Correction des exercices de TD

Table des matières

1 TD/TP1 : Preuves en logique du premier ordre

Exercice 1 (Logique propositionnelle)

Démontrer les propositions suivantes dans LJ et LK :

1. $A \Rightarrow B \Rightarrow A$
2. $(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C$
3. $A \wedge B \Rightarrow B$
4. $B \Rightarrow A \vee B$
5. $(A \vee B) \Rightarrow (A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow C$
6. $A \Rightarrow \bot \Rightarrow \neg A$
7. $\bot \Rightarrow A$
8. $(A \Leftrightarrow B) \Rightarrow A \Rightarrow B$
9. $(A \Leftrightarrow B) \Rightarrow B \Rightarrow A$
10. $(A \Rightarrow B) \Rightarrow (B \Rightarrow A) \Rightarrow (A \Leftrightarrow B)$

1.

$$\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\,\text{ax}}{A \vdash B \Rightarrow A}\,\Rightarrow_{\text{right}}}{\vdash A \Rightarrow B \Rightarrow A}\,\Rightarrow_{\text{right}}$$

Ici, la preuve en LJ est la même que la preuve en LK.

2.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{B, A \vdash B}\,\text{ax} \quad \dfrac{}{A \vdash A}\,\text{ax}}{A \Rightarrow B, A \vdash B}\,\Rightarrow_{\text{left}} \quad \dfrac{}{C, A \Rightarrow B, A \vdash C}\,\text{ax}}{B \Rightarrow C, A \Rightarrow B, A \vdash C}\,\Rightarrow_{\text{left}} \quad \dfrac{}{A \Rightarrow B, A \vdash A}\,\text{ax}}{A \Rightarrow B \Rightarrow C, A \Rightarrow B, A \vdash C}\,\Rightarrow_{\text{left}}}{\vdash (A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C}\,\Rightarrow_{\text{right}} \times 3$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{B, A \vdash B, C}\,\text{ax} \quad \dfrac{}{A \vdash A, B, C}\,\text{ax}}{A \Rightarrow B, A \vdash B, C}\,\Rightarrow_{\text{left}} \quad \dfrac{}{C, A \Rightarrow B, A \vdash C}\,\text{ax}}{B \Rightarrow C, A \Rightarrow B, A \vdash C}\,\Rightarrow_{\text{left}} \quad \dfrac{}{A \Rightarrow B, A \vdash C, A}\,\text{ax}}{A \Rightarrow B \Rightarrow C, A \Rightarrow B, A \vdash C}\,\Rightarrow_{\text{left}}}{\vdash (A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C}\,\Rightarrow_{\text{right}} \times 3$$

Preuve dans LK

3.

$$\dfrac{\dfrac{\dfrac{}{A, B \vdash B}\,\text{ax}}{A \wedge B \vdash B}\,\wedge_{\text{left}}}{\vdash A \wedge B \Rightarrow B}\,\Rightarrow_{\text{right}}$$

Ici, la preuve en LJ est la même que la preuve en LK.

4.

$$\dfrac{\dfrac{\dfrac{}{B \vdash B}\,\text{ax}}{B \vdash A \vee B}\,\vee_{\text{right}}}{\vdash B \Rightarrow A \vee B}\,\Rightarrow_{\text{right}}$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{}{B \vdash A, B}\,\text{ax}}{B \vdash A \vee B}\,\vee_{\text{right}}}{\vdash B \Rightarrow A \vee B}\,\Rightarrow_{\text{right}}$$

Preuve dans LK

5.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B, A \Rightarrow C, C \vdash C}\,\text{ax} \quad \dfrac{}{B, A \Rightarrow C \vdash B}\,\text{ax}}{B, A \Rightarrow C, B \Rightarrow C \vdash C}\,\Rightarrow_{\text{left}} \quad \dfrac{\dfrac{}{A, C, B \Rightarrow C \vdash C}\,\text{ax} \quad \dfrac{}{A, B \Rightarrow C \vdash A}\,\text{ax}}{A, A \Rightarrow C, B \Rightarrow C \vdash C}\,\Rightarrow_{\text{left}}}{A \vee B, A \Rightarrow C, B \Rightarrow C \vdash C}\,\vee_{\text{left}}}{\vdash (A \vee B) \Rightarrow (A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow C}\,\Rightarrow_{\text{right}} \times 3$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B, A \Rightarrow C, C \vdash C}\,\text{ax} \quad \dfrac{}{B, A \Rightarrow C \vdash C, B}\,\text{ax}}{B, A \Rightarrow C, B \Rightarrow C \vdash C}\,\Rightarrow_{\text{left}} \quad \dfrac{\dfrac{}{A, C, B \Rightarrow C \vdash C}\,\text{ax} \quad \dfrac{}{A, B \Rightarrow C \vdash C, A}\,\text{ax}}{A, A \Rightarrow C, B \Rightarrow C \vdash C}\,\Rightarrow_{\text{left}}}{A \vee B, A \Rightarrow C, B \Rightarrow C \vdash C}\,\vee_{\text{left}}}{\vdash (A \vee B) \Rightarrow (A \Rightarrow C) \Rightarrow (B \Rightarrow C) \Rightarrow C}\,\Rightarrow_{\text{right}} \times 3$$

Preuve dans LK

6.

$$\dfrac{\dfrac{}{A, \bot \vdash \neg A}\,\bot_{\text{left}}}{\vdash A \Rightarrow \bot \Rightarrow \neg A}\,\Rightarrow_{\text{right}} \times 2$$

Ici, la preuve en LJ est la même que la preuve en LK.

7.

$$\dfrac{\dfrac{}{\bot \vdash A}\,\bot_{\text{left}}}{\vdash \bot \Rightarrow A}\,\Rightarrow_{\text{right}}$$

Ici, la preuve en LJ est la même que la preuve en LK.

8.

$$\dfrac{\dfrac{\dfrac{}{A \vdash A}\,\text{ax} \quad \dfrac{}{B, A \vdash B}\,\text{ax}}{A \Leftrightarrow B, A \vdash B}\,\Leftrightarrow_{\text{left}}}{\vdash (A \Leftrightarrow B) \Rightarrow A \Rightarrow B}\,\Rightarrow_{\text{right}} \times 2$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{}{A \vdash A, B}\,\text{ax} \quad \dfrac{}{A, B \vdash B}\,\text{ax}}{A \Leftrightarrow B, A \vdash B}\,\Leftrightarrow_{\text{left}}}{\vdash (A \Leftrightarrow B) \Rightarrow A \Rightarrow B}\,\Rightarrow_{\text{right}} \times 2$$

Preuve dans LK

9.

$$\dfrac{\dfrac{\dfrac{}{B \vdash B}\,\text{ax} \quad \dfrac{}{A, B \vdash A}\,\text{ax}}{A \Leftrightarrow B, B \vdash A}\,\Leftrightarrow_{\text{left}}}{\vdash (A \Leftrightarrow B) \Rightarrow B \Rightarrow A}\,\Rightarrow_{\text{right}} \times 2$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{}{B \vdash A, B}\,\text{ax} \quad \dfrac{}{A, B \vdash A}\,\text{ax}}{A \Leftrightarrow B, B \vdash A}\,\Leftrightarrow_{\text{left}}}{\vdash (A \Leftrightarrow B) \Rightarrow B \Rightarrow A}\,\Rightarrow_{\text{right}} \times 2$$

Preuve dans LK

10.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \Rightarrow B, A, B \vdash A}\,\text{ax} \quad \dfrac{}{A \Rightarrow B, B \vdash B}\,\text{ax}}{A \Rightarrow B, B \Rightarrow A, B \vdash A}\,\Rightarrow_{\text{left}} \quad \dfrac{\dfrac{}{B \Rightarrow A, A \vdash A}\,\text{ax} \quad \dfrac{}{B \Rightarrow A, B, A \vdash B}\,\text{ax}}{A \Rightarrow B, B \Rightarrow A, A \vdash B}\,\Rightarrow_{\text{left}}}{A \Rightarrow B, B \Rightarrow A \vdash A \Leftrightarrow B}\,\Leftrightarrow_{\text{right}}}{\vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow A) \Rightarrow (A \Leftrightarrow B)}\,\Rightarrow_{\text{right}} \times 2$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \Rightarrow B, A, B \vdash A}\,\text{ax} \quad \dfrac{}{A \Rightarrow B, B \vdash A, B}\,\text{ax}}{A \Rightarrow B, B \Rightarrow A, B \vdash A}\,\Rightarrow_{\text{left}} \quad \dfrac{\dfrac{}{B \Rightarrow A, A \vdash B, A}\,\text{ax} \quad \dfrac{}{B \Rightarrow A, B, A \vdash B}\,\text{ax}}{A \Rightarrow B, B \Rightarrow A, A \vdash B}\,\Rightarrow_{\text{left}}}{A \Rightarrow B, B \Rightarrow A \vdash A \Leftrightarrow B}\,\Leftrightarrow_{\text{right}}}{\vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow A) \Rightarrow (A \Leftrightarrow B)}\,\Rightarrow_{\text{right}} \times 2$$

Preuve dans LK

Exercice 2 (Logique du premier ordre)

Démontrer les propositions suivantes dans LJ et LK (si la proposition n'admet pas de preuve intuitionniste, démontrer la proposition dans $\text{LJ}_{(\text{em})}$) :

1. $\forall x.P(x) \Rightarrow \exists y.P(y) \vee Q(y)$
2. $(\exists x.P(x) \vee Q(x)) \Rightarrow (\exists x.P(x)) \vee (\exists x.Q(x))$
3. $(\forall x.P(x)) \wedge (\forall x.Q(x)) \Rightarrow \forall x.P(x) \wedge Q(x)$
4. $(\forall x.P(x) \wedge Q(x)) \Rightarrow (\forall x.P(x)) \wedge (\forall x.Q(x))$
5. $(\forall x.\neg P(x)) \Rightarrow \neg(\exists x.P(x))$
6. $\neg(\forall x.P(x)) \Rightarrow \exists x.\neg P(x)$

