Algoritmos y Estructura de Datos I

Primer cuatrimestre 2012

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

TP OJOTA (Organización de Juegos Olímpicos) v1.0

Grupo 4

Integrante	LU	Correo electrónico
Nicolas Lasso	763/10	lasso.nico@gmail.com
Guido Tripodi	843/10	guido.tripodi@hotmail.com
Tomas Agustin Shaurli	671/10	zeratulzero@hotmail.com
Patricio Inzaghi	255/11	pinzaghi@dc.uba.ar

1. Aclaraciones

1. En el ejercicio 7, en caso de empate en medallas de oro, se compara las medallas de plata, si siguen siendo las mismas, se compara las medallas de bronce. En caso de tener las 3 medallas iguales se considera mejor al primero que se comparo. Si no hay ningun pais con medallas la lista resultante sera vacia.
2. En el ejercicio 12, consideramos a un patron, como la secuencia de paises hasta que aparece un elemento repetido sin incluirlo. Por Ej: [Argentina, USA, Canada, Argentina, Chile] el patron seria [Argentina, USA, Canada], en cambio en [Argentina, USA, USA, Argentina, Chile] el patron seria [Argentina, USA].
3. En el ejercicio 17, consideramos a un atleta prodigio, aquel que obtuvo una medalla de oro en la categoria pedida, a menor edad que el resto de los ganadores de los otros JJOO en esa misma categoria.

2. Resolucion

1. `problema entrenarNuevoDeporte` (a: Atleta, d: Deporte, c: $\mathbb{Z}$) {
   `requiere` $d \notin deportes(pre(a))$;
   `requiere` $0 \leq c \leq 100$;
   `modifica` a;
   `asegura` $nombre(a) == nombre(pre(a))$;
   `asegura` $sexo(a) == sexo(pre(a))$;
   `asegura` $añoNacimiento(a) == añoNacimiento(pre(a))$;
   `asegura` $naionalidad(a) == nacionalidad(pre(a))$;
   `asegura` $ciaNumber(a) == ciaNumber(pre(a))$;
   `asegura` $mismos(deportes(a), deportes(pre(a)) : d)$;
   `asegura` $(\forall d \in deportes(pre(a)))capacidad(a, d) == capacidad(pre(a), d)$;
   `asegura` $capacidad(a, d) == c$;
   }
2. `problema finalizarCompetencia` (c: Competencia, posiciones: [Atleta], control: [(Atleta, Bool)]) {
   `requiere` $(\forall a \in control)prm(a) \in participantes(c)$;
   `requiere` $(\forall a \in posiciones)a \in participantes(c)$;
   `requiere` $finalizada(c) == False$;
   `modifica` c;
   `asegura` $categoria(c) == categoria(pre(c))$;
   `asegura` $mismos(participantes(c), participantes(pre(c))$;
   `asegura` $finalizada(c)$;
   `asegura` $ranking(c) == posiciones$;
   `asegura` $mismos([prm(d)|d \leftarrow control], lesTocoControlAntiDoping(c))$;
   `asegura` $(\forall x \leftarrow control), lesDioPositivo(c, prm(x)) == sgd(x)$;
   }
3. `problema linfordChristie` (c: Competencia, a: Atleta) {
   `requiere` $finalizada(c) ==$ False;
   `requiere` $atletaPertenezca : a \in participantes(c)$;
   `modifica` c;
   `asegura` $categoria(c) == categoria(pre(c))$;
   `asegura` $finalizada(c) == finalizada(pre(c))$;
   `asegura` $mismos(a : participantes(c), participantes(pre(c)))$;
   }
4. `problema gananLosMasCapaces` (c: Competencia) = `result` : Bool {
   `requiere` $finalizada(c) == true$;
   `asegura` $Result == masCapaces(c)$;
   }

5. `problema sancionarTramposos` (c: Competencia) {

`requiere` $finalizada(c) == true$;

`modifica` c;

`asegura` $categoria(c) == categoria(pre(c))$;

`asegura` $finalizada(c) == finalizada(pre(c)$;

`asegura` $mismos(participantes(c), participantes(pre(c)))$;

`asegura` $mismos(lesTocoControlAntiDoping(c), lesTocoControlAntiDoping(pre(c)))$;

`asegura` $(\forall a \in participantes(c))leDioPositivo(c, a) == leDioPositivo(pre(c), a))$;

`asegura` $ranking(c) == borrarPositivos(pre(c))$;

`aux borrarPositivos` (c:competencia) : [Atleta] $= [x | x \leftarrow ranking(c), x \notin lesTocoControlAntiDoping(c) \vee (x \in lesTocoControlAntiDoping(c) \wedge leDioPositivo(c, x) == False)]$;

