Université Grenoble Alpes
MIAGE L3 – Programmations déclaratives

Contrôle Continu
Février 2018

Durée : 70 minutes, 1 document A4 autorisé, Sujet sur 1 recto simple.

I. red ! (barème indicatif : 6 points)

Le programme suivant n'a pas été documenté, sa sémantique et son utilisation sont incertaines.

```
red([],[],[]).
red([E],[E],[]).
red([E,E|L],[E|G],D):- red(L,G,D).
red([E,F|L],[E|G],[F|D]) :- dif(E,F), red(L,G,D).
```

(remarque : la troisième règle pourrait être écrite `red([E,F|L],[E|G],D):- eq(E,F), red(L,G,D).`*)*

Q1 Exécution. Quelles seront la/les réponses aux requêtes suivantes (les explications ne sont pas nécessaires) :

```
?- red([1,2,3],G,D).

?- red([A,A,A,A,A],G,D).

?- red([1,2,2,3,3,3,4],G,D).
```

Q2. Arbre d'exécution et terminaison. Donner les (~10) premiers nœuds de l'arbre d'exécution de la requête suivante (les nœuds d'échec ⊥ comptent aussi) :

```
?- red(L,[1,2],D).
```

Cette requête termine-t-elle ? si oui, avec combien de solutions.

II. Médiane (barème indicatif : 7 points)

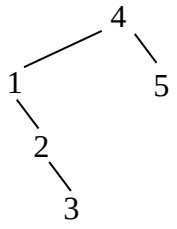
Un algorithme simple (naïf) pour trouver la médiane d'une liste d'entiers distincts consiste à enlever le minimum et le maximum de cette liste et recommencer sur la liste réduite obtenue, autant de fois que nécessaire jusqu'à ce qu'il ne reste qu'un (ou deux) entier(s), c'est alors (presque) fini : en supposant que la liste initiale est de longueur impaire, alors, à la fin, il ne reste qu'un seul entier, c'est la médiane.

Q1. Suppression du minimum et du maximum. Spécifiez et réalisez un prédicat qui produise, à partir d'une liste d'entiers, une liste réduite des mêmes entiers à laquelle on aura supprimé minimum et maximum.

Q2. Médiane. Spécifiez et réalisez un prédicat qui donne la médiane d'une liste d'entiers. Vous pourrez utiliser le prédicat précédent, ainsi que l'algorithme naïf proposé au début de l'énoncé.

Q3. Cas général. Sans changer vos deux programmes, si l'utilisateur souhaite travailler avec des listes de longueur paire, les propriétés de vos programmes (resp. : la terminaison, la complétude et la correction) sont-elles modifiées ?

III. Arbre binaire de recherche (barème indicatif : 7 points)



Un arbre binaire de recherche est un arbre binaire dans les éléments se répartissent à gauche et à droite de chaque nœud selon leur valeur : à gauche ceux qui sont plus petits que la valeur du nœud, à droite les autres. Ainsi en construisant un arbre binaire de recherche par insertions successives des valeurs 4, 1, 2, 3 puis 5 on obtient l'arbre binaire de recherche suivant (cf. ci-contre), représenté en ProLog par le terme [4,[1,[],[2,[],[3,[],[]]]],[5,[],[]]] où chaque nœud est de la forme [ValeurDuNoeud, SousArbreGauche, SousArbreDroit] et chaque feuille de la forme [ValeurDeLaFeuille, [], []].

Q1. Recherche. Spécifiez et réalisez un prédicat qui recherche si un entier est présent dans un arbre binaire de recherche (l'entier recherché et l'arbre sont a priori donnés/connus par l'utilisateur lors de l'exécution).

Q2. Insertion. Sur la base du prédicat précédent, spécifiez et réalisez un prédicat qui insère un entier dans un arbre de recherche à la place où il aurait dû être s'il avait été présent.

Q3. Parcours dans l'ordre croissant. Spécifiez et réalisez un prédicat qui parcourt un arbre binaire de recherche en produisant la liste des valeurs des nœuds et des feuilles parcourus de telle manière que cette liste d'entiers soit ordonnée par valeurs croissantes (sur l'exemple donné, le résultat sera la liste [1, 2, 3, 4, 5].