UNIVERSIDADE DA CORUÑA

# Tecnología de la programación
## Ingeniería Técnica en Informática de Sistemas. Curso 2009/10
### Convocatoria Extraordinaria de Septiembre (6 de Septiembre de 2010)

Nombre : .....

Apellidos : .....

D.N.I. : .....

*Instrucciones: En las preguntas 1 a 4 marcar todas las respuestas que sean correctas. Cada pregunta totalmente correcta se valorará con +0,5 puntos.*

**Pregunta 1**: Dada la expresión $\alpha : (\forall K \in [0, N] \wedge b[K] < 0 : (\forall J \in [0, K\rangle : b[J] < b[K]))$:

(a) es equivalente a $(\exists K \in [0, N] \wedge b[K] \geq 0 : (\forall J \in [0, K\rangle : b[J] < b[K]))$.

(b) Significa que los elementos negativos de $b$ están ordenados decrecientemente.

×(c) $\alpha_I^K = \alpha$

×(d) la sustitución $\alpha_i^N$ es igual a $(\forall K \in [0, i] \wedge b[K] < 0 : (\forall L \in [0, K\rangle : b[L] < b[K]))$.

(a) **No**. El predicado expuesto dice que existe un elemento positivo o cero, que es mayor que todos los anteriores a él, sean positivos o negativos. Pero no dice nada sobre la ordenación de los elementos negativos.

(b) **No**. El orden de los elementos negativos es creciente.

(c) **Sí**. Ya que La variable $K$ aparece ligada al primer cuantificador y por lo tanto la sustitución no se puede realizar, dejando $\alpha$ tal y como está.

(d) **Sí**. Las apariciones de la variable libre $N$ han sido sustituidas por $i$. Además, la variable cuantificada $J$ ha sido cambiada por otra variable ($L$), lo cual es independiente de cualquier sustitución.

**Pregunta 2**: Dado $S :$ do $i \neq 9 \rightarrow i := i + 1 | i = 3 \rightarrow i := 0$ od

(a) $\{T\}S\{i = 9\}$

×(b) $\{3 < i \wedge i < 9\}S\{i = 9\}$

×(c) Es no determinista para cualquier estado inicial $i \leq 3$

×(d) $T\{S\}i = 9$ (corrección parcial)

(a) **No**, $\{T\}S\{i = 9\}$ implica que $S$ termina siempre (para cualquier estado inicial) y esto no es verdad. Por ejemplo, si $i = 10$ al principio, el bucle se queda aumentando la $i$ indefinidamente.

(b) **Sí**, para los valores iniciales $i = 4; ...; 8$, el bucle siempre ejecuta la primera rama $i := i + 1$ y por tanto acaba llegando a $i = 9$, lo que lo hace terminar. Es necesario destacar que para terminar $\neg BB \equiv \neg(i \neq 9 \vee i = 3) \equiv (i = 9 \wedge i \neq 3) \equiv (i = 9)$. Para valores mayores que $9$ el bucle no termina, mientras que para valores $i \leq 3$ puede no terminar (si decide ejecutar siempre la segunda rama cada vez que $i$ alcanza $3$).

(c) **Sí**, dado que si empieza con $i \leq 3$ siempre llegará un momento en que los dos guardas $i \neq 9$ e $i = 3$ se vuelvan ciertos y deba elegir uno u otro.

(d) **Sí**, dado que, $\neg BB \equiv (i = 9)$, esto es, siempre que el bucle termina, lo hace con $i = 9$.

**Pregunta 3**: El programa **do** $B_1 \rightarrow$ **if** $B_2 \rightarrow S_1 | \neg B_2 \rightarrow S_2$ **fi od** siempre equivale a:

(a) **do** $B_1 \wedge B_2 \rightarrow S_1 | B_1 \wedge \neg B_2 \rightarrow S_2$ **od**

×(b) **do** $B_1$ **cand** $B_2 \rightarrow S_1 | B_1$ **cand** $\neg B_2 \rightarrow S_2$ **od**

(c) **do** $B_2$ **cand** $B_1 \rightarrow S_1 | \neg B_2$ **cand** $B_1 \rightarrow S_2$ **od**

(d) **do** $B_1 \rightarrow$ **if** $\neg B_2 \rightarrow S_2 | B_1 \rightarrow S_1$ **fi od**

(a) **No**. Porque podría darse el caso que $B_2$ fuese no definido. Como no se ha protegido $B_1$ con un **cand**, si $B_1$ es cierto ambas ramas devuelven no definido y el **do** aborta.

