Durée : 3 heures

Exercice 1 : Ensembles bien fondés

Étant donnés deux ordres bien fondés $(A, >_A)$ et $(B, >_B)$, en justifiant rapidement vos réponses, indiquer lesquels parmi les suivants sont aussi des ordres bien fondés :

1. $(A \times B, >_{A\times B})$ avec $(a, b) >_{A\times B} (a', b')$ si et seulement si $a >_A a'$ et $b >_B b'$.
2. $(A \times B, >_{A\times' B})$ avec $(a, b) >_{A\times' B} (a', b')$ si et seulement si ($a >_A a'$ et $b = b'$) ou ($a = a'$ et $b >_B b'$).
3. $(A \cup B, >_{A\cup B})$ avec $e >_{A\cup' B} e'$ si et seulement si ($e \in A$ et $e' \in A$ et $e >_A e'$) ou ($e \in B$ et $e' \in B$ et $e >_B e'$).

Exercice 2 : Construction par induction

1. Donner une base qui permette de construire $\mathbb{N}^*$ par induction en utilisant uniquement l'opérateur de construction $f : x \mapsto 2x$.
2. Donner une base et un ensemble de constructeurs permettant de définir l'ensemble des mots de la forme $(^n|)^m$ pour $n \in \mathbb{N}$ et $m \in \mathbb{N}$, c'est-à-dire avec n caractères (, un caractère | et enfin m caractères).
3. Donner une base et un ensemble de constructeurs permettant de définir l'ensemble des mots de la forme $(^n|)^m$ pour $(n, m) \in \mathbb{N}^2$ avec $n < m$.
4. Donner une base et un ensemble de constructeurs permettant de définir l'ensemble des mots bien parenthésé. C'est à dire des suites finies de caractères (ou) telle qu'il y ait autant de (que de) et que tout préfixe contienne un nombre supérieur ou égale de (au nombre de).
   Plus précisément, si on note $|u|_a$ le nombre de caractère a dans le chaîne de caractères u, il faut construire :

$$E = \{u = u_1u_2 \dots u_n, \quad n \in \mathbb{N}, \quad |u|_(= |u|_) \text{ et } \forall i \leqslant n, \quad |u_1u_2 \dots u_i|_) \leqslant |u_1u_2 \dots u_i|_(\}$$

   On considérera que la chaîne de caractère vide est un mot bien parenthésé.

Exercice 3 : Démonstration par induction d'une terminaison

On considère la fonction $f : \mathbb{N}^3 \longrightarrow \mathbb{N}$ définie par

$$\forall (x, y, z) \in \mathbb{N}^3, \quad f(x, y, z) = \begin{cases} y + z + 1 & \text{si } x = 0 \\ f(x - 1, 1, z + 1) & \text{si } x \neq 0 \text{ et } y = 0 \\ f(x - 1, y + 1, 1) & \text{si } x \neq 0,\ y \neq 0 \text{ et } z = 0 \\ f(x - 1, f(x, y - 1, f(x, y, z - 1)), x + y + z) & \text{sinon} \end{cases}$$

1. On munit $\mathbb{N}^3$ de l'ordre lexicographique. Rappeler la définition de cet ordre et montrer qu'il est bien ordonné.
2. Ecrire une fonction récursive OCaml de type `int -> int -> int -> int` calculant f.
3. Montrer que le calcul de `f x y z` termine quels que soient x, y, z dans $\mathbb{N}$.

Exercice 4 : Logique des propositions

Vous participez à un concours de mathématiques comportant une partie de raisonnement logique. Plusieurs orateurs font des déclarations et vous devez répondre à des questions en vous appuyant sur des informations déduites de ces déclarations. La règle suivante s'applique : « Les orateurs sont de trois natures : les véridiques, les menteurs et les changeants. Les véridiques disent toujours la vérité, les menteurs mentent toujours, et les changeants disent en alternance une vérité et un mensonge (c'est-à- dire, soit une vérité, puis un mensonge, puis une vérité, etc. ; soit un mensonge, puis une vérité, puis un mensonge, etc.). Pendant tout le concours, les orateurs ne peuvent pas changer de nature. »

Les épreuves comportent deux phases :

— Les différents orateurs font plusieurs déclarations dont l'analyse permet de déterminer la nature de chaque orateur (véridique, menteur, changeant commençant par dire la vérité, ou changeant commençant par dire un mensonge).
— Les orateurs font une seconde série de déclarations. Puis, vous devez répondre à des questions en exploitant les informations contenues dans ces déclarations.

Question .1. Dans la première phase, quel est le nombre minimum de déclarations que doit faire chaque orateur pour qu'il soit possible de déterminer sa nature ? Justifier votre réponse.

Question .2. Soit un orateur A qui fait une suite de n déclarations A_i. Proposer des formules du calcul des propositions A_V , A_M, A_{CV} et A_{CM} qui permettent de caractériser la nature de A (respectivement véridique, menteur, changeant commençant par dire la vérité, ou changeant commençant par dire un mensonge).

