UBA – Facultad de Ciencias Exactas y Naturales – Departamento de Computación

Algoritmos y Estructura de Datos I

Primer cuatrimestre de 2012

4 de Mayo de 2012

TPF OJOTA (Organización de Juegos Olímpicos Tp de Algoritmos 1) v1.0

1. Tipos

`tipo` Deporte = String ;
`tipo` Pais = String ;
`tipo` Sexo = Femenino, Masculino ;

2. Atleta

`tipo Atleta` {
  `observador nombre` (a: Atleta) : String ;
  `observador sexo` (a: Atleta) : Sexo ;
  `observador añoNacimiento` (a: Atleta) : $\mathbb{Z}$;
  `observador nacionalidad` (a: Atleta) : Pais ;
  `observador ciaNumber` (a: Atleta) : $\mathbb{Z}$;
  `observador deportes` (a: Atleta) : [Deporte] ;
  `observador capacidad` (a: Atleta, d: Deporte) : $\mathbb{Z}$;
    `requiere` $d \in deportes(a)$;
  `invariante` $sinRepetidos(deportes(a))$;
  `invariante` $ordenada(deportes(a))$;
  `invariante capacidadEnRango` : $(\forall d \leftarrow deportes(a))0 \leq capacidad(a, d) \leq 100$;
}

`problema nuevoA` (nom : String, s : Sexo, a : $\mathbb{Z}$, nac : Pais, cia : $\mathbb{Z}$) = `result` : Atleta {
  `asegura` $nombre(result) == nom$;
  `asegura` $sexo(result) == s$;
  `asegura` $añoNacimiento(result) == a$;
  `asegura` $nacionalidad(result) == nac$;
  `asegura` $ciaNumber(result) == cia$;
  `asegura` $deportes(result) == []$;
}

`problema nombreA` (a : Atleta) = `result` : String {
  `asegura` $nombre(a) == result$;
}

`problema sexoA` (a : Atleta) = `result` : Sexo {
  `asegura` $sexo(a) == result$;
}

`problema anioNacimientoA` (a : Atleta) = `result` : $\mathbb{Z}$ {
  `asegura` $añoNacimiento(a) == result$;
}

`problema nacionalidadA` (a : Atleta) = `result` : Pais {
  `asegura` $nacionalidad(a) == result$;
}

`problema ciaNumberA` (a : Atleta) = `result` : $\mathbb{Z}$ {
  `asegura` $ciaNumber(a) == result$;
}

`problema deportesA` (a : Atleta) = `result` : [Deporte] {
  `asegura` $deportes(a) == result$;

}

problema capacidadA (a : Atleta, d : Deporte) = result : $\mathbb{Z}$ {
  requiere $d \in deportes(a)$;
  asegura $capacidad(a, d) == result$;
}

problema entrenarDeporteA (a: Atleta, d: Deporte, c: $\mathbb{Z}$) = result : Atleta {
  requiere $0 \leq c \leq 100$;
  asegura $nombre(result) == nombre(a)$;
  asegura $sexo(result) == sexo(a)$;
  asegura $añoNacimiento(result) == añoNacimiento(a)$;
  asegura $nacionalidad(result) == nacionalidad(a)$;
  asegura $ciaNumber(result) == ciaNumber(a)$;
  asegura $mismos(deportes(result), sacarRepetidos(d : deportes(a)))$;
  asegura $(\forall x \leftarrow deportes(result), x \neq d) capacidad(result, x) == capacidad(a, x)$;
  asegura $capacidad(result, d) == c$;
}

