Université Bordeaux 1, Sciences et Technologies
UFR Mathématiques et Informatique

Maîtrise d'Informatique
Session de janvier 1999

Méthodes et Techniques de Programmation

Durée : deux heures.

Tous documents autorisés sauf photocopies de livres. Votre voisin n'est pas un document.

Rappel : Soyez bref et clair ! Personne n'est impressionné par la quantité !

Bon courage !

Une file (anglais: queue) est une structure de données dont les opérations sont :

`enq`	: insérer un élément à la fin de la file.
`deq`	: supprimer l'élément au début de la file, et renvoyer cet élément.
`empty`	: test pour determiner si la file est vide

Exercice 1. (5p)

Écrire une implémentation de la file en C. L'interface doit être impérative. L'implémentation doit utiliser un tableau considéré comme un tampon circulaire. La file doit pouvoir contenir un nombre arbitraire d'éléments.

Exercice 2. (5p)

Écrire une implémentation de la file en Java. L'interface doit être impérative. L'implémentation doit utiliser une liste circulaire simplement chaînée sans sentinelle.

Exercice 3. (5p)

Écrire une implémentation de la file en Common Lisp + CLOS. L'interface doit être fonctionnelle. L'implémentation doit utiliser une liste Lisp. Pour l'opération `deq`, utiliser les valeurs multiples pour renvoyer à la fois l'élément et la nouvelle file.

Exercice 4. (5p)

L'une des opérations (`enq` ou `deq`) de l'exercice 3 a un coût assez important. Écrire une macro Lisp `with-queues` permettant d'écrire des expressions similaire à la suivante :

```
(with-queues ((q1 my-queue-1) (q2 my-queue-2))
  (while (some-test)
    (enq q1 (deq q2)))
  (enq q1 234)
  (list q1 q2))
```

Le premier argument à la macro est une liste de variables temporaires initialisées (comme avec `let`) avec des expressions dont les valeurs doivent être des files. Les autres arguments sont le corps de la macro. Dans le corps, les variables temporaires contiennent des files *impératives*. Ces files sont initialement des copies impératives des files correspondantes utilisées dans les initialisations. Les opérations `enq` ou `deq` sont donc destructives dans le corps de la macro. La valeur de la dernière expression du corps est la valeur de l'expression `with-queues`. Avant l'exécution de la dernière expression du corps, les files contenues dans les variables temporaires sont converties en files fonctionnelles.

Implémenter la macro `with-queues`. Vous pouvez supposer l'existence de deux fonctions `functional-queue-to-imperative-queue` et `imperative-queue-to-functional-queue`

pour la conversion entre files fonctionnelles et files impératives. Vous pouvez aussi supposer que les opérations `enq` et `deq` sont implémentées sous la forme de fonctions génériques qui marchent à la fois sur des files fonctionnelles et des files impératives. L'expansion de l'expression ci-dessus sera donc par exemple :

```
(let ((q1 (functional-queue-to-imperative-queue my-queue-1))
      (q2 (functional-queue-to-imperative-queue my-queue-2)))
  (while (some-test)
    (enq q1 (deq q2)))
  (enq q1 234)
  (setf q1 (imperative-queue-to-functional-queue q1)
        q2 (imperative-queue-to-functional-queue q2))
  (list q1 q2))
```