Cours MOdélisation, Vérification et EXpérimentations
Exercices (avec les corrections)
Modélisation et vérification d'algorithmes en PlusCal
par Dominique Méry
12 février 2025

TD5 Les solutions PlusCal sont dans le dossier associé mais sont numérotées avec les préfixe appex4 avec un numéro qui correspond aux numéros des exercices.

TD5 Les solutions PlusCal sont dans le dossier associé mais sont numérotées avec les préfixe appex4 avec un numéro qui correspond aux numéros des exercices.

Exercice 1 ☑

Nous allons utiliser la fonctionnalité de traduction d'un algorithme PlusCal en un module TLA pour vérifier des algorithmes. Pour chaque question, on écrira un module TLA contenant une expression de l'algorithme puis on traduira par un module et ensuite on analysera le module obtenu par rapport à la correction partielle et l'absence d'erreurs à l'exécution. Cet exercice ressemble aux exercices précédents mais se focalise sur le langage algorithmique PlusCal qui est traduit automatiquement comme cela a été fait manuellement.

Question 1.1 *(appex4_1_1)*
Soit l'annotation suivante :

$\ell_1 : x = 10 \ \wedge \ y = z{+}x \ \wedge z = 2{\cdot}x$
$y := z{+}x$
$\ell_2 : x = 10 \ \wedge \ y = x{+}2{\cdot}10$

Traduire en PlusCal.

◇— **Solution de la question 1.1**

MODULE $tp4$

```
EXTENDS TLC, Integers, Naturals

(*
--algorithm ex1 {
variables x, y, z;
{
init : x := 10; z := 2*x; y := z+x;
l1 : y := z+x;
print <<x, y, z>>;

}
}
*)

i ≜
    ∧ pc = "l1" ⇒ x = 10 ∧ y = z+x ∧ z = 2*x
    ∧ pc = "Done" ⇒ x = 10 ∧ y = x+2*10
```

Fin 1.1

Question 1.2 *(appex4_1_2)*

On suppose que p est un nombre premier :

$\ell_1 : x = 2^p \ \wedge\ y = 2^{p+1} \ \wedge\ x{\cdot}y = 2^{2 \cdot p+1}$
$x := y{+}x{+}2^x$
$\ell_2 : x = 5{\cdot}2^p \ \wedge\ y = 2^{p+1}$

Question 1.3 *(appex4_1_3)*

$\ell_1 : x = 1 \ \wedge\ y = 12$
$x := 2{\cdot}y$
$\ell_2 : x = 1 \ \wedge\ y = 24$

Question 1.4 *(appex4_1_4)*

$\ell_1 : x = 11 \ \wedge\ y = 13$
$z := x; x := y; y := z;$
$\ell_2 : x = 26/2 \ \wedge\ y = 33/3$

Exercice 2 *(pluscal_max.tla)*
On considère l'algorithme correspondant au calcul du maximum de deux nombres.

Variables : X,Y,Z
Requires : $x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z}$
Ensures : $z_f = max(x_0, y_0)$

$\ell_0 : \{x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z}\}$
if $X < Y$ **then**
  $\ell_1 : \{x < y \wedge x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z}\}$
  $Z := Y;$
  $\ell_2 : \{x < y \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z} \wedge z = y_0\}$
else
  $\ell_3 : \{x \ geqy \wedge x = x_0 \wedge y = y_0 \wedge z = z_0 \wedge x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z}\}$
  $Z := X;$
  $\ell_4 : \{x \geq y \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z} \wedge z = x_0\}$
;
$\ell_5 : \{z = max(x_0, y_0) \wedge x = x_0 \wedge y = y_0 \wedge x_0, y_0 \in \mathbb{N} \ \wedge z_0 \in \mathbb{Z}\}$

Algorithme 1: maximum de deux nombres non annotée

Question 2.1 *Traduire cet algorithme annoté en un algorithme PlusCal.*

Question 2.2 *Montre que cette annotation est correcte pour des valeurs choisies de a et b.*

Question 2.3 *Montrer que cet algorithme est aprtiellement correct par rapport à sa précondition et à sa postcondition qu'il faudra énoncer.*

Question 2.4 *Montrer qu'il est sans erreur à l'exécution.*

Exercice 3 *(Exponentiation en PlusCal pluscal_exponentiation,appex5_2)*
Ecrire l'algorithme de l'exponentiation avec PlusCal.
On suppose que x_1 et x_2 sont des constantes.

precondition : $x_1 \in \mathbb{N} \wedge x_2 \in \mathbb{N} \wedge x_1 \neq 0$
postcondition : $z = {x_1}^{x_2}$
local variables : $y_1, y_2, y_3 \in \mathbb{Z}$

