INTELLIGENCE ARTIFICIELLE

SUPINFO International University

Sign of Success

3AIT – TP

Intelligence Artificielle

L'analyse d'un monde guidé

Programmation Fonctionnelle
Lisp
Python

Version 1.0
Last update: 19/04/2018
Use: Students/Staff
Author: Cyril Alexandre Pachon

SOMMAIRE

1 PREAMBULE : LES CONSIGNES GENERALES

Votre rendu se fera sous la forme d'un **dossier compressé** (**.zip**) nommé **[3AIT]-IDOpenCampus-NomDuCampus-Nom-Prénom-TP**. Votre dossier peut contenir **uniquement** des documents sous la forme **.pdf**, **.py** et **.lsp**.

Pour cet examen, **vous pouvez utiliser les supports de cours (.ppt, LABS)**. L'utilisation d'**internet est interdite**. Les outils **autorisés** sont les interpréteurs **Python** et **Lisp, pas d'autres outils**. Si votre surveillant(e) constate une tricherie, votre épreuve sera annulée et votre relevé de notes portera la mention de « **cheater** » pour cet examen.

2 LA PROGRAMMATION FONCTIONNELLE (6 POINTS)

Question1.1 (1 point) : Avec les sélecteurs ou les constructeurs, écrire la fonction récursive nommée **NB** qui compte le nombre d'atomes dans une liste d'atomes. Spécifiez tous les éléments de vos réalisations.

Vous disposez de la liste suivante L = ((1 2 3) (4 5 6) (7 8 9))

Question1.2 (3 points) : Avec les sélecteurs ou les constructeurs, écrire une fonction récursive nommée **VF** qui vérifie si une liste contient des sous-listes (uniquement de profondeur 1, comme la liste L) contenant le même nombre d'atomes dans chaque sous-liste (comme la liste L). Spécifiez tous les éléments de vos réalisations.

Question1.3 (2 points) : Avec les sélecteurs ou les constructeurs, écrire une fonction récursive nommée **VLA** qui vérifie si le nombre d'atomes dans chaque sous-liste de L est égal au nombre de sous-liste de L. Spécifiez tous les éléments de vos réalisations.

3 LE PROBLEME LISP (8 POINTS)

Un Système Expert contient l'expression suivante :

```
(and (defun E()
(lambda (A B)
(cond
  ((< A 20) (expt B 10))
  (T  "Confirmez-moi le résultat obtenu ?"))))
  (funcall (E) *read-base* *print-base*))
```

Question 2.1 (1 point) : Donnez le résultat de l'expression.

Question2.2 (2 points) : Expliquez ligne à ligne ce que fait l'expression donnée. Comment le résultat est-il produit ?

Question2.3 (1 point) : Faites une seule modification dans l'expression donnée pour que la chaine de caractères soit interprétée et affichée. Il n'est pas demandé de refaire l'expression, mais juste de modifier un des éléments de l'expression pour obtenir le résultat : "Confirmez-moi le résultat obtenu ?". Expliquer votre choix.

Question 2.4 (1,5 points) : Ecrire une fonction Lisp nommée **F** sans argument qui comporte notamment l'expression initiale pour correspondre aux résultats suivants :

(funcall (F) 0) → 0
(funcall (F) 5) → 9765625
(funcall (F) 3) → 59049
(funcall (F) 19) → 6131066257801
(funcall (F) 20) → " Confirmez-moi le résultat obtenu?"
"Oui il s'agit d'un 20"
(funcall (F) 25) → " Confirmez-moi le résultat obtenu?"
"Oui il s'agit d'un 25"

Question2.5 (1,5 points) : La fonction Lisp **expt** a été donnée dans l'expression initiale. Il faut maintenant la faire. Ecrire une fonction récursive Lisp nommée **expt2** qui produit la même interprétation que la fonction **expt**. Définissez tous les paramètres que vous allez utiliser.

Question 2.6 (1 point) : Modifiez la fonction **F** en incluant maintenant **expt2**. Votre fonction **F** doit être vérifiée. Il vous est demandé de proposer un ensemble de tests pour vérifier son bon fonctionnement.

4 LE SYTEME PYTHON (6 POINTS)

Vous disposez de la fonction Python suivante :

```
F = lambda L : L.reverse()
```

Question 3.1 (1 point): Que fait la fonction **F**, comment éditer le résultat ?

Vous disposez des fonctions Python suivantes :

```
car = lambda liste: liste[0]
cdr = lambda liste: liste[1:]
membre = lambda a, liste : a in liste
```

Question 3.2 (2 points): Avec les fonctions données, et en gardant le principe de la programmation fonctionnelle, écrire une fonction en Python nommée **EG** d'arité 2 qui élimine toutes les valeurs à gauche d'une valeur donnée.

>>> EG('a', ['d','c','b']) → ['d', 'c', 'b']
>>> EG('a', ['d','c','a','b']) → ['a', 'b']
>>> EG('1', ['d','a','b']) → ['d', 'a', 'b']

Question 3.3 (3 points) Vous avez les résultats d'une suite **S** :

>>> S([]) → 1
>>> S(['1']) → 1
>>> S(['1','2']) → 2
>>> S(['1','2','3']) → 2
>>> S(['1','2','3','4']) → 24
>>> S(['1','2','3','4','5']) → 24
>>> S(['1','2','3','4','5','6']) → 720
>>> S(['1','2','3','4','5','6','7']) → 720
>>> S(['1','2','3','4','5','6','7','8']) → 40320
>>> S(['1','2','3','4','5','6','7','8','9']) → 40320
>>> S(['1','2','3','4','5','6','7','8','9','10']) → 3628800
…….

En utilisant les fonctions **car** et **cdr**, écrire en Python la fonction **S** d'arité 1 qui réalise la suite.