DS6 (éléments de réponse)

Exercice 1

Question 1.

□ **1.1.** Posons $\Gamma = (\neg\varphi \vee \psi)$. On peut proposer l'arbre de déduction suivant.

$$\dfrac{\dfrac{\dfrac{}{\Gamma, \varphi \vdash \Gamma}\,ax \qquad \dfrac{}{\Gamma, \varphi, \psi \vdash \psi}\,ax}{\Gamma, \varphi \vdash \psi}\,\vee_e \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, \varphi, \neg\varphi \vdash \varphi}\,ax \qquad \dfrac{}{\Gamma, \varphi, \neg\varphi \vdash \neg\varphi}\,ax}{\Gamma, \varphi, \neg\varphi \vdash \bot}\,\neg_e}{\Gamma, \varphi, \neg\varphi \vdash \psi}\,\bot_e}{\Gamma \vdash (\varphi \to \psi)}\,\to_i}{\vdash (\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,\to_i$$

□ **1.2.** Construisons la table de vérité de $(\varphi \to \psi) \to (\neg\varphi \vee \psi)$.

φ	ψ	$(\varphi \to \psi)$	$(\neg\varphi \vee \psi)$	$(\varphi \to \psi) \to (\neg\varphi \vee \psi)$
F	F	V	V	V
F	V	V	V	V
V	F	F	F	V
V	V	V	V	V

Cette table indique que $(\varphi \to \psi) \to (\neg\varphi \vee \psi)$ est une tautologie. En logique classique, le système déductif défini par la déduction naturelle est *complet* : tout ce qui est sémantiquement vrai est syntaxiquement prouvable. Par conséquent, $\Vdash (\varphi \to \psi) \to (\neg\varphi \vee \psi)$ implique $\vdash (\varphi \to \psi) \to (\neg\varphi \vee \psi)$. Il existe donc une preuve du séquent $\vdash (\varphi \to \psi) \to (\neg\varphi \vee \psi)$.

Question 2.

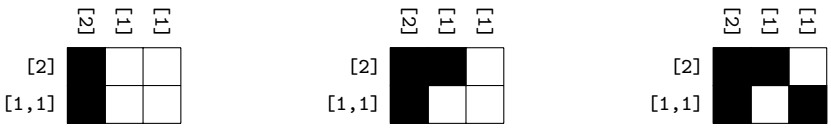
□ **2.1.** Posons $\Gamma = (\forall x\ \varphi \vee \forall x\ \psi)$. On peut alors proposer l'arbre de déduction suivant.

$$\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \Gamma}\,ax \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, \forall x\ \varphi \vdash \forall x\ \varphi}\,ax}{\Gamma, \forall x\ \varphi \vdash \varphi}\,\forall_e}{\Gamma, \forall x\ \varphi \vdash (\varphi \vee \psi)}\,\vee_i}{\Gamma, \forall x\ \varphi \vdash \forall x\ (\varphi \vee \psi)}\,\forall_i \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, \forall x\ \psi \vdash \forall x\ \psi}\,ax}{\Gamma, \forall x\ \psi \vdash \psi}\,\forall_e}{\Gamma, \forall x\ \psi \vdash (\varphi \vee \psi)}\,\vee_i}{\Gamma, \forall x\ \psi \vdash \forall x\ (\varphi \vee \psi)}\,\forall_i}{\Gamma \vdash \forall x\ (\varphi \vee \psi)}\,\vee_e}{\vdash \Gamma \to \forall x\ (\varphi \vee \psi)}\,\to_i$$

□ **2.2.** La formule $\forall x\ (\varphi \vee \psi) \to (\forall x\ \varphi \vee \forall x\ \psi)$ n'est pas toujours sémantiquement vraie. Or, le système déductif défini par la déduction naturelle est *correct* : tout ce qui est syntaxiquement prouvable est sémantiquement vrai. Par conséquent, $\nVdash \forall x\ (\varphi \vee \psi) \to (\forall x\ \varphi \vee \forall x\ \psi)$ implique $\nvdash \forall x\ (\varphi \vee \psi) \to (\forall x\ \varphi \vee \forall x\ \psi)$. Il n'existe donc pas de preuve du séquent $\vdash \forall x\ (\varphi \vee \psi) \to (\forall x\ \varphi \vee \forall x\ \psi)$.

Exercice 2

Question 1. En suivant les indications numériques portées sur les bords de la gille, la première colonne doit comporter deux cases noires. Comme elle n'a que deux cases, celles-ci sont donc noires (figure de gauche ci-dessous).