$\ell_0 : \{y_1, y_2, y_3, z \in \mathbb{Z}\}$
$(y_1, y_2, y_3) := (x_1, x_2, 1)$;
$\ell_1 : \{y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
while $y_2 \neq 0$ **do**
  $\ell_2 : \{y_2 \neq 0 \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
  if $impair(y_2)$ **then**
    $\ell_3 : \{impair(y_2) \wedge y_2 \neq 0 \wedge \boxed{\ldots} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
    $y_2 := y_2 - 1$;
    $\ell_4 : \{y_2 \geq 0 \wedge pair(y_2) \wedge y_3 \cdot y_1 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
    $y_3 := y_3 \cdot y_1$;
    $\ell_5 : \{y_2 \geq 0 \wedge pair(y_2) \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
  ;
  $\ell_6 : \{y_2 \geq 0 \wedge pair(y_2) \wedge \boxed{\ldots} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
  $y_1 := y_1 \cdot y_1$;
  $\ell_7 : \{y_2 \geq 0 \wedge pair(y_2) \wedge y_3 \cdot {y_1}^{y_2 \ div 2} = {x_1}^{x_2} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
  $y_2 := y_2 \ div \ 2$;
  $\ell_8 : \{y_2 \geq 0 \wedge \boxed{\ldots} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
;
$\ell_9 : \{y_2 = 0 \wedge \boxed{\ldots} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z}\}$
$z := y_3$;
$\ell_{10} : \{y_2 = 0 \wedge y_3 \cdot {y_1}^{y_2} = {x_1}^{x_2} \wedge y_1, y_2, y_3 \in \mathbb{N} \wedge z \in \mathbb{Z} \wedge z = {x_1}^{x_2}\}$

Algorithme 2: Algorithme de l'exponentitaion indienne annoté

◇— **Solution de l'exercice 3**

MODULE $tp3$

EXTENDS *Naturals*, *Integers*, *TLC*

CONSTANT *MAXINT*, u, v

(·
−*termination*
−*wfNext*
−−*algorithm Power* {
variables $x_1 = u$; 1st integer
$x2 = v$; 2nd integer
$y1 = 0$;
$y2 = 0$;
$y3 = 0$;
$z = 0$;

{
$l1$: *print* $\langle x_1, x_2 \rangle$;

$y1 := x_1$;
$y2 := x_2$;
$y3 := 1$;
$l2$: *while* $(y_2 \; / = \; 0)$ {

$L3$: *if* $(\; y_2 \,\%\, 2 \neq 0)$ {
$l4 : y_2 := y_2 - 1$;
$l5 : y_3 := y_3 \cdot y1$;
};
$l6 : y_1 := y_1 \cdot y1$;
$l7 : y_2 := y_2 \div 2$;
};
$l8 : z := y_3$;
$l9$: *print* $\langle x_1, x_2, z \rangle$;
}

}
·)

Modification History
Last modified Fri Feb 26 06 :26 :27 CET 2016 by mery
Created Wed Sep 09 17 :02 :47 CEST 2015 by mery

La traduction par l'outil donne alors le module suivant :

MODULE $tp3$

EXTENDS *Naturals*, *Integers*, *TLC*

CONSTANT *MAXINT*, u, v

(·
−*termination*

$-wfNext$
$--algorithm\ Power$ {
$variables\ x_1 = u;$ 1st integer
$x2 = v;$ 2nd integer
$y1 = 0;$
$y2 = 0;$
$y3 = 0;$
$z = 0;$

{
$l1:$ ***print*** $\langle x_1,\ x_2\rangle;$

$y1 := x_1;$
$y2 := x_2;$
$y3 := 1;$
$l2:$ ***while*** $(y_2 \mathrel{/=} 0)$ {

$L3:$ ***if*** $(\ y_2 \% 2 \neq 0)$ {
$l4: y_2 := y_2 - 1;$
$l5: y_3 := y_3 \cdot y1;$
};
$l6: y_1 := y_1 \cdot y1;$
$l7: y_2 := y_2 \div 2;$
};
$l8: z := y_3;$
$l9:$ ***print*** $\langle x_1,\ x_2, z\rangle;$
}

}
$\cdot)$

BEGIN TRANSLATION

VARIABLES $x_1,\ x_2,\ y_1,\ y_2,\ y_3,\ z,\ pc$

$vars \triangleq \langle\ x_1,\ x_2,\ y_1,\ y_2,\ y_3,\ z,\ pc\ \rangle$

$Init \triangleq$ Global variables
$\land\ x_1 = u$
$\land\ x_2 = v$
$\land\ y_1 = 0$
$\land\ y_2 = 0$
$\land\ y_3 = 0$
$\land\ z = 0$
$\land\ pc = \text{"l1"}$