1.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(x) \vdash P(x)}\,\text{ax}}{P(x) \vdash P(x) \vee Q(x)}\,\vee_{\text{right}}}{P(x) \vdash \exists y.P(y) \vee Q(y)}\,\exists_{\text{right}}}{\vdash P(x) \Rightarrow \exists y.P(y) \vee Q(y)}\,\Rightarrow_{\text{right}}}{\vdash \forall x.P(x) \Rightarrow \exists y.P(y) \vee Q(y)}\,\forall_{\text{right}}$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(x) \vdash P(x), Q(x)}\,\text{ax}}{P(x) \vdash P(x) \vee Q(x)}\,\vee_{\text{right}}}{P(x) \vdash \exists y.P(y) \vee Q(y)}\,\exists_{\text{right}}}{\vdash P(x) \Rightarrow \exists y.P(y) \vee Q(y)}\,\Rightarrow_{\text{right}}}{\vdash \forall x.P(x) \Rightarrow \exists y.P(y) \vee Q(y)}\,\forall_{\text{right}}$$

Preuve dans LK

2.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{Q(x) \vdash Q(x)}\,\text{ax}}{Q(x) \vdash (\exists x.Q(x))}\,\exists_{\text{right}}}{Q(x) \vdash (\exists x.P(x)) \vee (\exists x.Q(x))}\,\vee_{\text{right}} \quad \dfrac{\dfrac{\dfrac{}{P(x) \vdash P(x)}\,\text{ax}}{P(x) \vdash (\exists x.P(x))}\,\exists_{\text{right}}}{P(x) \vdash (\exists x.P(x)) \vee (\exists x.Q(x))}\,\vee_{\text{right}}}{P(x) \vee Q(x) \vdash (\exists x.P(x)) \vee (\exists x.Q(x))}\,\vee_{\text{left}}}{\exists x.P(x) \vee Q(x) \vdash (\exists x.P(x)) \vee (\exists x.Q(x))}\,\exists_{\text{left}}}{\vdash (\exists x.P(x) \vee Q(x)) \Rightarrow (\exists x.P(x)) \vee (\exists x.Q(x))}\,\Rightarrow_{\text{right}}$$

Preuve dans LJ

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{Q(x) \vdash P(x), Q(x)}\,\text{ax} \quad \dfrac{}{P(x) \vdash P(x), Q(x)}\,\text{ax}}{P(x) \vee Q(x) \vdash P(x), Q(x)}\,\vee_{\text{left}}}{P(x) \vee Q(x) \vdash \exists x.P(x), \exists x.Q(x)}\,\exists_{\text{right}} \times 2}{P(x) \vee Q(x) \vdash (\exists x.P(x)) \vee (\exists x.Q(x))}\,\vee_{\text{right}}}{\exists x.P(x) \vee Q(x) \vdash (\exists x.P(x)) \vee (\exists x.Q(x))}\,\exists_{\text{left}}}{\vdash (\exists x.P(x) \vee Q(x)) \Rightarrow (\exists x.P(x)) \vee (\exists x.Q(x))}\,\Rightarrow_{\text{right}}$$

Preuve dans LK

3.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(x), Q(x) \vdash Q(x)}\,\text{ax} \quad \dfrac{}{P(x), Q(x) \vdash P(x)}\,\text{ax}}{P(x), Q(x) \vdash P(x) \wedge Q(x)}\,\wedge_{\text{right}}}{\forall x.P(x), \forall x.Q(x) \vdash P(x) \wedge Q(x)}\,\forall_{\text{left}} \times 2}{\forall x.P(x), \forall x.Q(x) \vdash \forall x.P(x) \wedge Q(x)}\,\forall_{\text{right}}}{(\forall x.P(x)) \wedge (\forall x.Q(x)) \vdash \forall x.P(x) \wedge Q(x)}\,\wedge_{\text{left}}}{\vdash (\forall x.P(x)) \wedge (\forall x.Q(x)) \Rightarrow \forall x.P(x) \wedge Q(x)}\,\Rightarrow_{\text{right}}$$

Ici, la preuve en LJ est la même que la preuve en LK.

4.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(x), Q(x) \vdash Q(x)}\,\text{ax}}{P(x) \wedge Q(x) \vdash Q(x)}\,\wedge_{\text{left}}}{\forall x.P(x) \wedge Q(x) \vdash Q(x)}\,\forall_{\text{left}}}{\forall x.P(x) \wedge Q(x) \vdash \forall x.Q(x)}\,\forall_{\text{right}} \quad \dfrac{\dfrac{\dfrac{\dfrac{}{P(x), Q(x) \vdash P(x)}\,\text{ax}}{P(x) \wedge Q(x) \vdash P(x)}\,\wedge_{\text{left}}}{\forall x.P(x) \wedge Q(x) \vdash P(x)}\,\forall_{\text{left}}}{\forall x.P(x) \wedge Q(x) \vdash \forall x.P(x)}\,\forall_{\text{right}}}{\forall x.P(x) \wedge Q(x) \vdash (\forall x.P(x)) \wedge (\forall x.Q(x))}\,\wedge_{\text{right}}}{\vdash (\forall x.P(x) \wedge Q(x)) \Rightarrow (\forall x.P(x)) \wedge (\forall x.Q(x))}\,\Rightarrow_{\text{right}}$$

Ici, la preuve en LJ est la même que la preuve en LK.

5.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(x) \vdash P(x)}\,\text{ax}}{\neg P(x), P(x) \vdash}\,\neg_{\text{left}}}{\forall x.\neg P(x), P(x) \vdash}\,\forall_{\text{left}}}{\forall x.\neg P(x), \exists x.P(x) \vdash}\,\exists_{\text{left}}}{\forall x.\neg P(x) \vdash \neg(\exists x.P(x))}\,\neg_{\text{right}}}{\vdash (\forall x.\neg P(x)) \Rightarrow \neg(\exists x.P(x))}\,\Rightarrow_{\text{right}}$$

Ici, la preuve en LJ est la même que la preuve en LK.

6.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\neg P(x) \vdash \neg P(x)}\,\text{ax}}{\neg P(x) \vdash \exists x.\neg P(x)}\,\exists_{\text{right}}}{\neg\exists x.\neg P(x), \neg P(x) \vdash}\,\neg_{\text{left}}}{\neg\exists x.\neg P(x) \vdash \neg\neg P(x)}\,\neg_{\text{right}}}{\neg\exists x.\neg P(x) \vdash P(x)}\,\text{em}}{\neg\exists x.\neg P(x) \vdash \forall x.P(x)}\,\forall_{\text{right}}}{\neg(\forall x.P(x)), \neg\exists x.\neg P(x) \vdash}\,\neg_{\text{left}}}{\neg(\forall x.P(x)) \vdash \neg\neg\exists x.\neg P(x)}\,\neg_{\text{right}}}{\neg(\forall x.P(x)) \vdash \exists x.\neg P(x)}\,\text{em}}{\vdash \neg(\forall x.P(x)) \Rightarrow \exists x.\neg P(x)}\,\Rightarrow_{\text{right}}$$

Preuve dans $\text{LJ}_{(\text{em})}$
(pas de preuve dans LJ possible)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(x) \vdash P(x)}\,\text{ax}}{\vdash \neg P(x), P(x)}\,\neg_{\text{right}}}{\vdash \exists x.\neg P(x), P(x)}\,\exists_{\text{right}}}{\vdash \exists x.\neg P(x), \forall x.P(x)}\,\forall_{\text{right}}}{\neg(\forall x.P(x)) \vdash \exists x.\neg P(x)}\,\neg_{\text{left}}}{\vdash \neg(\forall x.P(x)) \Rightarrow \exists x.\neg P(x)}\,\Rightarrow_{\text{right}}$$

Preuve dans LK

Exercice 3 (Preuves en Coq)

Démontrer les propositions des exercices 1 et 2 en `Coq`.
On rappelle que pour lancer `Coq`, il suffit de se mettre dans un terminal et de taper la commande `coqide`, qui lance l'IDE de `Coq`.

▶ Voir TP1

2 TD/TP2 : Preuves en logique équationnelle

Exercice 1 (Preuves dans les CCC dans LJ_{EQ})

Dans les Catégories Cartésiennes Closes (CCC), tous les isomorphismes de types sont capturés par une théorie équationnelle démontrée complètement par Sergei Soloviev en 1983. Cette théorie est la suivante :

1. $\forall x, y.x \times y \doteq y \times x$
2. $\forall x, y, z.x \times (y \times z) \doteq (x \times y) \times z$
3. $\forall x, y, z.((x \times y) \to z) \doteq (x \to (y \to z))$
4. $\forall x, y, z.(x \to (y \times z)) \doteq (x \to y) \times (x \to z)$
5. $\forall x.x \times \mathbb{1} \doteq x$
6. $\forall x.(x \to \mathbb{1}) \doteq \mathbb{1}$
7. $\forall x.(\mathbb{1} \to x) \doteq x$

où $\mathbb{1}$ est une constante.
Démontrer dans cette théorie en utilisant LJ_{EQ} les propositions suivantes :

1. $\forall x, y.x \times (y \to \mathbb{1}) \doteq x$
2. $\forall x, y.((x \times \mathbb{1}) \times y) \doteq (y \times (\mathbb{1} \times x))$
3. $\forall x, y, z.((x \times \mathbb{1}) \to (y \times (z \times \mathbb{1}))) \doteq (((x \times \mathbb{1}) \to (z \to \mathbb{1}) \times z) \times (\mathbb{1} \to (x \to y)))$

1. Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash x \doteq x}\,\text{refl}}{\vdash x \times \mathbb{1} \doteq x}\,=_{\text{right}_2},\ (5)}{\vdash x \times (y \to \mathbb{1}) \doteq x}\,=_{\text{right}_2},\ (6),\ \sigma = [y/x]}{\vdash \forall x, y.x \times (y \to \mathbb{1}) \doteq x}\,\forall_{\text{right}} \times 2$$

2. Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash (x \times y) \doteq (x \times y)}\,\text{refl}}{\vdash (x \times y) \doteq (y \times x)}\,=_{\text{right}_2},\ (1)}{\vdash ((x \times \mathbb{1}) \times y) \doteq (y \times (x \times \mathbb{1}))}\,=_{\text{right}_2},\ (5)}{\vdash ((x \times \mathbb{1}) \times y) \doteq (y \times (\mathbb{1} \times x))}\,=_{\text{right}_2},\ (1)}{\vdash \forall x, y.((x \times \mathbb{1}) \times y) \doteq (y \times (\mathbb{1} \times x))}\,\forall_{\text{right}} \times 2$$

3. Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash (x \to y) \times (x \to z) \doteq (x \to y) \times (x \to z)}\,\text{refl}}{\vdash (x \to y) \times (x \to z) \doteq ((x \to z) \times (x \to y))}\,=_{\text{right}_2},\ (1),\ \sigma = [(x \to z)/x, (x \to y)/y]}{\vdash (x \to y) \times (x \to z) \doteq ((x \to z \times \mathbb{1}) \times (x \to y)))}\,=_{\text{right}_2},\ (5),\ \sigma = [z/x]}{\vdash (x \to y) \times (x \to z) \doteq ((x \to \mathbb{1} \times z) \times (x \to y)))}\,=_{\text{right}_2},\ (1),\ \sigma = [\mathbb{1}/x, z/y]}{\vdash (x \to y) \times (x \to z) \doteq ((x \to \mathbb{1} \times z) \times (\mathbb{1} \to (x \to y)))}\,=_{\text{right}_2},\ (7),\ \sigma = [x \to y/x]}{\vdash (x \to y) \times (x \to z) \doteq ((x \to (z \to \mathbb{1}) \times z) \times (\mathbb{1} \to (x \to y)))}\,=_{\text{right}_2},\ (6),\ \sigma = [z/x]}{\vdash (x \to (y \times z)) \doteq ((x \to (z \to \mathbb{1}) \times z) \times (\mathbb{1} \to (x \to y)))}\,=_{\text{right}_2},\ (4)}{\vdash (x \to (y \times (z \times \mathbb{1}))) \doteq ((x \to (z \to \mathbb{1}) \times z) \times (\mathbb{1} \to (x \to y)))}\,=_{\text{right}_2},\ (5),\ \sigma = [z/x]}{\vdash ((x \times \mathbb{1}) \to (y \times (z \times \mathbb{1}))) \doteq (((x \times \mathbb{1}) \to (z \to \mathbb{1}) \times z) \times (\mathbb{1} \to (x \to y)))}\,=_{\text{right}_2},\ (5)}{\vdash \forall x, y, z.((x \times \mathbb{1}) \to (y \times (z \times \mathbb{1}))) \doteq (((x \times \mathbb{1}) \to (z \to \mathbb{1}) \times z) \times (\mathbb{1} \to (x \to y)))}\,\forall_{\text{right}} \times 3$$

Exercice 2 (Preuves en arithmétique de Peano dans $\mathbf{LJ_{EQ}}$)

À la fin du 19ème siècle, Giuseppe Peano à proposé un ensemble d'axiomes (pour la plupart équationnels) pour l'arithmétique. Ces axiomes sont les suivants (nous avons omis un axiome qui est en fait un schéma d'axiome au premier ordre et qui représente la récurrence) :

1. $\forall x.\neg(s(x) \doteq o)$
2. $\forall x.\exists z.\neg(x \doteq o) \Rightarrow s(z) \doteq x$
3. $\forall x, y.s(x) \doteq s(y) \Rightarrow x \doteq y$
4. $\forall x.x + o \doteq x$
5. $\forall x, y.x + s(y) \doteq s(x + y)$
6. $\forall x.x \times o \doteq o$
7. $\forall x, y.x \times s(y) \doteq (x \times y) + x$

où o est une constante, s, $+$ et $\times$ des fonctions.
Démontrer dans cette théorie et en utilisant LJ_{EQ} les propositions suivantes :

— $1 + 2 \doteq 3$

— $2 + 2 \doteq 4$

— $2 \times 2 \doteq 4$

NB : $2 \equiv s(s(o))$.

— Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash s(s(s(o))) \doteq s(s(s(o)))}\,\text{refl}}{\vdash s(s(s(o)) + o)) \doteq s(s(s(o)))}\,=_{\text{right}_2},\ (4),\ \sigma = [s(o)/x]}{\vdash s(s(s(o) + s(o)) \doteq s(s(s(o)))}\,=_{\text{right}_2},\ (5),\ \sigma = [s(o)/x, o/y]}{\vdash s(s(o) + s(o)) \doteq s(s(s(o)))}\,=_{\text{right}_2},\ (5),\ \sigma = [s(o)/x, s(o)/y]}{\vdash s(o) + s(s(o)) \doteq s(s(s(o)))}$$

— Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash s(s(s(s(o)))) \doteq s(s(s(s(o))))}\,\text{refl}}{\vdash s(s(s(s(o) + o))) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (4),\ \sigma = [s(o)/x]}{\vdash s(s(s(s(o)) + s(o))) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (5),\ \sigma = [s(s(o))/x, o/y]}{\vdash s(s(s(o)) + s(s(o))) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (5),\ \sigma = [s(s(o))/x, s(o)/y]}{\vdash s(s(o)) + s(s(o)) \doteq s(s(s(s(o))))}$$

— Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash s(s(s(s(o)))) \doteq s(s(s(s(o))))}\,\text{refl}}{\vdash s(s(s(s(o) + o))) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (4),\ \sigma = [s(o)/x]}{\vdash s(s(s(s(o)) + s(o))) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (5),\ \sigma = [s(s(o))/x, o/y]}{\vdash s(s(s(o)) + s(s(o))) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (5),\ \sigma = [s(s(o))/x, s(o)/y]}{\vdash s(s(o)) + s(s(o)) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (4),\ \sigma = [o/x]}{\vdash s(s(o + o)) + s(s(o)) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (5),\ \sigma = [o/x, o/y]}{\vdash s(o + s(o)) + s(s(o)) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (5),\ \sigma = [o/x, s(o)/y]}{\vdash (o + s(s(o))) + s(s(o)) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (6),\ \sigma = [s(s(o))/x]}{\vdash ((s(s(o)) \times o) + s(s(o))) + s(s(o)) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (7),\ \sigma = [s(s(o))/x, o/y]}{\vdash (s(s(o)) \times s(s(o))) + s(s(o)) \doteq s(s(s(s(o))))}\,=_{\text{right}_2},\ (7),\ \sigma = [s(s(o))/x, s(o)/y]}{\vdash s(s(o)) \times s(s(o)) \doteq s(s(s(s(o))))}$$

Exercice 3 (Isomorphismes de types dans les CCC en Coq)

En Coq, la théorie équationnelle correspondant aux isomorphismes de types dans les CCC peut être implémentée comme suit :

```
Open Scope type_scope.

Section iso_axioms.

Variables A B C : Set.

Axiom Com : A * B = B * A.
Axiom Ass : A * (B * C) = A * B * C.
Axiom Cur : (A * B -> C) = (A -> B -> C).
Axiom Dis : (A -> B * C) = (A -> B) * (A -> C).
Axiom P_unit : A * unit = A.
Axiom AR_unit : (A -> unit) = unit.
Axiom AL_unit : (unit -> A) = A.

End iso_axioms.
```

1. Démontrer les lemmes suivants dans cette théorie en Coq :

   ```
   Lemma isos_ex1 : forall A B : Set, A * (B -> unit) = A.

   Lemma isos_ex2 : forall A B : Set, A * unit * B = B * (unit * A).

   Lemma isos_ex3 : forall A B C : Set,
       (A * unit -> B * (C * unit)) =
       (A * unit -> (C -> unit) * C) * (unit -> A -> B).
   ```

2. Écrire une tactique qui normalise les expressions selon les axiomes de la théorie (attention, tous les axiomes ne sont pas bons à prendre).
3. Démontrer les propositions précédentes à l'aide de cette tactique.

▶ Voir TP2

Exercice 4 (Arithmétique de Peano en Coq)

En `Coq`, la théorie de l'arithmétique de Peano peut être implémentée comme suit :

```
Section Peano.

Parameter N : Set.
Parameter o : N.
Parameter s : N -> N.
Parameters plus mult : N -> N -> N.
Variables x y : N.
Axiom ax1 : ~((s z) = o).
Axiom ax2 : exists z, ~(x = o) -> (s z) = x.
Axiom ax3 : (s x) = (s y) -> x = y.
Axiom ax4 : (plus x o) = x.
Axiom ax5 : (plus x (s y)) = (s (plus x y)).
Axiom ax6 : (mult x o) = o.
Axiom ax7 : (mult x (s y)) = (plus (mult x y) x).

End Peano.
```

1. Démontrer les propositions suivantes dans cette théorie en `Coq` :
   - $1 + 2 = 3$
   - $2 + 2 = 4$
   - $2 \times 2 = 4$
2. Écrire une tactique qui calcule automatiquement dans cette théorie.
3. Démontrer les propositions précédentes à l'aide de cette tactique.
4. Même question en utilisant la tactique `autorewrite` (voir la documentation).

▶ Voir TP2

3 TD/TP3 : Types inductifs

Exercice 1 (Fonctions et preuves inductives sur les entiers)

1. Écrire la fonction *mult* sur les entiers naturels $\mathcal{N}$.
2. Démontrer que : $\forall n \in \mathcal{N}.mult(2, n) = plus(n, n)$.
3. Démontrer que : $\forall n \in \mathcal{N}.mult(n, 2) = plus(n, n)$.

1. Soit $mult : \mathcal{N} \times \mathcal{N} \to \mathcal{N}$.
   (m_1) $\forall n \in \mathcal{N}.mult(0, n) = 0$.
   (m_2) $\forall p, n \in \mathcal{N}.mult((S\ p), n) = plus(mult(p, n), n)$.

