Feuille 1 : Logique propositionnelle

Exercice 1 : Négation

Exprimez les négations des propositions suivantes sans les faire précéder de : *il est faux que*. Pour ce faire, on traduira ces propositions, exprimées en langage naturel, en formules de la logique propositionnelle, après avoir déterminé les variables propositionnelles nécessaires.

Q 1.1 Ce quadrilatère n'est ni un losange, ni un rectangle.

Solution. Énoncé de la forme $\neg p \wedge \neg q$ avec
- $p =$ ce quadrilatère est un losange,
- et $q =$ ce quadrilatère est un rectangle.

La négation est $p \vee q$ d'où

Ce quadrilatère est un losange ou un rectangle.

Q 1.2 Lentier 522 n'est pas divisible par 3, mais il est divisible par 7.

Solution. Énoncé de la forme $\neg p \wedge q$ avec
- $p =$ l'entier 522 est divisible par 3,
- et $q =$ l'entier 522 est divisible par 7.

La négation est $p \vee \neg q$ d'où

L'entier 522 est divisible par 3 ou il n'est pas divisible par 7.

Q 1.3 S'il pleut demain ou s'il fait froid, je ne sortirai pas.

Solution. Énoncé de la forme $(p_1 \vee p_2) \Rightarrow \neg q$ avec
- $p_1 =$ il pleut demain,
- $p_2 =$ il fait froid demain,
- et $q =$ je sors demain.

Comme $(p_1 \vee p_2) \Rightarrow \neg q \equiv \neg(p_1 \vee p_2) \vee \neg q$, la négation est $(p_1 \vee p_2) \wedge q$, d'où

Il pleut demain ou il fait froid, mais je sors.

Exercice 2 : Associativité

Montrer que le connecteur d'implication $\Rightarrow$ n'est pas associatif.

Solution. Il suffit de considérer la valuation v dans laquelle toute variable est évaluée en 0 (faux). On a alors
- $[\![((p \Rightarrow q) \Rightarrow r)]\!]_v = 0$,
- et $[\![(p \Rightarrow (q \Rightarrow r))]\!]_v = 1$.

On en déduit que

$$((p \Rightarrow q) \Rightarrow r) \not\equiv (p \Rightarrow (q \Rightarrow r)).$$

Exercice 3 : Axiomes de $\equiv$

Q 3.1 Démontrer la distributivité de $\vee$ par rapport à $\wedge$, i.e. :

$$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$$

Q 3.2 Démontrer les lois de de Morgan

$$\neg(a \wedge b) \equiv \neg a \vee \neg b \text{ et } \neg(a \vee b) \equiv \neg a \wedge \neg b$$

Solution. Vérifier $a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$ revient à vérifier que pour toute valuation les formules $a \vee (b \wedge c)$, $(a \vee b) \wedge (a \vee c)$ prennent la même valeur. Cela revient également à vérifier que la formule $a \vee (b \wedge c) \Leftrightarrow (a \vee b) \wedge (a \vee c)$ prend la valeur 1 pour toute valuation. Pour les deux méthodes, on construit une table de vérités :

a	b	c	$b \wedge c$	$a \vee (b \wedge c)$	$a \vee b$	$a \vee c$	$(a \vee b) \wedge (a \vee c)$	$a \vee (b \wedge c) \Leftrightarrow (a \vee b) \wedge (a \vee c)$
0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	0	1
0	1	0	0	0	1	0	0	1
0	1	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1	1
1	0	1	0	1	1	1	1	1
1	1	0	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1

Pour les lois de de Morgan, on procède de même :

a	b	$\neg a$	$\neg b$	$a \vee b$	$\neg(a \vee b)$	$\neg a \wedge \neg b$	$a \wedge b$	$\neg(a \wedge b)$	$\neg a \vee \neg b$
0	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	0	0	1	1
1	0	0	1	1	0	0	0	1	1
1	1	0	0	1	0	0	1	0	0

On constate que les colonnes des formules $\neg(a \vee b)$ et $\neg a \wedge \neg b$ sont bien les mêmes ce qui implique que l'on a bien $\neg(a \vee b) \equiv \neg a \wedge \neg b$. De même on obtient l'identité de $\neg(a \wedge b) \equiv \neg a \vee \neg b$

Exercice 4 : Régime alimentaire

Alice mange la viande, le poisson et les légumes, mais pas les féculents.
Nous modélisons les plats à l'aide des quatre variables propositionnelles suivantes :
— v le plat contient de la viande,
— p le plat contient du poisson,
— l le plat contient des légumes,
— f le plat contient des féculents.

Q 4.1 Donner une formule utilisant ces variables qui permette de représenter les plats qu'Alice peut manger.

