```
Atleta atletaProdigio(const JJOO & j) {
   int i= j.competenciasFinalizadasConOroEnPodio().longitud()-1;
   // Estado e1;
   // vale i@e1==|competenciasConOroEnPodio(j)|-1;


   Atleta a = j.competenciasFinalizadasConOroEnPodio().iesimo(i).ranking().iesimo(0);
   // Estado e2;
   // vale a@e2==campeon(competenciasConOroEnPodio(j)[i@e1]);
   // implica
a@e2==campeon(competenciasConOroEnPodio(j)[|competenciasConOroEnPodio(j)|-1]);

   //vale  Pc:i==|competenciasConOroEnPodio(j)|-1  ∧  i  ≥ 0   ∧  a==a@e2;
   //Bc:i > 0;
   while (i>0){
      //invariante  I: 0 ≤ i ≤ |competenciasConOroEnPodio(j)|-1  ∧
      //((∀ k∈  [i..|competenciasConOroEnPodio(j)|-1])
anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) ≤   anioNacimiento(a))  ∧
      //esCampeon(a,j);

      //Estado eC;
      //vale Bc  ∧  I;
      i--;
      //estado pif;
      //vale i@pif == i@eC-1;
      //implica Pif: (∀ k∈  ( i@pif..|competenciasConOroEnPodio(j)|-1] )
anioNacimiento(campeon(competenciasConOroEnPodio(j)[k]))  ≤ anioNacimiento(a@pif)  ∧
esCampeon(a@pif,,j)

      //B : anioNacimiento(a@pif)  ≤
anioNacimiento(campeon(competenciasFinalizadasConOroEnPodio(j)[i@pif]));
      if (a.anioNacimiento() <=
j.competenciasFinalizadasConOroEnPodio().iesimo(i).ranking().iesimo(0).anioNacimiento
() ){
         //estado iif; vale Pif && B;
          a = j.competenciasFinalizadasConOroEnPodio().iesimo(i).ranking().iesimo(0);
          //estado if1;
         //vale H: a@if1 == campeon(competenciasFinalizadasConOroEnPodio(j)[i@pif])  ∧
anioNacimiento(a@pif) ≤ anioNacimiento(a@if1);
```

}
//estado qif;

//implica i@qif==i@pif;
//vale (B -> a@qif == a@if1) ∧ (¬B -> a@qif==a@pif);
//implica Qif: (∀ k ∈ [i @qif..|competenciasConOroEnPodio(j)|-1]) anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) ≤ anioNacimiento(a@qif) ∧ esCampeon(a@qif,j);

} //v: i ; c == 0;

//Qc: i == 0 ∧ ((∀ k ∈ [i..|competenciasConOroEnPodio(j)|-1]) anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) ≤ anioNacimiento(a)) ∧ esCampeon(a,j);

//estado fc; vale Qc;

return a;

//estado fin: vale a == a@fc ∧ result == a;
//implica Q;
}

1. Pc -> I
   Pc:i==|competenciasConOroEnPodio(j)|-1 ∧ i >=0 ∧ a == a@e2
   -> i==|competenciasConOroEnPodio(j)|-1\ ∧ i ≥ 0 ∧ a==campeon(competenciasConOroEnPodio(j)[|competenciasConOroEnPodio(j)|-1]

   Pc implica:

   a) 0 ≤ i ≤ |competenciasConOroEnPodio(j)|-1
   b) como a==campeon(competenciasConOroEnPodio(j)[|competenciasConOroEnPodio(j)|-1]:
   (∃ c ∈ competenciasConOroEnPodio(j)) a==campeon(c)
   implica:
   esCampeon(a,j);
   c) anioNacimiento(campeon(competenciasConOroEnPodio(j)[|competenciasConOroEnPodio(j)|-1]

) $\leq$ anioNacimiento(a)

implica:

($\forall$ k $\in$ [i..i]) anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) $\leq$ anioNacimiento(a),

por a:

(($\forall$ k $\in$[i..|competenciasConOroEnPodio(j)|-1]) anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) $\leq$ anioNacimiento(a))

Por lo tanto, Pc -> I = a $\wedge$ b $\wedge$ c.

