

TP 3: Classe FileDePriorite

L'objectif de ce TP est de développer une classe `FileDePriorite` du module `file_de_priorite.py`. Une fonction `cle`, initialisée par défaut à l'identité, sera utilisée pour comparer les éléments de cette file. On considère qu'un élément e_1 est prioritaire sur un autre élément e_2 si $cle(e_1) < cle(e_2)$. Si $e_1 == e_2$ et que e_1 a été enfilé avant e_2 alors e_1 sortira avant e_2 .

Cette classe proposera les méthodes suivantes :

- `__init__(self, elements=(), cle=lambda e:e)` qui permet d'initialiser la file avec des éléments;
- `est_vide(self)` qui permet de savoir si la file est vide;
- `enfiler(self, element)` qui permet d'enfiler un élément dans la file;
- `defiler(self)` qui permet de défiler l'élément le plus prioritaire, on obtient alors cet élément;
- `element(self)` qui permet d'obtenir l'élément le plus prioritaire sans le défiler;

De plus :

- on peut parcourir tous les éléments de la file du plus prioritaire au moins prioritaire;
- la représentation formelle doit permettre de recréer la file dans son état.

Voici quelques exemples d'utilisation de cette classe :

```
In [1]: from file_de_priorite import FileDePriorite
```

```
In [2]: f=FileDePriorite((3,9,1,4))
```

```
In [3]: f.est_vide()
```

```
Out[3]: False
```

```
In [4]: f.element()
```

```
Out[4]: 1
```

```
In [5]: f.enfiler(2)
```

```
In [6]: print(f.defiler())
```

```
1
```

```
In [7]: f.element()
```

```
Out[7]: 2
```

```
In [8]: for e in f: print(e)
```

```
2
```

```
3
```

```
4
```

```
9
```

```
In [9]: f
Out[9]: FileDePriorite(2, 3, 4, 9)
```