Q 4.2 Donner une valuation qui représente le poisson à la bordelaise qui contient est plat contenant du poisson et de légumes.

Q 4.3 Vérifier si Alice peut manger ce plat.

Solution. L'exercie a pour but de montrer qu'il faut ce méfier de l'équivalence intuitive de *et* et de $\wedge$ (il y a également des différences entre *ou* et $\vee$).
La formule demandée est $(v \vee p \vee l) \wedge \neg f$.
La valuation pour le poisson à la bordelaise est $v = 0$, $p = 1$, $l = 1$, $f = 0$.
La formule s'évalue alors à 1. Alice peut manger le poisson à la bordelaise.

Exercice 5 : Tombola

Vous décidez dacheter un billet de tombola. Le buraliste vous en présente cinq, de 1 à 5, et vous déclare :

(a) Si 5 est perdant, 1 est gagnant ;
(b) Si 4 est perdant, 2 est gagnant ;
(c) Si 3 est perdant, 5 aussi ;
(d) Si 1 est gagnant, 2 aussi ;
(e) Si 3 est gagnant, 4 est perdant.

Traduisez ces informations en formules de la logique propositionnelle. Peut-on en déduire le billet gagnant ?

Solution. Pour chacun des numéros $i = 1, \ldots, 5$ des billets, notons g_i la proposition affirmant que le billet n° i est gagnant.

(a) $\neg g_5 \Rightarrow g_1$
(b) $\neg g_4 \Rightarrow g_2$
(c) $\neg g_3 \Rightarrow \neg g_5$
(d) $g_1 \Rightarrow g_2$
(e) $g_3 \Rightarrow \neg g_4$

En considérant vraies les affirmations du buraliste, on peut envisager les deux situations du billet 3.

1. Si dans une valuation v on a $[\![g_3]\!]_v = 0$, alors on a aussi $[\![g_5]\!]_v = 0$ (affirmation (c)), et par conséquent $[\![g_1]\!]_v = 1$ (affirmation (a)) et donc $[\![g_2]\!]_v = 1$ (affirmation (d)).
2. Si dans une valuation v on a au contraire $[\![g_3]\!]_v = 1$, alors on a aussi $[\![g_4]\!]_v = 0$ (affirmation (e)) et donc $[\![g_2]\!]_v = 1$ (affirmation (b)).

Dans toute valuation v satisfaisant les cinq affirmations du buraliste, on a $[\![g_2]\!]_v = 1$, donc

$\boxed{\text{il faut choisir le billet n° 2.}}$

Exercice 6 : Déduction

On sait que :

1. Pierre joue au golf ou fait de l'alpinisme ou pratique la plongée.
2. Si Pierre ne joue pas au golf ou ne fait pas de plongée, il fait de l'alpinisme.
3. Si Pierre fait de la plongée, il ne joue pas au golf.
4. Si Pierre fait de l'alpinisme, alors il fait aussi de la plongée.

Quel(s) sport(s) pratique Pierre ?

Solution. Soient a, g et p les trois variables propositionnelles
— a = Pierre fait de l'alpinisme,
— g = Pierre joue au golf,
— et p = Pierre pratique la plongée.

La connaissance sur Pierre se traduit par

1. $a \vee g \vee p$
2. $(\neg g \vee \neg p) \Rightarrow a$
3. $p \Rightarrow \neg g$
4. $a \Rightarrow p$.

On établit la table de vérité de chacune de ces formules et on cherche pour quelle valuation les quatre formules sont vraies.

a	g	p	$a \vee g \vee p$	$(\neg g \vee \neg p) \Rightarrow a$	$p \Rightarrow \neg g$	$a \Rightarrow p$
0	0	0	0	0	1	1
0	0	1	1	0	1	1
0	1	0	1	0	1	1
0	1	1	1	1	0	1
1	0	0	1	1	1	0
$\boxed{1}$	$\boxed{0}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$	$\boxed{1}$
1	1	0	1	1	1	0
1	1	1	1	1	0	1

$\boxed{\text{Pierre pratique la plongée et l'alpinisme.}}$

Exercice 7 : Logique propositionnelle : simplification de programme

Le programme C suivant a été allégé en en éliminant le code. On cherche ici à savoir quelles propriétés sont vraies à un certain point du programme.

```
kwhile p((o!na o|| nbp) o ncp)
  p...
 p
p(l+m+mi1p)
kifp(na o|| nbp)
  p(l+m+mi2p) p...
 p
 kelsep
   kifp(nc o nbp)
     p(l+m+mi3p)...
   p
   kelsep
     p(l+m+mi4p)...
   p
 p
```

Les nombres entre parenthèse – (1), (2), (3), et (4) – servent à désigner des points du code. Les variables a, b, et c sont des variables à valeurs booléennes.

