Génie Logiciel Avancé

Année 2016-2017

Delphine Longuet, Thibaut Balabonski, Robin Pelle, Hadi Zaatiti
longuet@lri.fr, blsk@lri.fr, pelle@lri.fr, hadizaatiti@gmail.com

Examen

14 décembre 2016
Durée : 2 h 30

Seuls les transparents du cours (éventuellement annotés) sont autorisés.

Exercice 1 (barème indicatif : 11 points)
La fonction suivante calcule la multiplication de deux nombres entiers a et b avec b positif ou nul.

```
int mult(int a, int b) {
    int res = 0;
    while(b > 0) {
        if(b mod 2 != 0) {
            res = res + a;
            b = b - 1;
        }
        a = 2 * a;
        b = b/2;
    }
    return res;
}
```

1. Uniquement à partir de la spécification, donner exactement 3 tests pour la fonction `mult`. Pour chacun des tests, donner son objectif, des données d'entrée concrètes et le résultat attendu.

Partie 1. Critères sur le graphe de flot de contrôle

2. Donner le graphe de flot de contrôle de la fonction `mult`.
3. Donner une expression régulière représentant l'ensemble des chemins du graphe.

Toutes les instructions.

4. Donner le chemin le plus court permettant de satisfaire le critère « toutes les instructions ». On notera ce chemin `ch1`.
5. Par exécution symbolique, calculer la condition associée à ce chemin. Ce chemin est-il faisable ? Si oui, donner un test concret pour ce chemin, sinon, expliquer pourquoi.

Toutes les décisions. On considère l'ensemble de chemins $T = \{\texttt{ch1}\}$.

6. Compléter T par le plus court chemin permettant à T de satisfaire le critère « toutes les décisions »(également appelé « toutes les branches » ou « tous les arcs »). Justifier que le critère est satisfait.
7. Par exécution symbolique, calculer la condition associée à ce nouveau chemin. Est-il faisable ? Expliquer.
8. Déduire de la question précédente une condition suffisante pour qu'un chemin du graphe de flot de contrôle de cette fonction soit infaisable.
9. Ajouter finalement à T le plus court chemin faisable permettant à T de satisfaire le critère « toutes les décisions ». On notera ce chemin `ch2`.
10. Calculer par exécution symbolique la condition associée à ce chemin. Donner un test concret pour ce chemin.

Partie 2. Critères sur le graphe de flot de données

11. Annoter le graphe de flot de contrôle donné en question 2 par le flot de données de `mult`, c'est-à-dire les définitions et utilisations des variables.
12. Pour chacune des variables suivantes, donner une expression régulière représentant le flot de données pour cette variable. Préciser si le flot de données présente des anomalies ou non. En cas d'anomalie, expliquer s'il s'agit d'une erreur et pourquoi.
    (a) la variable b ;
    (b) la variable a.

Toutes les définitions.

13. Pour chacune des variables suivantes, donner un ensemble de chemins faisables permettant de satisfaire le critère « toutes les définitions ». Justifier soigneusement que le critère est satisfait. Remarque : il est possible d'utiliser les chemins de la partie 1.
    (a) la variable b ;
    (b) la variable a.

Toutes les utilisations.

14. Pour chacune des variables suivantes, donner un ensemble de chemins faisables permettant de satisfaire le critère « toutes les utilisations ». Justifier soigneusement que le critère est satisfait, ou expliquer pourquoi il ne peut pas l'être le cas échéant.
    (a) la variable res ;
    (b) la variable b.

Exercice 2 (barème indicatif : 6 points)
On veut prouver que le programme suivant effectue la multiplication de a par b pour $b \geq 0$.

```
x := 0;
y := 0;
WHILE y < b DO
   x := x + a;
   y := y + 1
```

Les règles du calcul de Hoare sont rappelées à la fin de l'énoncé. Dans vos réponses aux questions, **l'application d'une règle doit toujours être justifiée.**

1. Écrire la spécification du programme sous forme de pré et post-conditions.
2. Quel est le triplet de Hoare à prouver ?
3. Trouver un invariant Inv pour la boucle `WHILE`.

Les questions suivantes vous guident dans la preuve de la boucle `WHILE`, en partant des feuilles de l'arbre de preuve. Elles peuvent cependant être traitées indépendamment les unes des autres.

4. Dériver le triplet de Hoare :

$$\{x = a \times y \wedge y < b\}\ \texttt{x := x + a; y := y + 1}\ \{x = a \times y \wedge y \leq b\}$$

Indication : trouver une propriété S qui permette de démontrer le triplet

$$\{S\}\ \texttt{y := y + 1}\ \{x = a \times y \wedge y \leq b\}$$

5. On note `Loop` la boucle `WHILE` du programme. Compléter la preuve précédente afin de dériver le triplet :

$$\{x = a \times y \wedge y \leq b\}\ \texttt{Loop}\ \{x = a \times y \wedge y = b\}$$

6. Terminer la preuve de la boucle `WHILE` en complétant votre preuve de manière à dériver le triplet :

$$\{b \geq 0 \wedge x = 0 \wedge y = 0\}\ \texttt{Loop}\ \{x = a \times b\}$$

7. Donner un variant pour la boucle de ce programme, c'est-à-dire une expression toujours positive ou nulle et qui décroît strictement à chaque tour de boucle.

Exercice 3 (barème indicatif : 3 points)
On considère le programme suivant qui renvoie *true* si et seulement si l'entier v est présent dans le tableau t.

```
boolean contains(int t[], int v) {
    int i = 0;
    while(i < t.length) {
        if(t[i] == v) {
            return true;
        }
        i++;
    }
    return false;
}
```

1. Donner la spécification de ce programme en termes de pré et post-condition.
2. Donner un invariant pour la boucle.
   Indication : l'invariant est composé de deux propriétés, l'une portant sur le compteur de la boucle et l'autre portant sur la progression de la recherche de l'élément v dans le tableau. On rappelle que l'invariant, avec la négation de la condition du `while`, doit impliquer (une partie de) la post-condition.
3. Donner un variant pour la boucle, c'est-à-dire une expression toujours positive ou nulle et qui décroît strictement à chaque tour de boucle.

Calcul de Hoare

$$\frac{}{\{P\}\ \texttt{SKIP}\ \{P\}}\ \text{skip} \qquad \frac{}{\{P[x \mapsto exp]\}\ \texttt{x := exp}\ \{P\}}\ \text{affectation}$$

$$\frac{\{P \wedge cond\}\ \texttt{ins}_1\ \{Q\} \qquad \{P \wedge \neg cond\}\ \texttt{ins}_2\ \{Q\}}{\{P\}\ \texttt{IF cond THEN ins}_1\ \texttt{ELSE ins}_2\ \{Q\}}\ \text{ifthenelse}$$

$$\frac{\{P \wedge cond\}\ \texttt{ins}\ \{P\}}{\{P\}\ \texttt{WHILE cond DO ins}\ \{P \wedge \neg cond\}}\ \text{while}$$

$$\frac{P \Rightarrow P' \qquad \{P'\}\ \texttt{ins}\ \{Q'\} \qquad Q' \Rightarrow Q}{\{P\}\ \texttt{ins}\ \{Q\}}\ \text{consequence}$$

$$\frac{}{\{false\}\ \texttt{ins}\ \{P\}}\ \text{falseE} \qquad \frac{\{P\}\ \texttt{ins}_1\ \{Q\} \qquad \{Q\}\ \texttt{ins}_2\ \{R\}}{\{P\}\ \texttt{ins}_1\ \texttt{;}\ \texttt{ins}_2\ \{R\}}\ \text{sequence}$$