Algoritmos y Estructuras de Datos I - 1º cuatrimestre 2010

Práctico 4: Programación imperativa

El objetivo general de este práctico es afianzar los conceptos elementales de la programación imperativa. Por un lado, se pretende reforzar la noción de programa como **transformador de estados***, de manera intuitiva, a través de los ejercicios 1 y 2. Por otro lado, priorizando la interpretación de los programas como* **transformadores de predicados***, se busca consolidar la noción de corrección de programa, a través de la verificación de programas simples (ej. 3, 4, 5 y 9). Ademas se presentan las nociones de equivalencia y refinamiento de programas y se busca reflexionar principalmente sobre las ventajas prácticas que estas nociones introducen a la hora de escribir y verificar programas (ej. 6, 7, 9, 10, 11, 12).*

1. ¿Cuál es el valor que se computa en r, en cada uno de los siguientes programas?

(a)
$x := y;$
$\textbf{if } x = y \rightarrow r := \textit{true}$
$\Box\ x \neq y \rightarrow r := \textit{false}$
$\textbf{fi}$

(b)
$x := y;$
$\textbf{if } x \leq y \rightarrow r := \textit{true}$
$\Box\ x \geq y \rightarrow r := \textit{false}$
$\textbf{fi}$

(c)
$x, r := 2 * y, \textit{true};$
$\textbf{do } x > 0 \rightarrow x, r := x - 1, \neg r$
$\textbf{od}$

(d)
$\textbf{do } r < 0 \rightarrow r + 1$
$\Box\ \ r > 0 \rightarrow r - 1$
$\textbf{od}$

(e)
$x, r := 0, \textit{true};$
$\textbf{do } x > 0 \wedge r \rightarrow r := \textit{false}$
$\textbf{od}$
$r := \neg r$

(f)
$x, r := 2 * y, \textit{true};$
$\textbf{do } x > 0 \rightarrow x, r := x + 1, \neg r$
$\textbf{od}$
$r := \textit{false}$

2. Encuentre (informalmente) un programa imperativo que resuelva cada una de las siguientes especificaciones descriptas en lenguaje natural:

   a) Dado un valor inicial positivo X, la variable r es *true* si X es par, y *false* en caso contrario.
   b) Dado un valor inicial positivo X, la variable r almacena la suma de los primeros X numeros pares.
   c) Dado un valor inicial positivo X, el valor de r determina si X es un cuadrado perfecto.
   d) Dado un valor inicial positivo X, el valor de r determina si X es un número primo.
   e) Dado dos pares de valores iniciales N_1, D_1 y N_2, D_2 que representan los números racionales $\frac{N_1}{D_1}$ y $\frac{N_2}{D_2}$ respectivamente, el valor de r_n y r_d representa el producto de dichas fracciones como la fracción $\frac{r_n}{r_d}$. El resultado debe estar simplificado tanto como sea posible.
   f) Para valores iniciales enteros X e Y, en r se almacena el resultado de multiplicar X por Y utilizando sólo sumas.
   g) Dado valores iniciales positivos $PESOS$ y $CENTAVOS$ que representan una suma de dinero, las variables $r1$, $r2$, $r3$ y $r4$ almacenan valores que representan la forma de dar cambio de la suma inicial en monedas de 50, 25, 10 y 5 centavos respectivamente. Ademas r almacena el (posible) resto que no puede devolverse.
   h) Dado un valor inicial positivo X, r es el resultado de multiplicar los digitos de X.

3. *Calcule* una precondición P de modo que sean correctos los siguientes programas anotados, suponiendo que las variables x, y, z, q, r son de tipo Int, las variables i, j de tipo Nat y las variables a, b de tipo $Bool$:

   a) $\{P\}\ x := 8\ \{x = 8\}$
   b) $\{P\}\ x := 8\ \{x \neq 8\}$
   c) $\{P\}\ x := 9\ \{x = 7\}$
   d) $\{P\}\ x := x + 1;\ y := y - 2\ \{x + y = 0\}$
   e) $\{P\}\ x := x + 1;\ y := y - 1\ \{x * y = 0\}$
   f) $\{P\}\ x := x + 1;\ y := y - 1\ \{x + y + 10 = 0\}$

g) $\{P\}\ z := z * y;\ x := x - 1\ \{z * y^x = C\}$
h) $\{P\}\ x, y, z := 1, d, c\ \{x * x^y = c^d\}$
i) $\{P\}\ i, j := i + i, j;\ j := j + i\ \{i = j\}$
j) $\{P\}\ x := (x - y) * (x + y)\ \{x + y^2 = 0\}$
k) $\{P\}\ q, r := q + 1, r - y\ \{q * y + r = x\}$
l) $\{P\}\ a := a \equiv b;\ b := a \equiv b;\ a := a \equiv b\ \{(a \equiv B) \wedge (b \equiv A)\}$

