METODOLOGÍA DE LA PROGRAMACIÓN
EJERCICIOS DEL TEMA 2
ESPECIFICACIÓN Y DOCUMENTACIÓN DE PROGRAMAS

ÍNDICE

a) Fórmulas, predicados y variables libres y ligadas

Para cada enunciado hay que hacer lo siguiente:

- Escribir una aserción o fórmula de primer orden que represente lo expresado por el enunciado.
- Indicar cuáles son las variables libres y cuáles las ligadas en la fórmula.
- Dar el predicado asociado correspondiente.

En cada caso se indica cuál ha de ser el nombre del predicado asociado y si hay que utilizar predicados definidos con anterioridad.

1. **x es un número par (par)**
    - Nombre del predicado: par
    - Hacerlo sin utilizar predicados anteriores.

2. **x es un número impar (impar)**
    - Nombre del predicado: impar
    - Hacerlo de dos maneras:
        (i) Utilizando el predicado del ejercicio 1.
        (ii) Sin utilizar el predicado del ejercicio 1.

3. **x es una potencia entera de w (potencia)**
    Existe un número que es mayor o igual que cero y x es w elevado a ese número.
    - Nombre del predicado: potencia
    - Hacerlo sin utilizar predicados anteriores.

4. **x es una potencia entera de 2 (potdos)**
    - Nombre del predicado: potdos
    - Hacerlo de dos maneras:
        (i) Utilizando el predicado del ejercicio 3.
        (ii) Sin utilizar el predicado del ejercicio 3. (Existe un número que es mayor o igual que cero y x es 2 elevado a ese número)

5. **Todos los elementos de A(1..n) son mayores que x (todosmayores)**
    Todas las posiciones de A(1..n) contienen un valor mayor que x.
    - Nombre del predicado: todosmayores
    - Hacerlo sin utilizar predicados anteriores.

6. Algún elemento del vector A(1..n) es positivo (> 0) (algunpositivo)

En A(1..n) existe al menos una posición que contiene un valor positivo.

- Nombre del predicado: algunpositivo
- Hacerlo sin utilizar predicados anteriores.

7. Todos los elementos del vector A(1..n) son positivos (> 0) (todospositivos)

- Nombre del predicado: todospositivos
- Hacerlo de dos maneras:
  - (i) Utilizando el predicado del ejercicio 5.
  - (ii) Sin utilizar el predicado del ejercicio 5. (Todas las posiciones de A(1..n) contienen un valor positivo)

8. El vector A(1..n) solo contiene dígitos (digitos)

El vector A(1..n) sólo contiene dígitos. (Todas las posiciones de A(1..n) contienen un valor comprendido entre 0 y 9)

- Nombre del predicado: digitos
- Hacerlo sin utilizar predicados anteriores.

9. En A(1..n) hay x elementos que son pares (numpares)

El número de posiciones de A(1..n) que contienen un valor par es x.

- Nombre del predicado: numpares
- Hacerlo utilizando el predicado del ejercicio 1.

10. El vector A(1..n) solo contiene elementos pares (todospares)

- Nombre del predicado: todospares
- Hacerlo de dos maneras:
  - (i) Utilizando el predicado del ejercicio 9.
  - (ii) Utilizando el predicado del ejercicio 1. (Todas las posiciones de A(1..n) contienen un valor par)

11. La variable booleana p denota si todos los elementos de A(1..n) son pares (denotatodospares)

- Nombre del predicado: denotatodospares
- Hacerlo de dos maneras:
  - (i) Utilizando el predicado del ejercicio 10.
  - (ii) Utilizando el predicado del ejercicio 1. (p es True si y solamente si todas las posiciones de A(1..n) contienen un valor par)

12. x aparece v veces entre las posiciones pos1 y pos2 de A(1..n)

Los valores pos1 y pos2 definen una sección en A(1..n) que podría ser vacía, es decir, pos1 es un valor comprendido entre 1 y n + 1 y pos2 es un valor comprendido entre 0 y n, y x aparece v veces entre las posiciones comprendidas entre los valores pos1 y pos2 de A(1..n).

- Nombre del predicado: vecesseccion
- Hacerlo sin utilizar predicados anteriores. (pos1 es un valor comprendido entre 1 y n + 1, pos2 es un valor comprendido entre 0 y n y el número de posiciones de A(1..n) comprendidos entre pos1 y pos2 que contienen x es v)

13. x aparece v veces en A(1..n) (veces)

- Nombre del predicado: veces
- Hacerlo de <u>dos maneras</u>:
  - (i) Utilizando el predicado del ejercicio 12.
  - (ii) Sin utilizar predicados definidos previamente. (el número de posiciones de A(1..n) que contienen x es v)

14. En el vector A(1..n) todos los elementos comprendidos entre pos1 y pos2 son distintos (todosdistintos)

- Nombre del predicado: todosdistintos
- Hacerlo de <u>dos maneras</u>:
  - (i) Utilizando el predicado del ejercicio 12. (pos1 es un valor comprendido entre 1 y n + 1, pos2 es un valor comprendido entre 0 y n y para toda posición de A(1..n) comprendida entre pos1 y pos2 ocurre que el valor de esa posición aparece una vez entre las posiciones pos1 y pos2 de A(1..n))
  - (ii) Sin utilizar predicados definidos previamente. (pos1 es un valor comprendido entre 1 y n + 1, pos2 es un valor comprendido entre 0 y n y para toda posición de A(1..n) comprendida entre pos1 y pos2 ocurre que toda posición de ese intervalo distinta a ella contiene un valor diferente)

15. x aparece más veces que y en A(1..n) (masvecesque)

- Nombre del predicado: masvecesque
- Hacerlo sin utilizar predicados anteriores (el número de posiciones de A(1..n) que contienen el valor x es mayor que el número de posiciones de A(1..n) que contienen el valor y)

16. x e y son distintos y aparecen el mismo número de veces en A(1..n) (mismonumveces)

- Nombre del predicado: mismonumveces
- Hacerlo sin utilizar predicados anteriores. (x e y son distintos y el número de posiciones de A(1..n) que contienen el valor x es igual al número de posiciones de A(1..n) que contienen el valor y)

17. En el vector A(1..n) existen al menos dos posiciones que contienen el mismo valor (almenosdosrep)

- Nombre del predicado: almenosdosrep
- Hacerlo sin utilizar predicados anteriores. (Existe una posición de A(1..n) y existe otra posición distinta de A(1..n) tal que las dos contienen el mismo valor)

18. En el vector A(1..n) existen exactamente dos posiciones que contienen el mismo valor (justodosrep)

- Nombre del predicado: justodosrep
- Hacerlo sin utilizar predicados anteriores. (El número de posiciones de A(1..n) para los cuales existe otra posición distinta de A(1..n) conteniendo el mismo valor es 2)

19. En el vector A(1..n) todos los elementos aparecen al menos dos veces (todosdosvecesomas)

- Nombre del predicado: todosdosvecesomas
- Hacerlo sin utilizar predicados anteriores. (Para toda posición de A(1..n) existe otra posición distinta de A(1..n) que contiene el mismo valor)

20. En el vector A(1..n) todos los elementos aparecen exactamente dos veces (todosjustodosveces)

- Nombre del predicado: todosjustodosveces
- Hacerlo de dos maneras:
  (i) Utilizando el predicado del ejercicio 13. (Para toda posición de A(1..n) ocurre que el elemento de esa posición aparece 2 veces en A(1..n))
  (ii) Sin utilizar predicados definidos previamente. (Para toda posición de A(1..n) ocurre que el número de posiciones de A(1..n) que son distintas a dicha posición y que contienen el mismo valor que dicha posición es 1)

21. x es un número primo (primo)

Un número entero x es primo si es mayor o igual que 1 y si tiene exactamente dos divisores (el 1 no es primo).

- Nombre del predicado: primo
- Hacerlo sin utilizar predicados anteriores. (x es mayor o igual que 1 y el número de valores comprendidos entre 1 y x (ambos inclusive) y que dividen a x es 2)

22. z es igual a la suma de una secuencia de enteros consecutivos que empieza por 1 (sumaconsecutivos)

Ejemplo: 10 = 1 + 2 + 3 + 4

- Nombre del predicado: sumaconsecutivos
- Hacerlo sin utilizar predicados anteriores. (Existe un valor mayor o igual que 1 tal que la suma de los números que van desde 1 hasta ese valor es z)

23. (sumaintervalo)

x y w definen un subintervalo no vacío en el intervalo [1..n] y s es la suma de los elementos de A(1..n) que están entre las posiciones x y w (ambas incluidas).

Ejemplo:

Suponiendo que x = 3 e w = 6, s sería la suma de los elementos que están en negrita.

A

90	22	**1**	**1**	**1**	**-70**	62	9
1	2	3	4	5	6	7	8

s = 1 + 1 + 1 + (-70)

- Nombre del predicado: sumaintervalo
- Hacerlo sin utilizar predicados anteriores. (x es mayor o igual que 1 y menor o igual que w, w es mayor o igual que x y menor o igual que n y la suma de los elementos que están en las posiciones de A(1..n) que van desde la posición x hasta la posición w es igual a s)

24. (sumaintervalo2)

x e y definen un subintervalo no vacío en el intervalo [1..**n**] y **n** es la suma de los elementos de A(1..n) que están entre las posiciones x e y (ambas incluidas).

Ejemplo:

Suponiendo que x = 3 e y = 6, n (esto es, 8) sería la suma de los elementos que están en negrita.

A

90	22	**6**	**-4**	**5**	**1**	62	9
1	2	3	4	5	6	7	8

n = 6 + (-4) + 5 + 1

- Nombre del predicado: sumaintervalo2
- Hacerlo utilizando el predicado del ejercicio 23.

25. s es la suma de los elementos que ocupan las posiciones pares de A(1..n) (sumapospares)

Ejemplo:

A

90	**22**	1	**1**	1	**-70**	62	**9**
1	2	3	4	5	6	7	8

s = 22 + 1 + (-70) + 9

- Nombre del predicado: sumapospares
- Utilizar el predicado del ejercicio 1. (la suma de los elementos que están en las posiciones pares de A(1..n) es igual a s)

26. s es la suma de los elementos pares de A(1..n) (sumapares)

Ejemplo:

A

90	**22**	1	1	1	**-70**	**62**	9
1	2	3	4	5	6	7	8

s = 90 + 22 + (-70) + 62

- Nombre del predicado: sumapares
- Utilizar el predicado del ejercicio 1. (la suma de los elementos de A(1..n) que son pares es igual a s)

27. s es la suma de los elementos de A(1..n) que son mayores que x (sumamayores)

Ejemplo:

Suponiendo que x = 20, s sería la suma de los elementos que están en negrita.

A

90	**22**	1	1	1	-70	**62**	9
1	2	3	4	5	6	7	8

s = 90 + 22 + 62

- Nombre del predicado: sumamayores
- Hacerlo sin utilizar predicados anteriores. (la suma de los elementos de A(1..n) que son mayores que x es igual a s)

28. A(1..n) solo contiene elementos positivos (≥ 1) y sp es el producto de los elementos primos de A(1..n) (productoprimos)

Ejemplo:

A

17	22	1	1	**2**	**7**	62	9
1	2	3	4	5	6	7	8

sp = 17 * 2 * 7

- Nombre del predicado: productoprimos
- Hacerlo utilizando los predicados de los ejercicios 7 y 21. (todos los elementos de A(1..n) son positivos y sp es el producto de los elementos primos de A(1..n))

29. s es la suma de los elementos de A(1..n) que tienen al siguiente número en la siguiente posición (sumacontiguos)

Ejemplo:

A

20	22	**1**	2	1	**7**	**8**	9
1	2	**3**	4	5	**6**	**7**	8

s = 1 + 7 + 8

- Nombre del predicado: sumacontiguos
- Hacerlo sin utilizar predicados anteriores. (la suma de aquellos elementos de A(1..n) para los cuales se cumple que en la siguiente posición de A(1..n) se tiene dicho elemento más uno es igual a s)

30. Entre las posiciones pos1 y pos2 de A(1..n) x aparece por lo menos una vez (aparece)

- Nombre del predicado: aparece
- Hacerlo sin utilizar predicados anteriores. <u>Dos posibilidades</u>:
  - (i) pos1 es mayor o igual que 1 y menor o igual que pos2 y a su vez pos2 es menor o igual que n y existe una posición de A(1..n) que es mayor o igual que pos1 y menor o igual que pos2 y que contiene el valor x.
  - (ii) pos1 es mayor o igual que 1 y menor o igual que pos2 y a su vez pos2 es menor o igual que n y el número de posiciones de A(1..n) que están comprendidas entre pos1 y pos2 y que contienen el valor x es mayor o igual que 1.

31. x e y son distintos y aparecen en A(1..n) (aparecenambos)

- Nombre del predicado: aparecenambos
- Hacerlo utilizando el predicado del ejercicio 30.

32. x y w son distintos y todos los elementos de A(1..n) son mayores que x y distintos de w (mayoresdistintos)

- Nombre del predicado: mayoresdistintos
- Hacerlo de dos maneras:
  (i) Utilizando los predicados de los ejercicios 5 y 30. (x y w son distintos y todos los elementos de A(1..n) son mayores que x y w no aparece en A(1..n))
  (ii) Sin utilizar predicados definidos previamente. (x y w son distintos y para toda posición de A(1..n) ocurre el valor contenido en esa posición es mayor que x y distinto de w)

33. En A(1..n) hay dos múltiplos de x y tres de w (dostresmult)

- Nombre del predicado: dostresmult
- Hacerlo sin utilizar predicados anteriores. (el número de posiciones de A(1..n) que contienen valores que al dividirlos por x dan resto 0 es igual a 2 y el número de posiciones de A(1..n) que contienen valores que al dividirlos por w dan resto 0 es igual a 3)

34. Cada elemento de A(1..n) es múltiplo positivo (> 0) de la posición que ocupa (multposicion)

Ejemplo:

8	4	12	12	10	24	14	8
1	2	3	4	5	6	7	8

- Nombre del predicado: multposicion
- Hacerlo sin utilizar predicados anteriores. (para toda posición de A(1..n) ocurre que el valor contenido en la posición es mayor que 0 y al dividirlo por el número que indica la posición el resto es 0)

35. En todas las posiciones pares de A(1..n) hay un número positivo (> 0) (posparespositivos)

Ejemplo:

8	6	-12	2	10	14	-1	8
1	2	3	4	5	6	7	8

- Nombre del predicado: posparespositivos
- Hacerlo utilizando el predicado del ejercicio 1. (para toda posición par de A(1..n) ocurre que en esa posición se tiene un valor positivo)

36. En A(1..n) hay al menos un elemento que es mayor que x y también hay al menos un elemento que es menor que x (mayormenor)

- Nombre del predicado: mayormenor
- Hacerlo sin utilizar predicados anteriores. (Existe una posición en A(1..n) que contiene un valor mayor que x y existe una posición en A(1..n) que contiene un valor menor que x)

37. A partir de la posición pos de A(1..n) sólo hay un valor negativo y ningún cero (unnegninguncero)

- Nombre del predicado: unnegninguncero
- Hacerlo de <u>dos maneras</u>:
  (i) Utilizando el predicado del ejercicio 12. (pos es un valor comprendido entre 1 y n, el número de posiciones de A(1..n) comprendidos entre pos y n y que contienen un valor negativo es 1 y el valor 0 aparece 0 veces en la sección A(pos..n) de A(1..n))
  (ii) Sin utilizar predicados definidos previamente. (pos es un valor comprendido entre 1 y n, el número de posiciones de A(1..n) comprendidos entre pos y n y que contienen un valor negativo es 1 y el número de posiciones de la sección A(pos..n) de A(1..n) que contienen el valor 0 es 0)

38. pos es la primera posición que ocupa x en A(1..n) (primerapos)

- Nombre del predicado: primerapos
- Hacerlo de <u>dos maneras</u>:
  (i) Utilizando el predicado del ejercicio 30. (pos es un valor comprendido entre 1 y n, en la posición pos de A(1..n) se tiene el valor x y x no aparece entre las posiciones 1 y pos – 1 de A(1..n))
  (ii) Sin utilizar predicados definidos previamente. (pos es un valor comprendido entre 1 y n, en la posición pos de A(1..n) se tiene el valor x y para toda posición comprendida entre 1 y pos – 1 se cumple que el valor contenido en la posición no es x)

39. La variable booleana esta denota si x aparece en A(1..n) (denotaaparece)

- Nombre del predicado: denotaaparece
- Hacerlo de <u>dos maneras</u>:
  (i) Utilizando el predicado del ejercicio 30. (la variable *esta* vale True si y sólamente si x aparece en A(1..n))
  (ii) Sin utilizar predicados definidos previamente. (la variable *esta* vale True si y sólamente si existe una posición en A(1..n) que contiene el valor x)

40. La variable booleana *antes* denota si x aparece en A(1..n) antes de la posición pos (denotaantes)

- Nombre del predicado: denotaantes
- Hacerlo utilizando el predicado del ejercicio 30. (pos es un valor comprendido entre 1 y n y la variable booleana *antes* vale True si y sólamente si x aparece entre las posiciones 1 y pos – 1 de A(1..n))

41. z es el número de pares de ceros consecutivos de un vector de enteros A(1..n) (paresceros)

Ejemplo 1:

Dado el siguiente vector:

8	23	0	0	0	-57	14	0	94	35
1	2	3	4	5	6	7	8	9	10

Hay que considerar que en este vector hay 2 pares de ceros consecutivos. El primer par lo componen los ceros que están en las posiciones 3 y 4, y el segundo lo componen los ceros que están en las posiciones 4 y 5.

