Algoritmos y Estructuras de Datos II

Primer Cuatrimestre de 2015

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

Trabajo Prï¿½ctico 1

Especificaciï¿½n

Integrante	LU	Correo electrónico
BENITEZ, Nelson	945/13	nelson.benitez92@gmail.com
ROIZMAN, Violeta	273/11	violeroizman@gmail.com
Vï¿½ZQUEZ, Jï¿½sica	318/13	jesis_93@hotmail.com
ZAVALLA, Agustï¿½n	670/13	nkm747@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Índice

1. TAD AS

TAD AS

géneros as

igualdad observacional

$$(\forall facu, facu' : \text{as}) \left(facu =_{\text{obs}} facu' \iff \begin{pmatrix} \text{campus(facu)=campus(facu')} \\ \wedge \text{ seguridad(facu)=seguridad(facu')} \\ \wedge \quad (\forall \ \textit{pos}\text{:p})\text{(posValida(campus(facu),p))} \\ \text{hayEst?(facu,p)} \iff \text{hayEst?(facu',p)} \\ \wedge \quad (\forall \ \textit{pos}\text{:p})\text{(posValida(campus(facu),p))} \\ \text{hayHippie?(facu,p)} \iff \text{hayHippie?(facu',p)} \\ \wedge \ (\forall \ \textit{seg}\text{:s})(\text{s} \in \text{seguridad(a))} \\ \text{(\#capturas(facu,s)=\#capturas(facu',s)} \\ \wedge \text{ \#sanciones(facu,s)=\#sanciones(facu',s))} \end{pmatrix} \right)$$

usa CAMPUS,BOOL,NAT,TUPLA,SEG

exporta AS,generadores, observadores,#hippies,#estudiantes,#masVigilante

observadores básicos

campus : as ⟶ campus

seguridad : as ⟶ conj(seguridad)

hayEst? : as a × pos p ⟶ bool $\{posValida(campus(a), p)\}$

hayHippie? : as a × pos p ⟶ bool $\{posValida(campus(a), p)\}$

#capturas : as a × seg s ⟶ nat $\{s \in seguridad(a)\}$

#sanciones : as a × seg s ⟶ nat $\{s \in seguridad(a)\}$

generadores

nueva : campus × conj(seguridad) ⟶ as
{(∀ *segs*:e) posValida(c,pos(e)) ∧ (∀ *segs*:s,s1) id(s)!=id(s1) ⇒ pos(s)!=pos(s1)}

moverEst : as a × pos pe × pos pd ⟶ as
$\left\{\begin{array}{l} posValida(campus(a), pe) \quad \wedge_{\text{L}} \quad hayEst?(a, pe) \quad \wedge \quad adyacente(campus(a), pe, pd) \quad \wedge \\ posValidaPersona(as, pd) \end{array}\right\}$

nuevoHippie : as a × pos p ⟶ as $\{posIngreso(campus(a), p) \wedge posValidaPersona(a, p)\}$

nuevoEst : as a × pos p ⟶ as $\{posIngreso(campus(a), p) \wedge posValidaPersona(a, p)\}$

sacarEst : as a × pos p ⟶ as $\{posValida(campus(a), p) \wedge_{\text{L}} hayEst?(a, p) \wedge posIngreso(a, p)\}$

otras operaciones

haySeg? : as a × pos p ⟶ bool {posValida(campus(as),p)}

posValidaPersona : as a × pos p ⟶ bool {posValida(campus(as),p)}

posIngreso : as a × pos p ⟶ bool {posValida(campus(as),p)}

moverTodos : as a × conj(seguridad) $segs$ ⟶ conj(seguridad)

moverSeg : as a × seguridad seg × pos $posSig$ ⟶ seguridad

proximasPosiciones : as a × conj(pos) $minPos$ × pos $posAct$ ⟶ conj(pos)
$\{\neg(\emptyset?(minPos)) \wedge_{\text{L}} posValida(campus(a), posAct) \wedge posicionesValidas(campus(a), minPos)\}$

hippiesMasCerca : as a × seguridad seg ⟶ conj(pos) $\{seg \in seguridad(a) \wedge hayHippies(a)\}$

encerrado : as a × pos p ⟶ bool $\{posValida(campus(as), p) \wedge hayEst?(p)\}$

#hippies : as a ⟶ nat

#estudiantes : as a ⟶ nat

#masVigilante : as a $\longrightarrow$ nat

contarHippies : as a × conj(pos) $poss$ $\longrightarrow$ nat

contarEstudiantes : as a × conj(pos) $poss$ $\longrightarrow$ nat

#masCapturas : as a × conj(seg) $segs$ $\longrightarrow$ conj(seg) {($\forall$ $segs$:s) s $\in$ $seguridad(a)$}