3. Competencia

tipo Competencia {
  observador categoria (c: Competencia) : (Deporte, Sexo) ;
  observador participantes (c: Competencia) : [Atleta] ;
  observador finalizada (c: Competencia) : Bool ;
  observador ranking (c: Competencia) : [Atleta] ;
      requiere $finalizada(c)$;
  observador lesTocoControlAntiDoping (c: Competencia) : [Atleta] ;
      requiere $finalizada(c)$;
  observador leDioPositivo (c: Competencia, a: Atleta) : Bool ;
      requiere $finalizada(c) \land a \in lesTocoControlAntiDoping(c)$;

  invariante participaUnaSolaVez : $sinRepetidos(ciaNumbers(participantes(c)))$;
  invariante participantesPertenecenACat :
     $(\forall p \leftarrow participantes(c)) prm(categoria(c)) \in deportes(p) \land sgd(categoria(c)) == sexo(p)$;
  invariante elRankingEsDeParticipantesYNoHayRepetidos :
     $finalizada(c) \Rightarrow incluida(ranking(c), participantes(c))$;
  invariante seControlanParticipantesYNoHayRepetidos :
     $finalizada(c) \Rightarrow incluida(lesTocoControlAntiDoping(c), participantes(c))$;
}

problema nuevaC (d: Deporte, s: Sexo, as: [Atleta]) = result : Competencia {
  requiere sonDeEstaCategoria : $(\forall a \leftarrow as) d \in deportes(a) \land s == sexo(a)$;
  requiere $noHayRepetidos(ciaNumbers(as))$;
  asegura $categoria(result) == (d, s)$;
  asegura $mismos(participantes(result), as)$;
  asegura $\neg finalizada(result)$;
}

problema categoriaC (c : Competencia) = result : (Deporte,Sexo) {
  asegura $categoria(c) == result$;
}

problema participantesC (c : Competencia) = result : [Atleta] {
  asegura $mismos(participantes(c), result)$;
}

problema finalizadaC (c : Competencia) = result : Bool {
  asegura $finalizada(c) == result$;
}

problema rankingC (c : Competencia) = result : [Atleta] {
  requiere $finalizada(c)$;
  asegura $ranking(c) == result$;

}

problema lesTocoControlAntiDopingC (c : Competencia) = result : [Atleta] {
  requiere $finalizada(c)$;
  asegura $mismos(lesTocoControlAntiDoping(c), result)$;
}

problema leDioPositivoC (c : Competencia, a : Atleta) = result : Bool {
  requiere $finalizada(c)$;
  requiere $a \in lesTocoControlAntiDoping(c)$;
  asegura $leDioPositivo(c, a) == result$;
}

problema finalizarC (c: Competencia, posiciones: [$\mathbb{Z}$], control: [($\mathbb{Z}$, Bool)]) = result : Competencia {
  requiere $\neg finalizada(c)$;
  requiere $incluida(posiciones, ciaNumbers(participantes(c)))$;
  requiere $incluida(primeros(control), ciaNumbers(participantes(c)))$;
  asegura seMantieneCategoria : $categoria(result) == categoria(c)$;
  asegura seMantienenenParticipantes : $mismos(participantes(result), participantes(c))$;
  asegura finalizo : $finalizada(result)$;
  asegura rankingOrdenado : $ciaNumbers(ranking(result)) == posiciones$;
  asegura quienesSeControlan : $mismos(ciaNumbers(lesTocoControlAntiDoping(result)), primeros(control))$;
  asegura resultadosDeControl : $(\forall x \leftarrow control) leDioPositivo(c, elAtleta(participantes(c), prm(x))) \Leftrightarrow sgd(x)$;
  aux elAtleta (as: [Atleta], x:$\mathbb{Z}$) : Atleta $= [a | a \leftarrow as, ciaNumber(a) == x]_0$;
}

problema linfordChristieC (c: Competencia, a: Atleta) = result : Competencia {
  requiere noFinalizada : $\neg finalizada(c)$;
  requiere esParticipante : $a \in participantes(c)$;
  asegura seMantieneCategoria : $categoria(result) == categoria(c)$;
  asegura soloUnoDescalificado : $mismos(a : participantes(result), participantes(c))$;
  asegura noFinalizada : $\neg finalizada(result)$;
}

problema gananLosMasCapacesC (c: Competencia) = result : Bool {
  requiere seConocenResultados : $finalizada(c)$;
  asegura $result == ordenada(reverso(capacidades(ranking(c), deporte(c))))$;
}