Ejemplo 2:

Dado el siguiente vector:

74	0	0	0	0	6	-68	0	0	-40
1	2	3	4	5	6	7	8	9	10

Hay que considerar que en este array hay 4 pares de ceros consecutivos. El primer par lo componen los ceros que están en las posiciones 2 y 3, el segundo lo componen los ceros que están en las posiciones 3 y 4, el tercero lo componen los que están en las posiciones 4 y 5, y el cuarto lo componen los que están en las posiciones 8 y 9.

- Nombre del predicado: paresceros
- Hacerlo sin utilizar predicados anteriores. (z es el número de posiciones de A(1..n) comprendidas entre 1 y n – 1 y que contienen un 0 y además se cumple que también se tiene un 0 en la siguiente posición)

42. El vector de enteros A(1..n) está formado por potencias enteras de 2 (todospotdos)

Ejemplo:

El siguiente vector está formado por potencias enteras de 2:

8	1	2	32	8	16	16	16	128	1
1	2	3	4	5	6	7	8	9	10

Las potencias enteras son aquellas que se obtienen con exponente entero y mayor o igual que 0. A continuación tenemos la descomposición correspondiente a cada valor:

2^3	2^0	2^1	2^5	2^3	2^4	2^4	2^4	2^7	2^0
1	2	3	4	5	6	7	8	9	10

- Nombre del predicado: todospotdos
- Hacerlo utilizando el predicado del ejercicio 4. (para toda posición de A(1..n) ocurre que en esa posición se tiene una potencia entera de 2)

43. x es el mínimo elemento de A(1..n) (minimo)

- Nombre del predicado: minimo
- Hacerlo sin utilizar predicados anteriores. (Existe una posición comprendida entre 1 y n que contiene el valor x y para toda posición de A(1..n) ocurre que el valor contenido en la posición es mayor o igual que x)

44. i y j definen una sección no vacía en A(1..n) y x es el máximo elemento de la sección A(i..j) de A(1..n) (maximoseccion)

Ejemplo:

Supongamos que A es un vector de 10 elementos y que el valor de las variables i y j es 4 y 8 respectivamente.

A

7	58	-60	**0**	**33**	**28**	**16**	**7**	549	95
1	2	3	**4**	**5**	**6**	**7**	**8**	9	10

Sección A(i..j)

- Nombre del predicado: maximoseccion
- Hacerlo sin utilizar predicados anteriores. (i es un valor comprendido entre 1 y n, j es un valor comprendido entre i y n, existe una posición comprendida entre i y j que contiene x y para toda posición comprendida entre i y j se tiene que el elemento contenido en la posición es menor o igual que x)

45. i y j definen una sección no vacía en A(1..n) y en A(1..n) solo hay números enteros positivos (≥ 1) y la sección A(i..j) del array A(1..n) contiene x elementos primos (numprimosseccion)

Ejemplo:
Supongamos que A es un array de 10 elementos y que el valor de las variables i y j es 2 y 6 respectivamente. La sección A(i..j) contiene dos primos (el 17 y el 23):

A

7	**58**	<ins>**17**</ins>	**21**	<ins>**23**</ins>	**64**	16	7	549	95
1	**2**	**3**	**4**	**5**	**6**	7	8	9	10

Sección A(i..j)

- Nombre del predicado: numprimosseccion
- Utilizar los predicados de los ejercicios 7 y 21. (i es un valor comprendido entre 1 y n, j es un valor comprendido entre i y n, en A(1..n) todos los elementos son positivos y el número de posiciones comprendidas entre i y j que contienen elementos primos es x)

46. pos y pos + 1 son los índices de los dos últimos elementos consecutivos iguales que tiene A(1..n) (ultimosiguales)

- Nombre del predicado: ultimosiguales
- Hacerlo sin utilizar predicados anteriores. (pos es un valor comprendido entre 1 y n – 1, los elementos de las posiciones pos y pos + 1 de A(1..n) son iguales y para toda posición comprendida entre pos + 1 y n – 1 se cumple que el elemento de la posición y el de la siguiente posición son distintos)

47. pos es una posición del vector A(1..n) y A(1..n) no tiene elementos repetidos hasta la posición pos incluida (distintoshastaposicion)

- Nombre del predicado: distintoshastaposicion
- Hacerlo utilizando el predicado del ejercicio 14. (pos es un valor comprendido entre 1 y n y todos los elementos comprendidos entre las posiciones 1 y pos de A(1..n) son distintos)

48. x aparece en A(1..n) y todas sus apariciones están seguidas al principio (apareceizquierda)

- Nombre del predicado: apareceizquierda
- Hacerlo de <u>dos maneras</u>:
  (i) Utilizando el predicado del ejercicio 12. (Existe una posición comprendida entre 1 y n tal que x aparece en todas las posiciones de la sección que va desde la posición 1 hasta dicha posición y x aparece 0 veces en la sección que va desde la siguiente posición de dicha posición hasta n)
  (ii) Sin utilizar predicados definidos previamente. (Existe una posición comprendida entre 1 y n tal que todas las posiciones comprendidas entre 1 y dicha posición contienen x y todas las posiciones que van a partir de dicha posición contienen un valor distinto de x)

49. De aparecer x en A(1..n), todas sus apariciones están seguidas al principio (apareceizquierda2)

- Nombre del predicado: apareceizquierda2
- Hacerlo de <u>dos maneras</u>:
  (i) Utilizando el predicado del ejercicio 12. (Existe un valor comprendido entre 0 y n tal que x aparece en todas las posiciones de la sección que va desde la posición 1 hasta dicha posición y x aparece 0 veces en la sección que va desde la siguiente posición de dicha posición hasta n)
  (ii) Sin utilizar predicados definidos previamente. (Existe una posición comprendida entre 0 y n tal que todas las posiciones comprendidas entre 1 y dicha posición contienen x y todas las posiciones que van a partir de dicha posición contienen un valor distinto de x)

50. A(1..n) = (a_n, a_1, …, a_{n-1}) (rotacionderecha)

En este caso hay que expresar cuál es la relación entre el subíndice de los elementos a_i y la posición que ocupan en el vector. Decir que A(1..n) = (a_n, a_1, …, a_{n-1}) es lo mismo que decir que el vector A tiene los siguientes elementos:

A

a_n	a_1	a_2	…	a_{n-3}	a_{n-2}	a_{n-1}
1	2	3	…	n – 2	n – 1	n

- Nombre del predicado: rotacionderecha
- Hacerlo sin utilizar predicados anteriores. (En la posición 1 se tiene el valor a_n y para toda posición comprendida entre 2 y n se tiene el valor a con subíndice igual a la posición menos 1)

51. A(1..n) = (a_2 , …, a_n , a_1) (rotacionizquierda)

Hay que expresar cuál es la relación entre el subíndice de los elementos a_i y la posición que ocupan en el vector. Decir que A(1..n) = (a_2 , …, a_n, a_1) es lo mismo que decir que el vector A tiene los siguientes elementos:

A

a_2	a_3	a_4	…	a_{n-1}	a_n	a_1
1	2	3	…	n – 2	n – 1	n

- Nombre del predicado: rotacionizquierda
- Hacerlo sin utilizar predicados anteriores. (En la posición n se tiene el valor a_1 y para toda posición comprendida entre 1 y n – 1 se tiene el valor a con subíndice igual a la posicíon más 1)

52. Ningún elemento de A(1..n) aparece en B(1..m) (disjuntos)

- Nombre del predicado: disjuntos
- Hacerlo de <u>dos maneras</u>:
  (i) Utilizando el predicado del ejercicio 30. (Para toda posición de A(1..n) ocurre que el elemento de esa posición no aparece en B(1..m))
  (ii) Sin utilizar predicados definidos previamente. (Para toda posición de A(1..n) ocurre que no existe ninguna posición de B(1..m) que contenga el mismo valor)

53. En A(1..n) hay menos negativos que positivos y ningún cero (maspositivos)

- Nombre del predicado: maspositivos
- Utilizar el predicado del ejercicio 30. (El número de posiciones de A(1..n) que contienen un valor negativo es menor que el número de posiciones que contienen un valor positivo y el valor 0 no aparece en A(1..n))

54. A(1..n) es una permutación de B(1..n) (permutacion)

Un array A(1..n) es una permutación de otro array B(1..n) cuando los dos tienen los mismos elementos y el mismo número de veces pero no necesariamente ocupando las mismas posiciones.

Ejemplo:

A

7	80	-60	3	0	0	0	7	50	30
1	2	3	4	5	6	7	8	9	10

B

0	0	30	3	-60	7	80	7	50	0
1	2	3	4	5	6	7	8	9	10

A(1..10) es una permutación de B(1..10).

- Nombre del predicado: permutacion
- Hacerlo sin utilizar predicados anteriores. (Para toda posición k comprendida entre 1 y n ocurre que el número de posiciones de A(1..n) que contienen el valor A(k) es igual al número de posiciones de B(1..n) que contienen el valor A(k))

55. A(1..n) es capicúa o palíndromo (palindromo)

Ejemplo 1:

A

7	10	-60	3	0	0	3	-60	10	7
1	2	3	4	5	6	7	8	9	10

Ejemplo 2:

A

'a'	'b'	'c'	'd'	'c'	'b'	'a'
1	2	3	4	5	6	7

- Nombre del predicado: palindromo
- Hacerlo sin utilizar predicados anteriores. (Para toda posición de A(1..n) ocurre que el elemento de la posición simétrica es igual al elemento de dicha posición)

Dada una posición i de A(1..n), su posición simétrica es n – i + 1.

56. A(1..n) tiene al menos un par de elementos consecutivos diferentes (almenosunpardif)

- Nombre del predicado: almenosunpardif
- Hacerlo sin utilizar predicados anteriores. (Existe una posición comprendida entre 1 y n – 1 tal que el valor de esa posición y el de la siguiente posición son diferentes)

57. A(1..n) tiene <u>exactamente</u> un par de elementos consecutivos diferentes (justounpardif)

- Nombre del predicado: justounpardif
- Hacerlo sin utilizar predicados anteriores. (El número de posiciones comprendidas entre 1 y n – 1 tal que el valor de esa posición y el de la siguiente posición son diferentes es 1)

58. np es el número de posiciones que tienen el mismo valor en A(1..n) y B(1..m) (numposmismovalor)

Ejemplo:
Los vectores A y B tienes 3 posiciones (2, 5 y 6) con el mismo valor:

A

26	**38**	-54	7	**1**	**-70**	44	9
1	**2**	3	4	**5**	**6**	7	8

B

0	**38**	1	6	**1**	**-70**	80	310	44	1
1	**2**	3	4	**5**	**6**	7	8	9	10

- Nombre del predicado: numposmismovalor
- Hacerlo sin utilizar predicados anteriores. (El número de posiciones comprendidas entre 1 y n y 1 y m tal que el valor de la posición coincide en A(1..n) y B(1..m) es np)

59. x aparece el mismo número de veces en A(1..n) y en B(1..p) (mismasvecesvectores)

- Nombre del predicado: mismasvecesvectores
- Hacerlo sin utilizar predicados anteriores. (El número de posiciones de A(1..n) que contienen x es igual al número de posiciones de B(1..p) que contienen x)

60. El vector A(1..n) contiene dígitos y representa al número natural x, pudiendo tener ceros a la izquierda (numero)

Ejemplo:
Un número natural es un número entero no negativo.

Los siguientes dos vectores son dos representaciones válidas del número natural 1280.

0	0	0	1	2	8	0
1	2	3	4	5	6	7

$$1280 = 0 \times 10^6 + 0 \times 10^5 + 0 \times 10^4 + 1 \times 10^3 + 2 \times 10^2 + 8 \times 10^1 + 0 \times 10^0$$

1	2	8	0
1	2	3	4

$$1280 = 1 \times 10^3 + 2 \times 10^2 + 8 \times 10^1 + 0 \times 10^0$$

- Nombre del predicado: numero
- Hacerlo utilizando el predicado del ejercicio 8. (A(1..n) solo contiene dígitos y x es igual a la suma de cada dígito multiplicado por 10 elevado a n menos la posición del dígito)

61. El vector A(1..n) contiene dígitos y representa a un número capicúa. Solo se admitirá un 0 en la posición 1 si el vector tiene un único elemento y representa por tanto al número 0 (numcapicua)

Ejemplo 1:

El vector A(1..10) de este ejemplo contiene dígitos y representa a un número capicúa.

A

8	3	9	7	1	1	7	9	3	8
1	2	3	4	5	6	7	8	9	10

Ejemplo 2:

El vector A(1..7) de este ejemplo contiene dígitos y representa a un número capicúa.

A

6	2	0	5	0	2	6
1	2	3	4	5	6	7

Ejemplo 3:

El vector A(1..1) de este ejemplo representa al número 0 (que es capicúa).

A

0
1

Ejemplo 4:

El siguiente vector no es adecuado por tener ceros a la izquierda:

0	0	0	5	8	2	8	5
1	2	3	4	5	6	7	8

Ejemplo 5:

El siguiente vector tampoco es adecuado por tener ceros a la izquierda:

0	0	0	6	2	6	0	0	0
1	2	3	4	5	6	7	8	9

- Nombre del predicado: numcapicua
- Hacerlo utilizando los predicados de los ejercicios 8 y 55. (A(1..n) contiene solo dígitos, es palíndromo y si el elemento de la posición 1 es 0 entonces n es igual a 1)

62. B(1..n) contiene el número de apariciones de cada elemento de A(1..n) (contieneveces)

Ejemplo:

El vector B(1..10) de este ejemplo contiene el número de apariciones de cada elemento del vector A(1..10):

A

8	3	8	8	1	1	-70	59	8	20
1	2	3	4	5	6	7	8	9	10

B

4	1	4	4	2	2	1	1	4	1
1	2	3	4	5	6	7	8	9	10

Así tenemos que el 8 aparece cuatro veces, el 3 una vez, etc.

