

5.4

Implémenter une file avec Python

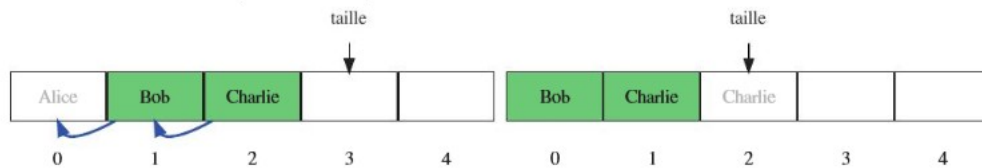
NSI TERMINALE - JB DUTHOIT

5.4.1 Implémenter une file avec un tableau

On veut ici implémenter une file avec un tableau d'une capacité limitée.

On représente la file par un tuple `tableau, taille`, avec `tableau` un tableau et `taille` un tableau contenant une seule valeur, correspondant à la taille de la file.

Pour ajouter `elt` à la file, il suffit d'ajouter `elt` à l'indice `taille` (sauf si elle est pleine...)



Exercice 5.19

1. Créer la fonction `creer_file(c)` qui renvoie un tuple :

```
>>> file = creer_file(5)
>>> file
([None, None, None, None, None], [0])
```

2. Créer la fonction `enfiler(file,elt)` qui enfiler `elt` à `file`

```
>>> enfiler(file, 'Alice')
>>> enfiler(file, 'Bob')
>>> enfiler(file, 'Charlie')
>>> file
(['Alice', 'Bob', 'Charlie', None, None], [3])
```

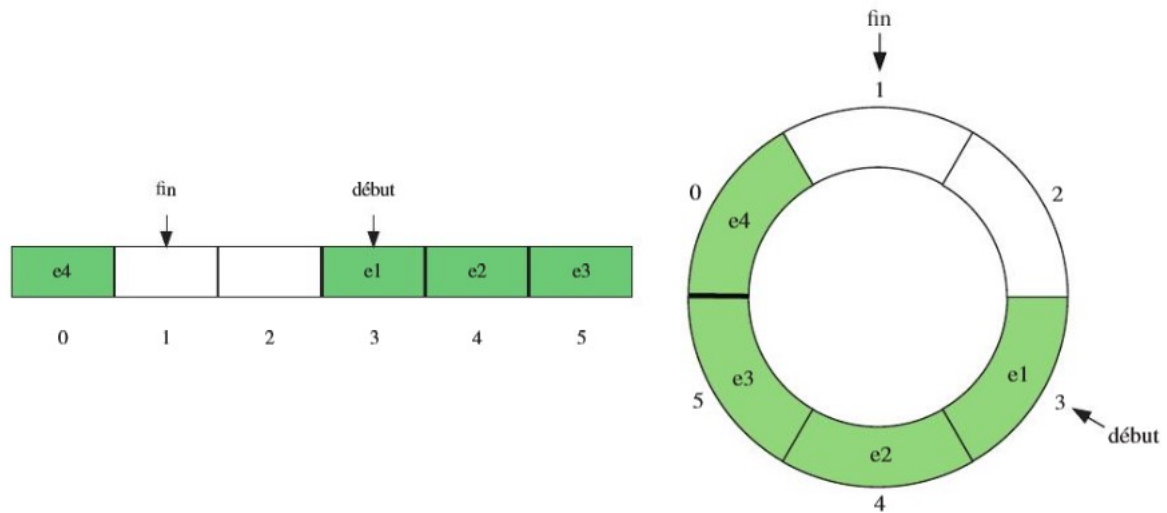
3. Créer la fonction `defiler(file)` qui défile `file` :

```
>>> defiler(file)
'Alice'
>>> defiler(file)
'Bob'
>>> file
(['Charlie', None, None, None, None], [0])
```

5.4.2 Implémenter une file avec un tableau circulaire

L'implémentation précédente est couteuse car pour défiler un élément, il faut recopier tous les éléments restants.

On considère ici un tableau comme un anneau dont la file occupe les éléments de l'indice `debut` à l'indice `fin - 1`.