La première ligne doit comporter deux cases noires contiguës. La première case étant déjà noircie, seule la deuxième peut également l'être (figure du milieu ci-dessus). Enfin, la deuxième ligne doit comporter deux cases noires non contiguës. La première case l'étant déjà et la ligne ne comportant que trois cases, seule la troisième case peut être noircie (figure de droite ci-dessus). Ce qui établit l'unicité de la solution de H.

Question 2.

□ **2.1.** Le prédicat L_0 exprime qu'*exactement deux cases contiguës de la première ligne de H sont noires.* Dressons la table de vérité qui présente les valeurs de vérité attribuées à x_0, x_1 et x_2 et la valeur de vérité de L_0.

x_0	x_1	x_2	L_0
F	F	F	F
F	F	V	F
F	V	F	F
F	V	V	V
V	F	F	F
V	F	V	F
V	V	F	V
V	V	V	F

Partant de cette table, on peut construire une formule logique φ sous forme normale disjonctive, qui décrit L_0. On a :

$$\varphi = (\neg x_0 \wedge x_1 \wedge x_2) \vee (x_0 \wedge x_1 \wedge \neg x_2)$$

On met cette formule sous forme normale conjonctive par quelques manipulation algébriques.

$$\begin{aligned}\varphi &= (x_1) \wedge [(\neg x_0 \wedge x_2) \vee (x_0 \wedge \neg x_2)] \\ &= (x_1) \wedge [(\neg x_0 \vee x_0) \wedge (\neg x_0 \vee \neg x_2) \wedge (x_2 \wedge x_0) \wedge (x_2 \wedge \neg x_2)] \\ &= (x_1) \wedge [\top \wedge (\neg x_0 \vee \neg x_2) \wedge (x_2 \wedge x_0) \wedge \top] \\ &= (x_1) \wedge (x_0 \vee x_2) \wedge (\neg x_0 \vee \neg x_2)\end{aligned}$$

□ **2.2.** Le prédicat C_1 exprime qu'*exactement une case de la deuxième colonne de H est noire.* Ce qui permet la construction de la table de vérité suivante.

x_1	x_4	C_1
F	F	F
F	V	V
V	F	V
V	V	F

On en déduit une formule ψ sous forme normale conjonctive qui décrit C_1.

$$\psi = (x_1 \vee x_4) \wedge (\neg x_1 \vee \neg x_4)$$

Question 3.

□ **3.1.** Avec $\varphi = (x_1) \wedge (x_0 \vee x_2) \wedge (\neg x_0 \vee \neg x_2)$, on peut proposer l'arbre de dérivation suivant.

$$\dfrac{\dfrac{}{\varphi \vdash \varphi}\ ax}{\varphi \vdash x_1}\ \wedge_e$$

□ **3.2.** Avec $\psi = (x_1 \vee x_4) \wedge (\neg x_1 \vee \neg x_4)$, on peut proposer l'arbre de dérivation suivant, en deux parties.

$$\dfrac{\dfrac{\dfrac{}{\psi, x_1, x_4 \vdash \psi}\ ax}{\psi, x_1, x_4 \vdash \neg x_1 \vee x_4}\ \wedge_e \qquad \dfrac{}{\psi, x_1, x_4, \neg x_1 \vdash \neg x_1}\ ax \qquad \dfrac{\dfrac{}{\psi, x_1, x_4, \neg x_4 \vdash \neg x_4}\ ax \qquad \dfrac{}{\psi, x_1, x_4, \neg x_4 \vdash x_4}\ ax}{\psi, x_1, x_4, \neg x_4 \vdash \neg x_1}\ \neg_e}{\psi, x_1, x_4 \vdash \neg x_1}\ \vee_e$$

$$\dfrac{\dfrac{}{\psi, x_1, x_4 \vdash x_1}\ ax \qquad \dfrac{\vdots}{\psi, x_1, x_4 \vdash \neg x_1}}{\psi, x_1 \vdash \neg x_4}\ \neg_i$$

□ **3.3.** En suivant l'énoncé, on pose $\psi' = (x_2 \vee x_5) \wedge (\neg x_2 \vee \neg x_5)$. Raisonnons par l'absurde en supposant qu'il existe un arbre de preuve de $\varphi \wedge \psi' \to \neg x_2$. Le système déductif défini par la déduction naturelle étant correct, $\varphi \wedge \psi' \to \neg x_2$ est une tautologie.
Considérons la valuation v telle que $v(x_0) = \mathsf{F}$, $v(x_1) = \mathsf{V}$, $v(x_2) = \mathsf{V}$ et $v(x_5) = \mathsf{F}$. Alors c'est un modèle de φ et de ψ' : $v^*(\varphi) = \mathsf{V}$ et $v^*(\psi') = \mathsf{V}$ où v^* prolonge v sur l'ensemble des formules logiques. Par conséquent $v^*(\varphi \wedge \psi') = \mathsf{V}$. Mais $v^*(\varphi \wedge \psi' \to \neg x_2) = \mathsf{F}$. Il existe donc une valuation qui ne satisfait pas $\varphi \wedge \psi' \to \neg x_2$. Cette dernière n'est pas une tautologie et l'hypothèse selon laquelle il existerait un arbre de preuve de $\varphi \wedge \psi' \to \neg x_2$ est donc fausse.