- Nombre del predicado: contieneveces
- Hacerlo de <u>dos maneras</u>:
  (i) Utilizando el predicado del ejercicio 13. (Para toda posición de B(1..n) ocurre que el elemento de esa posición indica cuántas veces aparece el elemento de la misma posición de A en todo el vector A(1..n))
  (ii) Sin utilizar predicados definidos previamente. (Para toda posición de B(1..n) ocurre que el elemento de esa posición es igual al número de posiciones de A(1..n) que son iguales al elemento que está en dicha posición en A(1..n))

63. (subvectorseccion)

A(1..n) tiene al menos un elemento. El número de elementos de B(1..p) es mayor o igual que el de A(1..n). Las variables i y j definen un subintervalo no vacío de tamaño n en el intervalo [1..p]. A(1..n) es un subvector de B(1..p) que va desde la posición i hasta la j, es decir, todos los elementos de A aparecen seguidos y en el mismo orden en B desde la posición i hasta la j.

Ejemplo:

El vector A(1..4) de este ejemplo es un subvector del vector B(1..10) que va desde la posición 4 hasta la 7:

A

40	333	6	10
1	2	3	4

B

11	78	0	**40**	**333**	**6**	**10**	-68	4	1
1	2	3	**4**	**5**	**6**	**7**	8	9	10
			i			j			

A

- Nombre del predicado: subvectorseccion
- Hacerlo sin utilizar predicados anteriores. (n es mayor o igual que 1, p es mayor o igual que n, i es mayor o igual que 1 y menor o igual que n, j es mayor o igual que i y menor o igual que n, j menos i es n – 1, para toda posición k de A(1..n) ocurre que B(i + k – 1) es igual a A(k).

64. (subvector)

A(1..n) tiene al menos un elemento. El número de elementos de B(1..p) es mayor o igual que el de A(1..n). A(1..n) es un subvector de B(1..p), es decir, todos los elementos de A aparecen seguidos y en el mismo orden en B.

Ejemplo:

El vector A(1..4) de este ejemplo es un subvector del vector B(1..10):

A

40	333	6	10
1	2	3	4

B

11	78	0	**40**	**333**	**6**	**10**	-68	4	1
1	2	3	**4**	**5**	**6**	**7**	8	9	10

A

- Nombre del predicado: subvector
- Hacerlo de dos maneras:
  - (i) Utilizando el predicado del ejercicio 63. (Existe una posición h mayor o igual que 1 y menor o igual que p y existe una posición g mayor o igual que 1 y menor o igual que p tal que A(1..n) es un subvector de B(1..p) justo en la subsección definida por h y g)
  - (ii) Sin utilizar predicados definidos previamente. (n es mayor o igual que 1, p es mayor o igual que n, existe una posición h que es mayor o igual que 1 y menor o igual que n, existe una posición g que es mayor o igual que h y menor o igual que n, g menos h es n – 1, para toda posición k de A(1..n) ocurre que B(h + k – 1) es igual a A(k).

65. (almenosparcomun)

La variable booleana *mp* denota si A(1..n) y B(1..m) tienen por lo menos un par de elementos adyacentes en común (ocupando las mismas posiciones en ambos arrays).

Ejemplo:

A

44	**38**	**1**	1	1	**-70**	**62**	9
1	**2**	**3**	4	5	**6**	**7**	8

B

44	**38**	**1**	6	22	**-70**	**62**	1	-70	1
1	**2**	**3**	4	5	**6**	**7**	8	9	10

Los vectores A y B de este ejemplo tienen tres pares de elementos adyacentes en común ocupando las mismas posiciones:

- Posiciones 1 y 2.
- Posiciones 2 y 3.
- Posiciones 6 y 7.

➢ Nombre del predicado: almenosparcomun
➢ Hacerlo sin utilizar predicados anteriores.

66. (justoparcomun)

La variable booleana *b* denota si A(1..n) y B(1..m) tienen exactamente un par (no más) de elementos adyacentes en común (ocupando las mismas posiciones en ambos vectores).

Ejemplo:

Los vectores A y B de este ejemplo tienen exactamente un par de elementos adyacentes en común ocupando las mismas posiciones: Posiciones 6 y 7.

A

90	22	1	1	1	**-70**	**62**	9
1	2	3	4	5	**6**	**7**	8

B

44	38	1	6	22	**-70**	**62**	1	-70	1
1	2	3	4	5	**6**	**7**	8	9	10

➢ Nombre del predicado: justoparcomun
➢ Hacerlo sin utilizar predicados anteriores.

67. (almenosparcomunpos)

La variable booleana *dp* denota si A(1..n) y B(1..m) tienen por lo menos un par de elementos adyacentes en común (pudiendo ocupar distintas posiciones en ambos vectores).

Ejemplo:

A

44	**38**	1	1	**6**	**-70**	**62**	9
1	**2**	3	4	**5**	**6**	**7**	8

B

20	**44**	**38**	**6**	**-70**	**62**	5	1	-70	1
1	**2**	**3**	**4**	**5**	**6**	7	8	9	10

Los vectores A y B de este ejemplo tienen tres pares adyacentes en común aunque ocupando distintas posiciones:

- Posiciones A(1), A(2) y B(2), B(3).
- Posiciones A(5), A(6) y B(4), B(5).
- Posiciones A(6), A(7) y B(5), B(6).

➢ Nombre del predicado: almenosparcomunpos
➢ Hacerlo sin utilizar predicados anteriores.

68. (justoparcomunpos)

La variable booleana c denota si A(1..n) y B(1..m) tienen exactamente un par (no más) de elementos adyacentes en común (pudiendo ocupar distintas posiciones en ambos vectores).

Ejemplo:

A

90	22	1	1	1	**-70**	**62**	9
1	2	3	4	5	**6**	**7**	8

B

44	38	**-70**	**62**	22	18	5	2	-70	1
1	2	**3**	**4**	5	6	7	8	9	10

Los vectores A y B de este ejemplo tienen exactamente un par adyacente en común aunque ocupando distintas posiciones: Posiciones A(6), A(7) y B(3), B(4).

➢ Nombre del predicado: justoparcomunpos
➢ Hacerlo sin utilizar predicados anteriores.

69. (primosconsec)

u y v son números primos consecutivos (u es menor que v, u es primo, v es primo y todo número comprendido entre u y v no es primo).

- Nombre del predicado: primosconsec
- Hacerlo utilizando el predicado del ejercicio 21.

70. (indicemax)

i es el índice del máximo elemento de C(1..m).

- Nombre del predicado: indicemax
- Hacerlo utilizando el predicado del ejercicio 44.

71. (indicemin)

i es el índice del mínimo elemento de C(1..m).

- Nombre del predicado: indicemin
- Hacerlo utilizando el predicado del ejercicio 43.

72. (creciente)

B(1..m) es C(1..m) ordenado crecientemente.

- Nombre del predicado: creciente
- Hacerlo utilizando el predicado del ejercicio 54.

73. (inverso)

C(1..m) es el inverso de B(1..m).

- Nombre del predicado: inverso
- Hacerlo sin utilizar predicados anteriores.

74. (norep)

C(1..m) no tiene elementos repetidos.

- Nombre del predicado: norep
- Hacerlo utilizando el predicado del ejercicio 14.

75. (nosecsumanula)

C(1..m) no tiene ninguna sección de suma nula.

- Nombre del predicado: nosecsumanula
- Hacerlo utilizando el predicado del ejercicio 23.

76. (noparcerosconsec)

C(1..m) no tiene ningún par de ceros consecutivos.

- Nombre del predicado: noparcerosconsec
- Hacerlo utilizando el predicado del ejercicio 41.

77. (todosdosveces)

Cada elemento de C(1..m) aparece exactamente dos veces.

- Nombre del predicado: todosdosveces
- Hacerlo utilizando el predicado del ejercicio 20.

78. (tresdisjuntos)

C(1..n), D(1..m) y E(1..p) no tienen ningún elemento en común.

- Nombre del predicado: tresdisjuntos
- Hacerlo de dos formas: primero utilizando el predicado del ejercicio 13 y luego utilizando el predicado del ejercicio 52.

79. (seleccionpositivos)

C(1..m) tiene al menos un elemento positivo (> 0), en B(1..p) todos los elementos son positivos (> 0) y distintos y todos los elementos de B aparecen en C.

- Nombre del predicado: seleccionpositivos
- Hacerlo utilizando los predicados de los ejercicios 6, 7, 14 y 30.

80. (particion)

C(1..m) y D(1..p) tienen al menos un elemento cada uno y definen una partición de E(1..q), es decir, que juntando C y D se obtiene E.

Ejemplo:

C

90	22	1
1	2	3

D

44	38	1	6	22
1	2	3	4	5

E

90	22	1	44	38	1	6	22
1	2	3	4	5	6	7	8

- Nombre del predicado: particion
- Hacerlo utilizando el predicado del ejercicio 63.

81. (primerosprimosorden)

A(1..n) tiene los primeros n números primos ordenados crecientemente.

- Nombre del predicado: primerosprimosorden
- Hacerlo utilizando el predicado del ejercicio 69. (En la posición 1 se tiene el valor 2(que es el primer primo) y para cada posición comprendida entre 1 y n – 1 el elemento de esa posición y el de la siguiente posición son primos consecutivos)

82. (maximominimounavez)

max es el máximo elemento de B(1..m), *min* el mínimo y tanto el mínimo como el máximo aparecen sólo una vez.

- Nombre del predicado: maximominimounavez
- Hacerlo utilizando los predicados de los ejercicios 13, 43 y 44.

b) Especificación pre-post de programas

1. Contar en c el número de posiciones iguales de A(1..n) y B(1..n)

Dar la **especificación pre-post** del siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n) donde n es mayor o igual que 1, calcula en la variable c cuántas posiciones iguales tienen los dos vectores:

```
{φ} ≡ ???
i : = 1;
c : = 0;
while i ≤ n loop
        if A(i) = B(i) then c : = c + 1; end if;
        i : = i + 1;
end loop;
{ψ} ≡ ???
```

2. Decidir en w si A(1..n) y B(1..n) tienen alguna posición igual

Dar la **especificación pre-post** del siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n) donde n es mayor o igual que 1, decide en la variable booleana w si los dos vectores tienen alguna posición igual:

```
{φ} ≡ ???
i : = 1;
w : = false;
while i ≠ n + 1 and not w loop
        w : = (A(i) = B(i));
        i : = i + 1;
end loop;
{ψ} ≡ ???
```

3. Decidir en *iguales* si todos los elementos de A(1..n) son iguales

Dar la **especificación pre-post** del siguiente programa que, dado un vector de enteros A(1..n) donde n es mayor o igual que 1, decide en la variable booleana iguales si todos los elementos son iguales:

```
{φ} ≡ ???
i : = 1;
iguales : = true;
while i ≠ n and iguales loop
        iguales : = (A(i) = A(i + 1));
        i : = i + 1;
end loop;
{ψ} ≡ ???
```

4. Decidir en d si el número de posiciones iguales de A(1..n) y B(1..n) es mayor que el de posiciones diferentes

Dar la **especificación pre-post** del siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n) donde n es mayor o igual que 1, decide en la variable booleana d si el número de posiciones iguales es mayor que el de posiciones diferentes:

```
{φ} ≡ ???
i : = 1;
c : = 0;
while i ≠ n + 1 loop
        if A(i) = B(i) then c : = c + 1; end if;
        i : = i + 1;
end loop;
d : = (c > (n div 2));
{ψ} ≡ ???
```

Donde div es la división entera.

5. Decidir en b si el número de unos de A(1..n) es mayor que el de ceros

Dar la **especificación pre-post** del siguiente programa que, dado un vector de enteros A(1..n) que solo contiene unos y ceros y donde n es mayor o igual que 1, decide en la variable booleana b si el número de unos es mayor que el de ceros:

```
{φ} ≡ ???
i : = 0;
c : = 0;
while i ≠ n loop
        if A(i + 1) = 1 then c : = c + 1; end if;
        i : = i + 1;
end loop;
b : = (c > (n div 2));
{ψ} ≡ ???
```

Donde div es la división entera.

6. Devolver en max el número de unos de A(1..n) si hay más unos y si no el número de ceros

Dar la **especificación pre-post** del siguiente programa que, dado un vector de enteros A(1..n) que solo contiene unos y ceros y donde n es mayor o igual que 1, devuelve en la variable max el número de unos si hay más unos que ceros y devuelve el número de ceros en caso contrario:

```
{φ} ≡ ???
i : = 0;
c : = 0;
u : = 0
while i ≠ n loop
        if A(i + 1) = 0 then c : = c + 1;
        else u : = u + 1; end if;
        i : = i + 1;
end loop;
if u > c then max : = u;
         else max : = c; end if;
{ψ} ≡ ???
```

7. Guardar en B(1..n) ceros donde en A(1..n) hay unos y unos donde en A(1..n) hay ceros

Dar la **especificación pre-post** del siguiente programa que, dado un vector de enteros A(1..n) que solo contiene ceros y unos y donde n es mayor o igual que 1, genera B(1..n) de tal forma que contiene ceros en las posiciones en las que A(1..n) tiene unos y contiene unos en las posiciones en las que A(1..n) tiene ceros:

```
{φ} ≡ ???
i : = 1;
while i ≠ n + 1 loop
        if A(i) = 0 then B(i) : = 1;
        else B(i) : = 0; end if;
        i : = i + 1;
end loop;
{ψ} ≡ ???
```

8. Intercambiar ceros por unos y unos por ceros en A(1..n)

Dar la **especificación pre-post** del siguiente programa que, dado un vector de enteros A(1..n) que solo contiene ceros y unos y donde n es mayor o igual que 1, intercambia ceros por unos y unos por ceros:

```
{φ} ≡ ???
i : = 1;
while i ≠ n + 1 loop
        if A(i) = 0 then A(i) : = 1;
        else A(i) : = 0; end if;
        i : = i + 1;
end loop;
{ψ} ≡ ???
```

c) Documentación de programas

1. Programa que calcula en C(1..n) la suma de A(1..n) y B(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que calcula en cada posición del vector C(1..n) la suma de los elementos correspondientes de los vectores A(1..n) y B(1..n). El valor de n será mayor o igual que 1:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≤ n loop
      (4) {Aserción intermedia}
      C(i) : =A(i) + B(i);
      (5) {Aserción intermedia}
      i : = i + 1;
      (6) {Aserción intermedia}
end loop;
(7) {Postcondición}
(8) Expresión cota E
```

2. Programa que intercambia los elementos de A(1..n) y B(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que intercambia los elementos de A(1..n) y B(1..n). El valor de n será mayor o igual que 1:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≤ n loop
      (4) {Aserción intermedia}
      aux : =A(i);
      (5) {Aserción intermedia}
      A(i) : =B(i);
      (6) {Aserción intermedia}
      B(i) : =aux;
      (7) {Aserción intermedia}
      i : = i + 1;
      (8) {Aserción intermedia}
end loop;
(9) {Postcondición}
(10) Expresión cota E
```

3. Programa que cuenta en v el número de veces que x aparece en el vector A(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que cuenta en v el número de veces que x aparece en el vector A(1..n):

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
v : = 0;
(3) {Aserción intermedia}
while (4) {Invariante} i ≤ n loop
      (5) {Aserción intermedia}
      if A(i) = x then (6) {Aserción intermedia}
                 v : = v + 1;
      end if;
      (7) {Aserción intermedia}
      i : = i + 1;
      (8) {Aserción intermedia}
end loop;
(9) {Postcondición}
(10) Expresión cota E
```

4. Programa que cuenta en v el número de veces que x aparece en el vector A(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que cuenta en v el número de veces que x aparece en el vector A(1..n):

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
v : = 0;
(3) {Aserción intermedia}
while (4) {Invariante} i < n loop
      (5) {Aserción intermedia}
      i : = i + 1;
      (6) {Aserción intermedia}
      if A(i) = x then (7) {Aserción intermedia}
                 v : = v + 1;
                 (8) {Aserción intermedia}
      end if;
      (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) Expresión cota E
```