2. On sait que $2 = (S\ (S\ 0))$. Ainsi :

$$\begin{aligned} mult((S\ (S\ 0)), n) &= plus(mult((S\ 0), n), n) && \text{par } m_2 \\ &= plus(plus(mult(0, n), n), n) && \text{par } m_2 \\ &= plus(plus(0, n), n) && \text{par } m_1 \\ &= plus(n, n) && \text{par } plus(0, n) = n \end{aligned}$$

3. On rappelle la définition de $plus$:

   $(p_1)\ \forall n \in \mathcal{N}.plus(0, n) = n$

   $(p_2)\ \forall p, n \in \mathcal{N}.plus((S\ p), n) = (S\ plus(p, n))$

   Lemme 1. $\forall n \in \mathcal{N}.plus(n, 0) = n$

   Démonstration. On prouve le lemme 1 par induction structurelle :

   Base Pour $n = 0$, on a $plus(0, 0) = 0$ par p_1. Par réflexivité, la propriété est vérifiée pour le cas de base.

   Induction On suppose que $plus(n, 0) = n$. Montrons que $plus((S\ n), 0) = (S\ n)$.

$$\begin{aligned} plus((S\ n), 0) &= (S\ plus(n, 0)) && \text{par } p_2 \\ &= (S\ n) && \text{par hypothèse d'induction} \end{aligned}$$

   Par réflexivité, la propriété est vérifiée $\forall n \in \mathcal{N}$.

   □

   Lemme 2. $\forall n, m \in \mathcal{N}.plus(m, (S\ n)) = (S\ plus(m, n))$

   Démonstration. On prouve le lemme 2 par induction structurelle :

   Base Pour $m = 0$, on a $plus(0, (S\ n)) = (S\ n)$ par p_1, et $(S\ plus(0, n)) = (S\ n)$. Par réflexivité, $(S\ n) = (S\ n)$ et donc la propriété est vérifiée pour le cas de base.

   Induction On suppose que la propriété est vraie pour un m quelconque, c'est à dire que $plus(m, (S\ n)) = (S\ plus(m, n))$. Montrons que $plus((S\ m), (S\ n)) = (S\ (S\ plus(m, n)))$.

$$\begin{aligned} plus((S\ m), (S\ n)) &= (S\ plus(m, (S\ n))) && \text{par } p_2 \\ &= (S\ (S\ plus(m, n)) && \text{par hypothèse d'induction} \end{aligned}$$

   Par réflexivité, on a bien $plus((S\ m), (S\ n)) = (S\ (S\ plus(m, n)))$, donc la propriété est vérifiée $\forall n, m \in \mathcal{N}$.

   □

   On peut maintenant montrer $\forall n \in \mathcal{N}.mult(n, 2) = plus(n, n)$ par induction :

   Base Pour $n = 0$, on a $mult(0, 2) = 0$ par m_1, et $plus(0, 0) = 0$ par p_1. Par réflexivité, $mult(0, 2) = plus(0, 0)$ et ainsi la propriété est vérifiée pour le cas de base.

   Induction On suppose que $\forall n \in \mathcal{N}.mult(n, 2) = plus(n, n)$. Montrons que $\forall n \in \mathcal{N}.mult((S\ n), 2) = plus((S\ n), (S\ n))$. On sait que $2 = (S\ (S\ 0))$.

$$\begin{aligned} mult((S\ n), (S\ (S\ 0))) &= plus(mult(n, 2), (S\ (S\ 0))) && \text{par } m_2 \\ &= plus(plus(n, n), (S\ (S\ 0))) && \text{par hypothèse d'induction} \\ &= (S\ plus(plus(n, n), (S\ 0))) && \text{par le lemme 2} \\ &= (S\ (S\ plus(plus(n, n), 0))) && \text{par le lemme 2} \\ &= (S\ (S\ plus(n, n))) && \text{par le lemme 1} \end{aligned}$$

De plus :

$$\begin{aligned} plus((S\ n),(S\ n)) &= (S\ plus(n,(S\ n))) && \text{par } p_2 \\ &= (S\ (S\ plus(n,n))) && \text{par le lemme 2} \end{aligned}$$

Par réflexivité, on conclut que $mult((S\ n),2) = plus((S\ n),(S\ n))$ et donc la propriété est vérifiée $\forall n \in \mathcal{N}$.

Exercice 2 (Fonctions et preuves inductives sur les listes)

1. Écrire la fonction rev qui inverse les éléments d'une liste.
2. Démontrer que : $\forall l \in \mathcal{L}, \forall e \in \mathcal{A}. rev(app(l,[e])) = e :: rev(l)$.
3. Démontrer que : $\forall l \in \mathcal{L}. rev(rev(l)) = l$.

1. On rappelle que l'ensemble $\mathcal{A}$ est le type des éléments de la liste. On peut définir $rev : \mathcal{L} \to \mathcal{L}$:
   (l_1) $rev([\,]) = [\,]$
   (l_2) $\forall e \in \mathcal{A}, \forall l \in \mathcal{L}. rev(e :: l) = app(rev(l),[e])$.
2. On rappelle la définition de app :
   (a_1) $\forall l \in \mathcal{L}. app([\,],l) = l$
   (a_2) $\forall e \in \mathcal{A}, \forall l_1, l_2 \in \mathcal{L}. app(e :: l_1, l_2) = e :: app(l_1, l_2)$.

   Lemme 3. $\forall l \in \mathcal{L}, \forall e \in \mathcal{A}. rev(app(l,[e])) = e :: rev(l)$

   Démonstration. On le démontre par induction.

   Base Pour $l = [\,]$ et $e \in \mathcal{A}$, on a $rev(app([\,],[e])) = rev([e]) = [e]$ et $e :: rev([\,]) = e :: [\,] = [e]$. Par réflexivité, $rev(app([\,],e)) = e :: rev([\,])$.

   Induction On suppose que $\forall l \in \mathcal{L}, \forall e_1 \in \mathcal{A}. rev(app(l,[e_1])) = e_1 :: rev(l)$. Montrons que $\forall l \in \mathcal{L}, \forall e_1, e_2 \in \mathcal{A}. rev(app(e_2 :: l,[e_1])) = e_1 :: rev(e_2 :: l)$.

$$\begin{aligned} rev(app(e_2 :: l,[e_1])) &= rev(e_2 :: app(l,[e_1])) && \text{par } a_2 \\ &= app(rev(app(l,[e_1])),[e_2]) && \text{par } l_2 \\ &= app(e_1 :: rev(l),[e_2]) && \text{par hypothèse d'induction} \\ &= e_1 :: app(rev(l),[e_2]) && \text{par } a_2 \\ &= e_1 :: rev(e_2 :: l) && \text{par } l_2 \end{aligned}$$

   Par réflexivité, $rev(app(e_2 :: l,[e_1])) = e_1 :: rev(e_2 :: l)$, donc la propriété est vérifiée $\forall e \in \mathcal{A}$ et $\forall l \in \mathcal{L}$.

   □

3. On démontre $\forall l \in \mathcal{L}. rev(rev(l)) = l$ par induction.

   Base Pour $l = [\,]$, on a $rev(rev([\,])) = rev([\,]) = [\,]$. Par réflexivité, la propriété est vérifiée pour le cas de base.

   Induction On suppose que $\forall l \in \mathcal{L}. rev(rev(l)) = l$. Montrons que $\forall e \in \mathcal{A}, \forall l \in \mathcal{L}. rev(rev(e :: l)) = e :: l$.

$$\begin{aligned} rev(rev(e :: l)) &= rev(app(rev(l),[e])) && \text{par } l_2 \\ &= e :: rev(rev(l)) && \text{par le lemme 3} \\ &= e :: l && \text{par hypothèse d'induction} \end{aligned}$$

   Par réflexivité, $rev(rev(e :: l)) = e :: l$, donc la propriété est vérifiée $\forall l \in \mathcal{L}$.

Exercice 3 (Type inductif des formules en logique)

1. Définir le type des formules en logique propositionnelle.
2. Écrire la fonction *sub*, qui rend l'ensemble des sous-formules d'une formule F.
3. Écrire la fonction *nbc*, qui rend l'ensemble des connecteurs d'une formule F.
4. Écrire le schéma d'induction structurelle des formules.
5. Démontrer que : $|sub(F)| \leq 2 \times nbc(F) + 1$ pour toute formule F.

1. Soit $\mathcal{V}$ l'ensemble des variables de propositions. Soit $\mathcal{F}$ l'ensemble des formules de logique propositionnelle. On a $\mathcal{F}$:
   - Si $A \in \mathcal{V}$, alors $A \in \mathcal{F}$,
   - $\top, \bot \in \mathcal{F}$,
   - $\forall \phi \in \mathcal{F}, \neg\phi \in \mathcal{F}$,
   - $\forall \phi_1, \phi_2 \in \mathcal{F}, \phi_1\ c\ \phi_2 \in \mathcal{F}$ (avec $c \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow\}$)
2. $sub : \mathcal{F} \to 2^{\mathcal{F}}$:
   - $sub(\top) = \{\top\}$; $sub(\bot) = \{\bot\}$,
   - $\forall A \in \mathcal{V}.sub(A) = \{A\}$,
   - $\forall \phi \in \mathcal{F}.sub(\neg\phi) = \{\neg\phi\} \cup sub(\phi)$,
   - $\forall \phi_1, \phi_2 \in \mathcal{F}.sub(\phi\ c\ \phi_2) = \{\phi_1\ c\ \phi_2\} \cup sub(\phi_1) \cup sub(\phi_2)$ (avec $c \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow\}$).
3. $nbc : \mathcal{F} \to \mathcal{N}$:
   - $nbc(\top) = 0$; $nbc(\bot) = 0$,
   - $\forall A \in \mathcal{V}.nbc(A) = 0$,
   - $\forall \phi \in \mathcal{F}.nbc(\neg\phi) = 1 + nbc(\phi)$,
   - $\forall \phi_1, \phi_2 \in \mathcal{F}.nbc(\phi\ c\ \phi_2) = 1 + nbc(\phi_1) + nbc(\phi_2)$ (avec $c \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow\}$).
4. $\forall P \in (\mathcal{F} \to Prop).(\forall A \in \mathcal{V}.P(A)) \Rightarrow (\forall \phi \in \mathcal{F}.P(\phi) \Rightarrow P(\neg\phi)) \Rightarrow (\forall \phi_1, \phi_2 \in \mathcal{F}.(P(\phi_1) \wedge P(\phi_2) \Rightarrow P(\phi_1 \wedge \phi_2) \wedge (P(\phi_1) \vee P(\phi_2) \Rightarrow P(\phi_1 \vee \phi_2)) \wedge (P(\phi_1) \Rightarrow P(\phi_2) \Rightarrow P(\phi_1 \Rightarrow \phi_2)) \wedge (P(\phi_1) \Leftrightarrow P(\phi_2) \Rightarrow P(\phi_1 \Leftrightarrow \phi_2)))) \Rightarrow \forall \phi \in \mathcal{F}.P(\phi)$
5. On démontre $\forall F \in \mathcal{F}.|sub(F)| \leq 2 \times nbc(F) + 1$ par induction :

   Base $F \in \mathcal{V}$ ou $F \in \{\top, \bot\}$, alors $sub(F) = \{F\}$ et $nbc(F) = 2 \times 0 + 1 = 1$. Par réflexivité, $|\{F\}| = 1$, donc la propriété est vérifiée pour le cas de base.

