cases du tableau. Ainsi, on peut écrire `t = Tableau(-10, 9, 42)` pour construire un tableau de vingt cases, indexées de −10 à 9 et toutes initialisées avec la valeur 42.
- Écrire une méthode `__len__(self)` qui renvoie la taille du tableau.
- Écrire une méthode `__getitem__(self, i)` qui renvoie l'élément d'indice i. De même, écrire une méthode `__setitem__(self, i, v)` qui modifie l'élément du tableau `self` d'indice `i` pour lui donner la valeur `v`.
  Ces deux méthodes doivent vérifier que l'indice `i` est bien valide et, dans le cas contraire, lever l'exception `IndexError` avec la valeur `i` en argument (c'est-à-dire `raise IndexError(i)`).
- Enfin, écrire une méthode `__str__(self)` qui renvoie une chaîne de caractères décrivant le contenu du tableau.

Exercice 2. — On veut définir une classe `TabBiDir` pour les tableaux bidirectionnels, dont une partie des éléments ont des indices positifs et une partie des éléments ont des indices négatifs, et qui sont extensibles aussi bien par la gauche que par la droite. Plus précisément, les indices d'un tel tableau bidirectionnel vont aller d'un indice `imin` à un indice `imax`, tous les deux inclus et tels que `imin` $\leqslant 0$ et $-1 \leqslant$ `imax`. Le tableau bidirectionnel vide correspond au cas où `imin` vaut 0 et `imax` vaut −1.
La classe `TabBiDira` pour attributs deux tableaux Python : un tableau `droite` contenant l'élément d'indice 0 et les autres éléments d'indices positifs, et un tableau `gauche` tel que `gauche[0]` continent l'élément d'indice −1 du tableau bidirectionnel, et `gauche[1]`, `gauche[2]`, etc. contiennent les éléments d'indices négatifs suivants, en progressant vers la gauche.

- Écrire un constructeur `__init__(self, g, d)` initialisant un tableau bidirectionnel contenant, dans l'ordre, les éléments des tableaux `g` et `d`. Le dernier élément de `g` (si `g` n'est pas vide), devra être calé sur l'indice −1 du tableau bidirectionnel, et le premier élément de `d` (si `d` n'est pas vide) sur l'indice 0.
  Écrire également des méthodes `imin(self)` `imax(self)` renvoyant respectivement l'indice minimum et l'indice maximum.
- Ajouter une méthode `append(self, v)` qui, comme son homonyme des « tableaux » Python, ajoute l'élément `v` à droite du tableau bidirectionnel `self`, et une méthode `prepend(self, v)` ajoutant cette fois l'élément `v` à gauche du tableau bidirectionnel `self`.
  Utiliser `append` sur un tableau bidirectionnel vide place l'élément à l'indice 0. Utiliser `prepend` sur un tableau bidirectionnel vide place l'élément à l'indice −1.
- Ajouter une méthode `__getitem__(self, i)` qui renvoie l'élément du tableau bidirectionnel `self` à l'indice `i`, et une méthode `__setitem__(self, i, v)` qui modifie l'élément du tableau `self` d'indice `i` pour lui donner la valeur `v`.
- Ajouter une méthode `__str__(self)` qui renvoie une chaîne de caractères décrivant le contenu du tableau.