5. Programa que decide en la variable booleana "iguales" si A(1..n) y B(1..n) son iguales o no

Documentar, con las aserciones que se indican, el siguiente programa que decide en la variable booleana "iguales" si A(1..n) y B(1..n) son iguales o no:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
iguales : = True;
(3) {Aserción intermedia}
while (4) {Invariante} i ≤ n and iguales loop
      (5) {Aserción intermedia}
      if A(i) ≠ B(i) then (6) {Aserción intermedia}
                    iguales : = False;
                   (7) {Aserción intermedia}
      end if;
      (8) {Aserción intermedia}
      i : = i + 1;
      (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) Expresión cota E
```

6. Programa que decide en la variable booleana "esta" si x aparece en el vector A(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que decide en la variable booleana "esta" si x aparece en el vector A(1..n):

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
esta : = False;
(3) {Aserción intermedia}
while (4) {Invariante} i ≠ n and not esta loop
      (5) {Aserción intermedia}
      i : = i + 1;
      (6) {Aserción intermedia}
      if A(i) = x then
                   (7) {Aserción intermedia}
                    esta : = True;
                   (8) {Aserción intermedia}
      end if;
      (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) Expresión cota E
```

7. Programa que calcula en z el producto de dos números naturales x e y

Documentar, con las aserciones que se indican, el siguiente programa que calcula en z el producto de dos números naturales x e y:

```
(1) {Precondición}
z : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} x ≠ 0 loop
      (4) {Aserción intermedia}
      z : = z + y;
      (5) {Aserción intermedia}
      x : = x – 1;
      (6) {Aserción intermedia}
end loop;
(7) {Postcondición}
(8) Expresión cota E
```

8. Programa que dado x ≥ 2 decide en la variable booleana "primo" si x es un número primo

Documentar, con las aserciones que se indican, el siguiente programa que dado x ≥ 2 decide en la variable booleana "primo" si x es un número primo:

```
(1) {Precondición}
d : = 2;
(2) {Aserción intermedia}
while (3) {Invariante} x mod d ≠ 0 loop
      (4) {Aserción intermedia}
      d : = d + 1;
      (5) {Aserción intermedia}
end loop;
(6) {Aserción intermedia}
primo : = (d = x);
(7) {Postcondición}
(8) Expresión cota E
```

9. Programa que dado un vector A(1..n) de enteros, cuenta en la variable cuantos el número de múltiplos de 5 que contiene

Documentar, con las aserciones que se indican, el siguiente programa que dado un vector A(1..n) de enteros, cuenta en la variable cuantos el número de múltiplos de 5 que contiene:

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
cuantos : = 0;
(3) {Aserción intermedia}
while (4) {Invariante} i < n loop
          (5) {Aserción intermedia}
          i : = i + 1;
          (6) {Aserción intermedia}
          if A(i) mod 5 = 0 then (7) {Aserción intermedia}
                                 cuantos : = cuantos + 1;
                                 (8) {Aserción intermedia}
          end if;
        (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) Expresión cota E
```

10. Programa que calcula en m el mínimo del vector de enteros A(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que calcula en m el mínimo del vector de enteros A(1..n):

```
(1) {Precondición}
m : = A(1);
(2) {Aserción intermedia}
k : = 1;
(3) {Aserción intermedia}
while (4) {Invariante} k < n loop
       (5) {Aserción intermedia}
       k : = k + 1;
       (6) {Aserción intermedia}
       if A(k) < m then (7) {Aserción intermedia}
                         m : = A(k);
                        (8) {Aserción intermedia}
       end if;
       (9) {Aserción intermedia}
end loop;
(10) {Postcondición}

(11) Expresión cota E
```

11. Programa que dado $n \geq 0$, calcula en x el valor del término s_n de la sucesión de Fibonacci

Documentar, con las aserciones que se indican, el siguiente programa que dado $n \geq 0$, calcula en x el valor del término s_n de la sucesión de Fibonacci ($s_0 = 0$, $s_1 = 1$ y $s_k = s_{k-1} + s_{k-2}$ para $k \geq 2$):

```
(1) {Precondición}
x : = 0;
(2) {Aserción intermedia}
y : = 1;
(3) {Aserción intermedia}
k : = 1;
(4) {Aserción intermedia}
while (5) {Invariante} k ≤ n loop
      (6) {Aserción intermedia}
      y : = y + x;
      (7) {Aserción intermedia}
      x : = y – x;
      (8) {Aserción intermedia}
      k : = k + 1;
      (9) {Aserción intermedia}
end loop;
(10) {Postcondición}

(11) Expresión cota E
```

12. Programa que calcula en f el factorial de un número natural x

Documentar, con las aserciones que se indican, los siguientes dos programas que calculan en f el factorial de un número natural x:

```
(1) {Precondición}
f : = 1;
(2) {Aserción intermedia}
t : = 1;
(3) {Aserción intermedia}
while (4) {Invariante} t ≤ x loop
      (5) {Aserción intermedia}
      f : = f ∗ t;
      (6) {Aserción intermedia}
      t : = t + 1;
      (7) {Aserción intermedia}
end loop;
(8) {Postcondición}

(9) Expresión cota E
```

```
(1) {Precondición}
f : = 1;
(2) {Aserción intermedia}
t : = x;
(3) {Aserción intermedia}
while (4) {Invariante} t ≥ 1 loop
      (5) {Aserción intermedia}
      f : = f ∗ t;
      (6) {Aserción intermedia}
      t : = t – 1;
      (7) {Aserción intermedia}
end loop;
(8) {Postcondición}

(9) Expresión cota E
```

13. Programa que calcula en neg cuántos números negativos hay en A(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que calcula en neg cuántos números negativos hay en A(1..n):

```
(1) {Precondición}
neg : = 0;
(2) {Aserción intermedia}
i : = 0;
(3) {Aserción intermedia}
while (4) {Invariante} i ≠ n loop
      if A(i + 1) < 0 then
                              (5) {Aserción intermedia}
                              neg : = neg + 1;
                              (6) {Aserción intermedia}
      end if;
      (7) {Aserción intermedia}
      i : = i + 1;
      (8) {Aserción intermedia}
end loop;
(9) {Postcondición}
(10) Expresión cota E
```

14. Programa que calcula en pos cuántos números positivos hay en A(1..n) y calcula en neg cuántos números negativos hay en A(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que calcula en pos cuántos números positivos hay en A(1..n) y calcula en neg cuántos números negativos hay en A(1..n):

```
(1) {Precondición}
pos : = 0;
neg : = 0;
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n loop
      if A(i + 1) > 0 then
                              (4) {Aserción intermedia}
                              pos : = pos + 1;
                              (5) {Aserción intermedia}
      elsif A(i + 1) < 0 then
                              (6) {Aserción intermedia}
                              neg : = neg + 1;
                              (7) {Aserción intermedia}
      end if;
      (8) {Aserción intermedia}
      i : = i + 1;
      (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) Expresión cota E
```

15. Programa que calcula en "neg" cuántos números negativos hay en A(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que calcula en "neg" cuántos números negativos hay en A(1..n)

```
(1) {Precondición}
neg : = 0;
(2) {Aserción intermedia}
i : = 0;
(3) {Aserción intermedia}
while (4) {Invariante} i ≠ n loop
      (5) {Aserción intermedia}
      i : = i + 1;
      (6) {Aserción intermedia}
      if A(i) < 0 then
                  (7) {Aserción intermedia}
                  neg : = neg + 1;
                  (8) {Aserción intermedia}
      end if;
      (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) Expresión cota E
```

16. Programa que decide en la variable booleana "inversa" si A(1..n) es la inversa de B(1..n)

Documentar, con las aserciones que se indican, el siguiente programa que decide en la variable booleana "inversa" si A(1..n) es la inversa de B(1..n):

```
(1) {Precondición}
inversa : = True;
(2) {Aserción intermedia}
i : = 1;
(3) {Aserción intermedia}
while (4) {Invariante} i ≠ n + 1 and inversa loop
      (5) {Aserción intermedia}
      if A(i) ≠ B(n – i + 1) then
                  (6) {Aserción intermedia}
                  inversa : = False;
      end if;
      (7) {Aserción intermedia}
      i : = i + 1;
      (8) {Aserción intermedia}
end loop;
(9) {Postcondición}
(10) Expresión cota E
```

17. Programa que calcula el número de posiciones para las cuales el siguiente elemento del vector es estrictamente menor

Documentar, con las aserciones que se indican, el siguiente programa que, dado un vector A(1..n) de enteros, calcula en la variable *decr* el número de posiciones para las cuales el siguiente elemento del vector es estrictamente menor.

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
decr : = 0;
(3) {Aserción intermedia}
while (4) {Invariante} i ≠ n – 1 loop
      (5) {Aserción intermedia}
      i : = i + 1;
      (6) {Aserción intermedia}
      if A(i) > A(i + 1) then
                        (7) {Aserción intermedia}
                        decr : = decr + 1;
                        (8) {Aserción intermedia}
      end if;
      (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) Expresión cota E
```

18. Programa que decide en la variable booleana p si A(1..n) contiene más elementos pares que impares

Documentar, con las aserciones que se indican, el siguiente programa que, dado un vector A(1..n) de enteros, decide en la variable booleana p si A(1..n) contiene más elementos pares que impares.

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
nump : = 0;
(3) {Aserción intermedia}
while (4) {Invariante} i ≠ n loop
      if A(i + 1) mod 2 = 0 then
                        (5) {Aserción intermedia}
                        nump : = nump + 1;
                        (6) {Aserción intermedia}
      end if;
      (7) {Aserción intermedia}
      i : = i + 1;
      (8) {Aserción intermedia}
end loop;
(9) {Aserción intermedia}
p : = (nump > (n / 2));
(10) {Postcondición}
(11) Expresión cota E
```

19. Predicado Invirtiendo(C(1..p), (c_1, c_2, …, c_h), p) y programa que invierte el vector A(1..n)

a) Escribir un predicado **Invirtiendo(C(1..h), (c_1, c_2, …, c_h), p)** que exprese que el vector C es como el vector (c_1, c_2, …, c_h) pero con la diferencia de que los elementos están invertidos hasta la posición p incluida:

C(1..h)

c_h	c_{h-1}	…	c_{h-p+1}	c_{p+1}	…	c_{h-p}	c_p	…	c_2	c_1
1	2	…	p	p + 1	…	h – p	h – p + 1	…	h – 1	h
Invertido				Sin invertir			Invertido			

Recordemos que el vector **(c_1, c_2, …, c_h)** representa al vector:

c_1	c_2		…	c_{h-1}	c_h
1	2	3	…	h – 1	h

b) **Documentar**, con el invariante y las aserciones que se indican y utilizando el predicado definido en el apartado anterior, el siguiente programa que invierte el vector A(1..n) que contiene al menos un elemento:

```
{Precondición} ≡ {n ≥ 1 ∧ ∀k(1 ≤ k ≤ n → A(k) = a_k)}
i : = 1;
(2) {Aserción intermedia}
while (1) {Invariante} i ≤ n / 2 loop
      aux : = A(i);
      (3) {Aserción intermedia}
      A(i) : = A(n – i + 1);
      (4) {Aserción intermedia}
      A(n – i + 1) : = aux;
      (5) {Aserción intermedia}
      i : = i + 1;
      (6) {Aserción intermedia}
end loop;
(7) {Postcondición}

(8) Expresión cota E
```

20. Predicado Nummayores(C(1..p), D(1..p), x) y programa que decide en la variable booleana b si más de la mitad de los elementos de A(1..n) son mayores que los de su misma posición en B(1..n)

a) Escribir un predicado **Nummayores(C(1..p), D(1..p), x)** que exprese que x es el número de posiciones para las cuales el elemento de C es mayor que el elemento de D.

b) Documentar con el invariante y las aserciones intermedias que se indican y utilizando el predicado definido en el apartado anterior, el siguiente programa que decide en la variable booleana b si más de la mitad de los elementos del vector de enteros A(1..n) son mayores que los de su misma posición en el array B(1..n):

```
(1) {Precondición}
i : = 1;
z : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≤ n loop
    if A(i) > B(i) then
                (4) {Aserción intermedia}
                 z : = z + 1;
                (5) {Aserción intermedia}
    end if;
    (6) {Aserción intermedia}
    i : = i + 1;
    (7) {Aserción intermedia}
end loop;
(8) {Aserción intermedia}
b : = z > (n / 2);
(9) {Postcondición}
(10) Expresión cota E
```

21. Predicados Ecreciente(C(1..m)) y Aparece(z, C(1..m)) y programa que obtiene en la sección B(1..j) de B(1..n) los índices de todas las apariciones de x en el vector A(1..n)

a) Escribir un predicado **Ecreciente**(C(1..m)) que exprese que C(1..m) está ordenado de manera creciente siendo cada elemento estrictamente menor que el siguiente.

b) Escribir un predicado **Aparece(z, C(1..m))** que exprese que el elemento z aparece en el vector C(1..m).

c) **Documentar**, con el invariante y las aserciones intermedias que se indican y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que obtiene en la sección B(1..j) de B(1..n) los índices de todas las apariciones de x en el vector A(1..n):

```
(1) {Precondición}
j : = 0;
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≤ n loop
     (4) {Aserción intermedia}
      if A(i) = x then
               (5) {Aserción intermedia}
               j : = j + 1;
               (6) {Aserción intermedia}
               B(j) : = i;
               (7) {Aserción intermedia}
     end if;
     (8) {Aserción intermedia}
      i : = i + 1;
     (9) {Aserción intermedia}
end loop;
(10) {Postcondición}

(11) Expresión cota E
```

22. Predicados Sumaintervalo(C(1..p), h, i, s) y Sumaigual(C(1..p), i) y programa que decide en eqsum si el vector A(1..n) de enteros es divisible en dos secciones de igual suma teniendo cada sección al menos un elemento

a) Escribir un predicado **Sumaintervalo(C(1..p), h, i, s)** que exprese que la suma de los elementos de C que van desde la posición h hasta la posición i es s, es decir que la suma de C(h..i) es s.

b) Escribir un predicado **Sumaigual(C(1..p), i)** que exprese que la suma de los elementos de C que van desde la posición 1 hasta la posición i es igual a la suma de los elementos que van desde la posición i + 1 hasta la n, es decir que la suma de C(1..i) es igual a la suma de C(i + 1..n).

c) Documentar, con los invariantes y las aserciones intermedias que se indican y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que decide (dejando la respuesta en la variable eqsum) si el vector A(1..n) de enteros (donde n ≥ 2) es divisible en dos secciones de igual suma teniendo cada sección al menos un elemento:

```
(1) {Precondición}
x : = A(1);
y : = 0;
k : = 2;
(2) {Aserción intermedia}
while (3) {Invariante} k ≤ n loop
      (4) {Aserción intermedia}
      y : = y + A(k);
      (5) {Aserción intermedia}
      k : = k + 1;
      (6) {Aserción intermedia}
end loop;
(7) {Aserción intermedia}
j : = 1;
(8) {Aserción intermedia}
while (9) {Invariante} x ≠ y and j < n – 1 loop
      (10) {Aserción intermedia}
      j : = j + 1;
      (11) {Aserción intermedia}
      x : = x + A(j);
      (12) {Aserción intermedia}
      y : = y – A(j);
      (13) {Aserción intermedia}
end loop;
(14) {Aserción intermedia}
eqsum : = (x = y);
(15) {Postcondición}
```

Dar también las expresiones cota E_1 y E_2 correspondientes a cada while.

23. Predicados Coinciden(D(1..r), x), Es_suma(D(1..r), s) y Es_producto(D(1..r), p) y programa que obtiene en c la mayor posición del intervalo 1..n tal que la suma y el producto de los elementos de A(1..n) coinciden hasta esa posición

a) Escribir un predicado **Coinciden(D(1..r), x)** que exprese que la suma de los elementos de D que van desde la posición 1 hasta la posición x coincide con el producto de esos mismos elementos.

b) Escribir un predicado **Es_suma(D(1..r), s)** que exprese que la suma de los elementos de D que van desde la posición 1 hasta la posición r es s.

c) Escribir un predicado **Es_producto(D(1..r), p)** que exprese que el producto de los elementos de D que van desde la posición 1 hasta la posición r es p.

d) Documentar, con el invariante y las aserciones intermedias que se indican y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que dado un vector A(1..n) de enteros, obtiene en c la mayor posición del intervalo 1..n tal que la suma y el producto de los elementos coinciden hasta esa posición:

```
(1) {Precondición}
i : = 1;
c : = 1;
suma : = A(1);
prod : = A(1);
(2) {Aserción intermedia}
while (3) {Invariante} i < n loop
      (4) {Aserción intermedia}
      i : = i + 1;
      (5) {Aserción intermedia}
      suma : = suma + A(i);
      (6) {Aserción intermedia}
      prod : = prod * A(i);
      (7) {Aserción intermedia}
      if suma = prod then (8) {Aserción intermedia}
                          c : = i;
                            (9) {Aserción intermedia}
      end if;
      (10) {Aserción intermedia}
end loop;
(11) {Postcondición}

(12) Expresión cota E
```

24. Predicados Crec(T(a..b)) y Decr(T(a..b)) y programa que decide si los elementos de A(1..n) están ordenados de manera estrictamente creciente hasta una cierta posición y de manera estrictamente decreciente a partir de esa posición

a) Escribir un predicado **Crec**(T(a..b)) que exprese que la sección T(a..b) es estrictamente creciente y otro predicado **Decr**(T(a..b)) que exprese que la sección T(a..b) es estrictamente decreciente.

b) **Documentar**, con las aserciones que se indican y utilizando los predicados definidos en el apartado anterior, el siguiente programa que decide si los elementos del vector A(1..n) están ordenados de manera estrictamente creciente hasta una cierta posición y de manera estrictamente decreciente a partir de esa posición. Por tanto, el programa decide si el vector A(1..n) consta de una sección estrictamente creciente seguida de una sección estrictamente decreciente, sin importar la longitud de cada una de ellas, pero abarcando entre las dos secciones todo el vector. El último elemento de la parte creciente será el primer elemento de la parte decreciente.