   Induction On suppose que $\forall \phi, \phi_1, \phi_2 \in \mathcal{F}.|sub(\phi)| \leq 2 \times nbc(\phi) + 1$.
   - $F = \neg\phi$, dans ce cas, $sub(F) = \{F\} \cup sub(\phi)$. On a $|sub(F)| = 1 + |sub(\phi)| \leq 1 + 2 \times nbc(\phi) + 1 = 2nbc(\phi) + 2$ et $nbc(F) = 1 + nbc(\phi)$, on a $2(nbc(F)) + 1 = 2nbc(\phi) + 2$, on a donc bien $|sub(F)| \leq 2(nbc(F)) + 1$.
   - $F = \phi_1\ c\ \phi_2$ avec $c \in \{\vee, \wedge, \Rightarrow, \Leftrightarrow\}$. Dans ce cas, $sub(F) = \{F\} \cup sub(\phi_1) \cup sub(\phi_2)$. On a $|sub(F)| = 1 + |sub(\phi_1)| + |sub(\phi_2)| \leq 1 + 2nbc(\phi_1) + 1 + 2nbc(\phi_2) + 1 = 2(nbc(\phi_1) + nbc(\phi_2)) + 3$ et $nbc(F) = 1 + nbc(\phi_1) + nbc(\phi_2)$, on a $2(nbc(F)) + 1 = 2(nbc(\phi_1) + nbc(\phi_2)) + 3$. On a donc bien $|sub(F)| \leq 2(nbc(F)) + 1$.

Exercice 4 (Relations inductives sur les listes)

1. Spécifier la relation « être une permutation de » pour deux listes.
2. Démontrer que la liste $[1; 2; 3]$ est une permutation de $[3; 2; 1]$.
3. Spécifier la relation « être triée » pour une liste.
4. Démontrer que la liste $[1; 2; 3]$ est triée.

1. On défini la relation inductive $is\_perm : \mathcal{L} \times \mathcal{L} \to Prop$ de la façon suivante :
   ($perm_1$) On a : $is\_perm([\,], [\,])$,
   ($perm_2$) On a aussi : $\forall e \in \mathcal{A}.is\_perm([e], [e])$.
   ($perm_3$) $\forall l_1, l_2, l_3 \in \mathcal{L}.is\_perm(l_1, app(l_2, l_3)) \Rightarrow is\_perm(e :: l_1, app(l_2, e :: l_3))$.
2. On veut montrer que $is\_perm([1; 2; 3], [3; 2; 1])$.
   — Par $perm_3$, on veut montrer $is\_perm([2; 3], [3; 2])$.
   — Par $perm_3$, on veut montrer que $is\_perm([2], [2])$.
   — Démontré en utilisant le cas de base.
3. On défini la relation inductive $is\_sorted : \mathcal{L} \to Prop$ de la façon suivante :
   ($sort_1$) On a $is\_sorted([\,])$,
   ($sort_2$) On a $\forall n \in \mathcal{N}.is\_sorted([n])$,
   ($sort_3$) $\forall e, n \in \mathcal{N}.\forall l \in \mathcal{L}.is\_sorted(l + +[e]) \wedge e \leq n \Rightarrow is\_sorted(l + +[e; n])$.
4. On veut montrer que $is\_sorted([1; 2; 3])$.
   — Par $sort_3$, comme $2 < 3$, on veut montrer $is\_sorted([1; 2])$.
   — Par $sort_3$, comme $1 < 2$, on veut montrer $is\_sorted([1])$.
   — Démontré en utilisant $sort_2$.

Exercice 5 (Preuves en Coq)

Faire les exercices 1, 2 et 3 en Coq.

▶ Voir TP3

Exercice 6 (Preuves en Coq)

1. Écrire la relation inductive is_even (vue en cours).
2. Écrire une tactique qui démontre des buts de la forme $is\_even(n)$.
3. Écrire une tactique qui démontre les buts de la forme $\neg is\_even(n)$.
4. Écrire une tactique qui démontre les buts précédents indifféremment.
5. Écrire la fonction f_{is_even} qui teste si un entier est pair.
6. Démontrer que la fonction f_{is_even} est correcte vis à vis de la relation is_even.

▶ Voir TP3

4 TD/TP4 : Preuves par induction

Exercice 1 (Fonction factorielle)

1. Spécifier la fonction factorielle à l'aide d'une relation inductive.
2. Écrire la fonction factorielle.
3. Écrire le schéma d'induction fonctionnelle associé à cette fonction.
4. Démontrer la correction de la fonction en utilisant le schéma d'induction structurel.
5. Démontrer la correction de la fonction en utilisant le schéma d'induction fonctionnel.
6. Démontrer la complétude de la fonction en utilisant le schéma d'induction sur la relation.
7. Répondre aux questions précédentes en utilisant Coq.

1. Soit la relation inductive $is\_fact : \mathbb{N} \times \mathbb{N} \to Prop$:
   ($fact_1$) On a : $is\_fact(0, 1)$;
   ($fact_2$) $\forall n, p \in \mathbb{N}. is\_fact(n, p) \Rightarrow is\_fact((S\ n), p \times (S\ n))$.
2. On peut définir $fact : \mathbb{N} \to \mathbb{N}$:
   (f_1) $fact(0) = 1$
   (f_2) $\forall p \in \mathbb{N}. fact((S\ p)) = (S\ p) \times fact(p)$
3. $\forall P \in \mathbb{N} \times \mathbb{N} \to Prop. P(0, 1) \Rightarrow (\forall n \in \mathbb{N}. P(n, fact(n)) \Rightarrow P((S\ n), (S\ n) \times fact(n))) \Rightarrow \forall n \in \mathbb{N}. P(n, fact(n))$
4. On veut montrer que $\forall n, p \in \mathbb{N}. fact(n) = p \Rightarrow is\_fact(n, p)$ par induction structurelle :
   Base Soit $n = 0$. $\forall p \in \mathbb{N}. fact(0) = p \Rightarrow is\_fact(0, p)$. Or, $fact(0) = 1$, donc on veut montrer que $\forall p \in \mathbb{N}. 1 = p \Rightarrow is\_fact(0, p)$. On peut donc remplacer p par 1, et on doit démontrer que $is\_fact(0, 1)$, ce qui est le cas de base de la spécification inductive de is_fact.
   Induction On suppose que $\forall n, p \in \mathbb{N}. fact(n) = p \Rightarrow is\_fact(n, p)$. On veut montrer que $\forall n, p \in \mathbb{N}. fact((S\ n)) = (S\ n) \times p \Rightarrow is\_fact((S\ n), (S\ n) \times p)$:
   $$\begin{aligned} is\_fact((S\ n), (S\ n) \times p) &= is\_fact((S\ n), fact((S\ n))) && \text{par } fact((S\ n)) = (S\ n) \times p \\ &= is\_fact((S\ n), fact(n) \times (S\ n)) && \text{par la définition de } fact \\ &= is\_fact(n, fact(n)) && \text{par } fact_2 \\ &= fact(n) = fact(n) && \text{par } p = fact(n) \end{aligned}$$
   Par réflexivité, on a $is\_fact((S\ n), (S\ n) \times p) = \top$ si et seulement si $is\_fact(n, fact(n))$.
5. On veut montrer que $\forall n, p \in \mathbb{N}. fact(n) = p \Rightarrow is\_fact(n, p)$ par induction fonctionnelle :
   Base On doit montrer que $fact(0) = 1 \Rightarrow is\_fact(0, 1)$. Par f_1, $1 = 1 \Rightarrow is\_fact(0, 1)$, qui devient $\top \Rightarrow \top$ par $fact_1$. Le cas de base est donc vérifié.
   Induction On suppose $\forall n, p \in \mathbb{N}. fact(n) = p \Rightarrow is\_fact(n, p)$, et que $fact(n) \times (S\ n) = p$, et on veut montrer que $is\_fact((S\ n), p)$:
   $$\begin{aligned} is\_fact((S\ n), p) &= is\_fact((S\ n), fact(n) \times (S\ n)) && \text{par } p = fact(n) \times (S\ n) \\ &= is\_fact(n, fact(n)) && \text{par } fact_2 \\ &= fact(n) = fact(n) && \text{par l'hypothèse d'induction} \end{aligned}$$
   Par réflexivité, on a vérifié la propriété pour tout n.
6. On veut montrer que $\forall n, p \in \mathbb{N}. is\_fact(n, p) \Rightarrow fact(n) = p$ par induction sur la relation :

Base On doit montrer que $is\_fact(0,1) \Rightarrow fact(0) = 1$. Par f_1, on a $1 = 1$, et par réflexivité, le cas de base est vérifié.