problema sancionarTrampososC (c: Competencia) = result : Competencia {
  requiere seConocenResultados : $finalizada(c)$;
  asegura seMantieneCategoria : $categoria(result) == categoria(c)$;
  asegura seMantienenParticipantes : $mismos(participantes(result), participantes(c))$;
  asegura sigueFinalizada : $finalizada(result)$;
  asegura drogadosDescalificados : $ranking(result) == [\, a \,|\, a \leftarrow ranking(c), noLoDescubrenDopado(a, c) \,]$;
  asegura seMantieneControl : $mismos(lesTocoControlAntiDoping(result), lesTocoControlAntiDoping(c))$;
  asegura mismosResultadosControl :
    $(\forall a \leftarrow lesTocoControlAntiDoping(result)) leDioPositivo(result, a) \Leftrightarrow leDioPositivo(c, a)$;

  aux noLoDescubrenDopado (a: Atleta, c: Competencia) : Bool =
    $a \notin lesTocoControlAntiDoping(c) \vee \neg leDioPositivo(c, a)$;
}

4. JJOO

tipo JJOO {
  observador año (j: JJOO) : $\mathbb{Z}$;
  observador atletas (j: JJOO) : [Atleta] ;
  observador cantDias (j: JJOO) : $\mathbb{Z}$;
  observador cronograma (j: JJOO, dia: $\mathbb{Z}$) : [Competencia] ;
      requiere $1 \leq dia \leq cantDias(j)$;
  observador jornadaActual (j: JJOO) : $\mathbb{Z}$;

  invariante atletasUnicos : $sinRepetidos(ciaNumbers(atletas(j)))$;

```
  invariante unaDeCadaCategoria : (∀i, k ← [0.. |competencias(j)|), i ≠ k)
      categoria(competencias(j)_i) ≠ categoria(competencias(j)_k) ;
  invariante competidoresInscriptos : (∀c ← competencias(j))incluida(participantes(c), atletas(j)) ;
  invariante jornadaValida : 1 ≤ jornadaActual(j) ≤ cantDias(j) ;
  invariante finalizadasSiiYaPasoElDia : lasPasadasFinalizaron(j) ∧ lasQueNoPasaronNoFinalizaron(j) ;
}

problema nuevoJ (año: ℤ, as: [Atleta], cron: [[Competencia]]) = result : JJOO  {
  requiere sinRepetidos(ciaNumbers(as)) ;
  requiere (∀cs ← concat(cron))(¬(∃i, j ← [0..|concat(cron)|), i ≠ j)categoria(cs_i) == categoria(cs_j))) ;
  requiere (∀cs ← concat(cron))incluida(participantes(cs), as) ;
  requiere |cron| ≥ 1 ;
  asegura año == año(result) ;
  asegura mismos(atletas(result), as) ;
  asegura |cron| == cantDias(result) ;
  asegura (∀j ← [0..|cron|))mismos(cron_j, cronograma(result, j + 1)) ;
  asegura jornadaActual(result) == 1 ;
}

problema anioJ (j: JJOO) = result : ℤ  {
  asegura año(j) == result ;
}

problema atletasJ (j: JJOO) = result : [Atleta]  {
  asegura mismos(atletas(j), result) ;
}

problema cantDiasJ (j: JJOO) = result : ℤ  {
  asegura cantDias(j) == result ;
}

problema cronogramaJ (j: JJOO, d: ℤ) = result : [Competencia]  {
  requiere 1 ≤ d ≤ cantDias(j) ;
  asegura cronograma(j, d) == result ;
}

problema jornadaActualJ (j: JJOO) = result : ℤ  {
  asegura jornadaActual(j) == result ;
}

problema dePaseoJ (j: JJOO) = result : [Atleta]  {
  asegura noParticipanEnNinguna : mismos(result, fueronAPasear(j)) ;

  aux fueronAPasear (j: JJOO) : [Atleta]  = [ a | a ← atletas(j), ¬(∃c ← competencias(j))a ∈ participantes(c) ] ;
}

problema medalleroJ (j: JJOO) = result : [(Pais, [ℤ])]  {
  asegura paisesConMedallas : mismos(primeros(result), paisesQueGanaron(j)) ;
  asegura cantidadMedallasCorrecta : (∀m ← result) |sgd(m)| == 3 ∧
      sgd(m)_0 == |filtrarPorPais(medallistasOro(j), prm(m))| ∧
      sgd(m)_1 == |filtrarPorPais(medallistasPlata(j), prm(m))| ∧
      sgd(m)_2 == |filtrarPorPais(medallistasBronce(j), prm(m))| ;
  asegura bienOrdenada : (∀i ← (0.. |result|))masMedallas(sgd(result_{i−1}), sgd(result_i)) ;

  aux paisesQueGanaron (j: JJOO) : [Pais]  = sacarRepetidos(nacionalidades(medallistasOro(j) + +
      medallistasPlata(j) + +medallistasBronce(j))) ;
  aux masMedallas (x, y: [ℤ]) : Bool  = x_0 > y_0 ∨ (x_0 == y_0 ∧ x_1 > y_1) ∨ (x_0 == y_0 ∧ x_1 == y_1 ∧ x_2 ≥ y_2) ;
}

problema boicotPorDisciplinaJ (j: JJOO, cat: (Deporte, Sexo), p: Pais) = result : (ℤ, JJOO)  {
  requiere esCategoriaValida : (∃c ← competencias(j))categoria(c) == cat ;
  asegura soloCambiaCronograma : año(j) == año(sgd(result)) ∧ cantDias(j) == cantDias(sgd(result))
      ∧ jornadaActual(j) == jornadaActual(sgd(result)) ∧ mismos(atletas(j), atletas(sgd(result))) ;
  asegura mismaCantDeCompetencias : (∀d ← [1..cantDias(j)]) |cronograma(j, d)| == |cronograma(sgd(result), d)| ;
  asegura lasOtrasCompetenciasNoCambian : (∀d ← [1..cantDias(j)])(∀c ← cronograma(j, d), categoria(c) ≠ cat)
      laCompetenciaSeMantiene(sgd(result), d, c) ;
```

