Génie Logiciel Avancé

Année 2020-2021

Prof. Burkhart Wolff
wolff@lri.fr

Hugo Mlodecki
hugo.mlodecki@u-psud.fr

TD 9 - Preuve de programmes

Semaine du 4 décembre 2017

Exercice 1 (Preuve en logique de Hoare)

1. Si possible, dérivez les triplets de Hoare suivants en utilisant les règles d'inférence introduites dans le cours (ces règles sont rappelées au dos de cette page). On suppose l'arithetique des nombres entiers.
   (a) $\vdash \{x > 0\}$ `x:==x-1` $\{0 \leq x\}$
   (b) $\vdash \{0 \leq x \wedge x \ mod \ 5 > 5\}$ `x:==x*x` $\{x \ div \ 2 = 1\}$
   (c) $\vdash \{x \leq 0\}$ `WHILE x < 0 DO x:==x+1` $\{x = 0\}$
   (d) $\vdash \{S * Y^P = X^N\}$ `P:=P-1; S:=S*Y` $\{S * Y^P = X^N\}$
   (e) $\vdash \{x \leq -2\}$ `WHILE 0 < x*x DO x:==x+1` $\{x = 0\}$
2. (a) Donnez une spécification du programme ci-dessous sous forme de pré et post-conditions.
   (b) Vérifiez le programme avec la méthode de Hoare. Indication : l'invariant est composé de deux conditions, une portant sur `F` qui tend vers la post-condition et l'autre exprimant que `P` ne peut pas dépasser une certaine valeur.

```
P := 1;
F := 1;
WHILE P <= N DO
  F := F*P;
  P := P+1
```

Calcul de Hoare

$$\frac{}{\vdash \{P\}\ \texttt{SKIP}\ \{P\}}\ skip \qquad \frac{}{\vdash \{P[x \mapsto E]\}\ \texttt{x :== } E\ \{P\}}\ assignment$$

$$\frac{\vdash \{P \wedge cond\}\ c\ \{Q\} \qquad \vdash \{P \wedge \neg cond\}\ d\ \{Q\}}{\vdash \{P\}\ \texttt{IF}\ cond\ \texttt{THEN}\ c\ \texttt{ELSE}\ d\ \{Q\}}\ ifthenelse$$

$$\frac{\vdash \{P \wedge cond\}\ c\ \{P\}}{\vdash \{P\}\ \texttt{WHILE}\ cond\ \texttt{DO}\ c\ \{P \wedge \neg cond\}}\ while$$

$$\frac{P \to P' \qquad \vdash \{P'\}\ c\ \{Q'\} \qquad Q' \to Q}{\vdash \{P\}\ c\ \{Q\}}\ consequence$$

$$\frac{}{\vdash \{false\}\ c\ \{P\}}\ falseE \qquad \frac{\vdash \{P\}\ c\ \{Q\} \qquad \vdash \{Q\}\ d\ \{R\}}{\vdash \{P\}\ c; d\ \{R\}}\ sequence$$

Exercice 2
Donner la spécification sous-forme de pré et post-conditions, ainsi que les invariants de boucle pour les programmes suivants.

Factorielle de n

```
int fact(int n) {
  int f = 1;
  int i;

  for(i = 2; i <= n; i++) {
    f = f * i;
  }
  return f;
}
```

Minimum d'un tableau

```
int min(int t[],int n) {
  int i;
  int m = t[0];

  for(i = 1; i < n; i++) {
    if(t[i] < m) {
      m = t[i];
    }
  }
  return m;
}
```

Échange de valeurs

```
void swap(int t[], int i, int j) {
  int tmp = t[i];
  t[i] = t[j];
  t[j] = tmp;
}
```

Tri par sélection

```
void tri_selection(int t[], int n) {
  int i = 0;
  int min;

  while(i < n) {
    int min = i;
    int j = i+1;
    while (j < n) {
      if (t[j] < t[min]) {
        min = j;
      }
      j = j+1;
    }
    swap(t,i,min);
    i = i + 1;
  }
}
```