#maxCapturas : as a × conj(seg) $segs$ $\longrightarrow$ nat {($\forall$ $segs$:s) $\in$ $seguridad(a)$}

captura? : as a × pos p $\longrightarrow$ bool {posValida(campus(as),p)}

hippiesVecinos : as a × pos p $\longrightarrow$ nat {posValida(campus(as),p)}

HippieNatural : as a × pos p $\longrightarrow$ nat {posValida(campus(as),p)}

axiomas

campus(nueva($c, segs$)) $\equiv$ c

campus(moverEst(a, p_1, p_2)) $\equiv$ $campus(a)$

campus(nuevoEst(a, p_1)) $\equiv$ $campus(a)$

campus(nuevoHippie(a, p_1)) $\equiv$ $campus(a)$

campus(sacarEst(a, p_1)) $\equiv$ $campus(a)$

seguridad(nueva($c, segs$)) $\equiv$ $segs$

seguridad(moverEst(a, p_1, p_2)) $\equiv$ $moverTodos(a, seguridad(a))$

seguridad(nuevoEst(a, p_1)) $\equiv$ $moverTodos(a, seguridad(a))$

seguridad(nuevoHippie(a, p_1)) $\equiv$ $seguridad(a)$

seguridad(sacarEst(a, p_1)) $\equiv$ $seguridad(a)$

hayEst?(nueva($c, segs$),p) $\equiv$ $False$

hayEst?(nuevoEst(a, p_1),p) $\equiv$ **if** $p_1 = p$ **then** $True$ **else** $hayEst?(a, p)$ **fi**

hayEst?(moverEst(a, p_1, p_2),p) $\equiv$

```
if $p_1 = p$ then
    $False$
else
    if $p_2 = p$ then
        $\neg(hippiesVecinos(a, p_2) \geq 2)$
    else
        $hayEst?(a, p)$
    fi
fi
```

hayEst?(nuevoHippie(a, p_1),p) $\equiv$ $hayEst?(a, p)$

hayEst?(sacarEst(a, p_1),p) $\equiv$ **if** $p_1 = p$ **then** $False$ **else** $hayEst?(a, p)$ **fi**

hayHippie?(nueva($c, segs$),p) $\equiv$ $False$

hayHippie?((nuevoHippie(a, p_1),p) $\equiv$ **if** $p_1 = p$ **then** $True$ **else** $hayHippie?(a, p)$ **fi**

hayHippie?(nuevoEst(a, p_1),p) $\equiv$ $hayHippie?(a, p)$

hayHippie?(sacarEst(a, p_1),p) $\equiv$ $hayHippie?(a, p)$

#capturas(nueva($a, segs$),s) $\equiv$ 0

#capturas(moverEst(a, p_1, p_2),s) $\equiv$ $\#capturas(a, s)$

#capturas(nuevoHippie(a, p_1),s) $\equiv$

```
if $(adyacente(a, p_1, posSeg(a, s)) \wedge encerrado(a, p_1))$ then
    $1 + \#capturas(a, s)$
else
    $\#capturas(a, s)$
fi
```

