

Por ejemplo, si el contenido inicial de v es (con DNI's no realistas)

[<111,9.4>, <222,4.3>, <333,3.5>, <444,7.5>, <555,NP>, <666,4.0>, <777,3.7>]

y el contenido de l es

[<222,true>, <333,false>, <777,true>]

el contenido final de v tras ejecutar `aplica_reevaluación(l,v)` ha de ser

[<111,9.4>, <222,5.0>, <333,3.5>, <444,7.5>, <555,NP>, <666,4.0>, <777,5.0>]

Os recordamos la especificación Pre/Post de dos métodos de la clase `Estudiant` que es necesario usar en la implementación de `aplica_reevaluacion`:

```
int consultar_DNI() const;
// Pre: cierto
// Post: el resultado es el DNI del parámetro implícito

void modificar_nota(double nota);
// Pre: el parámetro implícito tiene nota, 0 <= nota <= nota_maxima()
// Post: la nota del parámetro implícito pasa a ser nota
```

Se pide:

- (a) (2.5 puntos) Implementar la acción `aplica_reevaluacion` **iterativamente**.
- (b) (2.5 puntos) Escribir el invariante y la función de cota de todos los bucles del apartado anterior y justificar su corrección.

SOLUCIÓN:

- Opción 1:

```
int i_num_errores(const BinTree<int> &a, const vector<int> &v,  
                 int i, int j)
```

- Opción 2:

```
int ii_num_errores(const BinTree<int> &a, const vector<int> &v,  
                  int &i)
```

Para la opción 1 es esencial tener en cuenta que el árbol dado a es completo, mientras que, para la opción 2, la solución desarrollada probablemente será válida (si es correcta) con cualquier árbol binario, no solo con aquellos que son completos.

Ninguna solución que use una inmersión distinta o altere la cabecera elegida se considerará válida. Asimismo, se valorarán muy negativamente las soluciones innecesariamente ineficientes en tiempo (si éste es mayor que proporcional al tamaño del árbol y del vector) o en espacio (por ejemplo, si se usara también algún vector auxiliar).

Se pide:

- (1 punto) Escribir la especificación Pre/Post de la función de inmersión elegida.
- (2 puntos) Implementar la función de inmersión elegida.
- (1.5 puntos) Justificar la corrección de la función de inmersión elegida incluyendo la demostración de que termina siempre.
- (0.5 puntos) Implementar la función original (no recursiva) `num_errores`.

SOLUCIÓN: