Année 2018/2019 — TD Preuve en calcul de Hoare

Bordeaux INP ENSEIRB MATMECA

LOGIQUE ET PREUVE DE PROGRAMMES
IF107

Filière : Informatique, Année : 1

Web: `http://herbrete.vvv.enseirb-matmeca.fr/IF107`

Règles du calcul de Hoare pour les while-programs :

$$\frac{}{\{R\}\ \texttt{skip}\ \{R\}}\ (skip) \qquad \frac{\{P \wedge B\}\ S_1\ \{R\} \quad \{P \wedge \neg B\}\ S_2\ \{R\}}{\{P\}\ \texttt{if}\ B\ \texttt{then}\ S_1\ \texttt{else}\ S_2\ \texttt{end}\ \{R\}}\ (if)$$

$$\frac{}{\{R[x \leftarrow t]\}\ \texttt{x := t}\ \{R\}}\ (aff.) \qquad \frac{\{I \wedge B\}\ S\ \{I\}}{\{I\}\ \texttt{while}\ B\ \texttt{do}\ S\ \texttt{end}\ \{I \wedge \neg B\}}\ (while)$$

$$\frac{\{P\}S_1\{Q\} \quad \{Q\}S_2\{R\}}{\{P\}\ S_1; S_2\ \{R\}}\ (seq.) \qquad \frac{P' \implies P \quad \{P\}\ S\ \{R\} \quad R \implies R'}{\{P'\}\ S\ \{R'\}}\ (cons.)$$

Exercice 1 (Validité de triplets de Hoare)

Prouver la validité des triplets suivants à l'aide du calcul de Hoare.

1. $\{y > 0\}\, x := y\, \{x > 0\}$
2. $\{y \geq 10\}\, x := y\, \{x > 0\}$
3. $\{z < -3\}\, y := 3 * z + 7; x := 2 * y\, \{x < z\}$
4. $\{x = X \wedge y = Y\}\, z := x; x := y; y := z\, \{x = Y \wedge y = X\}$
5. $\{x = X\}$ `if` $(x < 0)$ `then` $x := -x$ `else skip end` $\{x = |X|\}$
6. Que se passe-t-il lorsqu'on inverse les instructions $x := -x$ et `skip` dans le triplet précédent ?

♦

Exercice 2 (Nouvelle règle de while)

$$\frac{I \wedge B \implies P \quad \{P\}\, S\, \{I\} \quad I \wedge \neg B \implies R}{\{I\}\, \texttt{while}\ B\ \texttt{do}\ S\ \texttt{end}\, \{R\}}\ (while_2)$$

1. Prouver cette règle en utilisant le calcul de Hoare
2. Expliquer à quoi correspondent les deux obligations de preuve générées par la règle $(while_2)$

♦

> Dans les exercices suivants, on demontrera la terminaison des while-programs avec la méthode des ensembles bien fondés, et leur correction avec le calcul de Hoare, en utilisant la règle $(while_2)$ plutôt que la règle $(while)$.

Exercice 3 (La suite Fibonacci)

Prouver la terminaison et la correction de l'algorithme suivant qui calcule le terme de rang `n` de la suite de Fibonacci $(F_n)_{n\in\mathbb{N}}$.

$$\begin{cases} F_0 = F_1 = 1 \\ F_n = F_{n-1} + F_{n-2} \quad n \geq 2 \end{cases}$$

```
1  {n ∈ ℕ}
2  y := x; x := 1; k := 1;
3  while (k < n) do
4     t := y; y := x; x := x+t; k := k+1;
5  end
6  {x = F_n}
```

♦

Exercice 4 (Tri à bulles)

Prouver la correction et la terminaison du while-program ci-dessous qui tri le tableau `t` de taille `n` par la méthode du tri à bulles. On introduit les prédicats :

- $sorted(t, i, j)$ défini par $\forall k.(i \leq k < j \implies t[i] \leq t[i+1])$ qui est vrai si t est trié des indices i à j.
- $bigger(t, imax, i, j)$ défini par $\forall k.(i \leq k \leq j \implies t[k] \leq t[imax])$ qui est vrai si tous les éléments de t des indices i à j sont plus petits ou égaux à $t[imax]$.

ainsi que la fonction :

- $swap(t, i, j)$ qui permute les éléments d'indice i et j dans t. En notant $t' = swap(t, i, j)$, on a : $t'[i] = t[j] \wedge t'[j] = t[i] \wedge \forall k \in [0; n) \setminus \{i, j\}.t'[k] = t[k]$.

```
1   {n ∈ ℕ ∧ n > 0}
2   i := n−1;
3   while (i > 0) do
4     j := 1;
5     while (j ≤ i) do
6       if (t[j − 1] > t[j]) then
7         t := swap(t, j − 1, j)
8       end;
9       j := j+1
10    end;
11    i := i−1
12  end
13  {sorted(t, 0, n − 1)}
```

♦

Exercice 5 (Multiplication rapide)

Prouver la terminaison et la correction de l'algorithme suivant qui calcule le produit de a par b. NB : / calcule la division entière et % le reste de la division entière.

```
1  {a ∈ ℕ ∧ b ∈ ℕ}
2  p:=0; a':=a; b':=b;
3  while (b'>0) do
4    if (b'%2 = 1) then
5      p:=p+a'
6    else
7      skip
8    endif;
9    a':=a'*2;
10   b':=b'/2
11 endwhile
12 {p = a × b}
```

♦