#capturas(nuevoEst(a, p_1),s) $\equiv$ $\#capturas(a, s)$

#capturas(sacarEst(a, p_1),s) $\equiv$ $\#capturas(a, s)$

$\#$capturas(a,moverSeg(a, s, p_1)) $\equiv$ $\beta(posValida(campus(a), < \pi_1(p_1) + 1, \pi_2(p_1) >) \wedge_L (hayHippie?(a, < \pi_1(p_1) + 1, \pi_2(p_1) >) \wedge_L encerrado(a, < \pi_1(p_1) + 1, \pi_2(p_1) >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1) - 1, \pi_2(p_1) >) \wedge_L (hayHippie?(a, < \pi_1(p_1) - 1, \pi_2(p_1) >) \wedge_L encerrado(a, < \pi_1(p_1) - 1, \pi_2(p_1) >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) + 1 >) \wedge_L (hayHippie?(a, < \pi_1(p_1), \pi_2(p_1) + 1 >) \wedge_L encerrado(a, < \pi_1(p_1), \pi_2(p_1) + 1 >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) - 1 >) \wedge_L (hayHippie?(a, < \pi_1(p_1), \pi_2(p_1) - 1 >) \wedge_L encerrado(a, < \pi_1(p_1), \pi_2(p_1) - 1 >))) + \#capturas(a, s)$

```
#capturas(moverEst($a, p_1, p_2$),$s$)    ≡ if ($PosValida(campus(a), < \pi_1(posSeg)+1, \pi_2(posSeg) >$
                                          )) then
                                              if ($hayHippie(a, < \pi_1(posSeg) + 1, \pi_2(posSeg) >$))
                                              then
                                                  if ($captura?(a, < \pi_1(posSeg) + 1, \pi_2(posSeg) >$))
                                                  then
                                                      1
                                                  else
                                                      0
                                                  fi
                                              else
                                                  0
                                              fi
                                          else
                                              0
                                          fi

                                          +

                                          if ($PosValida(campus(a), < \pi_1(posSeg)-1, \pi_2(posSeg) >$
                                          )) then
                                              if ($hayHippie(a, < \pi_1(posSeg) - 1, \pi_2(posSeg) >$))
                                              then
                                                  if ($captura?(a, < \pi_1(posSeg) - 1, \pi_2(posSeg) >$))
                                                  then
                                                      1
                                                  else
                                                      0
                                                  fi
                                              else
                                                  0
                                              fi
                                          else
                                              0
                                          fi
                                          +

                                          if ($PosValida(campus(a), < \pi_1(posSeg), \pi_2(posSeg)+1 >$
                                          )) then
                                              if ($hayHippie(a, < \pi_1(posSeg), \pi_2(posSeg) + 1 >$))
                                              then
                                                  if ($captura?(a, < \pi_1(posSeg), \pi_2(posSeg) + 1 >$))
                                                  then
                                                      1
                                                  else
                                                      0
                                                  fi
                                              else
                                                  0
                                              fi
                                          else
                                              0
                                          fi
                                          +

                                          if ($PosValida(campus(a), < \pi_1(posSeg), \pi_2(posSeg)-1 >$
                                          )) then
                                              if ($hayHippie(a, < \pi_1(posSeg), \pi_2(posSeg) - 1 >$))
                                              then
                                                  if ($captura?(a, < \pi_1(posSeg), \pi_2(posSeg) - 1 >$))
                                                  then
                                                      1
                                                  else
                                                      0
```

#sanciones(nueva($a, segs$),s) $\equiv$ 0

#sanciones(moverEst(a, p_1, p_2),s) $\equiv$ $\#sanciones(a, s)$

#sanciones(nuevoHippie(a, p_1),s) $\equiv$ **if** $(cercanos?(a, p_1, posSeg(a, s)) \land_{\text{L}} (hayEst?(casilleroEnComun(a, p_1, posSeg(a, s))) \land encerrado(casilleroEnComun(a, p_1, posSeg(a, s)))))$
then
$1 + \#sanciones(a, s)$
else
$\#sanciones(a, s)$
fi

