V.3 La File

Une *File* (Queue) est un type permettant de représenter un ensemble de données, du même `type T`, à accès séquentiel et écriture non destructive.

L'accès aux données se fait dans l'ordre dans lequel elles ont été insérées.

Définition : *File*

On appelle ***File***, un ensemble formé d'un nombre variable de données sur lequel on effectue les opérations suivantes :

- ajouter une nouvelle donnée ;
- consulter la première donnée ajoutée et non supprimée depuis ;
- supprimer la première donnée ajoutée et non supprimée depuis ;
- savoir si l'ensemble est vide ou non ;

Spécification fonctionnelle

Fonctionnalités

```
init: → QueueT //crée une file (vide)
first: QueueT → T | Vide //retourne le premier de la file
deQueue: QueueT → QueueT x T //retire le premier de la file
enQueue: QueueT x T → QueueT //ajoute un nouvel élément à
la fin de la file
isEmpty: QueueT → Bool //vérifie si la File est vide
isFull: QueueT -> Bool // vérifie si la File est pleine
capacity: QueueT -> Int // nombre maximum d'éléments
count: QueueT -> Int // nombre d'éléments dans la File
```

Spécification fonctionnelle

Fonctionnalités

Q1: isEmpty(init()) == True

Q2: count(init()) == 0

Q3: first(q)==Vide $\Rightarrow$ isEmpty(q)

Q4: deQueue^n(deQueue^n(q,t_{k_n}))==Vide

Q5: first(enQueue(…(enQueue(q, t_0), t_1), …, t_n))==t_0

Q6: deQueue(q) $\Rightarrow$!isEmpty(q)

Q7: deQueue(enQueue^n(init(),t_i))==enQueue^{n-1}(init(),t_i)

Q8: count(enQueue(q,t)) == count(q)+1

Q9: (q_2,t) = deQueue(q_1) $\Rightarrow$ count(q_2) == count(q_1) - 1

Q10: isFull(q) $\Leftrightarrow$ count(q) == capacity(q)

Q11: isEmpty(q) $\Leftrightarrow$ count(q) == 0

Q12: capacity(init(n)) == n

Q13: enQueue(q,t) $\Rightarrow$!isFull(q)

Description logique

Puisqu'il faut stocker des éléments, on peut utiliser un tableau. En revanche on doit interdire l'accès direct aux éléments, cet accès ne peut se faire que par les fonctions `enQueue` et `deQueue`. Les données étant ajoutées à un bout de la file mais retiré à l'autre bout, contrairement à la pile, le début de la file ne sera pas tout le temps à l'indice 0. Il faut donc être capable de connaître le début et la fin de la file

QueueT : (storage: [T], beg: Int, end: Int, count: Int)

Exercice :

Écrivez le protocol swift décrivant le type abstrait QueueInt

Exercice :

Proposez une structure de donnée à base d'un tableau pour le type concret QueueInt.

Exercice :

Proposez une implémentation concrète du type abstrait QueueInt.