asegura boicotAEsaCat : $(\exists c \leftarrow cronograma(sgd(result), elDiaDeEsaCat(j, cat)))$
$igualSalvoBoicot(c, competenciaDeCat(j, cat), p)$;
asegura $prm(result) == |filtrarPorPais(participantes(competenciaDeCat(j, cat)), p)|$;

aux elDiaDeEsaCat (j: JJOO, cat: (Deporte, Sexo)) : $\mathbb{Z}$ =
$[\, d \,|\, d \leftarrow [1..cantDias(j)], (\exists c \leftarrow cronograma(j, d))categoria(c) == cat\,]_0$;
aux competenciaDeCat (j: JJOO, cat: (Deporte, Sexo)) : Competencia = $[\, c \,|\, c \leftarrow competencias(j), categoria(c) == cat\,]_0$;
aux igualSalvoBoicot (c, prec: Competencia, p: Pais) : Bool = $categoria(c) == categoria(prec) \wedge$
$mismos(participantes(c), sacarLosDePais(participantes(prec), p) \wedge finalizada(c) \Leftrightarrow finalizada(prec)$
$\wedge\, finalizada(c) \Rightarrow (ranking(c) == sacarLosDePais(ranking(prec), p) \wedge$
$mismos(lesTocoControlAntiDoping(c), sacarLosDePais(lesTocoControlAntiDoping(prec), p))$
$\wedge\, (\forall a \leftarrow lesTocoControlAntiDoping(c))leDioPositivo(c, a) \Leftrightarrow leDioPositivo(prec, a))$;
aux sacarLosDePais (as: [Atleta], p: Pais) : [Atleta] = $[\, a \,|\, a \leftarrow as, nacionalidad(a) \neq p\,]$;
}

problema losMasFracasadosJ (j: JJOO, p: Pais) = result : [Atleta] {
asegura $mismos(result, noGanaronMedallas(j, losMasParticipantes(j, atletasDelPais(j, p))))$;