Q 7.1 Donner pour chaque point du code (i) une formule de la logique propositionnelle φ_i utilisant les variables a, b et c qui exprime la condition qui doit être vérifiée à ce point du code.

Solution. Afin d'alléger les notations, nous utiliserons les noms suivants :

$$\theta_0 = \neg a \vee b,\ \theta_1 = \theta_0 \wedge c,\ \theta_2 = a \vee b,\ \theta_3 = c \wedge b$$

Pour atteindre le point (1), il faut être sorti de la boucle `while`, c'est-à-dire que la propriété $(\neg a \vee b) \wedge c$ doit être fausse. Ainsi la propriété vérifiée au point (1) est $\varphi_1 = \neg\theta_1$.
Pour atteindre le point (2), il faut avoir atteint le point (1) et que la condition $a \vee b$ soit vraie. Ainsi $\varphi_2 = \varphi_1 \wedge \theta_2$.
Pour atteindre le point (3), il faut avoir atteint le point (1), que la condition $a \vee b$ soit fausse et que la condition $c \wedge b$ soit vraie. Nous avons donc $\varphi_3 = \varphi_1 \wedge \neg\theta_2 \wedge \theta_3$.
Finalement pour atteindre le point (4), il faut avoir atteint le point (1) et que les conditions $a \vee b$, et $c \wedge b$ soient fausses. Nous avons donc $\varphi_4 = \varphi_1 \wedge \neg\theta_2 \wedge \neg\theta_3$.

Q 7.2 Comment déterminer si une partie de code ne peut pas être exécutée ?

Solution. En chaque point (i) le code ne peut pas être exécuté si la formule φ_i est contradictoire, i.e. n'est pas satisfiable.

Q 7.3 Déterminer quelles parties de code ne peuvent pas être exécutées.

Solution. Afin de déterminer quand les formules φ_i sont satisfiables, nous allons procéder en utilisant une table de vérité.

				θ_0	θ_1	φ_1	θ_2	φ_2			θ_3	φ_3		φ_4
a	b	c	$\neg a$	$\neg a \vee b$	$\theta_0 \wedge c$	$\neg\theta_1$	$a \vee b$	$\varphi_1 \wedge \theta_2$	$\neg\theta_2$	$\varphi_1 \wedge \neg\theta_2$	$c \wedge b$	$\varphi_1 \wedge \neg\theta_2 \wedge \theta_3$	$\neg\theta_3$	$\varphi_1 \wedge \neg\theta_2 \wedge \neg\theta_3$
0	0	0	1	1	0	1	0	0	1	1	0	0	1	1
0	0	1	1	1	1	0	0	0	1	0	0	0	1	0
0	1	0	1	1	0	1	1	1	0	0	0	0	1	0
0	1	1	1	1	1	0	1	0	0	0	1	0	0	0
1	0	0	0	0	0	1	1	1	0	0	0	0	1	0
1	0	1	0	0	0	1	1	1	0	0	0	0	1	0
1	1	0	0	1	0	1	1	1	0	0	0	0	1	0
1	1	1	0	1	1	0	1	0	0	0	1	0	0	0

Parmi les formules φ_1, φ_2, φ_3, φ_4 seule la formule φ_3 est toujours fausse. Ainsi le point (3) du code ne peut jamais être éxecuté.

Q 7.4 Simplifier le programme en un programme équivalent.

Solution.
Le programme peut ainsi être simplifié de la façon suivante :

```
kwhile p((o!na o|| nbp) o ncp)
  p...
 p
p(1+m+mi1p)
kifp(na o|| nbp)
  p(1+m+mi2p) p...
    p
 kelsep
   p(1+m+mi4p)...
     p
```

Exercice 8 : Principe d'induction

Dans cet exercice, nous considérons une formule φ construite à partir des variables $x_1, \ldots, x_n$. Étant données des formules $\psi_1, \ldots, \psi_n$ nous notons $\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]$ la formule obtenue à partir de φ en remplaçant les occurrences de x_i par ψ_i pour tout i dans $[1, n]$.

Q 8.1 Étant donnée une valuation ν, démontrer, en utilisant le principe d'induction, que $[\![\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi]\!]_\mu$ où μ est la valuation telle que $\mu(x_i) = [\![\psi_i]\!]_\nu$ pour tout i dans $[1, n]$.

Q 8.2 Supposons maintenant que nous disposons de formules $\theta_1, \ldots, \theta_n$ telles que pour tout i dans $[1, n]$, $\psi_i \equiv \theta_i$. Déduire de la question précédente que pour toutes formules φ_1 et φ_2 construites à partir de variables propositionnelles $x_1, \ldots, x_n$ si $\varphi_1 \equiv \varphi_2$ alors, $\varphi_1[\psi_1/x_1, \ldots, \psi_n/x_n] \equiv \varphi_2[\theta_1/x_1, \ldots, \theta_n/x_n]$.