(b) **Sí**. Al añadir el **cand** se evita el problema que acabamos de mencionar.

(c) **No**. Porque ahora en el **cand** estamos evaluando primero la condición que puede devolver no definido.

(d) **No**. Porque las ramas del **if** son distintas a las del programa original.

**Pregunta 4**: Dado el predicado $\alpha : (\forall I \in [0:3] \wedge b[I] = 0 : k = 0)$

×(a) $k$ está libre en $\alpha$.

(b) $\alpha$ equivale a $k = 0$.

×(c) $\alpha_p^I = \alpha$.

×(d) $\alpha$ equivale a $(\forall I \in [0:3] : b[I] = 0) \Rightarrow k = 0$.

(a) **Sí**, ya que $k$ aparece en $\alpha$ y no está cuantificada.

(b) **No**, si expandimos el $\forall I$ para todos sus valores, la expresión que obtenemos es, en realidad:

$$(b[0] = 0 \Rightarrow k = 0) \wedge (b[1] = 0 \Rightarrow k = 0) \wedge (b[2] = 0 \Rightarrow k = 0) \wedge (b[3] = 0 \Rightarrow k = 0)$$

lo que no equivale a $k = 0$. Por ejemplo, si $k \neq 0$ pero el array $b[0:3]$ no contiene ceros, la expresión de anterior sería cierta.

(c) **Sí**, dado que $\alpha$ no contiene apariciones libres de $I$.

(d) **Sí**. es una propiedad de los cuantificadores llamada *término constante*. En todo caso, se puede ver fácilmente por la explicación dada antes para la segunda respuesta.

**Pregunta 5**: (2 puntos) Calcula y simplifica: $wp(\text{"}b[i], b[j] := b[b[j]], b[k]; b[i] := b[k]\text{"}, b[i] \neq b[j])$.

$wp(\text{"}b[i], b[j] := b[b[j]], b[k]; b[i] := b[k]\text{"}, b[i] \neq b[j])$

$\equiv wp(\text{"}b[i], b[j] := b[b[j]], b[k]\text{"}, wp(b[i] := b[k]\text{"}, b[i] \neq b[j]))$

Primero calcularemos:

$wp(\text{"}b[i] := b[k]\text{"}, b[i] \neq b[j])$

$\equiv enrango(k, b)$ cand $dominio(b[k])$ cand $enrango(i, b)$ cand $(b[i] \neq b[j])^b_{(b;i:b[k])}$

Desarrollando el último término obtenemos:

$(b[i] \neq b[j])^b_{(b;i:b[k])}$

$\equiv (b; i : b[k])[i] \neq (b; i : b[k])[j]$

$\equiv b[k] \neq (b; i : b[k])[j]$

$\equiv (i = j \wedge b[k] \neq b[k]) \vee (i \neq j \wedge b[k] \neq b[j])$

$\equiv (i = j \wedge F) \vee (i \neq j \wedge b[k] \neq b[j]) \equiv (i \neq j \wedge b[k] \neq b[j])$

A continuación sustituimos en la expresión original y resolvemos:

$wp(\text{"}b[i], b[j] := b[b[j]], b[k]\text{"}, wp(\text{"}b[i] := b[k]\text{"}, b[i] \neq b[j]))$

$\equiv wp(\text{"}b[i], b[j] := b[b[j]], b[k]\text{"}, enrango(k, b) \text{ cand } dominio(b[k]) \text{ cand } enrango(i, b) \text{ cand } (i \neq j \wedge b[k] \neq b[j]))$