}

6. `problema clasificóTarde` (c:Competencia, a:Atleta) {

`requiere` $finalizada(c) == True$;

`requiere` $a \notin participantes(c)$;

`requiere` $(\exists d \leftarrow deportes(a)), d == prm(categoria(c)) \wedge sexo(a) == sgd(categoria(c))$;

`modifica` c;

`asegura` $categoria(c) == categoria(pre(c))$;

`asegura` $mismos(paticipantes(c), participantes(pre(c)) : a)$;

`asegura` $finalizada(c) == finalizada(pre(c))$;

}

7. `problema dePaseo` (j: JJOO) = `result` : [Atleta] {

`asegura` $mismos(result, noParticiparon(j))$;

`aux noParticiparon` (j: JJOO) : [Atleta] $= [x | k \leftarrow competencias(j), x \leftarrow atletas(j), x \notin participantes(k)]$;

}

8. `problema medallero` (j: JJOO) = `result` : [(País, [$\mathbb{Z}$])] {

`asegura` $mismos(result, extraerSinMedallas(j))$;

`asegura` $soloTresTiposDeMedallas : (\forall p \leftarrow result, |sgd(p)| == 3)$;

`asegura` $paisesConMedallas : (\forall p \leftarrow result, algunaMedalla(sgd(p)))$;

`asegura` $mejoresMedallasQueSiguiente :$
$(\forall x \leftarrow [0..|result| - 1), mejoresOIgualesMedallas(sgd(result_x), sgd(result_{x+1})))$;

`aux mejoresOIgualesMedallas` (m: [$\mathbb{Z}$], p: [$\mathbb{Z}$]) : Bool $= superaPorOro(m, p)$;

`aux superaPorOro` (m: [$\mathbb{Z}$], p: [$\mathbb{Z}$]) : Bool $=$
$ifThenElse(m_0 > p_0, \text{True}, ifThenElse(m_0 == p_0, superaPorPlata(m, p), \text{False}))$;

`aux superaPorPlata` (m: [$\mathbb{Z}$], p: [$\mathbb{Z}$]) : Bool $=$
$ifThenElse(m_1 > p_1, \text{True}, ifThenElse(m_1 == p_1, superaPorBronce(m, p), \text{False}))$;

`aux superaPorBronce` (m: [$\mathbb{Z}$], p: [$\mathbb{Z}$]) : Bool $=$
$ifThenElse(m_0 > p_0, \text{True}, m_0 == p_0)$;

`aux algunaMedalla` (m: [$\mathbb{Z}$]) : Bool $= (m_0 \neq 0 \vee m_1 \neq 0 \vee m_2 \neq 0)$;

}

9. `problema boicotPorDisciplina` (j: JJOO, cat: (Deporte, Sexo), p: País) = `result` : $\mathbb{Z}$ {

`requiere` $(\exists c \in competencias(j))categoria(c) == cat$;

`modifica` j;

`asegura` $año(j) == año(pre(j))$;

`asegura` $cantDias(j) == cantDias(pre(j))$;

`asegura` $mismos(atletas(j), atletas(pre(j)$;

`asegura` $(\forall d \leftarrow [0..cantDias(j)))mismos(cronograma(j, d), cronograma(pre(j), d))$;

`asegura` $jornadaactual(j) == jornadaactual(pre(j))$;

asegura $result == participantesDeLaCat(j, cat, p)$;

asegura $(\forall c \leftarrow competencias(j), categoria(c) == cat), (\forall a \leftarrow atletasDelPais(j, p))a \notin participantes(c)$;

aux participantesDeLaCat (j:JJOO, cat: (Deporte, Sexo), p: País) : $\mathbb{Z} = long([i|i \leftarrow atletasDelPais(j, p), x \leftarrow competencias(j), categoria(x) == cat \wedge i \in participantes(x))$;

}

10. problema losMasFracasados (j: JJOO, p: País) = result : [Atleta] {

asegura $mismos(result, masFracasados(j, p))$;

aux competenciasAtleta (j:JJOO, p:Pais, a:Atleta) : [Competencia] $= [x|x \leftarrow competencias(j), (\exists p \in participantes(x))p == a \wedge en(prm(categoria(x)), deportes(a))]$;

aux atletasFracasados (j:JJOO, p:Pais) : [(atleta, $\mathbb{Z}$)] $= [(y, |competenciasAtleta(j, p, y)|)| y \leftarrow atletasDelPais(j, p), c \leftarrow competenciasAtleta(j, y, p), \neg en(y, sub(ranking(c), 0, 2))]$;

aux masFracasados (j:JJOO, p:Pais) : [Atleta] $= [prm(y)|y \leftarrow atletasFracasados(j, p), k \leftarrow atletasFracasados(j, p), sgd(y) \geq sgd(k)]$;

}

11. problema liuSong (j: JJOO, a: Atleta, p: País) {

requiere $a \in atletas(j)$;

modifica j ;

asegura $año(j) = año(pre(j))$;

asegura $cantDias(j) == cantDias(pre(j))$;

asegura $jornadaActual(j) == jornadaActual(pre(j))$;

asegura $(\forall d \leftarrow [1..cantDias(j)))mismos(cronograma(j, d), cronograma(pre(j), d))$;

asegura $(\forall h \leftarrow atletas(j), ciaNumber(h) == ciaNumber(a))nacionalidad(h) == p$;

asegura $|atletas(j)| == |atletas(pre(j))|$;

asegura $mismos(atletas(j), reemplazaAtleta(a, atletas(pre(j))))$;

asegura $(\forall f \leftarrow atletas(j), ciaNumber(f) \neq ciaNumber(a))(\exists b \leftarrow atletas(pre(j)))f == b$;

asegura $(\forall c \leftarrow competencias(j), q \leftarrow participantes(c), ciaNumber(q) == ciaNumber(a))nacionalidad(q) == p$;

asegura $(\forall c \leftarrow competencias(j), p \leftarrow participantes(c), h \leftarrow participantes(c), ciaNumber(p) \neq ciaNumber(h)), p \neq h$;

asegura $(\forall c \leftarrow competencias(j), q \leftarrow ranking(c), finalizada(c) == True, ciaNumber(q) == ciaNumber(a)) nacionalidad(q) == p$;

asegura $(\forall c \leftarrow competencias(j), a \leftarrow ranking(c), finalizada(c))a \in participantes(c)$;

asegura $(\forall c \leftarrow competencias(j), q \leftarrow lesTocoControlAntiDoping(c), finalizada(c) == True, ciaNumber(q) == ciaNumber(a))nacionalidad(q) == p$;

asegura $(\forall c \leftarrow competencias(j), a \leftarrow lesTocoControlAntiDoping(c), finalizada(c)), a \in participantes(c)$;

asegura $(\forall c \leftarrow competencias(j), x \leftarrow competencias(pre(j)), p \leftarrow lesTocoControlAntiDoping(c)) (\exists a \leftarrow lesTocoControlAntiDoping(x))leDioPositivo(c, p) == leDioPositivo(x, a)$;

aux reemplazaAtleta (a: Atleta, ls: [Atleta]) : [Atleta] $= [if ciaNumber(x) == ciaNumber(a) then a else sex|x \in ls]$;

}

12. problema stevenBradbury (j: JJOO) = result : Atleta {

requiere $algunaCompetenciaFinalizada(j)$;

asegura $esElMenosCapaz$: $(\forall a \in atletasConOro(j), a \neq result)sumaDeCapacidades(result) \leq sumaDeCapacidades(a)$;

aux atletasConOro (j:JJOO) : [Atleta] $= [ranking(c)_0|c \in competencias(j), finalizada(c) \wedge |ranking(c)| > 0]$;

aux sumaDeCapacidades (a:Atleta) : $\mathbb{Z} = \sum[capacidad(a, d)|d \in deportes(a)]$;

}

13. problema uyOrdenadoAsíHayUnPatrón (j: JJOO) = result : Bool {

asegura $result == verificaPaises(j)$;

aux medallasDeOro (j:JJOO, p:Pais, d:Dia) : $\mathbb{Z} = long([x|x \leftarrow atletasDelPais(j, p), y \leftarrow cronograma(j, d), x \in participantes(y) \wedge x == ranking(y)_0])$;

aux paisesDelDia (j:JJOO, d:Dia) : [Pais] $= [nacionalidad(x)|y \leftarrow cronograma(j, d), x \leftarrow participantes(y)]$;

aux mejoresPaises (j:JJOO) : [Pais] $= [p | d \leftarrow [1..jornadaActual(j)] p \leftarrow extraerRepetidos(PaisesDelDia(j,d)), q \leftarrow extraerRepetidos(PaisesDelDia(j,d)), medallasDeOro(j,p,d) \geq medallasDeOro(j,q,d) \wedge p > q]$;

aux patronDePaises (j:JJOO) : [Pais] $= [sub(mejoresPaises(j),0,i-2) | i \leftarrow [0..|mejoresPaises|-1], \neg sinRepetidos(sub(mejoresPaises(j),0,i))]_0$;

aux verificaPaises (j:JJOO) : Bool $= \forall i \in [0..|mejoresPaises(j)|-1], ((i + |patronDePaises(j)|) \leq |mejoresPaises(j)|) \wedge paisesMasGrosos(j)_i == mejoresPaises(j)_{i+|patronDePaises(j)|}$;

}

14. problema sequíaOlímpica (j: JJOO) = result : [País] {

asegura $mismos(result, extraerRepetidos(elPeorPais(j)))$;

aux atletasDelPais (j:JJOO,p: País) : [Atleta] $= [x | x \leftarrow atletas(j), nacionalidad(x) == p]$;

aux paisesConMedallas (J:JJOO) : [(País,[Bool])] $= [(nacionalidad(x), [alguno(h \leftarrow (ranking(y)[0,1,2]) k \leftarrow (atletasDelPais(j, nacionalidad(x))), k == h)]) | x \leftarrow atletas(j), i \leftarrow [0..cantDias(j)], y \leftarrow cronograma(j,i)]$;

aux lugaresDistintos (j:JJOO, b:[Bool]) : $[\mathbb{Z}] = [x+1 | x \leftarrow [0..|b|-1], b_x \neq b_{x+1}]$;

aux paisesConDiasSeparados (j:JJOO) : [(Pais,[[Bool]])] $= [(prm(x), [sub(sgd(x), cons(0, lugresDistintos(j, sgd(x)))_h, (cons(0, lugresDistintos(j, sgd(x)))_{h+1}) | x \leftarrow paisesConMedallas(j), h \leftarrow [0..|cons(0, lugaresDistintos(j, sgd(x)))| - 2]]$;

aux elijoLosFracasados (j:JJOO) : [(pais,[[Bool]])] $= [(prm(x), [y]) | x \leftarrow paisesConDiasSeparados(j), (\forall y \leftarrow sgd(x), h \leftarrow y)(h == False))]$;

aux mayorCantidadDeDIas (j:JJOO) : [(pais, $\mathbb{Z}$)] $= [(prm(x), long(y)) | x \leftarrow elijoLosFracasados(j), y \leftarrow sgd(x), h \leftarrow sgd(x), long(y) \geq long(h)]$;

aux elPeorPais (j:JJOO) : [pais] $= [prm(x) | x \leftarrow mayorCantidadDeDias(j), y \leftarrow mayorCantidadDeDias(j), sgd(x) \geq sgd(y)]$;

}

15. problema transcurrirDia (j: JJOO) {

requiere $jornadaActual(j) < cantDias(j)$;

modifica j ;

asegura $cantDias(j) == cantDias(pre(j))$;

asegura $año(j) == año(pre(j))$;

asegura $mismos(atletas(j), atletas(pre(j)))$;

asegura $(\forall d \in [1...cantDias(pre(j))], d \neq jornadaActual(pre(j)))mismos(cronograma(pre(j),d), cronograma(j,d))$;

asegura $cambiaElDia : jornadaActual(j) == jornadaActual(pre(j)) + 1$;

asegura $rankingSegunMasCapaces : (\forall c \in cronograma(pre(j), jornadaActual(pre(j))), finalizada(c), |participantes(c)| > 0) masCapaces(c)$;

asegura $unControlPorCompetencia : (\forall c \in cronograma(pre(j), jornadaActual(pre(j))), \neg finalizada(c), |participantes(c)| > 0) |lesTocaControlAntiDoping(c)| == 1$;

asegura $ifThenElse(todosLosAntiDoping > 0, 0 \leq sonPositivos(j)/todosLosAntiDoping(j) \leq 0,05, true)$;

aux SonPositivos (j:JJOO) : $\mathbb{Z} = long([y | x \leftarrow cronograma(j, jornadaActual(pre(j))), y \leftarrow LesTocoControlAntiDoping(x), LeDioPositivo(x,y)])$;

aux todosLosAntiDoping (j:JJOO) : $\mathbb{Z} = long(concat([LesTocoControlAntiDoping(x) | x \leftarrow cronograma(j, jornadaActual(pre(j))), |participantes(x)| > 0]))$;

}

16. problema deportesNoOlimpicos (j:JJOO) = result : [deporte] {

asegura $mismos(result, noCompiten(j, todosLosDeportes(j)))$;

asegura $(\forall x \leftarrow result), (\exists a \leftarrow atletas(j)) x \in deportes(a)$;

asegura $(\forall x \leftarrow result, y \leftarrow competencias(j)) x \neq prm(categoria(y))$;

aux todosLosDeportes (j:JJOO) : [deporte] $= [deportes(a) | a \leftarrow atletas(j)]$;

aux noCompiten (j:JJOO, ds:[deporte]) : [deporte] $= [d | d \leftarrow extraerRepetidos(ds), c \leftarrow competencias(j), d \neq prm(categoria(c))]$;

}

17. `problema atletaProdigio` (js:[JJOO], cat:(Deporte, Sexo)) = `result` : Atleta {

`requiere` $(\exists j \leftarrow js, c \leftarrow competencias(j))prm(cat) == prm(categoria(c)) \land sgd(cat) == sgd(categoria(c))$;

`asegura` $result \in competidoresOroPorCategoria(js, cat)$;

`asegura` $(\forall a \leftarrow competidoresOroPorCategoria(js, cat), a \neq result)añoNacimiento(result) \geq añoNacimiento(a)$;

`asegura` $(\exists d \in deportes(result))d == prm(cat) \land sexo(result) == sgd(cat)$;

`aux competioresOroPorCategoria` (js:[JJOO], cat:(Deporte, Sexo)) : [Atleta] $= [ranking(c)_0 | j \leftarrow js, c \leftarrow competencias(j),$
$prm(categoria(c)) == prm(cat) \land sgd(categoria(c)) == sgd(cat)]$;

}

3. Tipos

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

4. 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` : $(\forall d \leftarrow deportes(a))0 \leq capacidad(a, d) \leq 100$;
}

5. 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` :
    $finalizada(c) \Rightarrow incluida(lesTocoControlAntiDoping(c), participantes(c))$;
}

6. 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` : $(\forall i, k \leftarrow [0..|competencias(j)|), i \neq k)$
    $categoria(competencias(j)_i) \neq categoria(competencias(j)_k)$;
  `invariante competidoresInscriptos` : $(\forall c \leftarrow competencias(j))incluida(participantes(c), atletas(j))$;
  `invariante jornadaValida` : $1 \leq jornadaActual(j) \leq cantDias(j)$;
  `invariante finalizadasSiiYaPasoElDia` : $lasPasadasFinalizaron(j) \land lasQueNoPasaronNoFinalizaron(j)$;

}

7. Auxiliares

aux algunaCompetenciaFinalizada (j:JJOO) : Bool $= (\exists x \in competencias(j))finalizada(x) == true \wedge |ranking(x)| > 0$;
aux ciaNumbers (as: [Atleta]) : $[\mathbb{Z}] = [\, ciaNumber(a) \mid a \leftarrow as \,]$;
aux competencias (j: JJOO) : [Competencia] $= [\, c \mid d \leftarrow [1..cantDias(j)], c \leftarrow cronograma(j, d) \,]$;
aux competenciasFinalizadas (j: JJOO) : [Competencia] $= [\, c \mid C \leftarrow [1..cantDias(j)], c \leftarrow cronograma(j, d), finalizada(c) \,]$;
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 listaPositivos (c:competencia) : [Atleta] $= [lesTocoControlAntiDoping(c), lesDioPositivo(c, x) ==$ True] ;
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 masCapaces (c:competencia) : Bool $= (\forall x \leftarrow [0..|ranking(c)|-1), capacidad(ranking(c)_x, prm(categoria(c))) > capacidad(ran$
aux atletasDelPais (j:JJOO,p: País) : [Atleta] $= [x | x \leftarrow atletas(j), nacionalidad(x) == p]$;
aux medallasDelPais (j:JJOO, p:Pais, pos:$\mathbb{Z}$) : $\mathbb{Z} = long([x | x \leftarrow atletasDelPais(j, p), y \leftarrow competencias(j),$
$x == ranking(y)_{pos}])$;
aux todosLosPaises (j:JJOO) : [Pais] $= [nacionalidad(x) | x \leftarrow atletas(j)]$;
aux extraerRepetidos (ls:[T]) : [T] $= [x | x \leftarrow [0..|ls|), \neg en(ls_x, sub(ls, 0, x))]$;
aux obtenerMedallero (j:JJOO) : [(País, [$\mathbb{Z}$])] $= [(x, [medallasDelPais(j, x, y)]) | x \leftarrow extraerRepetidos(todosLosPaises(j)),$
$y \leftarrow [0, 1, 2]]$;
aux extraerSinMedallas (j:JJOO) : [(País, [$\mathbb{Z}$])] $= [x | x \leftarrow obtenerMedallero(j), (\exists x \leftarrow sgd(x))x > 0]$;
aux atletasdelpaisout (cat: (Deporte,Sexo), p: Pais, j: JJOO) : Bool $= (x \leftarrow [0..|atletas(j)| - 1], p(cat(atletas(j)_x)) \neq$
$atletas(j)_x)$;