```
(1) {Precondición}
i : = 1; j : = n; seguir : = True;
while (2) {Invariante} i ≠ j and seguir loop
        (3) {Aserción intermedia}
        if A(j – 1) > A(j) then (4) {Aserción intermedia}
                                   j : = j – 1;
                                    (5) {Aserción intermedia}
        elsif A(j – 1) < A(j) (6) {Aserción intermedia}
                                   i : = i + 1;
                                 (7) {Aserción intermedia}
        else (8) {Aserción intermedia}
                seguir : = False;
               (9) {Aserción intermedia}
        end if;
        (10) {Aserción intermedia}
end loop;
(11) {Aserción intermedia}
b : = (i = j);
(12) {Postcondición}
(13) Expresión cota E
```

25. Predicados Crec(T(a..b)), Decr(T(a..b)) y Perm(T1(a..b), T2(c..d), T3(e..f)) y programa que, dados A(1..n) ordenado crecientemente y B(1..p) ordenado decrecientemente, calcula en C(1..n + p) la mezcla ordenada crecientemente de A(1..n) y B(1..p)

a) Escribir un predicado **Crec**(T(a..b)) que exprese que la sección T(a..b) del vector T es creciente, otro predicado **Decr**(T(a..b)) que exprese que la sección T(a..b) del vector T es decreciente y un tercer predicado **Perm**(T1(a..b), T2(c..d), T3(e..f)) que exprese que la sección T3(e..f) del vector T3 es una permutación de los elementos de las secciones T1(a..b) y T2(c..d) de los vectores T1 y T2 (que sea permutación quiere decir que en T3(e..f) aparecen todos los elementos de T1(a..b) y T2(c..d) y el mismo número de veces).

Ejemplo: Para T1(1..3) = (14, 9, 6), T2(1..7) = (2, 5, 5, 1, 14, 7, 20) y T3(1..8) = (0, 1, 14, 9, 7, 14, 5, 3), la sección T3(2..6) es una permutación de T1(1..2) y T2(4..6).

b) **Documentar**, con las aserciones que se indican y utilizando los predicados definidos en el apartado anterior, el siguiente programa que dados un vector A(1..n) (de enteros) ordenado crecientemente y un vector B(1..p) (de enteros) ordenado decrecientemente, calcula en C(1..n + p) la mezcla ordenada crecientemente de A(1..n) y B(1..p).

Ejemplo: Para A(1..6) = (3, 6, 8, 8, 13, 18) y B(1..3) = (9, 5, 3), el resultado sería C(1..9) = (3, 3, 5, 6, 8, 8, 9, 13, 18)

```
(1) {Precondición}
i : = 1; j : = p; k : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} k ≤ n + p loop
      (4) {Aserción intermedia}
      if i = n + 1 then (5) {Aserción intermedia}
                        C(k) : = B(j); j : = j – 1;
                        (6) {Aserción intermedia}
      elsif j = 0 then  (7) {Aserción intermedia}
                        C(k) : = A(i); i : = i + 1;
                        (8) {Aserción intermedia}
      elsif A(i) ≤ B(j) then  (9) {Aserción intermedia}
                        C(k) : = A(i); i : = i + 1;
                        (10) {Aserción intermedia}
      else (11) {Aserción intermedia}
            C(k) : = B(j);
            (12) {Aserción intermedia}
            j : = j – 1;
            (13) {Aserción intermedia}
      end if;
      (14) {Aserción intermedia}
      k : = k + 1;
      (15) {Aserción intermedia}
end loop;
(16) {Postcondición}
(17) Expresión cota E
```

26. Predicados Par(T(1..s)), Impar(T(1..s)) y Perm(T1(1..p), T2(1..q), T3(1..r)) y programa que devuelve en los vectores B(1..p) y C(1..q) los elementos pares e impares de A(1..n) respectivamente

a) Escribir un predicado **Par(T(1..s))** que exprese que todos los elementos de T(1..n) son pares.

b) Escribir un predicado **Impar(T(1..s))** que exprese que todos los elementos de T(1..n) son impares.

c) Escribir un predicado **Perm(T1(1..v), T2(1..w), T3(1..x))** que exprese que T1(1..v) es una permutación de los elementos de T2(1..w) y T3(1..x). Que sea permutación quiere decir que T1 tiene tantos elementos como T2 y T3 juntos y que cada elemento de T1 aparece en T1 tantas veces como en T2 y T3 juntos.

Ejemplo:
T1(1..7) = (9, 2, 2, 8, 0, 5, 8) es una permutación de T2(1..5) = (0, 2, 8, 2, 5) y T3(1..2) = (8, 9).

d) **Documentar,** con las aserciones que se indican y utilizando los predicados definidos en los apartados anteriores, el siguiente programa iterativo que dado un vector A(1..n), devuelve los vectores B(1..p) y C(1..q) que contienen los elementos pares e impares de A respectivamente.

Ejemplo:
Para A(1..6) = (8, 8, 5, 2, 0, 5) devolvería B(1..4) = (8, 8, 2, 0) y C(1..2) = (5, 5).

```
(1) {Precondición}
i : = 1;   p : = 0;   q : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
        (4) {Aserción intermedia}
        if A(i) mod 2 = 0
        then (5) {Aserción intermedia}
                B(p + 1) : = A(i);
                (6) {Aserción intermedia}
                p : = p + 1;
                (7) {Aserción intermedia}
        else  (8) {Aserción intermedia}
                C(q + 1) : = A(i);
                (9) {Aserción intermedia}
                q : = q + 1;
                (10) {Aserción intermedia}
        end if;
        (11) {Aserción intermedia}
        i : = i + 1;
        (12) {Aserción intermedia}
end loop;
(13) {Postcondición}
(14) Expresión cota E
```

27. Predicado modificado(C(1..p), D(1..p), x, z) y programa que genera una copia modificada de A(1..n) en el vector B(1..n), sustituyendo cada valor de A que sea menor que min por min y cada valor de A que sea mayor que max por max

a) Escribir un predicado **modificado(C(1..p), D(1..p), x, z)** que exprese que **x** es menor o igual que **z** y que el vector D es igual al vector C salvo en aquellas posiciones de C que contienen un valor menor que x o mayor que z, en cuyo caso en D se tiene el valor x o el z respectivamente.

Ejemplo:

x = 2
z = 10

C(1..8)	6	**-7**	**-10**	**20**	8	**43**	5	2
	1	2	3	4	5	6	7	8

D(1..8)	6	**2**	**2**	**10**	8	**10**	5	2
	1	2	3	4	5	6	7	8

b) **Documentar,** con las aserciones que se indican y utilizando el predicado definido en el apartado anterior, el siguiente programa iterativo que dado un vector A(1..n) de enteros y dos valores enteros *min* y *max* (tal que min ≤ max), genera una copia modificada de A(1..n) en el vector B(1..n), de tal forma que cada valor de A que sea menor que min es sustituido por min en B y cada valor de A que sea mayor que max es sustituido por max en B.

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
        if A(i) < min then (4) {Aserción intermedia}
                              B(i) : = min;
                              (5) {Aserción intermedia}
        elsif A(i) > max then (6) {Aserción intermedia}
                                B(i) : = max;
                                (7) {Aserción intermedia}
        else (8) {Aserción intermedia}
            B(i) : = A(i);
            (9) {Aserción intermedia}
        end if;
       (10) {Aserción intermedia}
       i : = i + 1;
       (11) {Aserción intermedia}
end loop;
(12) {Postcondición}
(13) Expresión cota E
```

28. Predicados pares(D(1..r)), impares(D(1..r)) y alternan(D(1..r), E(1..s), F(1..t)) y programa que, dados dos vectores de enteros A(1..n) y B(1..p) (siendo 1 ≤ n ≤ p) donde todos los elementos de A(1..n) son pares y los de B(1..p) impares, calcula en C(1..n + p) el vector que se obtiene alternando los elementos de A y de B mientras sea posible, y copiando al final aquellos elementos de B que hayan podido sobrar tras terminarse el vector A.

a) **Definir** el predicado **pares(D(1..r))** que expresa que todos los elementos de D(1..r) son pares.

b) **Definir** el predicado **impares(D(1..r))** que expresa que todos los elementos de D(1..r) son impares.

c) **Definir** un predicado **alternan(D(1..r), E(1..s), F(1..t))** que exprese que r es menor o igual que s, que t es igual a r más s, que los elementos de D son pares y los de E impares y que los elementos de D y E se alternan en F hasta donde ello es posible, viniendo a continuación los elementos de E que han sobrado.

Ejemplo 1:
En este caso D(1..3), E(1..5) y F(1..8) cumplen el predicado:

D(1..3)	8	20	6
	1	2	3

E(1..5)	5	7	3	9	3
	1	2	3	4	5

F(1..8)	8	5	20	7	6	3	9	3
	1	2	3	4	5	6	7	8

Ejemplo 2:
En este caso D(1..2), E(1..4) y F(1..6) no cumplen el predicado porque los elementos no se alternan:

D(1..2)	4	10
	1	2

E(1..4)	9	7	5	9
	1	2	3	4

F(1..6)	4	7	9	10	5	9
	1	2	3	4	5	6

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..p) (siendo $1 \leq n \leq p$) donde todos los elementos de A(1..n) son pares y los de B(1..p) impares, calcula en C(1..n + p) el vector que se obtiene alternando los elementos de A y de B mientras sea posible, y copiando al final aquellos elementos de B que hayan podido sobrar tras terminarse el vector A.

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
     (4) {Aserción intermedia}
     C((i * 2) – 1) : = A(i);
      (5) {Aserción intermedia}
     C(i * 2) : = B(i);
      (6) {Aserción intermedia}
     i : = i + 1;
     (7) {Aserción intermedia}
end loop;
(8) {Aserción intermedia}
while (9) {Invariante} i ≠ p + 1 loop
     (10) {Aserción intermedia}
     C(n + i) : = B(i);
      (11) {Aserción intermedia}
     i : = i + 1;
      (12) {Aserción intermedia}
end loop;
(13) {Postcondición}
(14) {Expresión cota del primer while}
(15) {Expresión cota del segundo while}
```

29. Predicados tres(C(1..r)) y previos(C(1..r), D(1..r)) y programa que, dado un vector de enteros A(1..n) que solo contiene ceros, unos y doses, calcula en B(1..n) el vector que se obtiene manteniendo los ceros y los unos y sustituyendo los doses por ceros o unos dependiendo de la cantidad previa de ceros y unos

a) **Definir** el predicado **tres(C(1..r))** que expresa que en C(1..r) solo aparecen los valores 0, 1 y 2.

Ejemplo 1:
En este caso C(1..5) cumple el predicado:

C(1..5)	1	2	2	1	0
	1	2	3	4	5

Ejemplo 2:
En este caso C(1..5) también cumple el predicado, porque no aparece ningún valor que no sea 0, 1 o 2:

C(1..5)	2	0	2	2	2
	1	2	3	4	5

b) **Definir** un predicado **previos(C(1..r), D(1..r))** que exprese que siempre que en una posición de C haya un 1 o un 0, en D se tiene lo mismo que en C y cuando en una posición de C se tiene un 2, en D se tendrá:

- Un 1 si en las posiciones previas de D hay menos unos que ceros.
- Un 0 en caso contrario.

Ejemplo:
En este caso C(1..8) y D(1..8) cumplen el predicado:

C(1..8)	0	**2**	0	**2**	1	**2**	1	1
	1	2	3	4	5	6	7	8

D(1..8)	0	**1**	0	**1**	1	**0**	1	1
	1	2	3	4	5	6	7	8

El primer y el segundo 2 de C han sido sustituidos por un 1 en D porque en cada caso en las posiciones previas de D hay menos unos que ceros. El tercer 2 de C ha sido sustituido por un 0 porque en las posiciones previas de D hay menos ceros que unos.

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dado un vector de enteros A(1..n), siendo n ≥ 1 y donde A(1..n) solo contiene ceros, unos y doses, calcula en B(1..n) el vector que se obtiene de la siguiente forma:
- En aquellas posiciones en las que en A se tiene un 1 o un 0, se guardará en B lo que haya en A.
- En aquellas posiciones en las que en A se tiene un 2, se guardará en B un 1 si en las posiciones previas de B hay menos unos que ceros, y se guardará un 0 en caso contrario.