$\equiv enrango(j, b) \text{ cand } dominio(b[j]) \text{ cand } enrango(b[j], b) \text{ cand } enrango(k, b) \text{ cand } dominio(b[b[j]]) \text{ cand } dominio(b[k])$
$\text{ cand } enrango(j, b) \text{ cand } enrango(i, b) \text{ cand } dominio(b[k]) \text{ cand } enrango(i, b) \text{ cand } enrango(k, b) \text{ cand}$
$(i \neq j \wedge b[k] \neq b[j])^b_{(b;i:b[b[j]];j:b[k])}$

$\equiv dominio(b[b[j]]) \text{ cand } dominio(b[k]) \text{ cand } enrango(j, b) \text{ cand } enrango(i, b) \text{ cand } enrango(k, b)$
$\text{ cand } (i \neq j \wedge b[k] \neq b[j])^b_{(b;i:b[b[j]];j:b[k])}$

Resolviendo el último término:

$(i \neq j \wedge b[k] \neq b[j])^b_{(b;i:b[b[j]];j:b[k])}$

$\equiv (i \neq j \wedge (b; i : b[b[j]]; j : b[k])[k] \neq (b; i : b[b[j]]; j : b[k])[j])$

$\equiv (i \neq j \wedge (b; i : b[b[j]]; j : b[k])[k] \neq b[k])$

$\equiv (j = k \wedge i \neq j \wedge b[k] \neq b[k]) \vee (j \neq k \wedge i \neq j \wedge (b; i : b[b[j]])[k] \neq b[k])$

$\equiv (j = k \wedge i \neq j \wedge F) \vee (j \neq k \wedge i \neq j \wedge (b; i : b[b[j]])[k] \neq b[k])$

$\equiv (j \neq k \wedge i \neq j \wedge (b; i : b[b[j]])[k] \neq b[k])$

$\equiv (i = k \wedge j \neq k \wedge i \neq j \wedge b[b[j]] \neq b[k]) \vee (i \neq k \wedge j \neq k \wedge i \neq j \wedge b[k] \neq b[k])$

$\equiv (i = k \wedge j \neq k \wedge i \neq j \wedge b[b[j]] \neq b[k]) \vee (i \neq k \wedge j \neq k \wedge i \neq j \wedge F)$

$\equiv (i = k \wedge j \neq k \wedge i \neq j \wedge b[b[j]] \neq b[k])$

Simplificando y sustituyendo en la expresión de partida la solución queda como sigue:

$\equiv dominio(b[b[j]]) \text{ cand } dominio(b[k]) \text{ cand } enrango(j, b) \text{ cand } enrango(i, b) \text{ cand } enrango(k, b)$
$\text{ cand } (i = k \wedge j \neq k \wedge b[b[j]] \neq b[k])$

**Pregunta 6**: (4,5 puntos) Dado un array $b[0 : n-1] : integer$ con $n > 0$ y sin valores repetidos, se desea incrementar en $1$ todos los valores menos el mínimo. Resolver el problema con un único bucle. Se pide:

6.1 Establecer una precondición y una postcondición adecuadas

6.2 Fijar una invariante y una función cota

6.3 Escribir el programa y anotarlo; y

6.4 Demostrar su corrección *total*

6.1 Establecer una precondición y una postcondición adecuadas

$\{Q : \ n > 0 \wedge b[0 : n-1] : integer \wedge \neg(\exists I, J \in [0, n\rangle : I \neq J \wedge b[I] = b[J]) \wedge b = B\}$

Es necesario destacar que no hay que exigir la existencia de un valor mínimo: eso está garantizado simplemente por tener el array al menos un elemento ($n > 0$). Para expresar la postcondición, necesitamos referirnos a la posición donde se encuentra el valor mínimo del array original, llamémosle M, esto es, $(\forall I \in [0, n\rangle : B[M] \leq B[I])$ o de forma abreviada $B[M] \leq B[0 : n-1]$.

