

Durée : 90 minutes. 1 document personnel manuscrit A4 autorisé. Sujet sur 1 recto simple.

I. Tri encore plus vite – en ProLog (barème indicatif : 6 points)

Un ingénieur pensait faire un tri croissant efficace en séparant les nombres pris deux à deux : plus grands vs plus petits selon l'algorithme suivant :

```
triV(L,T) :- separe(L,Lp,Lg), triV(Lp,Tp), triV(Lg,Tg), concat(Tp,Tg,T).
separe([E,F|L],[E|Lp],[F|Lg]) :- lt(E,F), separe(L,Lp,Lg).
separe([E,F|L],[F|Lp],[E|Lg]) :- gte(E,F), separe(L,Lp,Lg).
concat([E|S],T,[E|R]) :- concat(S,T,R).
```

Malheureusement, le programme `triV` est incomplet (il manque les cas « simples ») et incorrect (le problème principal vient de la concaténation, qui devrait être une fusion par interclassement).

Q1. Complétude. Améliorez le programme `triV` pour qu'il soit complet.

L'ingénieur a bien travaillé, lui aussi, son programme `triV` modifié est maintenant complet. Il reste incorrect, mais l'ingénieur a observé qu'il permet de trier au moins deux éléments correctement : le plus petit qui se trouve en début de liste, et le plus grand qui se trouve à la fin de la liste.

Q2. Correction. En supposant que le programme `triV` modifié est effectivement complet et qu'il permet au moins de trier le plus petit et le plus grand élément d'une liste, proposez un programme `tri` qui trie une liste complètement en utilisant `triV`. *Indication* : Si le premier et le dernier élément sont bien rangés, récursivement, si l'on trie l'intérieur de la liste, cela devrait donner une liste complètement triée.

II. Solitaire, sur la base de l'Awalée (barème indicatif : 7 points)

Pour cet exercice, choisir ProLog ou Erlang pour les mises en œuvre.

Le jeu de solitaire suivant consiste à prendre un tas de cailloux, comme dans le jeu Awalée, et commence en répartissant aléatoirement ces cailloux devant soi en petits tas (c'est l'initialisation du jeu, ce sera la donnée d'entrée du programme). Puis, tant que les petits tas devant soi changent, appliquer l'égrenage suivant : prendre le tas le plus à gauche et distribuer un à un ses cailloux sur les autres tas (de gauche à droite), un caillou par tas, et s'il reste des cailloux à la fin, faire des tas de 1 caillou à droite.

Exemple : situation initiale (une lettre = un caillou ; des espaces entre les tas) : ABCDE FG HIJ K

situation après égrenage du premier (gros) tas à gauche : AFG BHIJ CK D E

Dans la suite de l'exercice, une situation sera donnée par la liste des tas représentés par l'entier qui donne le nombre de cailloux. La situation initiale précédente sera donc donnée par la liste [5,2,3,1].

Q1. À la main. Continuer à la main, l'exécution du jeu, à partir de la situation initiale donnée dans l'exemple, jusqu'à ce que la situation se stabilise (tant que la situation change).

Q2. Égrenage. Spécifiez et réalisez un programme qui produit un égrenage.

Q3. Stabilisation. Spécifiez et réalisez un programme qui prolonge le jeu jusqu'à stabilisation.

Nota bene : dans l'Awalée, (le vrai), on joue à deux, alternativement, les tas de cailloux sont en cercle, etc...

III. Parallélisation - en Erlang (barème indicatif : 7 points)

Cet exercice cherche à déterminer une décomposition des entiers comme somme de grands nombres carrés.

Exemple : 15 est la somme de 9, 4, 1 et 1 (chacun de ces nombres est un carré). La fonction Erlang suivante permet d'obtenir le plus grand nombre carré inférieur ou égal à un entier N :

```
lePlusGrandCarre(N)      -> lPGC(N,N).
lPGC(N,D) when N >= (D*D) -> D*D ;
lPGC(N,D)                 -> lPGC(N,D-1).
```

Q1. Décomposition d'un entier sous forme de somme de grands carrés. Spécifiez et réalisez une fonction Erlang qui décompose un entier comme la somme de ses plus grands carrés (en donner la liste).

Indication : le premier carré est donné par la fonction `lePlusGrandCarre`, si ce carré est égal à l'entier, la décomposition est terminée, sinon continuer avec la différence entre cet entier et son plus grand carré.

Q2. Décomposition des entiers de 1 à N. Spécifiez et réalisez une fonction Erlang qui donne la décomposition des entiers de 1 à N comme somme de leurs plus grands carrés. Le résultat peut être donné comme une liste de listes, ex, le résultat pour 1 à 10 ce serait : [[1],[1,1],[1,1,1],[4],[4,1],[4,1,1],[4,1,1,1],[4,4],[9],[9,1]]

Q3. Parallélisation. Spécifiez et réalisez une fonction Erlang qui parallélise le calcul précédent pour un processeur multicœurs.