4. Calcule las expresiones E y F *(intuitivamente) más simples* de modo que las siguientes ternas de Hoare sean correctas:

   a) $\{A = q * B + r\}\ q := E;\ r := r - B\ \{A = q * B + r\}$
   b) $\{x * y + p * q = N\}\ x := x - p;\ q := F\ \{x * y + p * q = N\}$

5. Considere los siguientes programas que intercambian los valores de dos variables x e y de tipo Int:

$x, y := y, x$	$z := x;$ $x := y;$ $y := z$	$x := x - y;$ $y := x + y;$ $x := y - x$

   a) Especifique la pre y postcondición, y *verifique* los tres programas.
   b) Decimos que dos programas S y T son *equivalentes*, denotado por $S \simeq T$, si y solo si $wp.S.Q \equiv wp.T.Q$ para cualquier predicado Q. Demuestre que el primer programa es equivalente al tercero, valiéndose de las siguientes propiedades de sustitución sintáctica en predicados:
   - $(Q(x := E))(x := F) \equiv Q(x := E(x := F))$
     donde x es una (lista de) variable(s), y E y F son (listas de) expresiones bien definidas.
   - $Q(x := E) \equiv Q(x, y := E, y)$
     donde x e y son variables distintas.

   c) ¿Es el segundo programa equivalente a los demás? En caso negativo, ¿como podemos relajar la definición de equivalencia, de modo que sea satisfecha por este programa respecto a cualquiera de los otros dos?

6. *a*) Demuestre las siguientes equivalencias entre programas:
   1) $x := x\ \simeq$ **skip**
   2) S; **skip** $\simeq S$ y **skip**; $S \simeq S$
   3) S; **abort** $\simeq$ **abort** y **abort**; $S \simeq$ **abort**
   4) $(S;\ T);\ U \simeq S;\ (T;\ U)$

   b) Haciendo analogía con las propiedades del cálculo proposicional, ¿qué nombres podríamos darle a las propiedades (2), (3) y (4)?

7. *a*) Usando las propiedades del transformador de predicados *weakest precondition*, demuestre las siguientes propiedades:
   1) $\{P\}\ S\ \{Q\} \wedge [P_0 \Rightarrow P] \quad \Rightarrow \quad \{P_0\}\ S\ \{Q\}$
   2) $\{P\}\ S\ \{Q\} \wedge [Q \Rightarrow Q_0] \quad \Rightarrow \quad \{P\}\ S\ \{Q_0\}$
   3) $\{P\}\ S\ \{Q\} \wedge \{P\}\ S\ \{R\} \quad \equiv \quad \{P\}\ S\ \{Q \wedge R\}$
   4) $\{P\}\ S\ \{Q\} \wedge \{R\}\ S\ \{Q\} \quad \equiv \quad \{P \vee R\}\ S\ \{Q\}$

   b) Desde un punto de vista práctico, ¿qué aportan las propiedades anteriores a la hora de verificar la corrección de un programa?

8. Sean S, T, S_0, S_1 programas cualesquiera, y B_0, B_1 guardas cualesquiera. En cada caso, ¿son equivalentes entre sí los programas? En caso afirmativo demuéstrelo, en caso negativo dé un contraejemplo (instanciando los programas y las guardas).

    a) *i*)
    $x := E;$
    $y := F;$

    ii)
    $y := F;$
    $x := E;$

    b) *i*)
    if $B_0 \rightarrow S$
    $\square\ B_1 \rightarrow S$
    fi

    ii) S

    c) *i*)
    if $B_0 \rightarrow S; S_0; T$
    $\square\ B_1 \rightarrow S; S_1; T$
    fi

    ii)
    if $B_0 \rightarrow S; S_0$
    $\square\ B_1 \rightarrow S; S_1$
    fi;
    T

    iii)
    $S;$
    if $B_0 \rightarrow S_0; T$
    $\square\ B_1 \rightarrow S_1; T$
    fi

9. Demuestre que los siguientes programas anotados son correctos. En todos los casos las variables x, y son de tipo Int, y a, b de tipo $Bool$.