```
(1) {Precondición}
i : = 1; ceros : = 0; unos : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
    (4) {Aserción intermedia}
    if A(i) = 0 then (5) {Aserción intermedia}
                     B(i) : = 0;
                     (6) {Aserción intermedia}
                     ceros : = ceros + 1;
                     (7) {Aserción intermedia}
    elsif A(i) = 1 then (8) {Aserción intermedia}
          B(i) : = 1; unos : = unos + 1;
          (9) {Aserción intermedia}
    elsif unos < ceros then (10) {Aserción intermedia}
          B(i) : = 1; unos : = unos + 1;
          (11) {Aserción intermedia}
    else (12) {Aserción intermedia}
          B(i) : = 0; ceros : = ceros + 1;
          (13) {Aserción intermedia}
    end if;
    (14) {Aserción intermedia}
    i : = i + 1;
    (15) {Aserción intermedia}
end loop;
(16) {Postcondición}
(17) {Expresión cota}
```

30. Predicados bits(D(1..p)) e intercambiado(D(1..p), (d_1, d_2, …, d_p), E(1..p), (e_1, e_2, …, e_p), F(1..p)) y programa que intercambia los elementos de A y B que están en aquellas posiciones en las que C tiene un 1, dejando las demás posiciones tal como están

a) **Definir** un predicado **bits(D(1..p))** que exprese que en D(1..p) sólo hay unos y ceros.

b) **Definir** un predicado **intercambiado(D(1..r), (d_1, d_2, …, d_r), E(1..r), (e_1, e_2, …, e_r), F(1..r), pos)** que exprese que:

- en F(1..r) solo hay unos y ceros
- pos es mayor o igual que 0 y menor o igual que r
- en las posiciones k comprendidas entre 1 y pos (ambas inclusive) en las que en F hay un 0, en D tenemos d_k y en E tenemos e_k
- en las posiciones k comprendidas entre 1 y pos (ambas inclusive) en las que en F tenemos un 1, en D se tiene e_k y en E se tiene d_k.

Al dar este predicado hay que utilizar el predicado del apartado anterior (bits).

Ejemplo:
Los vectores D(1..7), E(1..7) y F(1..7) de este ejemplo cumplen:

intercambiado(D(1..7),(4, **5, 15**, 6, 5, 4, 23), E(1..7), (8, **9, 2**, 30, 7, 2, 10), F(1..7), 5):

D(1..7)	4	9	2	6	5	4	23
	1	2	3	4	5	6	7

E(1..7)	8	5	15	30	7	2	10
	1	2	3	4	5	6	7

F(1..7)	0	**1**	**1**	0	0	1	1
	1	2	3	4	5	6	7

Pero no cumplen

intercambiado(D(1..7),(4, **5, 15**, 6, 5, 4, 23), E(1..7), (8, **9, 2**, 30, 7, 2, 10), F(1..7), 6)

ni

intercambiado(D(1..7),(4, **5, 15**, 6, 5, 4, 23), E(1..7), (8, **9, 2**, 30, 7, 2, 10), F(1..7), 7)

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados tres vectores de enteros A(1..n), B(1..n) y C(1..n), donde C(1..n) solo contiene unos y ceros, intercambia los elementos de A y B que están en aquellas posiciones en las que C tiene un 1, dejando las demás posiciones tal como están:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
    (4) {Aserción intermedia}
    if C(i) = 1 then (5) {Aserción intermedia}
                    aux : = A(i);
                    (6) {Aserción intermedia}
                    A(i) : = B(i);
                    (7) {Aserción intermedia}
                    B(i) : = aux;
                    (8) {Aserción intermedia}
    end if;
    (9) {Aserción intermedia}
    i : = i + 1;
    (10) {Aserción intermedia}
end loop;
(11) {Postcondición}
(12) {Expresión cota}
```

31. Predicados creciente(C(1..r)), permutacion(D(1..r), (d_1, d_2, …, d_r)) y minimo_intervalo(E(1..r), g, h, pos) y programa que ordena A(1..n) dejando los elementos en orden creciente

a) **Definir** un predicado **creciente(C(1..r))** que exprese que los elementos de C(1..r) están ordenados de manera creciente (cada elemento es menor o igual que el siguiente).

b) **Definir** un predicado **permutacion(D(1..r), (d_1, d_2, …, d_r))** que exprese que D(1..r) es una permutación de (d_1, d_2, …, d_r), es decir, que cada elemento de D aparecen el mismo número de veces en D(1..r) que en (d_1, d_2, …, d_r).

c) **Definir** un predicado **minimo_intervalo(E(1..r), g, h, pos)** que exprese que:
- g es mayor o igual que 1 y menor o igual que r, h es mayor o igual que g y menor o igual que r
- pos es mayor o igual que g y menor o igual que h
- el elemento de la posición pos es el mínimo del intervalo definido por g y h.

<u>Ejemplo</u> **(para el predicado minimo_intervalo):**
El vector E(1..10) y los valores g = 3, h = 8 y pos = 6 cumplen el predicado **minimo_intervalo(E(1..10), 3, 8, 6)** porque el menor elemento entre las posiciones 3 y 8 está en la posición 6:

E(1..10)	14	0	10	10	4	**2**	6	20	-6	0
	1	2	3	4	5	6	7	8	9	10

En este intervalo el mínimo está en la posición 6

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dado un vector de enteros A(1..n) ordena el vector dejando los elementos en orden creciente:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
    (4) {Aserción intermedia}
    posmin : = posicion_del_minimo(A(1..n), i);
    (5) {Aserción intermedia}
    aux : = A(i);
    (6) {Aserción intermedia}
    A(i) : = A(posmin);
    (7) {Aserción intermedia}
    A(posmin) : = aux;
    (8) {Aserción intermedia}
    i : = i + 1;
    (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) {Expresión cota}
```

El programa utiliza una **función auxiliar**, *posicion_del_minimo*, que dados un vector y una posición del vector, devuelve la posición del menor elemento a partir de esa posición (se supone que esa función ya está definida).

<u>Ejemplo</u> (para la función auxiliar posicion_del_minimo):

Si tenemos el vector E(1..10)

E(1..10)	14	-20	10	10	4	2	6	20	**-6**	0
	1	2	3	4	5	6	7	8	9	10

posicion_del_minimo(E(1..10), 3) devuelve 9 porque el mínimo a partir de la posición 3 está en la posición 9.

32. Predicados menores(C(1..r), D(1..r), pos) y permutacion(E(1..r), F(1..r), (e_1, e_2, …, e_p), (f_1, f_2, …, f_r)) y programa que intercambia los elementos de A(1..n) y B(1..n) de aquellas posiciones en las que el elemento de A es mayor que el de B

a) **Definir** un predicado **menores(C(1..r), D(1..r), pos)** que exprese que cada elemento de C(1..r) comprendido entre las posiciones 1 y pos (ambas inclusive) es menor o igual que el elemento que ocupa la misma posición en D(1..r).

b) **Definir** un predicado **permutacion(E(1..r), F(1..r), (e_1, e_2, …, e_r), (f_1, f_2, …, f_r))** que exprese que E(1..r) y F(1..r) juntos conforman una permutación de (e_1, e_2, …, e_r) y (f_1, f_2, …, f_r) juntos, es decir:

- Para cada elemento de E(1..r) se cumple que el número de veces que aparece en E(1..r) más el número de veces que aparece en F(1..r) es igual al número de veces que aparece en (e_1, e_2, …, e_p) más el número de veces que aparece en (f_1, f_2, …, f_p).
- Para cada elemento de F(1..r) se cumple que el número de veces que aparece en E(1..r) más el número de veces que aparece en F(1..r) es igual al número de veces que aparece en (e_1, e_2, …, e_r) más el número de veces que aparece en (f_1, f_2, …, f_r).

Ejemplo (para permutación):
Los vectores E(1..5), F(1..5) de este ejemplo cumplen:

permutacion(E(1..5), F(1..5), (4, 4, 15, 6, 8), (8, 9, 7, 8, 7)):

E(1..5)	8	4	8	6	8
	1	2	3	4	5

F(1..5)	7	4	15	9	7
	1	2	3	4	5

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n), intercambia los elementos de A y B de aquellas posiciones en las que el elemento de A es mayor que el de B, dejando las demás posiciones tal como están:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
     (4) {Aserción intermedia}
     if A(i) > B(i) then (5) {Aserción intermedia}
                          aux : = A(i);
                          (6) {Aserción intermedia}
                          A(i) : = B(i);
                          (7) {Aserción intermedia}
                          B(i) : = aux;
                          (8) {Aserción intermedia}
     end if;
     (9) {Aserción intermedia}
     i : = i + 1;
     (10) {Aserción intermedia}
end loop;
(11) {Postcondición}
(12) {Expresión cota}
```

33. Predicados par(x) y recolocado(C(1..r), (c_1, c_2, …, c_r), pos) y programa que siempre que un elemento de A(1..n) de una posición par sea menor que el elemento de la posición impar anterior, los intercambia

a) **Definir** el predicado **par(x)** que exprese que x es par.

b) **Definir** (utilizando el predicado del apartado anterior) el predicado **recolocado(C(1..r), (c_1, c_2, …, c_r), pos)** que exprese que:

- r es par.
- pos es mayor o igual que 0 y menor o igual que r.
- Para cada posición par k que esté comprendida entre 2 y pos y se cumpla que c_k es menor que c_{k-1}, en C(k) se tiene c_{k-1} y en C(k – 1) se tiene c_k.
- Para cada posición par k que esté comprendida entre 2 y pos y se cumpla que c_k no es menor que c_{k-1}, en C(k) se tiene c_k y en C(k – 1) se tiene c_{k-1}.

Ejemplo (para recolocado):
Los vectores C(1..6) y (4, 2, 4, 8, 5, 1) de este ejemplo cumplen:

recolocado(C(1..6), (7, 2, 7, 8, 9, 1), 4):

C(1..6)	2	7	7	8	9	1
	1	2	3	4	5	6

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dado un vector de enteros A(1..n) donde n es par, siempre que un elemento de una posición par sea menor que el elemento de la posición impar anterior, los intercambia:

```
(1) {Precondición}
i : = 2;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 2 loop
    (4) {Aserción intermedia}
    if A(i – 1) > A(i) then    (5) {Aserción intermedia}
                               aux : = A(i – 1);
                               (6) {Aserción intermedia}
                               A(i – 1) : = A(i);
                               (7) {Aserción intermedia}
                               A(i) : = aux;
                               (8) {Aserción intermedia}
    end if;
    (9) {Aserción intermedia}
    i : = i + 2;
    (10) {Aserción intermedia}
end loop;
(11) {Postcondición}
(12) {Expresión cota}
```

34. Predicados perm(C(1..r), (c_1, c_2, …, c_r)) y principio(D(1..r), pos) y programa que siempre que un elemento de A(1..n) de una posición par sea menor que el elemento de la posición impar anterior, los intercambia

a) **Definir** un predicado **perm(C(1..r), (c_1, c_2, …, c_r))** que exprese que el vector C(1..r) es una permutación de (c_1, c_2, …, c_r). Que C(1..r) sea una permutación de (c_1, c_2, …, c_r) quiere decir que cada elemento de C aparece el mismo número de veces en C que en (c_1, c_2, …, c_r).

b) **Definir** el predicado **principio(D(1..r), pos)** que exprese que:
- r es mayor o igual que 1.
- El valor pos es mayor o igual que 1 y menor o igual que r. Dicho de otra forma, pos representa una posición del vector D(1..r).
- Todos los elementos comprendidos entre la posición 1 y la posición pos (ambas inclusive) son iguales al elemento D(1), es decir, son iguales al primer elemento del vector.
- Todos los elementos comprendidos entre la posición pos + 1 y la posición r (ambas inclusive) son distintos al elemento D(1), es decir, son distintos al primer elemento del vector.

Ejemplo (para principio):
El vector D(1..6) de este ejemplo y el valor 3 cumplen el predicado:

principio(D(1..6), 3)

D(1..6)	8	8	8	5	1	5
	1	2	3	4	5	6

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dado un vector de enteros A(1..n), recoloca los elementos de A(1..n) dejando todas las apariciones del valor A(1) en la parte izquierda del vector (que irá desde la posición 1 hasta la p) y todos los demás valores en la parte derecha (desde la posición p + 1 hasta la n):

```
(1) {Precondición}
i : = 2;   p : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
    (4) {Aserción intermedia}
    if A(i) = A(1) then     (5) {Aserción intermedia}
                            A(i) : = A(p + 1);
                            (6) {Aserción intermedia}
                            A(p + 1) : = A(1);
                            (7) {Aserción intermedia}
                            p : = p + 1;
                            (8) {Aserción intermedia}
    end if;
    (9) {Aserción intermedia}
    i : = i + 1;
    (10) {Aserción intermedia}
end loop;
(11) {Postcondición}
(12) {Expresión cota}
```

35. (Abril 2008 #1) Predicados nocerosunos(C(1..r)), bits(C(1..r)) y sustituido(D(1..r), (d_1, d_2, …, d_r), E(1..r), (e_1, e_2, …, e_r), pos) y programa que dados A(1..n) que no contiene ningún cero ni ningún 1 y B(1..n) que solo contiene unos y ceros, sustituye cada cero de B(1..n) por el elemento de A(1..n) que ocupa la misma posición dejando el 0 en esa posición de A(1..n)

a) **Definir** un predicado **nocerosunos(C(1..r))** que exprese que en C(1..r) no hay ningún 0 ni ningún 1.

b) **Definir** un predicado **bits(C(1..r))** que exprese que en C(1..r) solo hay unos y ceros.

c) **Definir** el predicado **sustituido(D(1..r), (d_1, d_2, …, d_r), E(1..r), (e_1, e_2, …, e_r), pos)** que exprese que:
- pos es un valor comprendido entre 0 y r.
- en las posiciones k comprendidas entre 1 y pos (ambas inclusive):
  - si e_k es 0, entonces D(k) es 0 y E(k) es igual a d_k
  - si e_k es 1, entonces D(k) es igual a d_k y E(k) es igual a e_k.

Ejemplo:

Los vectores D(1..8), (4, 9, 2, 6, 20, 0, 0, 1), E(1..8) y (0, 1, 1, 0, 0, 0, 0, 1) de este ejemplo cumplen:

sustituido(D(1..8), (**4, 9, 2, 6, 20**, 0, 0, 1), E(1..8), (**0, 1, 1, 0, 0**, 0, 0, 1), 5):

D(1..8)	**0**	**9**	**2**	**0**	**0**	18	-9	14
	1	**2**	**3**	**4**	**5**	6	7	8

E(1..8)	**4**	**1**	**1**	**6**	**20**	0	0	1
	1	**2**	**3**	**4**	**5**	6	7	8

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n), donde n ≥1, A(1..n) no contiene ningún cero ni ningún 1 y B(1..n) solo contiene unos y ceros, sustituye cada cero de B(1..n) por el elemento de A(1..n) que ocupa la misma posición dejando el 0 en esa posición de A(1..n):

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n loop
    (4) {Aserción intermedia}
    if B(i + 1) = 0 then (5) {Aserción intermedia}
                    B(i + 1) : = A(i + 1);
                    (6) {Aserción intermedia}
                    A(i + 1) : = 0;
                    (7) {Aserción intermedia}
    end if;
    (8) {Aserción intermedia}
    i : = i + 1;
    (9) {Aserción intermedia}
end loop;
(10) {Postcondición}
(11) {Expresión cota}
```

36. (Abril 2008 #2) Predicados pares(C(1..r)), impares(C(1..r)) y movidos(D(1..s), (d_1, d_2, …, d_s), E(1..s), (e_1, e_2, …, e_s), F(1..s), (f_1, f_2, …, f_s), pos) y programa que, dados tres vectores de enteros A(1..n), B(1..n) y C(1..n), donde A(1..n) contiene sólo pares y C(1..n) contiene solo impares, intercambia los elementos de A(1..n) y B(1..n) en las posiciones en las que B(1..n) contiene un número par e intercambia los elementos de A(1..n) y C(1..n) en las posiciones en las que B(1..n) contiene un número impar

a) **Definir** un predicado **pares(C(1..r))** que exprese que en C(1..r) todos son pares.
b) **Definir** un predicado **impares(C(1..r))** que exprese que en C(1..r) todos son impares.
c) **Definir** un predicado

movidos(D(1..s), (d_1, d_2, …, d_s), E(1..s), (e_1, e_2, …, e_s), F(1..s), (f_1, f_2, …, f_s), pos)

que exprese que:

- En D(1..s) todos los elementos son pares y en F(1..s) todos impares.
- pos es un valor comprendido entre 0 y s.
- en las posiciones k comprendidas entre 1 y pos (ambas inclusive):
    - si e_k es par, entonces D(k) es igual a e_k, E(k) es igual a d_k y F(k) es igual a f_k
    - si e_k es impar, entonces D(k) es igual a d_k, E(k) es igual a f_k y F(k) es igual a e_k

Al dar este predicado hay que utilizar los predicados de los apartados anteriores (pares e impares).

Ejemplo:
Los vectores D(1..8), (4, 6, 2, 8, 20, 18, 8, 14), E(1..8), (2, 8, 1, 2, 9, 2, 4, 9), F(1..8) y (3, 9, 3, 7, 11, 3, 15, 7) de este ejemplo cumplen:

movidos(D(1..8), (**4, 6, 2, 8, 20**, 18, 8, 14), E(1..8), (**2, 8, 1, 2, 9**, 2,4, 9), F(1..8), (**3, 9, 3, 7, 11**, 3, 15, 7), 5):

D(1..8)	**2**	**8**	**2**	**2**	**20**	18	8	14
	1	**2**	**3**	**4**	**5**	6	7	8

E(1..8)	**4**	**6**	**3**	**8**	**11**	2	4	9
	1	**2**	**3**	**4**	**5**	6	7	8

F(1..8)	**3**	**9**	**1**	**7**	**9**	3	15	7
	1	**2**	**3**	**4**	**5**	6	7	8

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados tres vectores de enteros A(1..n), B(1..n) y C(1..n), donde n ≥1, A(1..n) contiene solo pares y C(1..n) contiene solo impares, intercambia los elementos de A(1..n) y B(1..n) en las posiciones en las que B(1..n) contiene un número par e intercambia los elementos de A(1..n) y C(1..n) en las posiciones en las que B(1..n) contiene un número impar:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n + 1 loop
     (4) {Aserción intermedia}
     if B(i) mod 2 = 0 then (5) {Aserción intermedia}
                              aux := A(i);
                               (6) {Aserción intermedia}
                              A(i) : = B(i);
                               (7) {Aserción intermedia}
                              B(i) : = aux;
     else aux := C(i);   C(i) := B(i);   B(i) := aux;
     end if;
     (8) {Aserción intermedia}
     i : = i + 1;
end loop;
(9) {Postcondición}
(10) {Expresión cota}
```

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) **Definir** el predicado **par(x)** que exprese que x es par.
b) **Definir** (utilizando el predicado del apartado anterior) el predicado **interpar(C(1..r), (c_1, c_2, …, c_r), D(1..r), (d_1, d_2, …, d_r), pos)** que exprese que:
   - Para cada posición k que esté comprendida entre 1 y pos (ambas inclusive) y se cumpla que c_k y d_k son pares, entonces en C(k) se tiene d_k y en D(k) se tiene c_k.
   - Para cada posición k que esté comprendida entre 1 y pos (ambas inclusive) y se cumpla que por lo menos uno de los dos valores c_k o d_k no es par, entonces en C(k) se tiene c_k y en D(k) se tiene d_k.