aux atletasDelPais (j: JJOO, p: Pais) : [Atleta] = $[\, a \,|\, a \leftarrow atletas(j), nacionalidad(a) == p\,]$;
aux losMasParticipantes (j: JJOO, as: [Atleta]) : [Atleta] = $[\, a \,|\, a \leftarrow as,$
$(\forall x \leftarrow as)cantCompetencias(j, a) \geq cantCompetencias(j, x)\,]$;
aux cantCompetencias (j: JJOO, a: Atleta) : $\mathbb{Z}$ = $|[\, c \,|\, c \leftarrow competencias(j), a \in participantes(c)\,]|$;
aux noGanaronMedallas (j: JJOO, as: [Atleta]) : [Atleta] = $[\, a \,|\, a \leftarrow as, cantMedallas(j, a) == 0\,]$;
aux cantMedallas (j: JJOO, a: Atleta) : $\mathbb{Z}$ = $|[\, c \,|\, c \leftarrow competencias(j), estaEnElPodio(c, a)\,]|$;
aux estaEnElPodio (c: Competencia, a: Atleta) : Bool = $finalizada(c) \wedge (salioPrimero(c, a) \vee salioSegundo(c, a) \vee$
$salioTercero(c, a))$;
aux salioPrimero (c: Competencia, a: Atleta) : Bool = $|ranking(c)| \geq 1 \wedge ranking(c)_0 == a$;
aux salioSegundo (c: Competencia, a: Atleta) : Bool = $|ranking(c)| \geq 2 \wedge ranking(c)_1 == a$;
aux salioTercero (c: Competencia, a: Atleta) : Bool = $|ranking(c)| \geq 3 \wedge ranking(c)_2 == a$;
}

problema liuSongJ (j: JJOO, a: Atleta, p: País) = result : JJOO {
requiere estaLiu : $a \in atletas(j)$;
asegura loDemasIgual : $año(j) == año(result) \wedge cantDias(j) == cantDias(result)$
$\wedge\, jornadaActual(j) == jornadaActual(result)$;
asegura mismaCantidadAtletas : $|atletas(j)| == |atletas(result)|$;
asegura atletasIguales : $(\forall at1 \leftarrow atletas(j), \neg(at1 == a))at1 \in atletas(result)$;
asegura cambioLiu : $(\forall at1 \leftarrow atletas(j), at1 == a)(\exists at2 \leftarrow atletas(result))igualSalvoPais(at1, at2, p)$;
asegura mismaCantDeCompetencias : $(\forall d \leftarrow [1..cantDias(j)])\, |cronograma(j, d)| == |cronograma(result, d)|$;
asegura lasOtrasCompetenciasNoCambian : $(\forall d \leftarrow [1..cantDias(j)])(\forall c \leftarrow cronograma(j, d), a \notin participantes(c))$
$laCompetenciaSeMantiene(result, d, c)$;
asegura cambianLasDeLiu : $(\forall d \leftarrow [1..cantDias(j)])(\forall c \leftarrow cronograma(j, d), a \in participantes(c))$
$(\exists c2 \leftarrow cronograma(result, d))igualSalvoLiu(c, c2, a, p)$;

aux igualSalvoPais (at1: Atleta, at2: Atleta, p: Pais) : Bool = $nombre(at1) == nombre(at2) \wedge$
$sexo(at1) == sexo(at2) \wedge añoNacimiento(at1) == añoNacimiento(at2)$
$\wedge\, ciaNumber(at1) == ciaNumber(at2) \wedge deportes(at1) == deportes(at2)$
$\wedge\, (\forall d \leftarrow deportes(at1))capacidad(at1, d) == capacidad(at2, d) \wedge nacionalidad(at2) == p$;
aux igualSalvoLiu (c1: Competencia, c2: Competencia, a: Atleta, p: Pais) : Bool = $categoria(c1) == categoria(c2)$
$\wedge\, participantesYLiu(c1, c2, a, p) \wedge finalizada(c1) \Leftrightarrow finalizada(c2) \wedge finalizada(c1) \Rightarrow$
$(rankingYLiu(c1, c2, a, p) \wedge mismosControladosYLiu(c1, c2, a, p))$;
aux participantesYLiu (c1: Competencia, c2: Competencia, a: Atleta, p: Pais) : Bool =
$|participantes(c1)| == |participantes(c2)|$
$\wedge\, (\forall at1 \leftarrow participantes(c1), at1! = a)at1 \in participantes(c2)$
$\wedge\, (\forall at1 \leftarrow participantes(c1), at1 == a)(\exists at2 \leftarrow participantes(c2))igualSalvoPais(at1, at2, p)$;
aux rankingYLiu (c1: Competencia, c2: Competencia, a: Atleta, p: Pais) : Bool = $|ranking(c1)| == |ranking(c2)|$
$\wedge\, (\forall i \leftarrow [0..|ranking(c1)|), ranking(c1)_i! = a)ranking(c2)_i == ranking(c1)_i$
$\wedge\, (\forall i \leftarrow [0..|ranking(c1)|), ranking(c1)_i == a)igualSalvoPais(ranking(c1)_i, ranking(c2)_i, p)$;
aux mismosControladosYLiu (c1: Competencia, c2: Competencia, a: Atleta, p: Pais) : Bool =
$|lesTocoControlAntiDoping(c1)| == |lesTocoControlAntiDoping(c2)| \wedge$
$(\forall at1 \leftarrow lesTocoControlAntiDoping(c1), at1! = a)at1 \in lesTocoControlAntiDoping(c2) \wedge leDioPositivo(c1, at1) ==$
$leDioPositivo(c2, at1) \wedge$
$(\forall at1 \leftarrow lesTocoControlAntiDoping(c1), at1 == a)(\exists at2 \leftarrow lesTocoControlAntiDoping(c2))$
$igualSalvoPais(at1, at2, p) \wedge leDioPositivo(c1, at1) == leDioPositivo(c2, at2)$;

}

problema stevenBradburyJ (j: JJOO) = result : Atleta {
  requiere alguienGanoMedalla : $(\exists d \leftarrow [1..jornadaActual(j)])(\exists c \leftarrow cronograma(j,d), finalizada(c))\,|ranking(c)| > 0$;
  asegura ganoMedallaDeOro : $result \in primeros(ganadoresPorCategoria(j))$;
  asegura elMenosCapaz : $(\forall a \leftarrow primeros(ganadoresPorCategoria(j)))peorDesempeño(result, j) \leq peorDesempeño(a, j)$;

  aux ganadoresPorCategoria (j: JJOO) : [(Atleta, (Deporte, Sexo))] = $[\,(ranking(c)_0, categoria(c))\,|$
    $d \leftarrow [1..jornadaActual(j)], c \leftarrow cronograma(j,d), finalizada(c) \wedge |ranking(c)| > 0\,]$;
  aux peorDesempeño (a: Atleta, j: JJOO) : $\mathbb{Z}$ =
    $minimo([\,capacidad(a, prm(sgd(g)))\,|\,g \leftarrow ganadoresPorCategoria(j), prm(g) == a\,])$;
}

problema uyOrdenadoAsíHayUnPatrónJ (j: JJOO) = result : Bool {
  asegura $siguenSiempreElMismoOrden(losMejoresPaises(j)) == result$;

  aux losMejoresPaisesJ (j: JJOO) : [Pais] = $[\,mejorEseDia(j,i)\,|\,i \leftarrow [1..jornadaActual(j)], alguienGanoOro(j,i))\,]$;
  aux mejorEseDia (j: JJOO, d: $\mathbb{Z}$) : Pais = $[\,p\,|\,p \leftarrow paises(j),$
    $\neg(\exists p2 \leftarrow paises(j))(cantOro(j,p2,d) > cantOro(j,p,d) \vee (cantOro(j,p2,d) == cantOro(j,p,d) \wedge p2 < p))\,]_0$;
  aux cantOro (j: JJOO, p: Pais, d: $\mathbb{Z}$) : $\mathbb{Z}$ = $|[\,1\,|\,c \leftarrow cronograma(j,d), finalizada(c)$
    $\wedge\, ranking(c) \geq 1 \wedge nacionalidad(ranking(c)_0) == p\,]|$;
  aux alguienGanoOro (j: JJOO, d: $\mathbb{Z}$) : Bool = $(\exists c \leftarrow cronograma(j,d))finalizada(c) \wedge ranking(c) \geq 1$;
  aux siguenSiempreElMismoOrden (ps:[Pais]) : Bool = $(\forall i, j \leftarrow [0..\,|ps|-1), i < j \wedge ps_i == ps_j)ps_{i+1} == ps_{j+1}$;
}

problema sequíaOlímpicaJ (j: JJOO) = result : [País] {
  asegura $mismos(result, secosOlimpicos(j))$;

  aux secosOlimpicos (j: JJOO) : [País] = $[\,p\,|\,p \leftarrow paises(j), masDiasSinMedallas(j,p) == maxDiasSinMedallas(j)\,]$;
  aux masDiasSinMedallas (j: JJOO, p: País) : $\mathbb{Z}$ = $maxDif(0 : [\,i\,|\,i \leftarrow [1..jornadaActual(j)),$
    $GanoMedallaEseDia(j,p,i)\,] + +[jornadaActual(j)])$;
  aux maxDif (ls:[$\mathbb{Z}$]) : $\mathbb{Z}$ = $max([\,ls_i - ls_{i-1}\,|\,i \leftarrow [1..\,|ls|)\,])$;
  aux GanoMedallaEseDia (j: JJOO, p: Pais, i: $\mathbb{Z}$) : Bool = $(\exists c \leftarrow cronograma(j,i))$
    $(|ranking(c)| \geq 1 \wedge nacionalidad(ranking(c)_0) == p)$
    $\vee\, (|ranking(c)| \geq 2 \wedge nacionalidad(ranking(c)_1) == p)$
    $\vee\, (|ranking(c)| \geq 3 \wedge nacionalidad(ranking(c)_2) == p)$;
  aux maxDiasSinMedallas (j: JJOO) : $\mathbb{Z}$ = $max([\,masDiasSinMedallas(j,p)\,|\,p \leftarrow paises(j)\,])$;
}

problema transcurrirDiaJ (j: JJOO) = result : JJOO {
  requiere losJuegosNoTerminaron : $jornadaActual(j) < cantDias(j)$;
  asegura seMantieneAño : $año(j) == año(result)$;
  asegura seMantienenAtletas : $mismos(atletas(j), atletas(result))$;
  asegura seMantienenDias : $cantDias(j) == cantDias(result)$;
  asegura avanzaDia : $jornadaActual(result) == jornadaActual(j) + 1$;
  asegura mismaCantDeCompetencias : $(\forall d \leftarrow [1..cantDias(j)])\,|cronograma(j,d)| == |cronograma(result,d)|$;
  asegura cronogramaDeOtrosDiasNoCambia : $(\forall d \leftarrow [1..cantDias(j)], d \neq jornadaActual(j))$
    $(\forall c \leftarrow cronograma(j,d))laCompetenciaSeMantiene(result, d, c)$;
  asegura lasFinalizadasSeMantinen : $(\forall c \leftarrow cronograma(j, jornadaActual(j)), finalizada(c))$
    $laCompetenciaSeMantiene(result, jornadaActual(j), c)$;
  asegura finalizanCompetencias : $(\forall c \leftarrow cronograma(j, jornadaActual(j)), \neg finalizada(c))$
    $finaliza(result, c, jornadaActual(j))$;

  aux finaliza (j: JJOO, c: Competencia, dia: $\mathbb{Z}$) : Bool = $(\exists x \leftarrow cronograma(j, dia))categoria(x) == categoria(c) \wedge$
    $mismos(participantes(x), participantes(c)) \wedge finalizada(x) \wedge mismos(ranking(x), participantes(x)) \wedge$
    $ordenada(reverso(capacidades(ranking(x), deporte(x)))) \wedge$
    $|ranking(x)| \geq 1 \Rightarrow |lesTocoControlAntiDoping(x)| == 1$;
}