$l1 \triangleq \land\ pc = \text{"l1"}$
$\land\ PrintT(\langle x_1,\ x_2\rangle)$
$\land\ y_1' = x1$
$\land\ y_2' = x2$
$\land\ y_3' = 1$
$\land\ pc' = \text{"l2"}$
$\land$ UNCHANGED $\langle\ x_1,\ x_2,\ z\ \rangle$

$l2 \triangleq \land\ pc = \text{"l2"}$
$\land$ IF $y_2 \mathrel{/=} 0$
THEN $\land\ pc' = \text{"L3"}$

$\quad\quad \text{ELSE} \quad \land\ pc' = \text{"l8"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_1,\ y_2,\ y_3,\ z\,\rangle$

$L3 \triangleq \land\ pc = \text{"L3"}$
$\quad \land\ \text{IF}\ y_2\ \%\ 2\ \neq\ 0$
$\quad\quad \text{THEN} \ \land\ pc' = \text{"l4"}$
$\quad\quad \text{ELSE} \quad \land\ pc' = \text{"l6"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_1,\ y_2,\ y_3,\ z\,\rangle$

$l4 \triangleq \land\ pc = \text{"l4"}$
$\quad \land\ y_2' = y_2 - 1$
$\quad \land\ pc' = \text{"l5"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_1,\ y_3,\ z\,\rangle$

$l5 \triangleq \land\ pc = \text{"l5"}$
$\quad \land\ y_3' = y_3 \cdot y1$
$\quad \land\ pc' = \text{"l6"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_1,\ y_2,\ z\,\rangle$

$l6 \triangleq \land\ pc = \text{"l6"}$
$\quad \land\ y_1' = y_1 \cdot y1$
$\quad \land\ pc' = \text{"l7"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_2,\ y_3,\ z\,\rangle$

$l7 \triangleq \land\ pc = \text{"l7"}$
$\quad \land\ y_2' = (y_2 \div 2)$
$\quad \land\ pc' = \text{"l2"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_1,\ y_3,\ z\,\rangle$

$l8 \triangleq \land\ pc = \text{"l8"}$
$\quad \land\ z' = y3$
$\quad \land\ pc' = \text{"l9"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_1,\ y_2,\ y_3\,\rangle$

$l9 \triangleq \land\ pc = \text{"l9"}$
$\quad \land\ PrintT(\langle x_1,\ x_2, z\rangle)$
$\quad \land\ pc' = \text{"Done"}$
$\quad \land\ \text{UNCHANGED}\ \langle\, x_1,\ x_2,\ y_1,\ y_2,\ y_3,\ z\,\rangle$

$Next \triangleq l_1 \lor l_2 \lor L_3 \lor l_4 \lor l_5 \lor l_6 \lor l_7 \lor l_8 \lor l9$
$\quad\quad \lor$ Disjunct to prevent deadlock on termination
$\quad\quad\quad (pc = \text{"Done"} \land \text{UNCHANGED}\ vars)$

$Spec \triangleq Init \land \Box[Next]_{vars}$

$Termination \triangleq \Diamond(pc = \text{"Done"})$

END TRANSLATION

Modification History

Last modified Fri Feb 26 06 :26 :27 CET 2016 by mery

Created Wed Sep 09 17 :02 :47 CEST 2015 by mery

Fin 3

Exercice 4 *(squareroot)*
On considère l'algorithme $squareroot$ *calculant la racine carrée entière d'un nombre naturel* $x \in \mathbb{N}$.

VARIABLES $X, Y1, Y2, Y3, Z$

$pre(x0, y10, y20, y30, z0) \stackrel{def}{=}$
$U \stackrel{def}{=} (\ X, Y1, Y2, Y3, Z\ )$
$u0 \stackrel{def}{=} (\ x0, y10, y20, y30, z0\ )$
$post(x0, y10, y20, y30, z0, xf, y1f, y2f, y3f, zf) \stackrel{def}{=}$

REQUIRES $pre(x0, y10, y20, y30, z0)$
ENSURES $post(x0, y10, y20, y30, z0, xf, y1f, y2f, y3f, zf)$

