Algoritmos y Estructuras de Datos I - 1º cuatrimestre 2010

Práctico 6: Especificación y derivación de programas imperativos

*Este práctico apunta a la incorporación de las técnicas avanzadas de derivación de programas imperativos, en particular, de aquellas utilizadas para encontrar invariantes. Intenta además integrar el proceso de derivación, con las competencias de especificación e interpretación (*semántica*) de las formulas involucradas (pre y postcondición, e invariantes). El objetivo de los ítems finales (ej. 7 y 8) es mostrar la estrecha relación entre la programación imperativa y funcional, y presentar a esta última como técnica para la derivación de programas imperativos.*

1. Considere los siguientes programas anotados, donde la variable x es de tipo Int:

   i) $\{P\}$
   do $x \neq 0 \rightarrow x := x - 1$
   od
   $\{x = 0\}$

   ii) $\{P\}$
   do $x \neq 0 \rightarrow x := x - 2$
   od
   $\{x = 0\}$

   a) Determine en cada caso una precondición P, la más débil que ud. encuentre, de manera que se satisfaga la corrección de las anotaciones.

   b) El predicado P encontrado, ¿es *la precondición más débil*?

2. Dado $N \geq 0$, $x, y : Int$, y $n : Nat$ especifique y derive un programa que:

   a) calcule en N-ésimo número de Fibonacci.

   b) calcule el menor entero x tal que $x^3 + x \geq N$.

   c) calcule el mayor entero x tal que $x^3 + x \leq N$.

   d) determine si N es *compuesto* (es decir, si no es primo).

   e) calcule el *mínimo común múltiplo* entre X e Y.

3. Considere la siguiente especificación:

   $|[\ Var\ r : Bool$
   $Var\ xs, ys : List$
   $Const\ XS, YS : List$
   $xs, ys := XS, YS$
   $\{ N \geq 0 \wedge \#xs = N \wedge \#ys = N \}$
   S
   $\{ r = \langle \forall i\ :\ 0 \leq i < N\ :\ xs.i = ys.(N - 1 - i) \rangle \}$
   $]|$

   a) Describa en lenguaje natural qué valor se computa en la variable r.

   b) Derive un programa S que satisfaga la especificación anterior. Suponga que el lenguaje imperativo tiene soporte para variables de tipo lista y la operación . sobre las mismas, aunque las variables de este tipo no pueden ser modificadas.

4. Bajo las hipótesis del ejercicio 3b y dada una variable $xs : List$, especifique y derive un programa que:

   a) determine si todos los elementos de xs son positivos.

   b) determine si algún elemento de xs es positivo.

   c) guarde en una variable el máximo elemento de xs.

5. Si queremos rehacer los ejercicios 3 y 4 con arreglos en lugar de listas, ¿qué es necesario cambiar en el código, en las especificaciones y en las demostraciones?

6. Considere la siguiente especificación:

$$
\begin{array}{l}
|[\ Var\ a : Array[0..N)\ of\ Int \\
\quad Var\ b : Array[0..M)\ of\ Int \\
\{ \langle \forall i \ :\ 0 \leq i < N \ :\ 0 \leq a.i < M \rangle \} \\
\qquad S \\
\{ \langle \forall j \ :\ 0 \leq j < M \ :\ b.j = \langle \mathrm{N}\ i \ :\ 0 \leq i < N \ :\ a.i = j \rangle \rangle \} \\
|]
\end{array}
$$

   a) Describa en lenguaje natural en qué consisten los valores computados en el arreglo b.

   b) Derive un programa S que satisfaga la especificación anterior. Note que hay dos constantes que pueden ser reemplazadas por variables para encontrar un invariante. Según el orden de reemplazo analice ambas variantes y compare los programas obtenidos.

   c) Describa en lenguaje natural qué significan los invariantes obtenidos en el ítem anterior.

7. Derive programas imperativos que satisfagan las siguientes especificaciones. Recuerde que las constantes de especificación no pueden utilizarse en el código del programa.

   a)
$$
\begin{array}{l}
|[\ Var\ f, h : Array[0..N)\ of\ Int \\
\quad Const\ F : Array[0..N)\ of\ Int \\
\quad f := F \\
\quad \{ true \} \\
\qquad S \\
\quad \{ \langle \forall k \ :\ 0 \leq k < N \ :\ h.k = \langle \sum i \ :\ 0 \leq i \leq k \ :\ f.i \rangle \rangle \} \\
\quad]|
\end{array}
$$

   b)
$$
\begin{array}{l}
|[\ Var\ f : Array[0..N)\ of\ Int \\
\quad Const\ F : Array[0..N)\ of\ Int \\
\quad f := F \\
\quad \{ \langle \forall i \ :\ 0 \leq i < N \ :\ f.i = F.i \rangle \} \\
\qquad S \\
\quad \{ \langle \forall i \ :\ 0 \leq i < N \ :\ F.i = \langle \sum k \ :\ 0 \leq k < i \ :\ f.k \rangle \rangle \} \\
\quad]|
\end{array}
$$

8. Considere las siguientes funciones recursivas:

   i)
$$
\begin{array}{rcl}
fac.0 &=& 1 \\
fac.(n+1) &=& (n+1) \times fac.n
\end{array}
$$

   ii)
$$
\begin{array}{rcl}
g.x.0 &=& 0 \\
g.x.(n+1) &=& ((n+1)\ mod\ 2 = 0 \rightarrow (1+x) \times g.(x \times x).n \\
&& \square (n+1)\ mod\ 2 = 1 \rightarrow 1 + x \times g.x.(2 \times n) \\
&&)
\end{array}
$$

   iii)
$$
\begin{array}{rcl}
mcd.x.y &=& (x = y \rightarrow x \\
&& (x \neq y \rightarrow mcd.(max.x.y - min.x.y).(min.x.y) \\
&&)
\end{array}
$$

   iv)
$$
\begin{array}{rcl}
in.n.[\,] &=& false \\
in.n.(x \triangleright xs) &=& (x = n) \vee in.(n+x).xs
\end{array}
$$

   a) Determine si cada función es recursiva final, y en caso negativo, derive una función recursiva final que resuelva el mismo problema.

   b) Escriba programas imperativos anotados para cada una de las funciones anteriores. En el caso (*iv*) suponga que el lenguaje imperativo tiene soporte para el tipo lista y para las operaciones utilizadas.

9. Considere los siguientes problemas, cuya solución funcional se encuentra en el libro de la materia:

   i) Evaluación de un polinomio (pág. 189).

   ii) Problema del segmento de suma mínima (pág. 211).

   iii) Problema de la lista balanceada (pág. 197).

   a) Escriba un programa imperativo para cada uno de los problemas mencionados. En caso de ser necesario, suponga que el lenguaje imperativo tiene soporte para el tipo de listas y las operaciones involucradas.

   b) Dé una función de abstracción que permita implementar listas con arreglos. Implemente cada una de las operaciones utilizadas en el punto anterior. Reescriba los programas anteriores utilizando la implementación obtenida.