Ahora bien, dado que en la pre y postcondición no debemos usar *ninguna variable libre que no aparezca en el enunciado del problema*, necesitaremos referirnos a $M$ con un cuantificador existencial, es decir, $(\exists M \in [0,n\rangle : B[M] \leq B[0:n-1])$. Además, queremos que el mínimo no cambie de valor ($b[M] = B[M]$) y que las demás posiciones se incrementen en 1, esto es $(\forall I \in [0,n\rangle \wedge I \neq M : b[I] = B[I]+1)$. La postcondición final sería, por tanto:

$$\{R:\ (\exists M \in [0:n\rangle : B[M] \leq B[0:n-1] \wedge b[M] = B[M] \wedge (\forall I \in [0,n\rangle \wedge I \neq M : b[I] = B[I]+1)\ \}$$

6.2 Fijar una invariante y una función cota

Para fijar la invariante se puede aplicar la técnica habitual de reemplazar $n$ por una nueva variable $i$ en $R$:
$\{P:\ (\exists M \in [0:i\rangle : B[M] \leq B[0:i-1] \wedge b[M] = B[M] \wedge (\forall I \in [0,i\rangle \wedge I \neq M : b[I] = B[I]+1)\ \}$

Esto estaría indicando que, el valor más pequeño encontrado hasta la posición $i-1$ estaría en $M$ y además, ya habríamos sumado $1$ a todas las demás posiciones del array recorridas hasta ese momento. Por otro lado, parece evidente que nos convendrá convertir el $\exists M$ en una nueva variable libre, llamémosle $m$, que nos indique la posición del mínimo de la parte recorrida hasta el momento. Es necesario destacar que $m$ se moverá entre $0$ e $i-1$. Por último, dado que de la posición $i$ en adelante, el array aún no ha sido modificado, también añadiremos $b[i:n-1] = B[i:n-1]$. De este modo, la invariante y la función cota serían las siguientes:

$$\{P:\ 0 \leq m < i \leq n \wedge B[m] \leq B[0:i-1] \wedge b[M] = B[M] \wedge (\forall I \in [0,i\rangle \wedge I \neq M : b[I] = B[I]+1)$$
$$\wedge b[i:n-1] = B[i:n-1]\ \}$$

$$\{t:\ n-i\}$$

6.3 Escribir el programa y anotarlo:

Para construir el programa, cuando $i$ tome el valor $n$ podremos obtener directamente $R$ a partir de $P$. Obsérvese que el $\exists M$ en $R$ no es un problema: la variable $m$ cumplirá en ese momento lo exigido para $M$. Así pues, tendremos un bucle del estilo **do** $i \neq n \rightarrow$ .... **od** . Para la inicialización, buscaremos hacer cierta $P$ de la forma más directa posible. Es sencillo comprobar que eso se consigue con $i, m := 1, 0$ (recuérdese que el array tiene al menos un elemento). Para poder avanzar, el bucle deberá comprobar si el siguiente elemento $b[i]$ es mayor o menor que el mínimo actual $b[m]$. Si $b[i] > b[m]$ sabemos que $b[i]$ ya no será el mínimo del array y, por tanto, podemos incrementarlo en $1$. Por el contrario, si $b[i] < b[m]$, tendremos que $b[m]$ no era el mínimo, por lo que es necesario incrementarlo, a la vez que anotamos el nuevo mínimo haciendo $m := i$. Así, tendríamos el siguiente programa:

$$\begin{array}{l}
\{Q\} \\
S_0 : i, m := 1, 0; \\
\{P\} \\
\text{do } B_1 : i \neq n \text{ cand } b[i] > b[m] \rightarrow \{P \wedge B_1\} S_1 : b[i], i := b[i]+1, i+1 \\
\ \ \| \ B_2 : i \neq n \text{ cand } b[i] < b[m] \rightarrow \{P \wedge B_2\} S_2 : b[m], m, i := b[m]+1, i, i+1 \\
\{P\} \\
\text{od} \\
\{P \wedge \neg BB\} \\
\{R\}
\end{array}$$

6.4 Demostrar su corrección *total*

*a*) $Q \Rightarrow wp(S_0, P)$

El $wp$ resultante sería:

$$\overbrace{0 \leq 0 < 1}^{T} \leq n \wedge B[0] \leq B[0:0] \wedge b[0] = B[0] \wedge (\forall I \in [0,1\rangle \wedge I \neq 0 : b[I] = B[I]+1) \wedge b[1:n-1] = B[1:n-1]$$

$$\equiv 1 \leq n \wedge \overbrace{B[0] \leq B[0]}^{T\ (n>0)} \wedge b[0] = B[0] \wedge \underbrace{(\forall I \in \overbrace{[0,1\rangle}^{\emptyset} \wedge I \neq 0 : b[I] = B[I]+1)}_{T} \wedge b[1:n-1] = B[1:n-1]$$

$\equiv 1 \leq n \wedge b[0] = B[0] \wedge b[1:n-1] = B[1:n-1]$

$\equiv n > 0 \wedge b[0:n-1] = B[0:n-1]$ que se deduce directamente de $Q$.