5. Auxiliares

aux ciaNumbers (as: [Atleta]) : [$\mathbb{Z}$] = $[\,ciaNumber(a)\,|\,a \leftarrow as\,]$;
aux competencias (j: JJOO) : [Competencia] = $[\,c\,|\,d \leftarrow [1..cantDias(j)], c \leftarrow cronograma(j,d)\,]$;
aux incluida (l_1, l_2:[T]) : Bool = $(\forall x \leftarrow l_1)cuenta(x, l_1) \leq cuenta(x, l_2)$;
aux lasPasadasFinalizaron (j: JJOO) : Bool = $(\forall d \leftarrow [1..jornadaActual(j)))(\forall c \leftarrow cronograma(j,d))finalizada(c)$;

aux lasQueNoPasaronNoFinalizaron (j: JJOO) : Bool =
$(\forall d \leftarrow (jornadaActual(j)..cantDias(j)])(\forall c \leftarrow cronograma(j,d))\neg finalizada(c)$;
aux ordenada (l:[T]) : Bool = $(\forall i \leftarrow [0..|l|-1))l_i \leq l_{i+1}$;
aux sinRepetidos (l: [T]) : Bool = $(\forall i,j \leftarrow [0..|l|), i \neq j)l_i \neq l_j$;
aux capacidades (as: [Atleta], d: Deporte) : $[\mathbb{Z}]$ = $[\, capacidad(a,d) \mid a \leftarrow as \,]$;
aux deporte (c: Competencia) : Deporte = $prm(categoria(c))$;
aux filtrarPorPais (as: [Atleta], p: Pais) : [Atleta] = $[\, a \mid a \leftarrow as, nacionalidad(a) == p \,]$;
aux laCompetenciaSeMantiene (j: JJOO, d: $\mathbb{Z}$, c: Competencia) : Bool =
$(\exists x \leftarrow cronograma(j,d))categoria(x) == categoria(c) \wedge mismos(participantes(x), participantes(c))$
$\wedge\ finalizada(x) \Leftrightarrow finalizada(c) \wedge finalizada(x) \Rightarrow (ranking(x) == ranking(c) \wedge mismosControlados(x,c))$;
aux medallistasOro (j: JJOO) : [Atleta] = $[\, ranking(c)_0 \mid d \leftarrow [1..jornadaActual(j)], c \leftarrow cronograma(j,d),$
$finalizada(c) \wedge |ranking(c)| \geq 1 \,]$;
aux medallistasPlata (j: JJOO) : [Atleta] = $[\, ranking(c)_1 \mid d \leftarrow [1..jornadaActual(j)], c \leftarrow cronograma(j,d),$
$finalizada(c) \wedge |ranking(c)| \geq 2 \,]$;
aux medallistasBronce (j: JJOO) : [Atleta] = $[\, ranking(c)_2 \mid d \leftarrow [1..jornadaActual(j)], c \leftarrow cronograma(j,d),$
$finalizada(c) \wedge |ranking(c)| \geq 3 \,]$;
aux minimo (l: $[\mathbb{Z}]$) : $\mathbb{Z}$ = $[\, x \mid x \leftarrow l, (\forall y \leftarrow l)x \leq y \,]_0$;
aux mismosControlados (c_1, c_2: Competencia) : Bool =
$mismos(lesTocoControlAntiDoping(c_1), lesTocoControlAntiDoping(c_2)) \wedge$
$(\forall p \leftarrow lesTocoControlAntiDoping(c_1))leDioPositivo(c_1,p) \Leftrightarrow leDioPositivo(c_2,p)$;
aux nacionalidades (as: [Atleta]) : [Pais] = $[\, nacionalidad(a) \mid a \leftarrow as \,]$;
aux paises (j: JJOO) : [País] = $sacarRepetidos(nacionalidades(atletas(j)))$;
aux primeros (l: [(T,S)]) : [T] = $[\, prm(x) \mid x \leftarrow l \,]$;
aux reverso (l: [T]) : [T] = $[\, l_{|x|-i-1} \mid i \leftarrow [0..|l|) \,]$;
aux sacarRepetidos (l: [T]) : [T] = $[\, l_i \mid i \leftarrow [0..|l|), l_i \notin l_{[0..i)} \,]$;