Ejemplo (para interpar):
Los vectores C(1..6), (4, 9, 8, 2, 15, 20), D(1..6), (2, 10, 6, 8, 3, 12) de este ejemplo cumplen:

interpar(C(1..6), (**4, 9, 8**, 2, 15, 20), D(1..6), (**2, 10, 6**, 8, 3, 12), **3**)

pero no cumplen:

interpar(C(1..6), (**4, 9, 8, 2**, 15, 20), D(1..6), (**2, 10, 6, 8**, 3, 12), **4**)

porque los elementos de las posiciones 4 en (4, 9, 8, **2**, 15, 20) y (2, 10, 6, **8**, 3, 12) son pares pero en C(1..6) y D(1..6) no están intercambiados:

C(1..6)	2	9	6	2	15	20
	1	2	3	4	5	6

D(1..6)	4	10	8	8	3	12
	1	2	3	4	5	6

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente 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:

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≤ n loop
  (4) {Aserción intermedia}
  if A(i) mod 2 = 0 and B(i) mod 2 = 0 then
                              (5) {Aserción intermedia}
                              aux : = A(i);
                              (6) {Aserción intermedia}
                              A(i) : = B(i);
                              (7) {Aserción intermedia}
                              B(i) : = aux;
    end if;
    (8) {Aserción intermedia}
    i : = i + 1;
end loop;
(9) {Postcondición}
(10) {Expresión cota}
```

38. (Septiembre 2008) Predicados par(x) y movpares(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 el elemento de una posición sea par en A(1..n), lo intercambia con el que ocupa la misma posición en B(1..n), y cuando en A(1..n) se tiene un impar, lo sustituye por un -1 manteniendo igual el elemento de B(1..n)

a) **Definir** el predicado **par(x)** que exprese que x es par.
b) **Definir** (utilizando el predicado del apartado anterior) el predicado **movpares(C(1..r), (c_1, c_2, …, c_r), D(1..r), (d_1, d_2, …, d_r), pos)** que exprese que:
   - **r** es mayor o igual que 1.
   - **pos** es un valor mayor o igual que 0 y menor o igual que r.
   - Para cada posición k que esté comprendida entre 1 y pos (ambas inclusive) y se cumpla que c_k es par, entonces en C(k) se tiene d_k y en D(k) se tiene c_k.
   - Para cada posición k que esté comprendida entre 1 y pos (ambas inclusive) y se cumpla que c_k es impar, entonces en C(k) se tiene -1 y en D(k) se tiene d_k.

Ejemplo (para movido):

Los vectores C(1..6), (10, 5, 8, 15, 2, 17), D(1..6) y (11, 0, 6, 68, 33, 80) de este ejemplo cumplen:

movpares(C(1..6), (10, 5, 8, 15, 2, 17), D(1..6), (11, 0, 6, 68, 33, 80), **3**)

pero no cumplen

movpares(C(1..6), (10, 5, 8, 15, 2, 17), D(1..6), (11, 0, 6, 68, 33, 80), **4**)

porque en la posición 4 de C debería haber un -1:

C(1..6)	11	-1	6	15	2	17
	1	2	3	4	5	6

D(1..6)	10	0	8	68	33	80
	1	2	3	4	5	6

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n), siempre que el elemento de una posición sea par en A(1..n), lo intercambia con el que ocupa la misma posición en B(1..n), y cuando en A(1..n) se tiene un impar, lo sustituye por un -1 manteniendo igual el elemento de B(1..n). El valor de n será mayor o igual que 1:

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i < n loop
  (4) {Aserción intermedia}
  i : = i + 1;
  (5) {Aserción intermedia}
  if A(i) mod 2 = 0    then    (6) {Aserción intermedia}
                               aux : = A(i);
                               (7) {Aserción intermedia}
                               A(i) : = B(i);
                               (8) {Aserción intermedia}
                               B(i) : = aux;
                               (9) {Aserción intermedia}
  else   (10) {Aserción intermedia}
         A(i) : = -1;
         (11) {Aserción intermedia}
  end if;
  (12) {Aserción intermedia}
end loop;
(13) {Postcondición}
(14) {Expresión cota}
```

39. (Abril 2009 #1) Predicados par(x) y sumado(D(1..r), (d_1, d_2, …, d_r), E(1..r), (e_1, e_2, …, e_r), pos) y programa que suma los elementos pares de A(1..n) a los elementos de B(1..n) que ocupan la misma posición, dejando un 0 en esas posiciones de A(1..n)

a) **Definir** un predicado **par(x)** que exprese que x es par.

b) **Definir** el predicado **sumado(D(1..r), (d_1, d_2, …, d_r), E(1..r), (e_1, e_2, …, e_r), pos)** que exprese que:

- pos es un valor comprendido entre 0 y r.
- en las posiciones k comprendidas entre 1 y pos (ambas inclusive):
  - si d_k es par, entonces D(k) es 0 y E(k) es igual a $d_k + e_k$
  - si d_k no es par, entonces D(k) es igual a d_k y E(k) es igual a e_k.

Al dar este predicado hay que <u>utilizar el predicado del apartado a)</u>.

Ejemplo:
Los vectores D(1..8), (4, 9, 2, 6, 17, 4, 9, 2), E(1..8) y (7, 8, 2, 1, 3, 5, 15, 3) de este ejemplo cumplen:

sumado(D(1..8), (**4, 9, 2, 6, 17**, 4, 9, 2), E(1..8), (**7, 8, 2, 1, 3**, 5, 15, 3), 5):

D(1..8)	**0**	**9**	**0**	**0**	**17**	4	9	2
	1	**2**	**3**	**4**	**5**	6	7	8

E(1..8)	**11**	**8**	**4**	**7**	**3**	5	15	3
	1	**2**	**3**	**4**	**5**	6	7	8

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n), donde n ≥ 1, suma los elementos pares de A(1..n) a los elementos de B(1..n) que ocupan la misma posición, dejando además un 0 en esas posiciones de A(1..n):

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n loop
    (4) {Aserción intermedia}
    i : = i + 1;
    (5) {Aserción intermedia}
    if A(i) mod 2 = 0 then (6) {Aserción intermedia}
                           B(i) : = A(i) + B(i);
                           (7) {Aserción intermedia}
                           A(i) : = 0;
    end if;
end loop;
(8) {Postcondición}
(9) {Expresión cota}
```

40. (Abril 2009 #2) Predicados opuestos(C(1..r), D(1..r)) y cambioneg(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 A(1..n) y B(1..n) de signo opuesto, intercambia cada elemento negativo de A(1..n) por el elemento que ocupa la misma posición de B(1..n)

a) **Definir** el predicado **opuestos(C(1..r), D(1..r))** que exprese que:

- Para toda posición k comprendida entre 1 y r (ambas inclusive) se cumple que si C contiene un valor negativo (< 0) en esa posición, entonces D contiene un valor no negativo (≥ 0) en esa posición.
- Para toda posición k comprendida entre 1 y r (ambas inclusive) se cumple que si C contiene un valor no negativo (≥ 0) en esa posición, entonces D contiene un valor negativo (< 0) en esa posición.

Ejemplo:
Los vectores C(1..8) y D(1..8) de este ejemplo cumplen: opuestos(C(1..8), D(1..8))

C(1..8)	3	-8	-5	0	7	-4	-2	7
	1	2	3	4	5	6	7	8

D(1..8)	-25	0	4	-35	-12	10	15	-3
	1	2	3	4	5	6	7	8

b) **Definir** el predicado **cambioneg(C(1..r), (c_1, c_2, …, c_r), D(1..r), (d_1, d_2, …, d_r), pos)** que exprese que:

- pos es mayor o igual que 0 y menor o igual que r.
- en las posiciones k comprendidas entre 1 y pos (ambas inclusive):
  - si c_k es negativo, entonces C(k) es igual a d_k y D(k) es igual a c_k
  - si c_k no es negativo, entonces C(k) es igual a c_k y D(k) es igual a d_k.

Ejemplo:
Los vectores C(1..8), (5, -40, -5, 0, 7, -4, -24, 77), D(1..8) y (-60, -6, 4, -80, 5, 90, 15, -6) de este ejemplo cumplen:

cambioneg(C(1..8), (**5, -40, -5, 0, 7**, -4, -24, 77), D(1..8), (**-60, -6, 4, -80, 5**, 90, 15, -6), 5):

C(1..8)	**5**	**-6**	**4**	**0**	**7**	-4	-24	77
	1	**2**	**3**	**4**	**5**	6	7	8

D(1..8)	**-60**	**-40**	**-5**	**-80**	**5**	90	15	-6
	1	**2**	**3**	**4**	**5**	6	7	8

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que:

- <u>dados dos vectores</u> de enteros A(1..n) y B(1..n), donde **n ≥ 1** y donde A(1..n) y B(1..n) son de signo opuesto, es decir, en aquellas posiciones en las que **A(1..n) contiene un negativo** (< 0) ocurre que **B(1..n) contiene un no negativo** (≥ 0) y en aquellas posiciones en las que **A(1..n) contiene un no negativo** (≥ 0) ocurre que **B(1..n) contiene un negativo** (< 0),

- <u>intercambia</u> cada elemento negativo de A(1..n) por el elemento que ocupa la misma posición de B(1..n):

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n loop
    (4) {Aserción intermedia}
    if A(i + 1) < 0 then (5) {Aserción intermedia}
                    aux : = A(i +1);
                    (6) {Aserción intermedia}
                    A(i + 1) : = B(i + 1);
                    (7) {Aserción intermedia}
                    B(i + 1) : = aux;
    end if;
    (8) {Aserción intermedia}
    i : = i + 1;
end loop;
(9) {Postcondición}
(10) {Expresión cota}
```

41. (Junio 2009) Predicados par(x), parimpar(C(1..r), D(1..r)) y cambiadopar(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 A(1..n) y B(1..n) de signo opuesto, decrementa en 1 cada elemento impar de A(1..n) e incrementa en 1 cada elemento par de B(1..n)

a) **Definir** el predicado **par(x)** que exprese que x es par.

b) **Definir**, utilizando el predicado del apartado anterior, el predicado **parimpar(C(1..r), D(1..r))** que exprese que:
- Para toda posición k comprendida entre 1 y r (ambas inclusive) se cumple que si C contiene un valor par en esa posición, entonces D contiene un valor impar en esa posición.
- Para toda posición k comprendida entre 1 y r (ambas inclusive) se cumple que si C contiene un valor impar en esa posición, entonces D contiene un valor par en esa posición.

Ejemplo:
Los vectores C(1..8) y D(1..8) de este ejemplo cumplen: parimpar(C(1..8), D(1..8))

C(1..8)	4	8	7	0	9	5	-2	7
	1	2	3	4	5	6	7	8

D(1..8)	-25	3	18	7	16	10	15	2
	1	2	3	4	5	6	7	8

c) **Definir** el predicado
cambiadopar(C(1..r), (c_1, c_2, …, c_r), D(1..r), (d_1, d_2, …, d_r), pos)
que exprese que:
- pos es mayor o igual que 0 y menor o igual que r.
- para cada posición k comprendida entre 1 y pos (ambas inclusive):
  - si c_k no es par, entonces C(k) es igual a $c_k - 1$ y D(k) es igual a $d_k + 1$
  - si c_k es par, entonces C(k) es igual a c_k y D(k) es igual a d_k.

Ejemplo:
Los vectores C(1..8), (4, 8, 7, 0, 9, 5, -2, 7), D(1..8) y (-25, 3, 18, 7, 16, 10, 15, 2) de este ejemplo cumplen:

cambiadopar(C(1..8), (**4, 8, 7, 0, 9**, 5, -2, 7), D(1..8), (**-25, 3, 18, 7, 16**, 10, 15, 2), **5**):

C(1..8)	**4**	**8**	**6**	**0**	**8**	5	-2	7
	1	**2**	**3**	**4**	**5**	6	7	8

D(1..8)	**-25**	**3**	**19**	**7**	**17**	10	15	2
	1	**2**	**3**	**4**	**5**	6	7	8

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que:

- dados dos vectores de enteros A(1..n) y B(1..n), donde **n ≥ 1** y donde A(1..n) y B(1..n) cumplen que en aquellas posiciones en las que **A(1..n) contiene un par** ocurre que **B(1..n) contiene un impar** y en aquellas posiciones en las que **A(1..n) contiene un impar** ocurre que **B(1..n) contiene un par**,

- decrementa en 1 cada elemento impar de A(1..n) e incrementa en 1 cada elemento par de B(1..n):

