45. (Junio 2010) Predicados par(x) e interdos(C(1..r), (c_1, c_2, …, c_r), pos) y programa que dado un vector A(1..n), intercambia los elementos de las posiciones contiguas de dos en dos (posiciones 1 y 2, 3 y 4, 5 y 6, etc.). Si el número de elementos de A(1..n) es impar (n impar), el último elemento no se moverá. -- #

a) par(x) ≡ {x mod 2 = 0}

b) **interdos(C(1..r), (c_1, c_2, …, c_r), pos)** ≡

$\{(0 \le pos \le r) \wedge$
$\forall k\,(1 \le k \le pos \rightarrow ((par(k) \rightarrow (C(k) = c_{k-1} \wedge C(k-1) = c_k)) \wedge$
$\wedge\ ((k = pos \wedge \neg par(k)) \rightarrow C(k) = c_k)))\}$

Otra opción para b)

interdos(C(1..r), (c_1, c_2, …, c_r), pos) ≡

$\{(0 \le pos \le r) \wedge$
$\forall k\,((1 \le k \le pos \wedge par(k)) \rightarrow (C(k) = c_{k-1} \wedge C(k-1) = c_k)) \wedge$
$(\neg par(pos) \rightarrow C(k) = c_k)\}$

c) Las aserciones se darán en el orden en el que es más natural formularlas:

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

En la precondición se indica que el vector A tendrá por lo menos un elemento y que los valores iniciales de A los representaremos mediante *a*'s minúsculas con los correspondientes subíndices.

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

(9) {Postcondición} ≡
{interdos(A(1..n), (a_1, a_2, …, a_n), n)}

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

(3) {Invariante} ≡ {$(1 \le i \le n + 1) \wedge$ interdos(A(1..n), (a_1, a_2, …, a_n), i – 1)}

En el invariante se indica que se han hecho los intercambios posibles hasta la posición i – 1. Recordemos que por definición del predicado *interdos*, si i – 1 es par entonces se han intercambiado las posiciones 1 y 2, 3 y 4, etc. hasta i – 2 e i – 1. En cambio si i – 1 es impar, se han intercambiado las posiciones 1 y 2, 3 y 4, etc hasta i – 3 e i – 2 pero el elemento de la posición i – 1 no se ha movido.

(4) {Aserción intermedia} ≡ $\{(1 \le i \le n) \wedge$
$\text{interdos}(A(1..n), (a_1, a_2, \ldots, a_n), i - 1)\}$

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} ≡ $\{(1 \le i \le n) \wedge \text{par}(i) \wedge$
$\text{interdos}(A(1..n), (a_1, a_2, \ldots, a_n), i - 1)\}$

Por haber entrado en la rama **then** del **if** sabemos que i es par.

(6) {Aserción intermedia} ≡ $\{(1 \le i \le n) \wedge \text{par}(i) \wedge \text{aux} = A(i - 1) = a_{i-1} \wedge$
$\text{interdos}(A(1..n), (a_1, a_2, \ldots, a_n), i - 1)\}$

Tras ejecutarse la asignación aux : = A(i – 1); sabemos que ahora en aux tenemos lo mismo que en A(i – 1) y que es el valor inicial a_{i-1}. Pero el predicado *interdos* se cumple para i – 1. Con respecto a la posición i estamos a medias.

(7) {Aserción intermedia} ≡
$\{(1 \le i \le n) \wedge \text{par}(i) \wedge \text{aux} = a_{i-1} \wedge A(i - 1) = A(i) = a_i$
$\text{interdos}(A(1..n), (a_1, a_2, \ldots, a_n), i - 1)\}$

Tras ejecutarse la asignación A(i – 1) : = A(i); sabemos que ahora en aux ya no tenemos lo mismo que en A(i – 1) pero sí el valor inicial a_{i-1}. En A(i – 1) ahora tenemos lo mismo que en A(i) que es el valor inicial a_i. El predicado *interdos* se sigue cumpliendo para i – 1. Con respecto a la posición i seguimos estando a medias.

(8) {Aserción intermedia} ≡ $\{(1 \le i \le n) \wedge \text{par}(i) \wedge \text{aux} = a_{i-1} = A(i) \wedge$
$A(i - 1) = a_i \wedge$
$\text{interdos}(A(1..n), (a_1, a_2, \ldots, a_n), i)\}$

Tras ejecutarse la asignación A(i) : = aux; sabemos que ahora en aux y A(i) tenemos lo mismo, el valor inicial a_{i-1}. En A(i – 1) ahora ya no tenemos lo mismo que en A(i). En A(i – 1) tenemos el valor inicial a_i y en A(i) el valor inicial a_{i-1}. Como ya se ha terminado de hacer el intercambio, ahora el predicado *interdos* se cumple para i.

(9) {Aserción intermedia} ≡ {$(1 \le i \le n) \wedge$ interdos(A(1..n), $(a_1, a_2, \ldots, a_n)$, i)}

La diferencia entre (8) y (9) es que en (8) sabemos que se ha ido por la rama **then** y por tanto podemos asegurar que se cumple par(i) $\wedge$ aux = a_{i-1} = A(i) $\wedge$ A(i – 1) = a_i, pero en (9) 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(i) $\wedge$ aux = a_{i-1} = A(i) $\wedge$ A(i – 1) = a_i ya que si no se ha cumplido la condición del if, par(i) $\wedge$ aux = a_{i-1} = A(i) $\wedge$ A(i – 1) = a_i no será cierto.

(11) 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. Cuando el vector se recorre de izquierda a derecha el valor de E será "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.