37. (Junio 2008) Predicados par(x) e interpar(C(1..r), (c_1, c_2, …, c_r), D(1..r), (d_1, d_2, …, d_r), pos) y programa que, dados dos vectores de enteros A(1..n) y B(1..n) siempre que los elementos de una posición sean pares tanto en A(1..n) como en B(1..n), los intercambia -- #

a) **par(x)** ≡ $\{x \bmod 2 = 0\}$

b) **interpar(C(1..r), (c_1, c_2, …, c_r), D(1..r), (d_1, d_2, …, d_r), pos)** ≡
$\forall k\, (1 \le k \le pos \wedge par(c_k) \wedge par(d_k) \rightarrow (C(k) = d_k \wedge D(k) = c_k)) \wedge$
$\forall k\, (1 \le k \le pos \wedge (\neg par(c_k) \vee \neg par(d_k)) \rightarrow (C(k) = c_k \wedge D(k) = d_k))$

c)

(1) {Precondición} ≡ $\{n \ge 1 \wedge \forall k\, (1 \le k \le n \rightarrow (A(k) = a_k \wedge B(k) = b_k))\}$

En la precondición se indica que los vectores A y B tendrán por lo menos un elemento y que los valores iniciales de A y B los representaremos mediante *a*’s y b’s minúsculas con los correspondientes subíndices.

(2) {Aserción intermedia} ≡ $\{(1) \wedge i = 1\}$

(9) {Postcondición} ≡
$\{interpar(A(1..n), (a_1, a_2, \dots, a_n), B(1..n), (b_1, b_2, \dots, b_n), n)\}$

En la poscondición se indica que se han hecho todos los cambios hasta la posición n incluida, es decir, se ha recorrido todo el vector y se han hecho todos los cambios necesarios.

(3) {Invariante} ≡ $\{(1 \le i \le n + 1) \wedge$
$interpar(A(1..n), (a_1, a_2, \dots, a_n), B(1..n), (b_1, b_2, \dots, b_n), i - 1)\}$

En el invariante se indica que se han hecho todos los cambios hasta la posición i – 1 incluida, es decir, estamos situados en la posición i pero todavía no se ha analizado lo que ocurre en la posición i, por ello lo que sí está analizado es hasta la posición i – 1.

(4) {Aserción intermedia} ≡ $\{(1 \le i \le n) \wedge$
$interpar(A(1..n), (a_1, a_2, \dots, a_n), B(1..n), (b_1, b_2, \dots, b_n), i - 1)\}$

Por haber entrado en el while sabemos que se ha cumplido la condición del while y que i no es n + 1.

(5) {Aserción intermedia} ≡ $\{(1 \le i \le n) \wedge par(A(i)) \wedge par(B(i)) \wedge A(i) = a_i \wedge B(i) = b_i \wedge interpar(A(1..n), (a_1, a_2, \dots, a_n), B(1..n), (b_1, b_2, \dots, b_n), i - 1)\}$

Por haber entrado en la rama then del if sabemos que A(i) y B(i) son pares y que todavía coinciden con los valores iniciales a_i y b_i.
El punto (5) se puede abreviar como sigue:
(5) {Aserción intermedia} ≡ $\{(4) \wedge par(A(i)) \wedge par(B(i)) \wedge A(i) = a_i \wedge B(i) = b_i\}$

(6) {Aserción intermedia} ≡ {$(1 \le i \le n) \wedge$
interpar(A(1..n), $(a_1, a_2, \ldots, a_n)$, B(1..n), $(b_1, b_2, \ldots, b_n)$, $i - 1) \wedge$
par(A(i)) $\wedge$ par(B(i)) $\wedge$ $A(i) = a_i \wedge B(i) = b_i \wedge aux = A(i)$}

Tras ejecutarse la asignación aux : = A(i); sabemos que ahora en aux tenemos a_i que sigue coincidiendo con A(i). Pero de momento los cambios completos están hechos hasta la posición i – 1, en la posición i estamos a medias por lo que en el predicado interpar seguimos poniendo i – 1.

El punto (6) se puede abreviar como sigue:
(6) {Aserción intermedia} ≡ {(5) $\wedge$ aux = A(i)}

(7) {Aserción intermedia} ≡ {$(1 \le i \le n) \wedge$ par(a_i) $\wedge$ par(b_i) $\wedge$ $A(i) = B(i) = b_i$ $\wedge$ interpar(A(1..n), $(a_1, a_2, \ldots, a_n)$, B(1..n), $(b_1, b_2, \ldots, b_n)$, $i - 1) \wedge$
$aux = a_i$}

Tras ejecutarse la asignación A(i) : = B(i); sabemos que ahora en A(i) tenemos b_i. Ahora aux ya no es igual a A(i). De momento los cambios completos están hechos hasta la posición i – 1, en la posición i estamos a medias por lo que en el predicado interpar seguimos poniendo i – 1.