*b*) $\{P \wedge B_i\} S_i \{P\}$ para $1 \leq i \leq n$. O lo que es lo mismo $P \wedge B_i \Rightarrow wp(S_i, P)$, lo que nos lleva a demostrar:

1) $P \wedge B_1 \Rightarrow wp(S_1, P) \equiv P \wedge B_1 \Rightarrow wp(\text{"}b[i], i := b[i]+1, i+1\text{"}, P)$
El $wp$ resultante sería: $P^{b,i}_{b',i+1}$ donde $b' = (b; i : b[i]+1)$. Si continuamos desarrollando:

$$\equiv 0 \leq m < i+1 \leq n \quad (1)$$

$$\wedge B[m] \leq B[0:i] \quad (2)$$

$$\wedge b'[m] = B[m] \quad (3)$$

$$\wedge (\forall I \in [0, i+1\rangle \wedge I \neq m : b'[I] = B[I]+1) \quad (4)$$

$$\wedge b'[i+1:n-1] = B[i+1:n-1] \quad (5)$$

Probaremos que $P \wedge B_1$ implica cada una de estas fórmulas:
(1) $0 \leq m$ estaba en $P$, mientras que $m < i+1$ se obtiene de $m < i$ también en $P$. Por otro lado, de $i \leq n$ en $P$ y de $i \neq n$ en $B_1$ se obtiene que $i < n$, es decir $i+1 \leq n$.
(2) Se puede separar en $B[m] \leq B[0:i-1] \wedge B[m] < B[i]$. Por otro lado, en $P$ tenemos $b[m] = B[m]$ y $b[i] = B[i]$, mientras que por $B_1$ $b[i] > b[m]$. Por lo tanto de aquí, se puede deducir que $B[i] > B[m]$.
(3) $b'[m] = (b; i : b[i]+1)[m] = b[m]$ puesto que sabemos que $m < i$ por $P$. Así pues, (3) es equivalente a $b[m] = B[m]$ que se tiene directamente en $P$.
(4) Aislando el caso $I = i$ obtenemos:

$$(\forall I \in [0, i\rangle \wedge I \neq m : (b; i : b[i]+1)[I] = B[I]+1) \wedge (b; i : b[i]+1)[i] = B[i]+1$$

$$\equiv \underbrace{(\forall I \in [0, i\rangle \wedge I \neq m : b[I] = B[I]+1)}_{\text{en } P} \wedge b[i]+1 = B[i]+1$$

$$\equiv \underbrace{(\forall I \in [0, i\rangle \wedge I \neq m : b[I] = B[I]+1)}_{\text{en } P} \wedge \underbrace{b[i] = B[i]}_{P}$$

(5) Como el rango de posiciones no toca la posición modificada $i$, la fórmula equivale a $b[i+1 : n-1] = B[i+1 : n-1]$ que se deduce de $b[i : n-1] = B[i : n-1]$ en $P$.