Exercice 3

Question 1. La fonction `calc` calcule la somme des entiers stockés dans un tableau `arr`, entre les indices `idx_min` inclus et `idx_max` exclu, et stocke le résultat dans la référence `retval`. La fonction `main` appelle cette fonction avec un tableau exemple et affiche le résultat du calcul, stocké dans la référence `retval`. Dans l'exemple de l'énoncé, l'entier affiché est `31`.

Question 2.
□ **2.1.** Le calcul de la somme des entiers du tableau est stockée dans la référence `retval`. Si plusieurs *threads* contribuent à ce calcul, cette variable doit être protégée pour que chaque *thread* y ajoute son résultat. Un *mutex* peut assurer ce rôle de verrou.
□ **2.2.** On a $d_0 = 0$ et $d_k = n$.
□ **2.3.**
▷ **2.3.1.** Pour respecter les conditions de l'énoncé avec $n = 15$ et $k = 4$, on a nécessairement :

$$d_0 = 0 \qquad d_1 = 4 \qquad d_2 = 8 \qquad d_3 = 12 \qquad d_4 = 15$$

▷ **2.3.2.** On peut proposer la fonction suivante.

```
let idx n k i = i * (n / k) + min i (n mod k);;
```

Question 3. On modifie le programmation de l'énoncé pour protéger les calculs de la section critique, à savoir la mise à jour de `retval`.

```
let calc_thrd (arr, idx_min, idx_max, retval, mutex) =
  let s = ref 0 in
  for i = idx_min to idx_max-1 do
    s := !s + arr.(i)
  done;
  Mutex.lock mutex;
  retval := !s + !retval;
  Mutex.unlock mutex
;;

let main k =
  let arr = [|3;1;4;1;5;9;2;6|] in
  let n = Array.length arr in
  let s = ref 0 in
  let mutex = Mutex.create () in
  let thrd = Array.init k
             (fun i -> Thread.create calc_thrd
             (arr, idx n k i, idx n k (i+1), s, mutex))
  in
  for i = 0 to k-1 do Thread.join thrd.(i) done;
  Printf.printf "s = %d\n" !s
;;
```

Exercice 4

Question 1. Supposons que les deux *threads* souhaitent tous les deux entrer en section critique (SC). Par exemple, T_0 veut entrer en SC et il est le premier thread à passer *flag[0]* à *true* et *turn* à 1. T_1 veut entrer en SC, passe *flag[1]* à *true* puis modifie *turn* en lui attribuant la valeur 0. S'il est le dernier thread à modifier la variable *turn*, alors *(flag[1]) && (turn = 1)* est à *false*. La boucle *while* interne de T_0 s'arrête et T_0 peut entrer en SC. Dans le même temps, *(flag[0]) && (turn = 0)* est à *false* puisque *turn = 0* et T_1 ne peut pas entrer en SC. Quand T_0 quite la SC, *flag[0]* passe à *false* et *(flag[0]) && (turn = 0)* passe également à *false*. Si T_1 a la main avant que T_0 ne la reprenne, la boucle interne *while* de T_1 s'arrête et T_1 entre en SC. Si jamais T_0 ne rend par la main à T_1, il va passer *flag[0]* à *true* et *turn* à 1. Ce qui permet encore à T_1 d'entrer en SC tout en interdisant à T_0 d'y entrer.
Dans tous les cas, quand un *thread* entre en SC, l'autre ne peut y entrer. La contrainte d'*exclusion mutuelle* est donc satisfaite.

Question 2. D'après ce que nous venons de montrer, un *thread* peut toujours entrer en SC à un moment ou à un autre. En outre, dès qu'un *thread* a terminé, l'autre *thread* peut à son tour entrer en section critique. La contrainte d'*absence de famine* est donc satisfaite.

Question 3. La contrainte d'*absence de famine* implique celle d'*absence d'interblocage.*

Question 4. Enfin, là encore, l'analyse faite à la première question montre que chaque *thread* ne peut se faire doubler qu'au plus une fois par l'autre *thread.* La contrainte d'*attente bornée* est elle aussi satisfaite.