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. 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` $ordenada(deportes(a))$;
   `asegura` $capacidad(a, d) == c$;
   }
2. `problema finalizarCompetencia` (c: Competencia, posiciones: [Atleta], control: [(Atleta, Bool)]) {
   `modifica` c;
   `asegura` $categoria(c) == categoria(pre(c))$;
   `asegura` $participantes(c) == participantes(pre(c))$;

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 $atletaNoPertenece : a \notin participantes(c)$;
asegura $long(participantes(c)) == long(participantes(pre(c)) - 1$;
asegura $mismos(a : participantes(c), participantes(pre(c)))$;
}

4. problema gananLosMasCapaces (c: Competencia) = result : Bool {
requiere $finalizada(c) ==$ True ;
requiere $| ranking(c) | > 1$;
asegura $Result == masCapaces(c)$;

}

5. `problema sancionarTramposos` (c: Competencia) {
`modifica` c;
`asegura` $categoria(c) == categoria(pre(c))$;
`asegura` $finalizada(c)$;
`asegura` $participantes(c) == participantes(pre(c))$;
`asegura` $lesTocoControlAntiDoping(c) == lesTocoControlAntiDoping(pre(c))$;
`asegura` $leDioPositivo(c, a) == leDioPositivo(pre(c), pre(a))$;
`asegura` $ranking(c) == borrarPositivos(pre(c))$;
}

6. `problema dePaseo` (j: JJOO) = `result` : [Atleta] {
`requiere` $cantDias(j) \geq 1$;
`requiere` $|atletas(j)| > 0$;
`asegura` $result == [x | y \leftarrow cantDias(j), k \leftarrow cronograma(j, y), x \leftarrow atletas(j), x \notin participantes(k)]$;
}

7. `problema medallero` (j: JJOO) = `result` : [(País, [$\mathbb{Z}$])] {
`asegura` $paisesConMedallas : (\forall p \leftarrow result, algunaMedalla(sgd(p)))$;
`asegura` $soloTresTiposDeMedallas : (\forall p \leftarrow result, |sgd(p)| == 3)$;

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 $= ifThenElse(m_1 > p_1, \text{True}, ifThenElse(m_1 == p_1, superaPorBronce(m, p), Fa$
aux superaPorBronce (m: $[\mathbb{Z}]$, p: $[\mathbb{Z}]$) : Bool $= ifThenElse(m_0 > p_0, \text{True}, ifThenElse(m_0 == p_0, \text{True}, \text{False}))$;
}

8. problema boicotPorDisciplina (j: JJOO, cat: (Deporte, Sexo), p: País) = result : $\mathbb{Z}$ {
requiere $long(atletas(j)) > 0$;
modifica j ;
asegura $año(j) == año(pre(j))$;
asegura $cantDias(j) == cantDias(pre(j))$;
asegura $cronograma(j) == cronograma(pre(j))$;
asegura $jornadaactual(j) == jornadaactual(pre(j))$;
asegura $(\forall a \in atletasEnCategoria(j, cat, p))a \notin atletas(j)$;
asegura $result == |atletasEnCategoria(j, cat, p)|$;
}

9. problema losMasFracasados (j: JJOO, p: País) = result : [Atleta] {
asegura $ningunoGanoMedallas : (\forall a \in result)noGanoMedallas(j, a)$;

asegura $tienenMasCompetenciasQueElResto$: $(\forall x \in result, y \in atletasSinMedallasPorPais(j,p))competenciasDelAtleta($
$competenciasDelAtleta(j,y)$;

aux noGanoMedallas (j:JJOO,a:Atleta) : Bool $= (\forall c \in competencias(j), finalizada(c))a \neq ranking(c)_0 \wedge a \neq ranking(c)_1 \wedge$
$a \neq ranking(c)_2$;

aux competenciasDelAtleta (j:JJOO,a:Atleta) : Bool $= [c|c \in competencias(j), a \in participantes(c)]$;

aux atletasSinMedallasPorPais (j:JJOO,p:Pais) : Bool $= [a|a \in atletas(j), nacionalidad(a) == p, noGanoMedallas(j,a)]$;

}

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

requiere $a \in atletas(pre(j))$;

modifica a, j ;

asegura $nombre(a) == nombre(pre(a))$;

asegura $sexo(a) == sexo(pre(a))$;

asegura $añoNacimiento(a) == añoNacimiento(pre(a))$;

asegura $ciaNumber(a) == ciaNumber(pre(a))$;

asegura $depores(a) == deportes(pre(a))$;

asegura $nacionalidad(a) == p$;

asegura $año(j) == año(pre(j))$;

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

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

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

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

`aux quitarAtleta` (j:JJOO,a:Atleta) : [Atleta] $= [x|x \leftarrow atletas(j), x \neq a]$;

`asegura` $atletas(j) == (quitarAtleta(pre(j), pre(a)) : a)$;

}

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

`requiere` $algunaCompetenciaFinalizada(j)$;

`asegura` $(\exists dia \in [1...cantDias(j)], cron \in cronograma(j, dia), comp \in cron) result == ranking(comp)_0$;

`asegura` $(\forall a \in atletas(j), a \neq result) capacidad(result) \leq capacidad(a)$;

}

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

`aux competenciaspordia` (j:JJOO) : [Dia][Compentecia] $= dia, comp|dia \leftarrow [1..cantDias(j)$;