$\ell_0 : pre(u0) \wedge u = u0$
$(Y1, Y2, Y3) := (0, 1, 1)$
$\ell_1 : pre(u0) \wedge x = x0 \wedge z = z0 \wedge y2 = (y1{+}1)\cdot(y1{+}1) \wedge y3 = 2\cdot y1{+}1 \wedge y1\cdot y1 \leq x$
WHILE $Y2 \leq X$ ***DO***
ℓ_2 :
  $(Y1, Y2, Y3) := (Y1{+}1, Y2{+}Y3{+}2, Y3{+}2)$;
ℓ_3 : ***OD***;
ℓ_4 :
$Z := Y1$;
ℓ_5 :

Question 4.1 *Définir les deux assertions pre et post qui établissent le contrat de cet algorithme.*

Question 4.2 *Complétez cet algorithme en proposant trois assertions :*
— $P_{\ell_2}(u0, u)$
— $P_{\ell_3}(u0, u)$
— $P_{\ell_4}(u0, u)$
— $P_{\ell_5}(u0, u)$
Pour cela, le plus efficace est de définir clairement les les conditions de vérifications : pour chaque paire (ℓ, ℓ') *d'étiquettes correspondant à un pas élémentaire ; on vérifie la propriété suivante :*
$P_\ell(u0, u)\ \wedge\ cond_{\ell,\ell'}(u)\ \wedge\ u' = f_{\ell,\ell'}(u) \Rightarrow P_{\ell'}(u')$
Enoncez et vérifiez cette propriété pour les paires d'étiquettes suivantes : (ℓ_1, ℓ_2) *;*(ℓ_1, ℓ_4) *;* (ℓ_2, ℓ_3) *;* (ℓ_3, ℓ_2) *;* (ℓ_3, ℓ_4) *;* (ℓ_4, ℓ_5) *;*

Question 4.3 *Finalisez les vérifications an montrant que les conditions de vérification pour un contrat sont toutes vérifiées.*

Question 4.4 *On suppose que toutes les conditions de vérifications associées aux paires d'étiquettes successives de l'algorithme sont vérifiées. Quelles sont les deux conditions à montrer pour déduire que l'algorithme est partiellement correct par rapport aux pré et post conditions ? Vous donnerez explicitement les conditions et vous expliquerez pourquoi elles sont correctes. <*

Question 4.5 *Expliquer que cet algorithme est sans erreurs à l'exécution, si les données initiales sont dans un domaine à définir inclus dans le domaine des entiers informatiques c'est-à-dire les entiers codables sur n bits. L'ensemble des entiers informatiques sur n bits est l'ensemble noté* $\mathbb{Z}_n$ *et défini par* $\{i | i \in \mathbb{Z}\ \wedge\ -2^{n-1} \leq i\ \wedge\ i \leq 2^{n-1}{-}1\}$.

L'algorithme annoté est décrit par l'algorithme **??**

VARIABLES $X, Y1, Y2, Y3, Z$

$pre(x0, y10, y20, y30, z0) \stackrel{def}{=} \begin{cases} x0 \in \mathbb{N} \wedge x0 \neq 0 \\ y10, y20, y30, z0 \in \mathbb{Z} \end{cases}$

$U \stackrel{def}{=} (\ X, Y1, Y2, Y3, Z\ )$

$u0 \stackrel{def}{=} (\ x0, y10, y20, y30, z0\ )$

$post(x0, y10, y20, y30, z0, xf, y1f, y2f, y3f, zf) \stackrel{def}{=} (\ zf^2 \leq x0 \wedge x0 \leq (zf{+}1)^2$

$pre(u0) \stackrel{def}{=} pre(x0, y10, y20, y30, z0)$

$(u = u0) \stackrel{def}{=} x = x0 \wedge y1 = y10 \wedge y1 = y10 \wedge y2 = y20 \wedge y3 = y30 \wedge z = z0$

REQUIRES $pre(x0, y10, y20, y30, z0)$
ENSURES $post(x0, y10, y20, y30, z0, xf, y1f, y2f, y3f, zf)$

$\ell_0 : pre(u0) \wedge u = u0$
$(Y1, Y2, Y3) := (0, 1, 1)$
$\ell_1 : pre(u0) \wedge x = x0 \wedge z = z0 \wedge y2 = (y1{+}1) \cdot (y1{+}1) \wedge y3 = 2 \cdot y1{+}1 \wedge y1 \cdot y1 \leq x$
WHILE $Y2 \leq X$ ***DO***
$\ell_2 : P_{\ell_1}(u0, u) \wedge y2 \leq x$
  $(Y1, Y2, Y3) := (Y1{+}1, Y2{+}Y3{+}2, Y3{+}2);$
$\ell_3 : P_{\ell_1}(u0, u)$
OD;
$\ell_4 : P_{\ell_1}(u0, u) \wedge y2 > x$
$Z := Y1;$
$\ell_5 : pre(u0) \wedge x = x0 \wedge y2 = (y1{+}1) \cdot (y1{+}1) \wedge y3 = 2 \cdot y1{+}1 \wedge y1 \cdot y1 \leq x \wedge x < y2$