2. (I $\wedge$ ¬Bc) -> Qc

a. ¬Bc: i $\leq$ 0;
b. I -> 0 $\leq$ i $\leq$ |competenciasConOroEnPodio(j)|-1
c. Por a y b, i==0;

Por I por c, vale Qc.

3. El invariante se mantiene. I@eC -> I@qif;

Asumiento la correctitud del if (ver demostración posterior):

a. Por Bc, en el estado eC, vale I: 0<i@eC,
b. Por I, en el estado eC, vale 0 $\leq$ i@eC $\leq$ |competenciasConOroEnPodio(j)|-1
y por otro lado:
($\forall$ k $\in$ [i@ec..|competenciasConOroEnPodio(j)|-1]) anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) $\leq$ anioNacimiento(a@eC) $\wedge$ esCampeon(a@eC , j)

c. en el estado pif:
i@pif = i@ec-1,
i@pif +1 = i@ec,
0 < i@pif + 1
d. Por I, implica, 0 $\leq$ i@pif+1 $\leq$ |competenciasConOroEnPodio(j)|-1
e. Luego,

($\forall$ k $\in$ [(i@pif+1)..|competenciasConOroEnPodio(j)|-1]) anioNacimiento(campeon(competenciasConOroEnPodio (j) [k])) $\leq$ anioNacimiento(a)

$\wedge$ esCampeon(a@pif,j);

f. Luego,

($\forall$ k $\in$ (i@pif..|competenciasConOroEnPodio(j)|-1])
anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) $\leq$ anioNacimiento(a)
$\wedge$ esCampeon(a@pif , j);

Por, d y f, vale I@pif

g. Luego del if, vale Qif.

($\forall$ k $\in$ [i@qif..|competenciasConOroEnPodio(j)|-1])
anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) $\leq$ anioNacimiento(a)
$\wedge$ esCampeon(a@qif , j);

Lo cual implica I@qif.

4. La funcion variante disminuye.

La función variante es v : i .
La cota es c == 0,

En el estado qif: i@qif = i@pif,
En el estado pif: i@pif = i@eC -1,

Luego, v@pif < v@eC,

Por ende, la función variante disminuye.

5. (I $\wedge$ v $\leq$ c) -> ¬B (El ciclo termina)
Por v $\leq$ c, vale i $\leq$ 0, implica ¬B.

Correctitud del If

a. estado pif, implica:
a1. i@pif == i@eC-1
a2. ($\forall$ k$\in$ (i@pif..|competenciasConOroEnPodio(j)|-1])
anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) $\leq$ anioNacimiento(a@pif) $\wedge$
esCampeon(a@pif, ,j)

b. En el estado iif, B implica anioNacimiento(a@pif) $\leq$ anioNacimiento(campeon(competenciasFinalizadasConOroEnPodio(j)[i@pif]))

c.B implica a@if1 == campeon(competenciasFinalizadasConOroEnPodio(j)[i@pif]) $\wedge$ anioNacimiento(a@pif) $\leq$ anioNacimiento(a@if1).

Por b y c: B -> a@qif == a@if1

d. Ahora, ¬B, implica:
anioNacimiento(a@pif) $\geq$
anioNacimiento(campeon(competenciasFinalizadasConOroEnPodio(j)[i@pif]))

e.¬B implica: a@qif == Campeon(competenciasFinalizadasConOroEnPodio(j)[i@pif]) $\wedge$ anioNacimiento(a@pif) $\geq$ anioNacimiento(a@if1).

Por d y e, ¬B -> a@qif==a@pif)

f. Dado que ($\exists$ c $\in$ competenciasConOroEnPodio(j)) a@qif==campeon(c)
implica:
esCampeon(a@qif,,j);

De Pif y lo anterior, el estado qif (donde i@qif==i@pif) implica:

($\forall$ k $\in$ [i @qif..|competenciasConOroEnPodio(j)|-1])
anioNacimiento(campeon(competenciasConOroEnPodio(j)[k])) $\leq$ anioNacimiento(a@qif) $\wedge$
esCampeon(a@qif,j);

Hemos demostrado, aplicando el teorema del invariante, la correctitud y terminación del ciclo, y la correctitud del If, por lo tanto, hemos demostrado la correctitud y terminación del algoritmo.