Induction On suppose que $\forall n, m \in \mathbb{N}. is\_fact(n,m) \Rightarrow fact(n) = m$ et on veut montrer que $fact((S\ n)) = m \times (S\ n)$:

$$\begin{aligned} fact((S\ n)) &= fact(n) \times (S\ n) && \text{par } f_2 \\ &= m \times (S\ n) && \text{par l'hypothèse d'induction} \end{aligned}$$

On a bien $fact((S\ n)) = m \times (S\ n)$ et donc par réflexivité la propriété est vérifiée pour tout n.

7. ▶ Voir TP4

Exercice 2 (Fonction de parité)

Cet exercice est à faire entièrement en Coq

1. Écrire la relation inductive is_even vue en cours.
2. Écrire la fonction récursive f_{is_even} vue en cours.
3. Démontrer que : $\forall n \in \mathbb{N}. f_{is\_even}(n) = \top \Rightarrow is\_even(n)$.
4. Démontrer que : $\forall n \in \mathbb{N}. f_{is\_even}(n) = \bot \Rightarrow \neg is\_even(n)$.
5. Démontrer que : $\forall n \in \mathbb{N}. is\_even(n) \Rightarrow f_{is\_even}(n) = \top$.
6. Démontrer que : $\forall n \in \mathbb{N}. \neg is\_even(n) \Rightarrow f_{is\_even}(n) = \bot$.

▶ Voir TP4

Exercice 3 (Fonction de pgcd)

Cet exercice est à faire entièrement en Coq

1. Écrire la fonction gcd vue en cours.
2. Définir $divides(r, (a, b))$ qui exprime r divise a et b avec $r \in \mathbb{N}^*$ et $a, b \in \mathbb{N}$.
3. Démontrer que : $\forall a, b, r \in \mathbb{N}^*. gcd(a, b) = r \Rightarrow divides(r, (a, b))$.
4. Définir $bezout(r, (a, b))$ qui exprime qu'il existe $p, q \in \mathbb{Z}$ t.q. $p \times a + q \times b = r, r, a, b \in \mathbb{N}$.
5. Démontrer que : $\forall a, b, r \in \mathbb{N}^*. gcd(a, b) = r \Rightarrow bezout(r, (a, b))$.

▶ Voir TP4

5 TD/TP5 : Preuve de correction du tri par insertion

▶ Voir TP5

6 TD/TP6 : Prouver = Programmer

Exercice 1 (Logique implicative minimale)

Démontrer les propositions suivantes en utilisant les règles de la déduction naturelle vues en cours. Puis, transformer l'arbre de preuve en un arbre de typage pour le λ-calcul simplement typé. Dans chaque cas, vous indiquerez explicitement quel est le λ-terme représentant la preuve de la proposition.

1. $A \Rightarrow B \Rightarrow A$
2. $(A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C$

1. Preuve :

$$\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\,\text{ax}}{A \vdash B \Rightarrow A}\,\Rightarrow_I}{\vdash A \Rightarrow B \Rightarrow A}\,\Rightarrow_I$$

Arbre de typage correspondant :

$\dfrac{\dfrac{\dfrac{}{\iota_A, \iota_B \vdash \iota_A}\,\text{ax}}{\iota_A \vdash \iota_B \to \iota_A}\,\Rightarrow_I}{\vdash \iota_A \to \iota_B \to \iota_A}\,\Rightarrow_I$	$\dfrac{\dfrac{\dfrac{}{(x, \iota_A), (y, \iota_B) \vdash \iota_A}\,\text{ax}}{(x, \iota_A) \vdash \iota_B \to \iota_A}\,\Rightarrow_I}{\vdash \iota_A \to \iota_B \to \iota_A}\,\Rightarrow_I$	$\dfrac{\dfrac{\dfrac{(x, \iota_A) \in (x, \iota_A), (y, \iota_B)}{(x, \iota_A), (y, \iota_B) \vdash x : \iota_A}\,\text{Var}}{(x, \iota_A) \vdash \lambda y : \iota_B . x : \iota_B \to \iota_A}\,\text{Fun}}{\vdash \underbrace{\lambda x : \iota_A . \lambda y : \iota_b . x}_{\lambda\text{-terme}} : \iota_A \to \iota_B \to \iota_A}\,\Rightarrow_I$
Étape 1 : formules/types	Étape 2 : contexte	Étape 3 : preuves/termes

2. Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B \Rightarrow C}\,\text{ax} \quad \dfrac{}{\Gamma \vdash A}\,\text{ax}}{\Gamma \vdash B \Rightarrow C}\,\Rightarrow_E \quad \dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B}\,\text{ax} \quad \dfrac{}{\Gamma \vdash A}\,\text{ax}}{\Gamma \vdash B}\,\Rightarrow_E}{\Gamma = A \Rightarrow B \Rightarrow C, A \Rightarrow B, A \vdash C}\,\Rightarrow_I}{A \Rightarrow B \Rightarrow C, A \Rightarrow B \vdash A \Rightarrow C}\,\Rightarrow_I}{A \Rightarrow B \Rightarrow C \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow C}\,\Rightarrow_I}{\vdash (A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C}\,\Rightarrow_I$$

Arbre de typage correspondant :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(g, \iota_A \to \iota_B \to \iota_C) \in \Gamma}{\Gamma \vdash g : \iota_A \to \iota_B \to \iota_C}\,\text{Var} \quad \dfrac{(x, \iota_A) \in \Gamma}{\Gamma \vdash x : \iota_A}\,\text{Var}}{\Gamma \vdash g\ x : \iota_B \to \iota_C}\,\text{App} \quad \dfrac{\dfrac{(f, \iota_A \to \iota_B) \in \Gamma}{\Gamma \vdash f : \iota_A \to \iota_B}\,\text{Var} \quad \dfrac{(x, \iota_A) \in \Gamma}{\Gamma \vdash x : \iota_A}\,\text{Var}}{\Gamma \vdash f\ x : \iota_B}\,\text{App}}{\Gamma = (g, \iota_A \to \iota_B \to \iota_C), (f, \iota_A \to \iota_B), (x, \iota_A) \vdash g\ x\ f\ x : \iota_C}\,\text{App}}{(g, \iota_A \to \iota_B \to \iota_C), (f, \iota_A \to \iota_B) \vdash \lambda x : \iota_A . g\ x\ f\ x : \iota_A \to \iota_C}\,\text{Fun}}{(g, \iota_A \to \iota_B \to \iota_C) \vdash \lambda f : \iota_A \to \iota_B . \lambda x : \iota_A . g\ x\ f\ x : (\iota_A \to \iota_B) \to \iota_A \to \iota_C}\,\text{Fun}}{\vdash \underbrace{\lambda g : \iota_A \to \iota_B \to \iota_C . \lambda f : \iota_A \to \iota_B . \lambda x : \iota_A . g\ x\ f\ x}_{\lambda\text{-terme}} : (\iota_A \to \iota_B \to \iota_C) \to (\iota_A \to \iota_B) \to \iota_A \to \iota_C}\,\text{Fun}$$

Exercice 2 (Ajout de la conjonction)

Dans cet exercice, on va ajouter la conjonction « $\wedge$ »à la logique implicative minimale et voir ce que cela rajoute côté λ-calcul afin de préserver l'isomorphisme d'Howard. Côté règles de preuve, l'ajout de la conjonction rajoute les règles suivantes :

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \wedge_I$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \wedge_{E1} \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B} \wedge_{E2}$$

Dans le λ-calcul simplement typé, afin de capturer ce nouveau connecteur, nous devons modifier le langage des termes et le langage des types. Concernant les termes, nous devons rajouter les couples, ainsi que les projecteurs comme suit :

— Si t_1 et t_2 sont des termes, alors (t_1, t_2) est un terme.

— Si t est un terme, alors $\mathsf{fst}\ t$ et $\mathsf{snd}\ t$ sont des termes.

Côté langage des types, nous devons rajouter le produit cartésien de la manière suivante :

— Si τ_1 et τ_2 sont des types, alors $\tau_1 \times \tau_2$ est un type.

Les règles de typage de ces nouveaux termes sont définies comme suit :

$$\frac{\Gamma \vdash t_1 : \tau_1 \qquad \Gamma \vdash t_2 : \tau_2}{\Gamma \vdash (t_1, t_2) : \tau_1 \times \tau_2}\ \text{Tup}$$

$$\frac{\Gamma \vdash t : \tau_1 \times \tau_2}{\Gamma \vdash \mathsf{fst}\ t : \tau_1}\ \text{Fst} \qquad \frac{\Gamma \vdash t : \tau_1 \times \tau_2}{\Gamma \vdash \mathsf{snd}\ t : \tau_2}\ \text{Snd}$$

Sachant que dans l'isomorphisme de Curry-Howard, nous souhaitons faire correspondre les règles de preuve $\wedge_I$, $\wedge_{E1}$ et $\wedge_{E2}$, aux règles de typage Tup, Fst et Snd respectivement, répondre aux questions suivantes :

1. Écrire les fonctions de correspondance Φ et φ pour ce nouveau connecteur (le « $\wedge$ ») et les nouvelles règles de preuve correspondantes.
2. Donner les λ-termes correspondants aux preuves de $A \Rightarrow B \Rightarrow A \wedge B$ et $A \wedge B \Rightarrow A$.

