Travaux pratiques

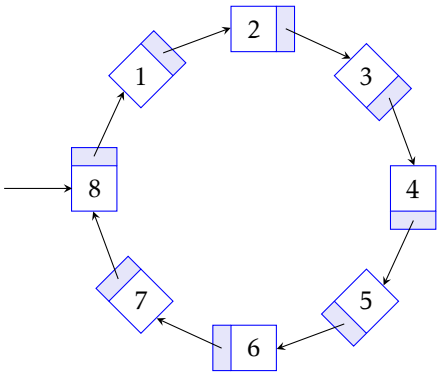
option informatique

Implémentation d'une file

Durant cette séance nous allons nous intéresser à deux implémentations alternatives des files : à l'aide d'une liste circulaire, puis à l'aide de deux piles.

Exercice 1. Implémentation d'une file à l'aide d'une liste circulaire

Une liste circulaire est une liste chaînée dont le dernier élément pointe vers le premier.



Afin de représenter une liste circulaire en Caml, on définit les types suivants :

```
type 'a cell = {Valeur: 'a; mutable Next: 'a cell} ;;
type 'a liste = Nil | Cellule of 'a cell ;;
```

Nous allons utiliser une liste circulaire pour représenter une file, ce qui nous amène à définir le type :

```
type 'a file = {mutable Queue: 'a liste} ;;
```

Par exemple, la liste circulaire dessinée ci-dessus représentera la file 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 (1 est la tête de la file donc le premier élément à en sortir, 8 la queue donc le dernier élément à y être entré).

Nous aurons enfin besoin d'une exception caractérisant une file vide :

```
exception Empty ;;
```

a) Définir une fonction **new** qui crée une file vide.

```
new : unit -> 'a file
```

b) Rédiger une fonction **peek** qui renvoie la valeur de la tête de la file, mais sans la supprimer de celle-ci. Dans le cas d'une file vide l'exception **Empty** sera déclenchée.

```
peek : 'a file -> 'a
```

c) Rédiger maintenant une fonction **take** qui renvoie la valeur de la tête de la file en la supprimant de celle-ci. Dans le cas d'une file vide l'exception **Empty** sera déclenchée.

```
take : 'a file -> 'a
```

d) Rédiger enfin une fonction **add** qui ajoute un élément à une file.

```
add : 'a -> 'a file -> unit
```

Exercice 2. Implémentation d'une file à l'aide de deux piles

Dans cet exercice nous allons utiliser les piles du module **stack**, dont on rappelle les primitives :

- **stack__t** est le type des piles définies dans ce module ;
- **stack__new** crée une pile vide ;
- **stack__push** empile un élément sur une pile ;
- **stack__pop** supprime le sommet de la pile et le retourne.

On définit alors le type *file* de la façon suivante :

```
type 'a file = {Pile1: 'a stack__t; Pile2: 'a stack__t} ;;
```

a) Définir une fonction **new** qui crée une file vide.

```
new : 'a file -> unit
```

b) L'ajout d'un élément de la file s'effectue en empilant ce dernier dans la première pile. Rédiger la fonction **add** correspondante.

```
add : 'a -> 'a file -> unit
```

c) Le retrait d'un élément de la file s'effectue en dépilant le sommet de la seconde pile si cette dernière est non vide. Mais que faut-il faire si cette dernière est vide ? En déduire la fonction **take** correspondante.

```
take : 'a file -> 'a
```