El punto (7) se puede abreviar como sigue:
(7) {Aserción intermedia} ≡ {(4) $\wedge$ par(a_i) $\wedge$ par(b_i) $\wedge$ $A(i) = B(i) = b_i$ $\wedge$ $aux = a_i$}
Por tanto hay que utilizar (4) ya que el (5) y el (6) no sirven.

(11) {Aserción intermedia} ≡ {$(1 \le i \le n) \wedge$ par(a_i) $\wedge$ par(b_i) $\wedge$ $A(i) = b_i \wedge$ $B(i) = a_i$ $\wedge$ interpar(A(1..n), $(a_1, a_2, \ldots, a_n)$, B(1..n), $(b_1, b_2, \ldots, b_n)$, $i - 1) \wedge$
$aux = a_i$}

Tras ejecutarse la asignación B(i) : = aux; sabemos que ahora en B(i) tenemos a_i. Ahora B(i) ya no es igual a A(i).

El punto (11) se puede abreviar como sigue:
(11) {Aserción intermedia} ≡ {(4) $\wedge$ par(a_i) $\wedge$ par(b_i) $\wedge$ $A(i) = b_i \wedge B(i) = aux$ $\wedge$ $aux = a_i$}

Pero hay otra opción para el punto (11). La cuestión es que el predicado
interpar(A(1..n), $(a_1, a_2, \ldots, a_n)$, B(1..n), $(b_1, b_2, \ldots, b_n)$, $i - 1)$
dice que en las posiciones que van desde la posición 1 hasta la i – 1 ya se han hecho los intercambios necesarios y por la formula par(a_i) $\wedge$ par(b_i) $\wedge$ $A(i) = b_i \wedge B(i) = aux \wedge aux = a_i$ ya sabemos que también se ha hecho el intercambio de la posición i. Por tanto tenemos que ya se han hecho los intercambios necesarios en las posiciones que van desde la posición 1 hasta la i y eso se puede expresar poniendo i (en vez de i – 1) en el predicado interpar:

(11) {Aserción intermedia} ≡ {$(1 \le i \le n) \wedge$ par(a_i) $\wedge$ par(b_i) $\wedge$
interpar(A(1..n), $(a_1, a_2, \ldots, a_n)$, B(1..n), $(b_1, b_2, \ldots, b_n)$, $i) \wedge$

aux = a_i

Ahora ya se han completado los cambios hasta la posición i incluida, pero no es necesario poner A(i) = b_i y B(i) = a_i ya que eso está dicho mediante el predicado interpar al poner i.

(8) {Aserción intermedia} ≡ {$(1 \le i \le n) \wedge$
interpar(A(1..n), ($a_1, a_2, \ldots, a_n$), B(1..n), ($b_1, b_2, \ldots, b_n$), i)}

La diferencia entre (11) y (8) es que en (11) sabemos que se ha ido por la rama **then** y por tanto podemos asegurar que par(a_i) ∧ par(b_i) ∧ aux = a_i, pero en (8) no sabemos si se ha ido por la rama then o si no se ha cumplido la condición del if y por ello no podemos asegurar que se cumpla par(a_i) ∧ par(b_i) ∧ aux = a_i.

(12) {Aserción intermedia} ≡ {$(2 \le i \le n + 1) \wedge$
interpar(A(1..n), ($a_1, a_2, \ldots, a_n$), B(1..n), ($b_1, b_2, \ldots, b_n$), i – 1)}

Tras ejecutar la asignación i : = i + 1; al haberse incrementado el valor de i cambian los límites del intervalo de i y también hay que decir que los cambios están hechos hasta la posición i – 1.

(10) E = n + 1 – i

La expresión cota nos indica, siempre que estemos en el punto donde se cumple el invariante, cuántas vueltas quedan por dar. Es "el último valor que tomará i" menos "i". La expresión E es al fin y al cabo la distancia entre n + 1 y la variable i. Cuando i crece, la distancia disminuye y el número de vueltas pendientes decrece.

Las partes coloreadas resaltan los cambios que hay de una aserción a otra.