1. Correspondance des formules/types : $\Phi(A \wedge B) = \Phi(A) \times \Phi(B)$.
   Correspondance des preuves/termes :
   — Si la preuve π est de la forme :

$$\frac{\dfrac{\pi_1}{\Gamma \vdash A} \qquad \dfrac{\pi_2}{\Gamma \vdash B}}{\Gamma \vdash A \wedge B} \wedge_I$$

   alors $\varphi(\pi) = \varphi(\pi_1) \times \varphi(\pi_2)$
   — Si la preuve π est de la forme :

$$\frac{\dfrac{\pi'}{\Gamma \vdash A \wedge B}}{\Gamma \vdash A} \wedge_{E1}$$

   alors $\varphi(\pi) = \mathsf{fst}\ \varphi(\pi')$
   — Si la preuve π est de la forme :

$$\dfrac{\dfrac{\pi'}{\Gamma \vdash A \wedge B}}{\Gamma \vdash B}\wedge_{E2}$$

alors $\varphi(\pi) = \mathsf{snd}\ \varphi(\pi')$

2. Preuve :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\,\text{ax} \quad \dfrac{}{A, B \vdash B}\,\text{ax}}{A, B \vdash A \wedge B}\wedge_I}{A \vdash B \Rightarrow A \wedge B}\Rightarrow_I}{\vdash A \Rightarrow B \Rightarrow A \wedge B}\Rightarrow_I$$

Arbre de typage correspondant :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(x, \iota_A) \in (x, \iota_A), (y, \iota_B)}{(x, \iota_A), (y, \iota_B) \vdash x : \iota_A}\,\text{Var} \quad \dfrac{(y, \iota_B) \in (x, \iota_A), (y, \iota_B)}{(x, \iota_A), (y, \iota_B) \vdash y : \iota_B}\,\text{Var}}{(x, \iota_A), (y, \iota_B) \vdash (x, y) : \iota_A \times \iota_B}\,\text{Tup}}{(x, \iota_A) \vdash \lambda y : \iota_B.(x, y) : \iota_B \to \iota_A \times \iota_B}\,\text{Fun}}{\vdash \underbrace{\lambda x : \iota_A.\lambda y : \iota_B.(x, y)}_{\lambda\text{-terme}} : \iota_A \to \iota_B \to \iota_A \times \iota_B}\,\text{Fun}$$

Preuve :

$$\dfrac{\dfrac{\dfrac{}{A \wedge B \vdash A \wedge B}\,\text{ax}}{A \wedge B \vdash A}\wedge_{E1}}{\vdash A \wedge B \Rightarrow A}\Rightarrow_I$$

Arbre de typage correspondant :

$$\dfrac{\dfrac{\dfrac{\dfrac{(x, \iota_A \times \iota_B) \in (x, \iota_A \times \iota_B)}{(x, \iota_A \times \iota_B) \vdash x : \iota_A \times \iota_B}\,\text{Var}}{(x, \iota_A \times \iota_B) \vdash \mathsf{fst}\ x : \iota_A}\,\text{Fst}}{\vdash \underbrace{\lambda x : \iota_A \times \iota_B.\mathsf{fst}\ x}_{\lambda\text{-terme}} : \iota_A \times \iota_B \to \iota_A}\,\text{Fun}$$

Exercice 3 (Extraction de la fonction factorielle)

L'objectif de cet exercice est de montrer comment il est possible d'extraire des fonctions à partir de preuves en Coq en utilisant l'exemple de la fonction factorielle en particulier.
Pour ce faire, nous allons utiliser le type sig, qui est équivalent au « il existe » mais adapté à l'extraction. La notation pour utiliser ce type est la suivante : `{x : A | P x}`, qui signifie « il existe un x de type A vérifiant $P(x)$ ».

1. Spécifier le comportement de la fonction factorielle en utilisant une relation inductive. Nous appellerons cette relation is_fact et elle prendra deux arguments (l'entier dont on veut calculer la factorielle et le résultat).
2. Démontrer le lemme suivant :

   ```
   Lemma fact : forall n : nat, { v : nat | is_fact n v }.
   ```

   Une fois la preuve terminée, vous devrez sauvegarder la preuve avec la commande Defined (qui permet de préparer l'extraction).
3. Extraire la fonction factorielle de la preuve de lemme fact effectuée précédemment. Pour ce faire, il faut importer la bibliothèque dédiée à l'extraction, puis utiliser la commande d'extraction (par défaut, l'extraction produit du code OCaml) comme suit :

   ```
   Require Extraction.
   Extraction fact.
   ```

   Cette commande n'extraie que la fonction correspondant au lemme fact. Pour extraire toutes les dépendances (notamment le type des entiers naturels), il faut utiliser la commande suivante :

   ```
   Recursive Extraction fact.
   ```

   Utiliser cette commande et exécuter le code produit dans un toplevel OCaml sur plusieurs exemples (attention, les entiers utilisés correspondent aux entiers extraits de Coq, à savoir les entiers de Peano).

▶ Voir TP6.

Exercice 4 (Extraction de l'égalité sur les entiers naturels)

Cet exercice est similaire au précédent. Nous nous proposons d'extraire une fonction qui teste l'égalité entre deux entiers naturels.
Pour ce faire, nous allons utiliser le type sumbool (que nous avons déjà vu au TD/TP précédent), qui est équivalent au « ou »mais adapté à l'extraction. On rappelle que la notation pour utiliser ce type est la suivante : `{A} + {B}`, qui signifie « A ou B ».

1. Démontrer le lemme suivant :

   ```
   Lemma eq_nat : forall n m : nat, {n = m} + {n <> m}.
   ```

   Pour faire cette preuve, on doit faire une induction structurelle sur n et m. Cette double induction peut se faire soit en enchaînant les inductions, soit en utilisant la tactique (double induction n m). À notre également qu'il existe une tactique en Coq qui démontre les lemmes de décidabilité de l'égalité automatiquement, il s'agit de la tactique `decide equality` (vous pouvez la tester sur notre exemple). Comme dans l'exercice précédent, une fois la preuve terminée, vous devrez sauvegarder la preuve avec la commande `Defined` afin de préparer l'extraction.
2. Extraire la fonction d'égalité sur les entiers naturels et la tester sur plusieurs exemples en OCaml. Comme la fonction en Coq retourne un objet de type sumbool (et non un booléen), iul est à noter que la fonction OCaml retournera des objets de ce type extrait, à savoir les valeurs Left (correspondant à true) et Right (correspondant à false).

▶ Voir TP6.

Exercice 5 (Extraction de l'inversion d'une liste)

Cet exercice est similaire aux deux exercices précédents. Nous nous proposons d'extraire une fonction qui inverse l'ordre des éléments d'une liste d'entiers naturels.

1. Spécifier le comportement de la fonction qui inverse l'ordre des éléments d'une liste d'entiers naturels en utilisant une relation inductive. Nous appellerons cette relation `is\_rev` et elle prendra deux arguments (la liste d'entiers naturels dont on veut inverser l'ordre des éléments et le résultat).
2. Démontrer le lemme suivant :

   ```
   Lemma rev : forall l : list nat, {l' : list nat | is_rev l l' }.
   ```

   Comme dans les deux exercices précédents, une fois la preuve terminée, vous devrez sauvegarder la preuve avec la commande `Defined` afin de préparer l'extraction.
3. Extraire la fonction qui inverse l'ordre des éléments d'une liste d'entiers naturels et la tester sur plusieurs exemples en OCaml.

▶ Voir TP6.

7 TD/TP7 : Preuve de programmes impératifs

Exercice 1 (Sémantique)

Évaluer les programmes suivants en utilisant les règles d'évaluation vues en cours :
- if $x \geq 0$ then $y := x$ else $y := -x$ dans les environnements $(x, 2), (y, 0)$ et $(x, -2), (y, 0)$.
- while $i < 3$ do $(x := x + i; i := i + 1)$ dans l'environnement $(i, 1), (x, 0)$.

1. Évaluation dans l'environnement $(x, 2), (y, 0)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{(x,2) \in E}{E \vdash x \rightsquigarrow 2}\,\mathbb{V} \quad \dfrac{0 \in \mathbb{Z}}{E \vdash 0 \rightsquigarrow 0}\,\mathbb{Z}}{E \vdash x \geq 0 \rightsquigarrow 2 \geq_{\mathbb{Z}} 0 = \top}\,\geq \quad \dfrac{y \in dom(E) \quad \dfrac{(x,2) \in E}{E \vdash x \rightsquigarrow 2}\,\mathbb{V}}{E \vdash y := x \rightsquigarrow E \leftarrow (y, 2)}\,:=}{E = (x,2),(y,0) \vdash \text{if } x \geq 0 \text{ then } y := x \text{ else } y := -x \rightsquigarrow (x,2),(y,2)}\,\text{if}_{\top}$$

Évaluation dans l'environnement $(x, -2), (y, 0)$ ($-_{\mathbb{1}}$ est l'opérateur du moins unaire) :

$$\dfrac{\dfrac{\dfrac{\dfrac{(x,2) \in E}{E \vdash x \rightsquigarrow -2}\,\mathbb{V} \quad \dfrac{0 \in \mathbb{Z}}{E \vdash 0 \rightsquigarrow 0}\,\mathbb{Z}}{E \vdash x \geq 0 \rightsquigarrow -2 \geq_{\mathbb{Z}} 0 = \bot}\,\geq \quad \dfrac{y \in dom(E) \quad \dfrac{\dfrac{(x,2) \in E}{E \vdash x \rightsquigarrow -2}\,\mathbb{V}}{E \vdash -x \rightsquigarrow 2}\,-_{\mathbb{1}}}{E \vdash y := -x \rightsquigarrow E \leftarrow (y, 2)}\,:=}{E = (x,-2),(y,0) \vdash \text{if } x \geq 0 \text{ then } y := x \text{ else } y := -x \rightsquigarrow (x,2),(y,2)}\,\text{if}_{\bot}$$

2. Évaluation dans l'environnement $(i, 1), (x, 0)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{(i,1) \in E}{E \vdash i \rightsquigarrow 1}\,\mathbb{V} \quad \dfrac{3 \in \mathbb{Z}}{E \vdash 3 \rightsquigarrow 3}\,\mathbb{Z}}{E \vdash i < 3 \rightsquigarrow 1 <_{\mathbb{Z}} 3 = \top}\,< \quad \dfrac{\dfrac{x \in dom(E) \quad \dfrac{\dfrac{(x,0) \in E}{E \vdash x \rightsquigarrow 0}\,\mathbb{V} \quad \dfrac{(i,1) \in E}{E \vdash i \rightsquigarrow 1}\,\mathbb{V}}{E \vdash x + i \rightsquigarrow 0 +_{\mathbb{Z}} 1}\,+}{E \vdash x := x + i \rightsquigarrow E_1 = E \leftarrow (x, 1)}\,:= \quad \dfrac{i \in dom(E_1) \quad \dfrac{\dfrac{(i,1) \in E_1}{E_1 \vdash i \rightsquigarrow 1}\,\mathbb{V} \quad \dfrac{1 \in \mathbb{Z}}{E_1 \vdash 1 \rightsquigarrow 1}\,\mathbb{Z}}{E_1 \vdash i + 1 \rightsquigarrow 1 +_{\mathbb{Z}} 1}\,+}{E_1 \vdash i := i + 1 \rightsquigarrow E_2 = E_1 \leftarrow (i, 2)}\,:=}{E \vdash x := x + i; i := i + 1 \rightsquigarrow E_2}\,; \quad \Pi_1}{E = (i,1),(x,0) \vdash \text{while } i < 3 \text{ do } (x := x + i; i := i + 1) \rightsquigarrow (i,3),(x,3)}\,\text{while}_{\top}$$