NB :

— pour rappel, $\varphi \equiv \psi$ signifie que pour toute valuation ν $[\![\varphi]\!]_\nu = [\![\psi]\!]_\nu$,

— cet exercice démontre que $\equiv$ est une *congruence* sur les formules.

Solution. Nous démontrons par induction sur φ que $[\![\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi]\!]_\mu$.

Si φ est égale à $\bot$, alors $\varphi[\psi_1/x_1, \ldots, \psi_n/x_n] = \bot = \varphi[\psi_1/x_1, \ldots, \psi_n/x_n]$ et ainsi, avons bien que dans ce cas $[\![\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = 0 = [\![\varphi]\!]_\mu$.

Supposons maintenant que φ soit égale à la variable propositionnelle x_i. Dans ce cas, $\varphi[\psi_1/x_1, \ldots, \psi_n/x_n] = \psi_i$ et $[\![\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\psi_i]\!]_\nu = [\![x_i]\!]_\mu = [\![\varphi]\!]_\mu$. Nous avons bien que $[\![\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi]\!]_\mu$.

Dans le cas où φ est de la forme $\varphi_1 \operatorname{op} \varphi_2$ avec op dans $\{\wedge, \vee, \Rightarrow, \Leftrightarrow\}$, nous savons par hypothèse d'induction que $[\![\varphi_1[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi_1]\!]_\mu$ et que $[\![\varphi_2[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi_2]\!]_\mu$. Par conséquent quelque soit op, nous obtenons que

$$[\![\varphi_1[\psi_1/x_1, \ldots, \psi_n/x_n] \operatorname{op} \varphi_2[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi_1 \operatorname{op} \varphi_2]\!]_\mu$$

autrement dit que $[\![\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi]\!]_\mu$.

Les cas où φ est de la forme (φ') ou $\neg\varphi'$ se traitent de façon similaire.

En conclusion, pour toute formule φ nous avons bien, par principe d'induction, que $[\![\varphi[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi]\!]_\mu$.

Nous voulons maintenant démontrer que $\varphi_1[\psi_1/x_1, \ldots, \psi_n/x_n] \equiv \varphi_2[\theta_1/x_1, \ldots, \theta_n/x_n]$. Pour cela, nous fixons une valuation ν et d'après ce qui précède, nous savons que $[\![\varphi_1[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi_1]\!]_{\mu_1}$ et $[\![\varphi_2[\theta_1/x_1, \ldots, \theta_n/x_n]]\!]_\nu = [\![\varphi_2]\!]_{\mu_2}$ lorsque l'on pose que :

— $\mu_1(x_i) = [\![\psi_i]\!]_\nu$ pour tout i dans $[1, n]$,

— $\mu_2(x_i) = [\![\theta_i]\!]_\nu$ pour tout i dans $[1, n]$,

Or par hypothèse, pour tout i dans $[1, n]$, $\psi_i \equiv \theta_i$ et ainsi $\mu_1(x_i) = [\![\psi_i]\!]_\nu = [\![\theta_i]\!]_\nu = \mu_2(x_i)$. Ceci implique que $\mu_1 = \mu_2$. Or par hypothèse, $\varphi_1 \equiv \varphi_2$ et donc $[\![\varphi_1]\!]_{\mu_1} = [\![\varphi_2]\!]_{\mu_1} = [\![\varphi_2]\!]_{\mu_2}$.

En conclusion nous obtenons l'identité suivante :

$$[\![\varphi_1[\psi_1/x_1, \ldots, \psi_n/x_n]]\!]_\nu = [\![\varphi_1]\!]_{\mu_1} = [\![\varphi_2]\!]_{\mu_2} = [\![\varphi_2[\theta_1/x_1, \ldots, \theta_n/x_n]]\!]_\nu$$

et ainsi comme ν est arbitraire, nous avons finalement montré que $\varphi_1[\psi_1/x_1, \ldots, \psi_n/x_n] \equiv \varphi_2[\theta_1/x_1, \ldots, \theta_n/x_n]$.

NB : il faut noter que l'identité démontrée est intéressante parce qu'elle montre que l'équivalence de formule (la relation $\equiv$) commute avec la substitution. Il s'agit ainsi d'une congruence. Il peut être intéressant de filer la comparaison avec la spécification de programmes. En effet, si celle-ci est suffisamment précise, on peut remplacer tout programme par un autre programme la réalisant, au même titre que pour les formules, on peut remplacer une formule réalisant une fonction booléenne par n'importe quelle autre réalisant la même fonction.