Université Bordeaux 1 Licence S&T Mention Informatique S5 2007/2008

UE INF353 **Programmation 3**

Programmation Fonctionnelle et Symbolique

Devoir surveillé No 2

Tous documents autorisés. Vendredi 23 Novembre 2007
Durée : 1h.
Le barème est donné à titre **indicatif**.
Chacun des quatre exercices doit être **rédigé** sur une feuille **séparée** sur laquelle doit être inscrit votre nom et votre numéro de groupe. Vous devez rendre exactement **une feuille** (ou copie), éventuellement blanche, **par exercice**.

Exercice 1 *(4pts)*

Écrire une fonction itérative `entier (chiffres &optional base)` qui retourne l'entier ayant pour chiffres en base `base` (par défaut 2) les chiffres de la liste `chiffres`. Exemples :

```
CL-USER> (entier '(1 1 1 0))
14
CL-USER> (entier '(1 1 1 0) 10)
1110
CL-USER> (entier '(1 2 0) 3)
15
```

Exercice 2 *(4pts)*

Soit f une fonction numérique d'un argument. Soit la série

$$\Sigma_{i=0}^{p} f^i(u_0) = u_o + f(u_o) + f(f(u_0)) + \ldots + f^n(u_0)$$

Écrire une fonction itérative `serie (u0 f p)` qui calcule cette série.

Exercice 3 *(5pts)*

Écrire une fonction itérative `next-line (line)` qui étant donné un tableau `line` correspondant à une ligne du triangle de Pascal retourne un tableau contenant la ligne suivante du triangle. Exemples :

```
CL-USER> (next-line #(1))
#(1 1)
CL-USER> (next-line (next-line #(1)))
#(1 2 1)
CL-USER> (next-line (next-line (next-line #(1))))
#(1 3 3 1)
```

Exercice 4 *(5pts)*

On considère une table de hachage `*repliques*` dont les entrées *(clé, valeur)* sont telles que *clé* est un symbole et *valeur* une liste de symboles. Une clé représente un mot (ou une phrase), la valeur associée représente une liste de réponses possibles à ce mot.

```
(defparameter *repliques* (make-hash-table :test #'eq))
(defun ajout-table (mot reponses table)
  (setf (gethash mot table) reponses))
(ajout-table 'hello '(salut bonjour) *repliques*)
(ajout-table 'salut '(comment\ vas-tu?) *repliques*)
(ajout-table 'bonjour '(comment\ vas-tu?) *repliques*)
(ajout-table 'comment\ vas-tu? '(pas\ mal
                                 bien\ et\ toi?
                                 ca\ va
                                 super!) *repliques*)
```

Cette table permettra de simuler un dialogue aléatoire (s'il y a plusieurs réponses possibles à un mot, comme pour `'comment vas-tu?` par exemple).

1. Écrire une fonction `mot-suivant (mot table)` qui retourne une réponse choisie aléatoirement parmi les réponses possibles à de mot et `NIL` si le mot n'est pas dans la table. Exemples :

   ```
   CL-USER> (mot-suivant 'comment\ vas\-tu? *repliques*)
   |PAS MAL|
   CL-USER> (mot-suivant 'comment\ vas\-tu? *repliques*)
   |CA VA|
   ```

2. En utilisant la fonction `mot-suivant`, écrire une fonction `simul-dialogue (mot table)` qui simule un dialogue démarrant avec le mot `mot` et utilisant les réponses contenues dans la table `table`. Le mot suivant est la réponse au mot précédent. Le dialogue s'affiche et s'arrête quand le mot courant n'a pas de réponse.

```
CL-USER> (simul-dialogue 'hello *repliques*)

HELLO
SALUT
COMMENT\ VAS-TU?
PAS\ MAL
NIL

CL-USER> (simul-dialogue 'salut *repliques*)

SALUT
COMMENT\ VAS-TU?
BIEN\ ET\ TOI?
NIL

CL-USER> (simul-dialogue 'hello *repliques*)

HELLO
SALUT
COMMENT\ VAS-TU?
CA\ VA
NIL
```