, c $\leftarrow cronograma(j, dia), sgd(cronograma)_c == sgd(cronograma)_{c+1} aux\ \ mejoresAtletas\ (j : JJOO)\ : [Dia][Atleta] = [dia, ranking(c)_0 | dia \leftarrow prm(competenciaspordia(j)), c \in sgd(competenciaspordia(j))]$;

`aux paisesoro` (j: JJOO) : [Dia][Pais][$\int$] $= [dia, nacionalidad(a), \int |dia \leftarrow prm(mejoresAtletas(j)), a \in sgd(mejoresAtletas(j))]$;

`aux mejorpaisdeldia` (j: JJOO) : [Dia][$\int$] $= [dia, \int |dia \leftarrow prm(paisesoro(j)), p \leftarrow sgd(paisesoro(j)), cuenta()]$;

`aux paisordenado` (j: JJOO) : Bool $= x \leftarrow mejorpaisdeldia(j), h \leftarrow [paisordenado_{x+1}...|paisordenado|-1], k \leftarrow [paisordenado_j...|p$ 2], $paisordenado_x == paisordenado_j \wedge paisordenado_x == paisordenado_{k+(|paisordenado|_{x-}|paisordenado|_j}$;

}

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

`aux todosLosPaises` (J:JJOO) : [País] = $[nacionalidad(x)|x \leftarrow atletas(j)]$;

`asegura` $sinRepetidos(todosLosPaises(j))$;

}

14. `problema transcurrirDia` (j: JJOO) {

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

`modifica` j ;

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

`asegura` $año(j) == año(pre(j))$;

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

`asegura` $(\forall d \in [1...cantDias(pre(j))])cronograma(pre(j), d) == cronograma(j, d))$;

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

`asegura` $rankingSegunMasCapaces : (\forall c \in cronograma(pre(j), jornadaActual(pre(j)))masCapaces(c)$;

`asegura` $unControlPorCompetencia : (\forall c \in cronograma(pre(j), jornadaActual(pre(j)))|lesTocaControlAntiDoping(c)| ==$
$1)$;

`asegura` $dopingPositivoMax5Porciento : (\exists a \in atletasLesTocoAntiDoping(j, jornadaActual(pre(j))))0 \leq |atletasLesDioP$
$100/|atletasLesTocoAntiDoping(j, jornadaActual(pre(j)))| \leq 5)$;

`aux atletasLesTocoAntiDoping` (J:JJOO,d:$\mathbb{Z}$) : [Atleta] $= [lesTocoControlAntiDoping(c)_0 | c \in cronograma(j,d)]$ **;**

`aux atletasLesDioPositivo` (J:JJOO,d:$\mathbb{Z}$) : [Atleta] $= [lesTocoControlAntiDoping(c)_0 | c \in cronograma(j,d), lesTocoContr$
$leDioPositivo(c)]$ **;**

}

2. Tipos

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

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

4. 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) ∧ a ∈ lesTocoControlAntiDoping(c) ;

  invariante participaUnaSolaVez : sinRepetidos(ciaNumbers(participantes(c))) ;
  invariante participantesPertenecenACat :
     (∀p ← participantes(c))prm(categoria(c)) ∈ deportes(p) ∧ sgd(categoria(c)) == sexo(p) ;
  invariante elRankingEsDeParticipantesYNoHayRepetidos :
     finalizada(c) ⇒ incluida(ranking(c), participantes(c)) ;
  invariante seControlanParticipantesYNoHayRepetidos :
     finalizada(c) ⇒ incluida(lesTocoControlAntiDoping(c), participantes(c)) ;
}
```

5. 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 : $lasPasadasFinalizaron(j) \wedge lasQueNoPasaronNoFinalizaron(j)$;
}

6. Auxiliares

aux atletasEnCategoria (j:JJOO, cat: Categoria, p: País) : [Atleta] = $[a|a \in atletas(j), nacionalidad(a) == p \wedge sexo(a) == sgd(cat) \wedge prm(cat) \in deportes(a))]$;
aux algunaCompetenciaFinalizada (j:JJOO) : Bool = $(\exists dia \in [1...cantDias(j)], cron \in cronograma(j, dia), comp \in cron) finalizada(comp)$;

aux algunaMedalla (m: $[\mathbb{Z}]$) : Bool $= (m_0 \neq 0 \vee m_1 \neq 0 \vee m_2 \neq 0)$;
aux borrarPositivos (c:competencia) : [Atleta] $= [x | x \leftarrow ranking(c), (\forall y \leftarrow listaPositivos(c))y \neq x]$;
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] $= [lesTocoControlAntiDoping(c), lesDioPositivo(c, x) == \text{True}]$;
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 $= (\forall x \leftarrow [0..|ranking(c)|-1), capacidad(ranking(c)_x) > capacidad(ranking(c)_{x+1}))$;
aux atletasdelpaisout (cat: (Deporte,Sexo), p: Pais, j: JJOO) : Bool $= (x \leftarrow [0..|atletas(j)| - 1], p(cat(atletas(j)_x)) \neq atletas(j)_x)$;

7. Aclaraciones

1. En el ejercicio 4 requerimos que el ranking sea mayor a uno ya que sino no quedaria sin efecto la funcion auxiliar ya que de esta manera no se podrian comparar los atletas porque habria uno solo.
2. 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.