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≤ n loop
  (4) {Aserción intermedia}
  if A(i) mod 2 ≠ 0 then
                    (5) {Aserción intermedia}
                    A(i) : = A(i) – 1;
                    (6) {Aserción intermedia}
                    B(i) : = B(i) + 1;
                    (7) {Aserción intermedia}
  end if;
  (8) {Aserción intermedia}
  i : = i + 1;
end loop;
(9) {Postcondición}
(10) {Expresión cota}
```

42. (Septiembre 2009) Predicados nocerosunos(C(1..r)), bits(C(1..r)) y sustituido(D(1..q), (d_1, d_2, …, d_q), E(1..q), (e_1, e_2, …, e_q), pos) y programa que, dados dos vectores de enteros A(1..n) y B(1..n), donde A(1..n) no contiene ningún cero ni ningún 1 y B(1..n) solo contiene unos y ceros, sustituye cada cero de B(1..n) por el elemento de A(1..n) que ocupa la misma posición dejando el 0 en esa posición de A(1..n)

a) **Definir** un predicado **nocerosunos(C(1..r))** que exprese que en C(1..n) no hay ningún 0 ni ningún 1.
b) **Definir** un predicado **bits(C(1..r))** que exprese que en C(1..n) solo hay unos y ceros.
c) **Definir** el predicado
sustituido(D(1..q), (d_1, d_2, …, d_q), E(1..q), (e_1, e_2, …, e_q), pos)
que exprese que:
- pos es un valor mayor o igual que 0 y menor o igual que q.
- en las posiciones k comprendidas entre 1 y pos (ambas inclusive):
  - si e_k es 0, entonces D(k) es 0 y E(k) es igual a d_k
  - si e_k es 1, entonces D(k) es igual a d_k y E(k) es igual a e_k.

Ejemplo:
Los vectores D(1..8), (4, 9, 2, 6, 20, 18, -9, 14), E(1..8) y (0, 1, 1, 0, 0, 0, 0, 1) de este ejemplo cumplen:

sustituido(D(1..8), (**4, 9, 2, 6, 20**, 18, -9, 14), E(1..8), (**0, 1, 1, 0, 0**, 0, 0, 1), **5**):

D(1..8)	**0**	**9**	**2**	**0**	**0**	18	-9	14
	1	**2**	**3**	**4**	**5**	6	7	8

E(1..8)	**4**	**1**	**1**	**6**	**20**	0	0	1
	1	**2**	**3**	**4**	**5**	6	7	8

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados dos vectores de enteros A(1..n) y B(1..n), donde n ≥ 1, A(1..n) no contiene ningún cero ni ningún 1 y B(1..n) solo contiene unos y ceros, sustituye cada cero de B(1..n) por el elemento de A(1..n) que ocupa la misma posición dejando el 0 en esa posición de A(1..n):

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n loop
    (4) {Aserción intermedia}
    i : = i + 1;
    (5) {Aserción intermedia}
    if B(i) = 0 then (6) {Aserción intermedia}
                     B(i) : = A(i);
                     (7) {Aserción intermedia}
                     A(i) : = 0;
                     (8) {Aserción intermedia}
    end if;
end loop;
(9) {Postcondición}
(10) {Expresión cota}
```

43. (Abril 2010 #1) Predicados par(x), impares(D(1..p)) y parpar(E(1..r), (e_1, e_2, …, e_r), F(1..r), (f_1, f_2, …, f_r), G(1..r), (g_1, g_2, …, g_r), pos) y programa que, dados tres vectores de enteros A(1..n), B(1..n) y C(1..n) y sabiendo que n ≥ 1 y que al principio todos los elementos de C(1..n) son impares, intercambia los elementos de B(1..n) y C(1..n) que ocupan la misma posición siempre que los elementos de esa posición en A(1..n) y B(1..n) sean números pares.

a) Definir el predicado **par(x)** que expresa que x es par.

b) Definir el predicado **impares(D(1..p))** que expresa que todos los elementos del vector D(1..p) son impares. Utilizar el predicado del apartado a).

c) Definir el predicado **parpar(E(1..r), (e_1, e_2, …, e_r), F(1..r), (f_1, f_2, …, f_r), G(1..r), (g_1, g_2, …, g_r), pos)** que expresa lo siguiente:
- pos es mayor o igual que 0 y menor o igual que r.
- En las posiciones k comprendidas entre 1 y pos (las posiciones 1 y pos incluidas), si E(k) es par entonces F(k) es impar.
- En las posiciones k comprendidas entre 1 y pos (las posiciones 1 y pos incluidas):
  - Si e_k y f_k son pares, entonces E(k) es igual a e_k, F(k) es igual a g_k y G(k) es igual a f_k.
  - Si e_k o f_k (al menos uno de ellos) es impar, entonces E(k) es igual a e_k, F(k) es igual a f_k y G(k) es igual a g_k.

Al definir este predicado hay que <u>utilizar el predicado del apartado</u> <u>a)</u>.

Ejemplo:
Los vectores E(1..8), (**4, 9, <u>10</u>, <u>12</u>, 17**, 30, 4, 2), F(1..8), (**7, 8, <u>6</u>, <u>4</u>, 3**, 39, 18, 3), G(1..8) y (**19, 11, <u>25</u>, <u>1</u>, 3**, 5, 15, 3) de este ejemplo cumplen lo siguiente:

parpar(E(1..8), (**4, 9, <u>10</u>, <u>12</u>, 17**, 30, 4, 2), F(1..8), (**7, 8, <u>6</u>, <u>4</u>, 3**, 18, 39, 3),
G(1..8), (**19, 11, <u>25</u>, <u>1</u>, 3**, 43, 15, 3), **5**):

donde E(1..8), F(1..8) y G(1..8) son los siguientes vectores:

E(1..8)	**4**	**9**	**<u>10</u>**	**<u>12</u>**	**17**	30	4	2
	1	**2**	**3**	**4**	**5**	6	7	8

F(1..8)	**7**	**8**	**<u>25</u>**	**<u>1</u>**	**3**	18	39	3
	1	**2**	**3**	**4**	**5**	6	7	8

G(1..8)	**19**	**11**	**<u>6</u>**	**<u>4</u>**	**3**	43	15	3
	1	**2**	**3**	**4**	**5**	6	7	8

d) **Documentar** dando las fórmulas que se cumplen en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa. Este programa, dados tres vectores de enteros A(1..n), B(1..n) y C(1..n) y sabiendo que $n \geq 1$ y que al principio todos los elementos de C(1..n) son impares, intercambia los elementos de B(1..n) y C(1..n) que ocupan la misma posición siempre que los elementos de esa posición en A(1..n) y B(1..n) sean números pares.

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n loop
    (4) {Aserción intermedia}
    if A(i + 1) mod 2 = 0 and B(i + 1) mod 2 = 0
    then (5) {Aserción intermedia}
        aux : = B(i + 1);
        (6) {Aserción intermedia}
        B(i + 1) : = C(i + 1);
        (7) {Aserción intermedia}
        C(i + 1) : = aux;
    end if;
    (8) {Aserción intermedia}
    i : = i + 1;
end loop;
(9) {Postcondición}
(10) {Expresión cota E}
```

44. (Abril 2010 #2) Predicados bits(E(1..r)) y rotado(F(1..r), (f_1, f_2, …, f_r), G(1..r), (g_1, g_2, …, g_r), H(1..r), (h_1, h_2, …, h_r), Q(1..r), pos) y programa que, dados cuatro vectores A(1..n), B(1..n), C(1..n) y D(1..n) donde al principio D(1..n) solo contiene ceros y unos, si C(i) es 0 se rota A(i) → B(i) → C(i) → A(i) y si C(i) es 1 se rota A(i) → C(i) → B(i) → A(i)

a) Definir el predicado **bits(E(1..r))** que expresa que en cada posición de E(1..r) hay un 0 o un 1.

b) Definir el predicado **rotado(F(1..r), (f_1, f_2, …, f_r), G(1..r), (g_1, g_2, …, g_r), H(1..r), (h_1, h_2, …, h_r), Q(1..r), pos)** que expresa lo siguiente:

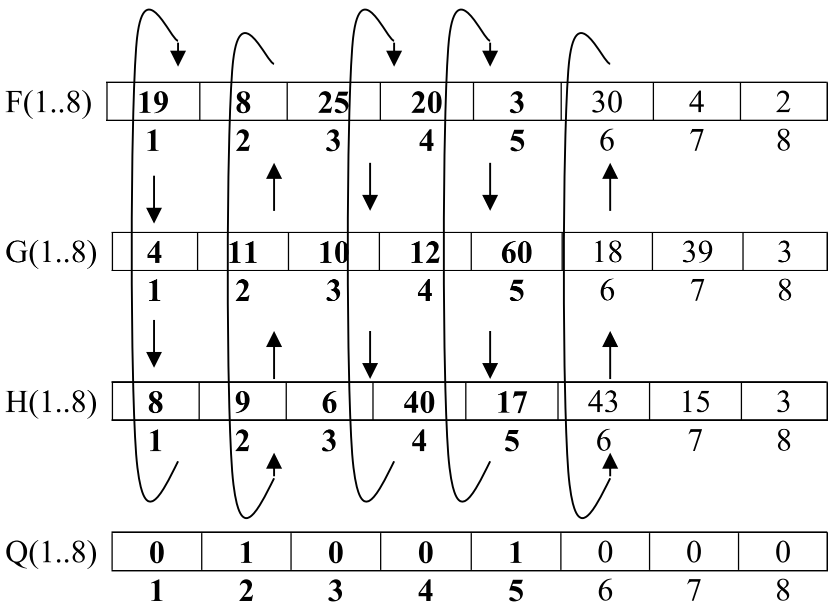
- pos es mayor o igual que 0 y menor o igual que r.
- En cada posición de Q(1..r) hay un 0 o un 1.
- En las posiciones k comprendidas entre 1 y pos (las posiciones 1 y pos incluidas):
  - Si Q(k) es igual a 0, entonces F(k) es igual a h_k, G(k) es igual a f_k y H(k) es igual a g_k.
  - Si Q(k) es igual a 1, entonces F(k) es igual a g_k, G(k) es igual a h_k y H(k) es igual a f_k.

Al definir este predicado hay que utilizar el predicado del apartado a).

Ejemplo:

Los vectores F(1..8), (4, 9, 10, 12, 17, 30, 4, 2), G(1..8), (7, 8, 6, 4, 3, 18, 39, 3), H(1..8), (19, 11, 25, 1, 3, 5, 15, 3) y Q(1..8) de este ejemplo cumplen lo siguiente:

rotado(F(1..8), (**4, 9, 10, 12, 17**, 30, 4, 2), G(1..8), (**7, 8, 6, 40, 3**, 18, 39, 3), H(1..8), (**19, 11, 25, 20, 60**, 43, 15, 3), Q(1..8), **5**):



c) **Documentar** dando las fórmulas que se cumplen en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que, dados cuatro vectores de enteros A(1..n), B(1..n), C(1..n) y D(1..n) y sabiendo que $n \geq 1$ y que al principio D(1..n) solo contiene ceros y unos, hace lo siguiente:
   - Si una posición i de D(1..n) contiene el valor 0 rota hacia abajo los elementos que ocupan esa posición en los otros tres vectores, es decir, los elementos se mueven de la siguiente forma: A(i) → B(i) → C(i) → A(i).
   - En cambio si una posición i de D(1..n) contiene el valor 1, el programa hace rotar hacia arriba los elementos que ocupan esa posición en los otros tres vectores, es decir, los elementos se mueven de la siguiente forma: A(i) → C(i) → B(i) → A(i).

```
(1) {Precondición}
i : = 0;
(2) {Aserción intermedia}
while (3) {Invariante} i ≠ n loop
    (4) {Aserción intermedia}
    i : = i + 1;
    (5) {Aserción intermedia}
    if D(i) = 0
    then (6) {Aserción intermedia}
        aux : = C(i);
        (7) {Aserción intermedia}
        C(i) : = B(i);
        (8) {Aserción intermedia}
        B(i) : = A(i);
        (9) {Aserción intermedia}
        A(i) : = aux;
    else
        aux : = A(i);
        A(i) : = B(i);
        B(i) : = C(i);
        C(i) : = aux;
    end if;
end loop;
(10) {Postcondición}
(11) {Expresión cota E}
```

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) **Definir** el predicado **par(x)** que exprese que x es par.

b) **Definir** el predicado **interdos(C(1..r), (c_1, c_2, …, c_r), pos)** que exprese que:
- pos es mayor o igual que 0 y menor o igual que r.
- para cada posición k comprendida entre 1 y pos (ambas inclusive):
  - si k es par, entonces C(k) es igual a c_{k-1} y C(k – 1) es igual a c_k
  - si k es igual a pos e impar, entonces C(k) es igual a c_k

Ejemplo:
Los vectores C(1..8) y (4, 8, 7, 0, 9, 5, -2, 7) de este ejemplo cumplen:

interdos(C(1..8), (**4, 8, 7, 0, 9**, 10, -2, 7), **5**)
interdos(C(1..8), (**4, 8, 7, 0,** 9, 10, -2, 7), **4**)

C(1..8)	**8**	**4**	**0**	**7**	**9**	10	-2	7
	1	**2**	**3**	**4**	**5**	6	7	8

c) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que dado un vector de enteros A(1..n), donde n ≥ 1, 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.

```
(1) {Precondición}
i : = 1;
(2) {Aserción intermedia}
while (3) {Invariante} i ≤ n loop
  (4) {Aserción intermedia}
  if i mod 2 = 0 then
                    (5) {Aserción intermedia}
                    aux : = A(i – 1);
                    (6) {Aserción intermedia}
                    A(i – 1) : = A(i);
                    (7) {Aserción intermedia}
                    A(i) : = aux;
                    (8) {Aserción intermedia}
  end if;
  (9) {Aserción intermedia}
  i : = i + 1;
end loop;
(10) {Postcondición}
(11) {Expresión cota}
```

46. (Septiembre 2010) Predicados par(x), mulcuatro(x) y intercuatro(D(1..r), (d_1, d_2, …, d_r), pos) y programa que, dado un vector A(1..n), donde n es múltiplo de cuatro, intercambia los elementos de las posiciones pares de dos en dos (posiciones 2 y 4, 6 y 8, etc.)

a) **Definir** el predicado **par(x)** que exprese que x es par.

b) **Definir** el predicado **mulcuatro(x)** que exprese que x es múltiplo de 4.

c) **Definir** el predicado

intercuatro(D(1..r), (d_1, d_2, …, d_r), pos)

que exprese que:

- r es mayor o igual que 1 y además es múltiplo de 4.
- pos es mayor o igual que 0 y menor o igual que r y además múltiplo de 4.
- para cada posición k comprendida entre 1 y pos (ambas inclusive):
  - si k es múltiplo de 4, entonces D(k) es igual a d_{k-2} y D(k – 2) es igual a d_k
  - si k no es múltiplo de 4 pero es par, entonces D(k) es igual a d_{k+2} y D(k + 2) es igual a d_k
  - si k no es par, entonces D(k) es igual a d_k.

Ejemplo:

Los vectores D(1..12) y (5, 9, 2, 10, 20, 18, -9, 17, 0, 3, 6, 1) de este ejemplo cumplen:

intercuatro(D(1..12), (**5, 9, 2, 10, 20, 18, -9, 17**, 0, 3, 6, 1), **8**)

D(1..12)	**5**	**10**	**2**	**9**	**20**	**17**	**-9**	**18**	0	3	6	1
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	9	10	11	12

También se cumple lo siguiente:

intercuatro(D(1..12), (**5, 9, 2, 10,** 20, 18, -9, 17, 0, 3, 6, 1), **4**)

Pero las siguientes tres fórmulas no se cumplen, son false:

intercuatro(D(1..12), (5, 9, 2, 10, 20, 18, -9, 17, 0, 3, 6, 1), **9**)
Es False porque 9 no es múltiplo de 4.

intercuatro(D(1..12), (5, 9, 2, 10, 20, 18, -9, 17, 0, 3, 6, 1), **6**)
Es False porque 6 no es múltiplo de 4

intercuatro(D(1..12), (5, 9, 2, 10, 20, 18, -9, 17, 0, 3, 6, 1), **12**)
Es False porque las posiciones 10 y 12 no cumplen lo establecido por el predicado.

d) **Documentar**, en los puntos indicados y utilizando los predicados definidos en los apartados anteriores, el siguiente programa que dado un vector de enteros A(1..n), donde n ≥ 1 y además n es múltiplo de cuatro, intercambia los elementos de las posiciones pares de dos en dos (posiciones 2 y 4, 6 y 8, etc.).

```
(1) {Precondición}
i : = 4;
(2) {Aserción intermedia}
while (3) {Invariante} i <= n loop
    (4) {Aserción intermedia}
    aux : = A(i – 2);
    (5) {Aserción intermedia}
    A(i – 2) : = A(i);
    (6) {Aserción intermedia}
     A(i) : = aux;
    (7) {Aserción intermedia}
     i : = i + 4;
end loop;
(8) {Postcondición}
(9) {Expresión cota}
```