    a)
    $\{true\}$
    if $x \geq 1 \rightarrow x := x + 1$
    $\square\ x \leq 1 \rightarrow x := x - 1$
    fi
    $\{x \neq 1\}$

    b)
    $\{true\}$
    if $x \geq y \rightarrow$ **skip**
    $\square\ x \leq y \rightarrow x, y := y, x$
    fi
    $\{x \geq y\}$

    c)
    $\{true\}$
    $x, y := y * y, x * x;$
    if $x \geq y \rightarrow x := x - y$
    $\square\ x \leq y \rightarrow y := y - x$
    fi
    $\{x \geq 0 \wedge y \geq 0\}$

    d)
    $\{true\}$
    if $\neg a \vee b \rightarrow a := \neg a$
    $\square\ a \vee \neg b \rightarrow b := \neg b$
    fi
    $\{a \vee b\}$

    e)
    $\{N \geq 0\}$
    $x := 0;$
    do $x \neq N \rightarrow x := x + 1$
    od
    $\{x = N\}$

    f)
    $\{N \geq 0\}$
    $x, y := 0, 0;$
    do $x \neq 0 \rightarrow x := x - 1$
    $\square\ y \neq N \rightarrow x, y := N, y + 1$
    od
    $\{x = 0 \wedge y = N\}$

10. Una relación más *debil* que la equivalencia de programas es la *implicacion* entre programas (también llamada *refinamiento*). Decimos que un programa S *implica* un programa T (o T es un *refinamiento* de S) cuando $wp.S.Q \Rightarrow wp.T.Q$ para cualquier predicado Q, y lo denotamos con $S \sqsubseteq T$.

    a) Demuestre los siguientes refinamientos:

    i)
    if $B \rightarrow S$
    fi
    $\sqsubseteq S$

    ii)
    if $B_0 \rightarrow S_0$
    $\square\ B_1 \rightarrow S_1$
    fi
    $\sqsubseteq$
    if $B_0 \rightarrow S_0$
    $\square\ \neg B_0 \rightarrow S_1$
    fi

    iii)
    if $B_0 \rightarrow S_0$
    $\square\ B_1 \rightarrow S_1$
    $\square\ B_2 \rightarrow S_2$
    fi
    $\sqsubseteq$
    if $B_0 \rightarrow S_0$
    $\square\ \neg B_0 \rightarrow$ **if** $B_1 \rightarrow S_1$
    $\square\ B_2 \rightarrow S_2$
    fi
    fi

    iv)
    if $B_0 \rightarrow S_0$
    $\square\ B_1 \rightarrow S_1$
    fi
    $\sqsubseteq$
    if $B_0 \wedge \neg B_1 \rightarrow S_0$
    $\square\ B_1 \rightarrow S_1$
    fi

    b) En términos prácticos, ¿qué aporta la noción de refinamiento respecto a la verificación de programas? es decir, si se ha demostrado $\{P\}\ S\ \{Q\}$ y $S \sqsubseteq S'$, ¿qué se puede decir de S' respecto a P y Q?

    c) ¿Qué relación hay entre las nociones de equivalencia y refinamiento de programas?

    d) Revise el ejercicio 5 (*a* y *b*) ¿podría haberlo resuelto haciendo menos cuentas?

11. Demuestre las siguientes equivalencias entre programas. Para los ítems b y c en los que no podemos calcular la *weakest precondition*, ¿qué definición alternativa de *equivalencia* podemos utilizar?

a) $\textbf{if}\ \mathit{false} \to S\ \textbf{fi}\ \simeq\ \textbf{abort}$

b) $\textbf{do}\ \mathit{false} \to S\ \textbf{od}\ \simeq\ \textbf{skip}$

$$c)\quad \begin{array}{l} \textbf{do}\ B_0 \to S_0 \\ \square\ \ B_1 \to S_1 \\ \textbf{od} \end{array} \quad \simeq \quad \begin{array}{l} \textbf{do}\ B_0 \vee B_1 \to \\ \quad \textbf{if}\ B_0 \to S_0 \\ \quad \square\ B_1 \to S_1 \\ \quad \textbf{fi} \\ \textbf{od} \end{array}$$

12. Muchos lenguajes de programación actuales imponen restricciones sobre las guardas de los ciclos **do** (por ejemplo permitiendo una única guarda) y condicionales **if** (por ejemplo permitiendo solamente guardas disjuntas). Interprete cómo las relaciones entre programas $10a$ (*iii*), $10a$ (*iv*) y $10\,c$ permiten sortear esas restricciones.