Principe du tableau circulaire

Exercice 5.20

1. Créer la fonction `creer_file(c)` qui renvoie un tuple :

```
>>> file = creer_file(5)
>>> file
([None, None, None, None, None], [0],[1])
```

2. Créer la fonction `enfiler(file,elt)` qui enfile `elt` à `file`

```
>>> enfiler(file,'Alice')
>>> enfiler(file,'Bob')
>>> enfiler(file,'Charlie')
>>> file
(['Alice', 'Bob', 'Charlie', None, None], [0],[3])
```

3. Créer la fonction `defiler(file)` qui défile `file` :

```
>>> defiler(file)
'Alice'
>>> defiler(file)
'Bob'
>>> file
([None, None, 'Bob', None, None], [2],[3])
```

5.4.3 Implémenter une file avec les tableaux dynamiques python

Voici comment implémenter facilement une file avec les méthodes des tableaux dynamiques python (type `list`) :

```
ma_file = []
# Pour créer une file vide
ma_file.append(ma_valeur)
```

```
# permet d'enfiler une valeur
ma_file.pop(0)
# permet de défiler une valeur
len(ma_file)
# renvoie la longueur de ma_file
```

● Exercice 5.21

Implémenter une file avec un tableau dynamique. On créera pour cela la fonction `creer_file()`, `est_vide(file)`, `enfiler(file,elt)` et `defiler(file)`.

L'objectif de l'exercice suivant est d'implémenter des files avec des objets. L'objet sera ici implémenté avec des tableaux dynamiques.

● Exercice 5.22

Créer la classe `File` qui dispose des méthodes suivantes :

- `estvide()` qui renvoie `True` si la file est vide, `False` sinon.
- `enfiler(elt)` qui permet d'enfiler un élément à la file
- `defiler()` qui défile un élément et le renvoie.
- `affiche()` qui affiche la file

5.4.4 Construire une file avec deux piles

Principe

Soient `p1` et `p2` deux piles vides.

Principe pour enfiler

1. Avant d'enfiler `elt`, on déplace les éventuels éléments présents dans `p2` vers `p1` par dépilages et empilages successifs.
2. On enfile `elt` dans `p2` (qui est vide)
3. On déplace chaque élément de `p1` vers `p2` par dépilages et empilages successifs.

Principe pour défiler

1. Pour défiler, il suffit de dépiler sur `p2`.

Exemple pour enfiler 1, puis 2 puis 3 :

1. Initialisation :

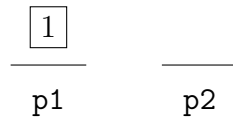
$\overline{\quad}$ $\overline{\quad}$
 $p1$ $p2$

$\overline{\quad}$ $\overline{\boxed{1}}$
 $p1$ $p2$

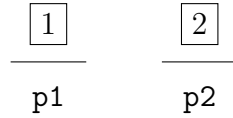
2. On enfile 1 par empilage de 1 sur `p2` :

3. Pour enfiler 2 :

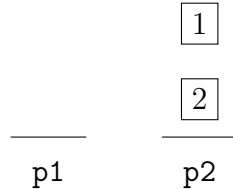
- a) On déplace 1 sur `p1` :



b) On enfiler 2 par empilage de 2 sur p2 :

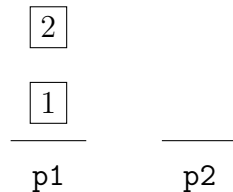


c) On déplace 1 sur p2 :

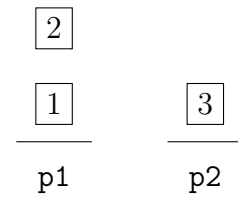


4. Pour enfiler 3 :

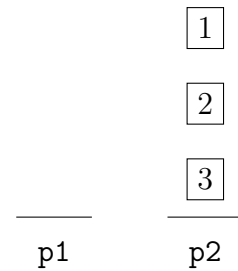
a) On déplace les éléments de p2 sur p1 :



b) On enfiler 3 par empilage de 3 sur p2 :



c) On déplace les éléments de p1 sur p2 :



5. Pour défiler, il suffit de dépiler successivement sur p2 !

Exercice 5.23

On considère une classe `Pile` construite précédemment, et disposant des méthodes `est_vide()`, `empiler(elt)`, `depiler()`.

On considère deux piles `p1` et `p2`.

Avec ces deux piles, on souhaite, en utilisant le principe précédent, créer une file.

On crée une file vide par la création des deux piles vides `p1` et `p2`

1. Créer la pile `p1` et la pile `p2`.
2. Créer la fonction `enfiler(elt)`, avec `elt` l'élément à enfiler, en utilisant seulement les piles `p1` et `p2`
3. Créer la fonction `defiler()`, en utilisant seulement les piles `p1` et `p2`
4. Tester votre file !