Exercice 5 *(pluscal_division.tla)*
On considère l'algorithme suivant :

START
$\{x_1 \geqq 0 \wedge x_2 > 0\}$
$(y_1, y_2, y_3) \leftarrow (x_1, 0, x_2)$;
while $y_3 \leqq y_1$ **do** $y_3 \leftarrow 2y_3$;
while $y_3 \neq x_2$ **do**
  begin $(y_2, y_3) \leftarrow (2y_2, y_3/2)$;
    if $y_3 \leqq y_1$ **do** $(y_1, y_2) \leftarrow (y_1 - y_3, y_2 + 1)$
  end;
$(z_1, z_2) \leftarrow (y_1, y_2)$
$\{0 \leqq z_1 < x_2 \wedge x_1 = z_2 x_2 + z_1\}$
HALT

Question 5.1 *Montrer que cet algorithme est aprtiellement correct par rapport à sa précondition et à sa postcondition qu'il faudra énoncer. Pour cela, on traduira cet algorithme sous forme d'un module à partir du lanage PlusCal.*

Question 5.2 *Montrer qu'il est sans erreur à l'exécution.*

Question 5.3 *L'algorithme n'est pas réellement annoté suffisamment pour permettre une vérification complète de la correction partielle et dlabsence d'erreurs à l'exécution. En utilisant l'algorithme PlusCal annoter cet algorithme en vérifiant au fur et à mesure la bonne annotation.*

Sommaire On rappelle qu'un contrat pour la correction partielle d'un petit programme est donné par les éléments ci-dessou en colonne de gauche et que les conditions de vérification associées sont définies par le texte de la colonne de droite.

Contrat de la correction partielle

requires $pre(x_0)$
ensures $post(x_0, x_f)$
variables type X
  begin
  $0 : P_0(x_0, x)$
  $\texttt{instruction}_0$
  $1 : P_i(x_0, x)$
  $\texttt{instruction}_1$
  $f : P_f(x_0, x)$
  end

Conditions de vérification
- $pre(x_0) \wedge x = x_0 \Rightarrow P_0(x_0, x)$
- $pre(x_0) \wedge P_f(x_0, x) \Rightarrow post(x_0, x)$
- Pour toute paire d'étiquettes ℓ, ℓ' telle que $\ell \longrightarrow \ell'$, on vérifie que pour toutes les valeurs $x, x' \in \textsc{Memory}$
$$\left(\begin{array}{l} \left(\begin{array}{l} pre(x_0) \wedge P_\ell(x_0, x)) \\ \wedge cond_{\ell,\ell'}(x) \wedge x' = f_{\ell,\ell'}(x) \end{array} \right) \\ \Rightarrow P_{\ell'}(x_0, x') \end{array} \right),$$

Exercice 6 *En utilisant le contrat ci-dessus, confirmer ou infirmer les annotations suivantes :*

Question 6.1

$\ell_1 : x = 9 \ \wedge\ y = z{+}x$
$y := x{+}9$
$\ell_2 : x = 9 \ \wedge\ y = x{+}9$

requires $x0 = 9 \wedge y0 = z0{+}x0 \wedge x0, y0, z0 \in \mathbb{Z}$
ensures $xf = x0 \wedge yf = xf{+}9$
variables `int` X, Y, Z
begin
$0 : x = x0 \wedge y = \wedge z = z0 \wedge x0 = 9 \wedge y0 = z0{+}x0 \wedge x0, y0, z0 \in \mathbb{Z}$
$Y := X{+}9$
$f : x = 9 \ \wedge\ y = x{+}9$
end

Question 6.2

$\ell_1 : x = 1 \ \wedge\ y = 3 \ \wedge\ x{+}y = 12$
$x := y{+}x$
$\ell_2 : x = 567 \ \wedge\ y = 34$

requires $x0 = 1 \wedge y0 = 3 \wedge x0{+}y0 = 12 \wedge x0, y0 \in \mathbb{Z}$
ensures $xf = 567 \wedge yf = 34$
variables `int` X, Y
begin
$0 : x = x0 \wedge y = y0 \wedge x0 = 1 \wedge y0 = 3 \wedge x0{+}y0 = 12 \wedge x0, y0 \in \mathbb{Z}$
$X := Y{+}X$
$f : x = 567 \ \wedge\ y = 34$
end

Question 6.3 *(ex2_7.tla)*
Appliquer PlusCal pour vérifier ce contrat.