Preuve de Π_1 :

$$\dfrac{\dfrac{\dfrac{\dfrac{(i,2) \in E_2}{E_2 \vdash i \rightsquigarrow 2}\,\mathbb{V} \quad \dfrac{3 \in \mathbb{Z}}{E_2 \vdash 3 \rightsquigarrow 3}\,\mathbb{Z}}{E_2 \vdash i < 3 \rightsquigarrow 2 <_{\mathbb{Z}} 3 = \top}\,< \quad \dfrac{\dfrac{x \in dom(E_2) \quad \dfrac{\dfrac{(x,1) \in E_2}{E_2 \vdash x \rightsquigarrow 1}\,\mathbb{V} \quad \dfrac{(i,2) \in E_2}{E_2 \vdash i \rightsquigarrow 2}\,\mathbb{V}}{E_2 \vdash x + i \rightsquigarrow 1 +_{\mathbb{Z}} 2}\,+}{E_2 \vdash x := x + i \rightsquigarrow E_3 = E_2 \leftarrow (x, 3)}\,:= \quad \dfrac{i \in dom(E_3) \quad \dfrac{\dfrac{(i,2) \in E_3}{E_3 \vdash i \rightsquigarrow 2}\,\mathbb{V} \quad \dfrac{2 \in \mathbb{Z}}{E_3 \vdash 1 \rightsquigarrow 1}\,\mathbb{Z}}{E_3 \vdash i + 1 \rightsquigarrow 2 +_{\mathbb{Z}} 1}\,+}{E_3 \vdash i := i + 1 \rightsquigarrow E_4 = E_3 \leftarrow (i, 3)}\,:=}{E_2 \vdash x := x + i; i := i + 1 \rightsquigarrow E_4}\,; \quad \Pi_2}{\Pi_1 = E_2 \vdash \text{while } i < 3 \text{ do } (x := x + i; i := i + 1) \rightsquigarrow (i,3),(x,3)}\,\text{while}_{\top}$$

Preuve de Π_2 :

$$\dfrac{\dfrac{\dfrac{(i,3) \in dom(E_4)}{E_4 \vdash i \rightsquigarrow 3}\,\mathbb{V} \quad \dfrac{3 \in \mathbb{Z}}{E_4 \vdash 3 \rightsquigarrow 3}\,\mathbb{Z}}{E_4 \vdash i < 3 \rightsquigarrow 3 <_{\mathbb{Z}} 3 = \bot}\,<}{\Pi_2 = E_4 \vdash \text{while } i < 3 \text{ do } (x := x + i; i := i + 1) \rightsquigarrow E_4}\,\text{while}_{\bot}$$

Exercice 2 (Logique de Hoare)

Démontrer la validité des triplets de Hoare suivants :

1. $\{x = 0\}\ x := x + 1; x := x + 1\ \{x = 2\}$;
2. $\{x = 1 \wedge y = 2\}\ t := x; x := y; y := t\ \{x = 2 \wedge y = 1\}$;
3. $\{x \geq 0\}$ if $x \geq 0$ then $y := 1$ else $y := 2\ \{y = 1\}$;
4. $\{x \geq 0\}$ if $x\ != 0$ then $x := x - 1$ else $x := x + 1\ \{x \geq 0\}$;
5. $\{\}$ while $x\ != 0$ do $x := x - 1\ \{x = 0\}$.

1. Preuve :

$$\dfrac{\dfrac{}{\{x+1=1\}\ x := x+1\ \{x=1\}}:= \quad x = 0 \Rightarrow x+1=1}{\{x=0\}\ x := x+1\ \{x=1\}}\ \text{Aff} \qquad \dfrac{\dfrac{}{\{x+1=2\}\ x := x+1\ \{x=2\}}:= \quad x = 1 \Rightarrow x+1=2}{\{x=1\}\ x := x+1\ \{x=2\}}\ \text{Aff}}{\{x=0\}\ x := x+1; x := x+1\ \{x=2\}}\ ;$$

2. Preuve :

$$\dfrac{\dfrac{}{\{x=1\wedge y=2\}\ t := x\ \{x=1\wedge y=2\wedge t=1\}}:= \qquad \dfrac{\dfrac{}{\{x=1\wedge y=2\wedge t=1\}\ x := y\ \{x=2\wedge y=2\wedge t=1\}}:= \quad \dfrac{}{\Pi_1}:=}{\{x=1\wedge y=2\wedge t=1\}\ x := y; y := t\ \{x=2\wedge y=1\}}\ ;}{\{x=1\wedge y=2\}\ t := x; x := y; y := t\ \{x=2\wedge y=1\}}\ ;$$

Avec $\Pi_1 = \{x = 2 \wedge y = 2 \wedge t = 1\}\ y := t\ \{x = 2 \wedge y = 1\}$.

3. Preuve :

$$\dfrac{\dfrac{\dfrac{}{\{1=1\}\ y := 1\ \{y=1\}}:= \quad x \geq 0 \Rightarrow 1=1}{\{x\geq 0\}\ y := 1\ \{y=1\}}\ \text{Aff} \qquad \dfrac{\dfrac{}{\{2=1\}\ y := 2\ \{y=1\}}:= \quad x \geq 0 \wedge x < 0 \Rightarrow 2=1}{\{x\geq 0\wedge x<0\}\ y := 2\ \{y=1\}}\ \text{Aff}}{\{x\geq 0\}\ \text{if } x\geq 0 \text{ then } y := 1 \text{ else } y := 2\ \{y=1\}}\ \text{if}$$

4. Preuve :

$$\dfrac{\dfrac{\dfrac{}{\{x-1\geq 0\}\ x := x-1\ \{x\geq 0\}}:= \quad x\geq 0\wedge x\neq 0 \Rightarrow x-1\geq 0}{\{x\geq 0\wedge x\neq 0\}\ x := x-1\ \{x\geq 0\}}\ \text{Aff} \qquad \dfrac{\dfrac{}{\{x+1=1\}\ x := x+1\ \{x\geq 0\}}:= \quad x\geq 0\wedge x=0 \Rightarrow x+1=1}{\{x\geq 0\wedge x=0\}\ x := x+1\ \{x\geq 0\}}\ \text{Aff}}{\{x\geq 0\}\ \text{if } x\ != 0 \text{ then } x := x-1 \text{ else } x := x+1\ \{x\geq 0\}}\ \text{if}$$

5. Preuve :

$$\dfrac{\dfrac{\dfrac{}{\{x-1=x-1\}\ x := x-1\ \{x=x-1\}}:= \quad x\neq 0 \Rightarrow x-1=x-1 \qquad x=x-1\Rightarrow\top}{\{x\neq 0\}\ x := x-1\ \{\}}}{\{\}\ \text{while } x\ != 0 \text{ do } x := x-1\ \{x=0\}}\ \text{while}$$

Exercice 3 (Logique de Hoare)

Démontrer que le programme suivant implémente la fonction factorielle :

```
{}
i := 0;
r := 1;
while i != n do
    i := i + 1;
    r := r * i;
{ r = n! }
```

Preuve :

$$\dfrac{\dfrac{}{\{0=0\}\ i := 0\ \{i=0\}}:= \qquad \top \Rightarrow 0=0}{\{\}\ i := 0\ \{i=0\}}\ \text{Aff} \qquad \dfrac{\dfrac{\dfrac{}{\{1=1\}\ r := 1\ \{i=0 \wedge r=1\}}:= \qquad i=0 \Rightarrow 1=1 \qquad i=0 \wedge r=1 \Rightarrow r=i!}{\{i=0\}\ r := 1\ \{r=i!\}}\ \text{Aff} \qquad \Pi_1}{\{i=0\}\ r := 1;\ \text{while } i \neq n \text{ do } (i := i+1; r := r \times i)\ \{r=n!\}}\ ;}{\{\}\ i := 0; r := 1;\ \text{while } i \neq n \text{ do } (i := i+1; r := r \times i)\ \{r=n!\}}\ ;$$

Preuve de Π_1 :

$$\dfrac{\dfrac{\Pi_2 \qquad \dfrac{}{\{r \times i = i!\}\ r := r \times i\ \{r=i!\}}:= \qquad r=(i-1)! \Rightarrow r \times i = i!}{\{r=(i-1)!\}\ r := r \times i\ \{r=i!\}}\ \text{Aff}}{\{r=i! \wedge i \neq n\}\ i := i+1; r := r \times i\ \{r=i!\}}\ ; \qquad r=i! \wedge i=n \Rightarrow r=n!}{\{r=i!\}\ \text{while } i \neq n \text{ do } (i := i+1; r := r \times i)\ \{r=i! \wedge i=n\}}\ \text{while}}{\{r=i!\}\ \text{while } i \neq n \text{ do } (i := i+1; r := r \times i)\ \{r=n!\}}\ \text{Aff}$$

Preuve de Π_2 :

$$\dfrac{\dfrac{}{\{i+1=k+1 \wedge r=k!\}\ i := i+1\ \{i=k+1 \wedge r=k!\}}:= \qquad \begin{array}{c} r=i! \wedge i \neq n \Rightarrow i+1=k+1 \wedge r=k! \\ i=k+1 \wedge r=k! \Rightarrow r=(i-1)! \end{array}}{\{r=i! \wedge i \neq n\}\ i := i+1\ \{r=(i-1)!\}}\ \text{Aff}$$