2) $P \wedge B_2 \Rightarrow wp(S_2, P) \equiv P \wedge B_2 \Rightarrow wp(\text{"}b[m], m, i := b[m]+1, i, i+1\text{"}, P)$

El $wp$ resultante sería:

$wp(\text{"}b[m], m, i := b[m+1], i, i+1\text{"}, P) \equiv P^{b,m,i}_{b',i,i+1}$

donde, en este caso, $b' = (b; m : b[m]+1)$

$$\equiv 0 \leq i < i+1 \leq n \quad (6)$$

$$\wedge B[i] \leq B[0:i] \quad (7)$$

$$\wedge b'[i] = B[i] \quad (8)$$

$$\wedge (\forall I \in [0 : i+1\rangle \wedge I \neq i : b'[I] = B[I]+1) \quad (9)$$

$$\wedge b'[i+1:n-1] = B[i+1:n-1] \quad (10)$$

Probaremos aquí también que $P \wedge B_2$ implica cada una de estas fórmulas:
(6) Por un lado, en $P$ tenemos $0 \leq i \leq n$. Como $i < i+1$ es siempre cierto, nos falta probar que $i+1 \leq n$. Esto último se deduce de $P$ y de $i \neq n$ en $B_2$.
(7) Como $B[i] \leq B[i]$ es tautología, (7) equivale a $B[i] \leq B[0:i-1]$. Ahora bien, es necesario destacar que en $P$ tenemos $b[m] = B[m]$ y $b[i] = B[i]$. Como en $B_2$ tenemos $b[i] < b[m]$, esto implica $B[i] < B[m]$. Por otro lado, en $P$ tenemos $B[m] \leq B[0:i-1]$, con lo que llegamos a $B[i] < B[m] \leq B[0:i-1]$, es decir, $B[i] < B[0:i-1]$ que es más fuerte que (7).
(8) Como tenemos que $m < i$ en $P$, $b'[i] = (b; m : b[m]+1)[i] = b[i]$. Por tanto, (8) es equivalente a $b[i] = B[i]$ que se deduce de $P$.
(9) es equivalente a:

$$\equiv (\forall I \in [0:i] \wedge I \neq i : b'[I] = B[I] + 1)$$

$$\equiv (\forall I \in [0:i\rangle : b'[I] = B[I] + 1)$$

Como en $P$ tenemos que $0 \leq m < i$, podemos separar el caso $I = m$ de la siguiente forma:

$$\equiv (\forall I \in [0:i\rangle \wedge I \neq m : b'[I] = B[I] + 1) \wedge b'[m] = B[m] + 1$$

$$\equiv (\forall I \in [0:i\rangle \wedge I \neq m : b[I] = B[I] + 1) \wedge b[m] + 1 = B[m] + 1$$

$$\equiv (\forall I \in [0:i\rangle \wedge I \neq m : b[I] = B[I] + 1) \wedge b[m] = B[m]$$

que se obtiene directamente de $P$.
(10) Como tenemos $m < i$ en $P$, podemos concluir que $b'[i+1:n-1] = b[i+1:n-1]$ y por tanto (10) equivale simplemente a $b[i+1:n-1] = B[i+1:n-1]$ que se deduce de $P$.

*c*) $P \wedge \neg BB \Rightarrow R$

Calculamos primero $BB$:

$$BB \equiv B_1 \vee B_2$$

$$\equiv i \neq n \text{ cand } b[i] > b[m] \vee i \neq n \text{ cand } b[i] < b[m]$$

$$\equiv i \neq n \text{ cand } (b[i] > b[m] \vee b[i] < b[m])$$

$$\equiv i \neq n \text{ cand } b[i] \neq b[m]$$

Por tanto, $\neg BB \equiv i = n \text{ cor } b[i] = b[m]$.

Ahora bien, cuando consideramos $P \wedge \neg BB$, como tenemos en $P$ que $b[i] = B[i]$, $b[m] = B[m]$ y $m < i$, si $b[i] = b[m]$ tendríamos que $B[i] = B[m]$ para dos posiciones diferentes del array original $B$, lo que contradice $Q$. Por tanto, $P \wedge \neg BB$ equivale simplemente a $P \wedge i = n$ y es fácil ver que esto implica directamente $R$ cuando se toma $M = m$ para el cuantificador existencial.

*d*) $P \wedge BB \Rightarrow (n - i > 0)$
Es inmediato ya que $BB$ implica $i \neq n$ mientras que $P$ implica $i \leq n$, de donde se obtiene $i < n$ o, lo que es lo mismo $0 < n - i$.

*e*) $\{P \wedge B_i\} t_1 := t; S_i \{t < t_1\}$ para $1 \leq i \leq 2$.

