# 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 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. Si no hay ningun pais con medallas la lista resultante sera vacia.
3. 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 ].

# 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` $ordenada(deportes(a))$;
   `asegura` $capacidad(a, d) == c$;
   }
2. `problema finalizarCompetencia` (c: Competencia, posiciones: [Atleta], control: [(Atleta, Bool)]) {
   `requiere` $|participantes(c)| > 0$;
   `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` $(\forall a \in ranking(c))a \in participantes(c)$;
   `asegura` $ranking(c) == posiciones$;
   `asegura` $(\forall a \in control)prm(a) \in lesTocoControlAntiDoping(c)$;
   `asegura` $(\forall a \in lesTocoControlAntiDoping(c))a \in participantes(c)$;
   `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` $finalizada(c) == \text{True}$;
`requiere` $| ranking(c) | > 1$;
`requiere` $|participantes(c)| > 1$;
`asegura` $(\forall a \in ranking(c))a \in participantes(c)$;
`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` $participantes(c) == participantes(pre(c))$;
`asegura` $lesTocoControlAntiDoping(c) == lesTocoControlAntiDoping(pre(c))$;
`asegura` $(\forall a \in participantes(c))leDioPositivo(c, a) == leDioPositivo(pre(c), a))$;
`asegura` $ranking(c) == borrarPositivos(pre(c))$;
`aux borrarPositivos` (c:competencia) : [Atleta] $= [x | x \leftarrow ranking(c), (\forall y \leftarrow listaPositivos(c))y \neq x]$;
}

6. `problema dePaseo` (j: JJOO) = `result` : [Atleta] {
`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, [$\mathbb{Z}$])] {
`requiere` $|atletas(j)| > 0$;
`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 $=$
$ifThenElse(m_1 > p_1, \text{True}, ifThenElse(m_1 == p_1, superaPorBronce(m, p), \text{False}))$;
`aux superaPorBronce` (m: [$\mathbb{Z}$], p: [$\mathbb{Z}$]) : Bool $=$
$ifThenElse(m_0 > p_0, \text{True}, ifThenElse(m_0 == p_0, \text{True}, \text{False}))$;
`aux algunaMedalla` (m: [$\mathbb{Z}$]) : Bool $= (m_0 \neq 0 \vee m_1 \neq 0 \vee m_2 \neq 0)$;
}

8. `problema boicotPorDisciplina` (j: JJOO, cat: (Deporte, Sexo), p: País ) = `result` : $\mathbb{Z}$ {
`requiere` $long(atletas(j)) > 0$;
`requiere` $(\forall a \in atletasEnCategoria(j, cat, p))nacionalidad(a) == p$;
`requiere` $(\exists c \in competencias(j))categoria(c) == cat$;
`modifica` $j$;
`asegura` $año(j) == año(pre(j))$;
`asegura` $cantDias(j) == cantDias(pre(j))$;
`asegura` $mismos(cronograma(j), cronograma(pre(j)))$;
`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] {

`requiere` $(\exists a \in atletas(j))nacionalidad(a) == p$ ;

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

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

`aux atletasFracasados` (j:JJOO, p:Pais) : [(atleta, $\mathbb{Z}$)] $= [(y, |competenciasAtleta(j, p, y)|)|$
$y \leftarrow atletasDelPais(j, p), c \leftarrow competenciasAtleta(j, y, p), en(y, sub(ranking(c), 3, |ranking(c) - 1|))]$ ;

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

`asegura` $result == masFracasados(j, p)$ ;

}

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

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

`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` $mismos(deportes(a), deportes(pre(a)))$ ;

`asegura` $cambioNacionalidad : nacionalidad(a) == p$ ;

`asegura` $año(j) == año(pre(j))$ ;

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

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

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

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

`asegura` $mismosAtletas : (\forall x \in atletas(pre(j)))ciaNumber(x) \in ciaNumbers(j)$ ;

}

11. `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] $= [ranking(c)_0|c \in competencias(j)]$ ;

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

}

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

`asegura` $result == verificaPaises(j)$ ;

`aux competenciasPorDia` (j:JJOO) : [($\mathbb{Z}$, [Competencia])] $= [(dia, c)|dia \leftarrow [1..jornadaActual(j)],$
$c == mismos(cronograma(j, dia))]$ ;

`aux oroPorDia` (j:JJOO) : [($\mathbb{Z}$, [Atleta])] $= [(dia, a)|h \leftarrow competenciasPorDia(j), dia \leftarrow prm(h), x \leftarrow sgd(h),$
$i \leftarrow [0..|x| - 1], a \leftarrow ranking(x_i)_0]$ ;

`aux paisesPorDia` (j:JJOO) : [($\mathbb{Z}$,[Pais])] $= [(dia, p)|x \leftarrow oroPorDia(j), dia \leftarrow prm(x), h \leftarrow sgd(x),$
$i \leftarrow [0..|h| - 1], p \leftarrow nacionalidad(h_i)]$ ;

`aux mejoresPaises` (j:JJOO) : [($\mathbb{Z}$, [Pais])] $= [(dia, p)|x \leftarrow paisesPorDia(j), dia \leftarrow prm(x), p \leftarrow sgd(x),$
$i \leftarrow [0..|sgd(x)| - 2], (cuenta(p, sgd(x)) \geq cuenta(sgd(x)_{i+1}, sgd(x)))]$ ;

`aux paisesMasGrosos` (j:JJOO) : [Pais] $= [p|x \leftarrow mejoresPaises(j), p \leftarrow sgd(x), i \leftarrow [0..|sgd(x)| - 2], p \leq sgd(x)_{i+1}]$ ;

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

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

}

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

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

`asegura` $result == elPeorPais(j)$ ;

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

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

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

`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] $= [prm(x) | x \leftarrow mayorCantidadDeDias(j),$
$y \leftarrow mayorCantidadDeDias(j), sgd(x) \geq sgd(y)]$ ;

}

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

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

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

`asegura` $lasPasadasFinalizaron(pre(j))$ ;

`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` $dopingPositivoMax5Porciento : (\exists a \in atletasLesTocoAntiDoping(j, jornadaActual(pre(j))))$
$0 \leq porcentajePositivos(j) \leq 5)$ ;

`aux porcentajePositivos` (j:JJOO) : $\mathbb{R} = |atletasLesDioPositivo(j, jornadaActual(pre(j)))| * 100$
$/|atletasLesTocoAntiDoping(j, jornadaActual(pre(j)))|$ ;

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

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

}

# 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 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 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 $= (\forall x \leftarrow [0..|ranking(c)|-1), capacidad(ranking(c)_x) > capacidad(ranking(c)_{x+1}))$ ;
aux atletasdelpaisout (cat: (Deporte,Sexo), p: Pais, j: JJOO) : Bool $= (x \leftarrow [0..|atletas(j)| - 1], p(cat(atletas(j)_x)) \neq atletas(j)_x)$ ;