Vous participez à une première épreuve avec un orateur A qui fait les déclarations suivantes :
— J'aime le rouge mais pas le bleu.
— Soit j'aime le rouge, soit j'aime le vert.
— Si j'aime le rouge et le vert, alors j'aime le bleu.
Nous noterons R, V et B les variables propositionnelles associées au fait que l'orateur aime le rouge, le vert ou le bleu. Nous noterons A_1, A_2 et A_3 les formules propositionnelles associées aux déclarations de A.

Question .3. Représenter les déclarations de l'orateur sous la forme de formules du calcul des propositions A_1, A_2 et A_3 dépendant des variables R, V et B.

Question .4. Appliquer les formules permettant de caractériser la nature des orateurs A_V , A_M, A_{CV} et A_{CM} que vous avez proposées pour la question I.2 pour l'orateur A dépendant des variables A_1, A_2 et A_3.

Question .5. En utilisant le calcul des propositions (résolution avec les tables de vérité), déterminer la nature de l'orateur A. Quelles sont les couleurs qu'aime A ?

Vous participez à une seconde épreuve avec trois orateurs G, H et I. Vous avez déterminé dans la première phase avec succès que G est un menteur, que H est un véridique et que I est un changeant sans savoir s'il doit dire la vérité ou un mensonge pour sa déclaration suivante. Ceux-ci font les déclarations :
I : Le losange est visible.
G : Le cercle n'est visible que si le losange est visible.
I : Le triangle n'est pas visible.
H : Soit le cercle est visible, soit le triangle est visible.
Nous noterons G_1, H_1, I_1 et I_2 les formules propositionnelles associées aux déclarations des orateurs G, H et I dans cette seconde épreuve. Nous noterons C, L et T les variables propositionnelles associées au fait que le cercle, le losange ou le triangle soit visible.

Question .6. Représenter les déclarations des orateurs sous la forme de formules du calcul des propositions G_1, H_1, I_1 et I_2 dépendant des variables C, L et T.

Question .7. Représenter les informations sur la nature des orateurs sous la forme d'une formule du calcul des propositions dépendant des variables G_1, H_1, I_1 et I_2.

Question .8. En utilisant le calcul des propositions (résolution avec les formules de De Morgan), déterminer quelle est (ou quelles sont) la (ou les) figure(s) visible(s) ainsi que la nature exacte de I l'orateur changeant.

Exercice 5 : Tris de listes chaînées

1. Ecrire en OCaml une fonction récursive `insertion x l` de type `'a -> 'a list -> 'a list` qui à un élément `x` et une liste chaînée `l` d'éléments du même type triée dans l'ordre croissant renvoie une liste contenant les éléments de `l` avec en plus `x` et toujours triée dans l'ordre croissant.
2. Ecrire une fonction récursive `tri_insertion l` de type `'a list -> 'a list` qui à une liste `l` renvoie une liste composée des même éléments mais triée dans l'ordre croissant. Cette fonction devra appeler la fonction précédente.
3. Donner la complexité de `tri_insertion l` dans le pire cas et le meilleur cas en fonction de la taille de la liste `l`. Justifier votre réponse.
4. Écrire une fonction récursive `divise` de signature :

```
divise : 'a list -> 'a list -> 'a list -> 'a list * 'a list
```

qui prend en entrée trois listes `liste1`, `liste2` et `liste` et qui ajoute alternativement à `liste1` et à `liste2` les éléments de `liste`, puis renvoie le couple (`liste1`,`liste2`) une fois la liste `liste` épuisée.

5. Écrire une fonction récursive `fusionne` de signature :

```
fusionne : 'a list -> 'a list -> 'a list
```

qui prend en entrée deux listes triées (la fonction ne vérifiera pas ce point) et qui renvoie la liste triée obtenue en fusionnant les deux listes. La complexité de l'algorithme devrait être en $O(n)$ avec n le nombre total d'éléments présents dans les deux listes.

6. Écrire une fonction récursive `tri_fusion` de signature :

```
tri_fusion : 'a list -> 'a list
```

Exercice 6 : Une fonction récursive

1. On considère la fonction récursive suivante, qui calcule les termes d'une suite $(u_n)_{n\in\mathbb{N}}$:

```
let rec u n =
   match n with
      | 0 -> 4
      | 1 -> 7
      | _ -> 3 * u (n-1) + 2 * (u (n-2))
;;
```

   (a) Que valent u_0 et u_1 ? Expliciter une relation de récurrence vérifiée par la suite u.

   (b) Montrer que l'appel `u n` termine pour toute entrée $n \in \mathbb{N}$.

   (c) Déterminer le nombre d'appels de fonctions nécessaires au calcul de `u n`.

   (d) En déduire la complexité de la fonction `u` ?

2. Une amélioration :

   (a) Ecrire une fonction récursive `u_term` qui prend en entrée trois entiers a, b et n et qui, pour $m \in \mathbb{N}$, renvoie u_{n+m} lorsque $a = u_m$ et $b = u_{m+1}$.

   (b) En déduire une fonction `u2` calculant les termes de la suite u plus rapidement que `u`.

   (c) Déterminer le nombre d'opérations arithmétiques et le nombre d'appels de fonctions nécessaires au calcul de `u2 n`.

   (d) Quelle est la complexité de la fonction `u2` ?