Calculamos el $wp$ correspondiente para la primera rama:

1) $S_1 : b[i], i := b[i] + 1, i + 1$

$wp(\text{"}t_1 := t; wp(\text{"}b[i], i := b[i] + 1, i + 1\text{"}, n - i < t_1))$

$((n - i < t_1)^{b,i}_{(b;i:b[i]+1),i+1})^{t_1}_{n-i}$

$\equiv n - (i + 1) < t_1)^{t_1}_{n-i}$

$\equiv n - i - 1 < n - i \equiv -1 < 0 \equiv T$

2) Para la segunda rama ($S_2$), el desarrollo es idéntico al de la primera rama dado que en la cota sólo interviene la variable $i$ y ésta se modifica del mismo modo.

**Pregunta 7**: (1,5 puntos) Dada la precondición $Q : a > 0 \wedge b > 0$ y el programa **do** $a > b \rightarrow a := a - b | b > a \rightarrow b := b - a$ **od** , fijar invariante y cota para demostrar que el programa termina, y demostrar que dicho programa termina.

En primer lugar vamos a hacer un estudio de cómo varían *a* y *b* en cada rama.

| | a | b |
|---|---|---|
| Rama 1 | $\downarrow$ | $=$ |
| Rama 2 | $=$ | $\downarrow$ |

Como ambas variables, cuando cambian, lo hacen siempre en la misma dirección, en principio una posible cota sería $a+b$ (dado que ambas decrecen). Sólo nos faltaría garantizar que esa cota se mantiene por encima de cero al entrar en el bucle. La condición de entrada es $BB \equiv (a > b \vee a < b) \equiv (a \neq b)$, es decir, que las variables sean distintas. Ambas comienzan siendo estrictamente mayores que cero, y siempre que son distintas, restamos la menor a la mayor, por lo que ambas quedan con valor positivo. Tomaremos como invariante y función cota las siguientes:

$\{P : \ a > 0 \wedge b > 0 \ \}$
$\{t : \ a + b\}$

Demostramos ahora que el programa termina:

1 $Q \Rightarrow P$

Probamos que la invariante es cierta al empezar. Esto es obvio, dado que $Q = P$.

Probamos ahora que $P$ se mantiene en cada iteración del bucle.

1.1 $P \wedge a > b \Rightarrow wp(\text{"}a := a - b\text{"}, P)$

Calculamos el $wp$: $wp(\text{"}a := a - b\text{"}, a > 0 \wedge b > 0)$

$\equiv a - b > 0 \wedge b > 0$

$\equiv a > b \wedge b > 0$ que es cierto por $B_1 : a > b$ y por $P : b > 0$.

1.2 $P \wedge b > a \Rightarrow wp(\text{"}b := b - a\text{"}, P)$

Calculamos el $wp$: $wp(\text{"}b := b - a\text{"}, a > 0 \wedge b > 0)$

$\equiv a > 0 \wedge b - a > 0$

$\equiv a > 0 \wedge b > a$ que es cierto por $B_2 : b > a$ y por $P : a > 0$.

4 $P \wedge BB \Rightarrow t > 0$
En realidad, de $P$ ya se deduce que $a + b > 0$.

5.1 $\{P \wedge B_i\} t_1 := t; S_i \{t < t_1\}$ para $1 \leq i \leq 2$.

Calculamos el $wp$ correspondiente para cada rama:

1) $S_1 : a := a - b$

$wp(\text{"}t_1 := t; wp(\text{"}a := a - b\text{"}, t < t_1))$

$((a + b < t_1)^{a}_{a-b})^{t_1}_{a+b}$

$\equiv (a - b + b < t_1)^{t_1}_{a+b}$

$\equiv a < a + b$

$\equiv 0 < b$, que es cierto por P.

2) $S_2 : b := b - a$

$wp(\text{"}t_1 := t; wp(\text{"}b := b - a\text{"}, t < t_1))$

$((a + b < t_1)^{b}_{b-a})^{t_1}_{a+b}$

$\equiv (a + b - a < t_1)^{t_1}_{a+b}$

$\equiv b < a + b$

$\equiv 0 < a$ que es cierto por P.