#sanciones(nuevoEst(a, p_1),s) $\equiv$ $\#sanciones(a, s)$

#sanciones(sacarEst(a, p_1),s) $\equiv$ $\#sanciones(a, s)$

#sanciones(a,moverSeg(a, s, p_1)) $\equiv$ $\beta(posValida(campus(a), < \pi_1(p_1) + 1, \pi_2(p_1) >) \land_{\text{L}} (hayEst?(a, < \pi_1(p_1) + 1, \pi_2(p_1) >) \land_{\text{L}} encerrado(a, < \pi_1(p_1) + 1, \pi_2(p_1) >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1) - 1, \pi_2(p_1) >) \land_{\text{L}} (hayEst?(a, < \pi_1(p_1) - 1, \pi_2(p_1) >) \land_{\text{L}} encerrado(a, < \pi_1(p_1) - 1, \pi_2(p_1) >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) + 1 >) \land_{\text{L}} (hayEst?(a, < \pi_1(p_1), \pi_2(p_1) + 1 >) \land_{\text{L}} encerrado(a, < \pi_1(p_1), \pi_2(p_1) + 1 >))) +$
$\beta(posValida(campus(a), < \pi_1(p_1), \pi_2(p_1) - 1 >) \land_{\text{L}} (hayEst?(a, < \pi_1(p_1), \pi_2(p_1) - 1 >) \land_{\text{L}} encerrado(a, < \pi_1(p_1), \pi_2(p_1) - 1 >))) + \#sanciones(a, s)$

```
#sanciones(moverEst($a, p_1, p_2$),$s$)    ≡ if ($PosValida(campus(a), < \pi_1(posSeg)+1, \pi_2(posSeg) >)$) then
    if ($hayEst(a, < \pi_1(posSeg) + 1, \pi_2(posSeg) >)$) then
        if ($captura?(a, < \pi_1(posSeg) + 1, \pi_2(posSeg) >)$) then
            1
        else
            0
        fi
    else
        0
    fi
else
    0
fi

+

if ($PosValida(campus(a), < \pi_1(posSeg)-1, \pi_2(posSeg) >)$) then
    if ($hayEst(a, < \pi_1(posSeg) - 1, \pi_2(posSeg) >)$) then
        if ($captura?(a, < \pi_1(posSeg) - 1, \pi_2(posSeg) >)$) then
            1
        else
            0
        fi
    else
        0
    fi
else
    0
fi
+

if ($PosValida(campus(a), < \pi_1(posSeg), \pi_2(posSeg)+1 >)$) then
    if ($hayEst(a, < \pi_1(posSeg), \pi_2(posSeg) + 1 >)$) then
        if ($captura?(a, < \pi_1(posSeg), \pi_2(posSeg) + 1 >)$) then
            1
        else
            0
        fi
    else
        0
    fi
else
    0
fi
+

if ($PosValida(campus(a), < \pi_1(posSeg), \pi_2(posSeg)-1 >)$) then
    if ($hayEst(a, < \pi_1(posSeg), \pi_2(posSeg) - 1 >)$) then
        if ($captura?(a, < \pi_1(posSeg), \pi_2(posSeg) - 1 >)$) then
            1
        else
            0
        fi
    else
        0
    fi
else
```

```
moverTodos(a,segs)    ≡ if ($\emptyset?(segs)$) then
                            $\emptyset$
                        else
                            if ($hayHippies?(a)$) then
                                $Ag(moverTodos(a, sinUno(segs)),$
                                $moverSeg(a, dameUno(segs),$
                                $dameUno(proxPosiciones$
                                $(hippiesMasCerca(a, dameUno(segs))))))$
                            else
                                $moverIngreso(a, segs)$
                            fi
                        fi
moverIngreso(a,segs)  ≡ if $\emptyset?(segs)$ then
                            $\emptyset$
                        else
                            if ($alto(campus(a)) - 1) - \pi_2(dameUno(segs)) >$
                            $\pi_2(dameUno(segs))$ then
                                $ag(moverIngreso(a, sinUno(segs)), mover(dameUno(segs), <$
                                $(\pi_1(dameUno(segs)), \pi_2(segs) - 1) >))$
                            else
                                if ($alto(campus(a)) - 1) - \pi_2(dameUno(segs)) <$
                                $\pi_2(dameUno(segs))$ then
                                    $ag(moverIngreso(a, sinUno(segs)), mover(dameUno(seg$
                                    $(\pi_1(dameUno(segs)), \pi_2(segs) + 1) >))$
                                else
                                    $ag(moverIngreso(a, sinUno(segs)), mover(dameUno(seg$
                                    $dameUno(\{< (\pi_1(dameUno(segs)), \pi_2(segs) -$
                                    $1) >, < (\pi_1(dameUno(segs)), \pi_2(segs)+1) >)\}))$
                                fi
                            fi
                        fi
moverSeg(a,seg,nPos)  ≡ if ($distMan(campus(a), \pi_2(seg), nPos) \geq 2$
                        $\vee \neg(posValida(campus(a), nPos)))$ then
                            $seg$
                        else
                            if $\#sanciones(a, seg) < 3$ then
                                $< \pi_1(seg), nPos >$
                            else
                                $seg$
                            fi
                        fi
```

```
proximasPosiciones(hscerca, posSeg)    ≡ if $\emptyset?(hscerca)$ then
                                             $\emptyset$
                                         else
                                             if $\pi_1(dameUno(hscerca)) > \pi_1(posSeg)$ then
                                                 if $\pi_2(dameUno(hscerca)) > \pi_2(posSeg)$ then
                                                     $\{< \pi_1(posSeg) + 1, \pi_2(posSeg) >,$
                                                     $< \pi_1(posSeg), \pi_2(posSeg) + 1 >\}$
                                                     $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                 else
                                                     if $\pi_2(dameUno(hscerca)) < \pi_2(posSeg)$ then
                                                         $\{< \pi_1(posSeg) + 1, \pi_2(posSeg) >,$
                                                         $< \pi_1(posSeg), \pi_2(posSeg) - 1 >\}$
                                                         $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                     else
                                                         $\{< \pi_1(posSeg) + 1, \pi_2(posSeg) >\}$
                                                         $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                     fi
                                                 fi
                                             else
                                                 if $\pi_1(dameUno(hscerca)) < \pi_1(posSeg)$ then
                                                     if $\pi_2(dameUno(hscerca)) > \pi_2(posSeg)$ then
                                                         $\{< \pi_1(posSeg) - 1, \pi_2(posSeg) >,$
                                                         $< \pi_1(posSeg), \pi_2(posSeg) + 1 >\}$
                                                         $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                     else
                                                         if $\pi_2(dameUno(hscerca)) < \pi_2(posSeg)$
                                                         then
                                                             $\{< \pi_1(posSeg) - 1, \pi_2(posSeg) >,$
                                                             $< \pi_1(posSeg), \pi_2(posSeg) - 1 >\}$
                                                             $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                         else
                                                             $\{< \pi_1(posSeg) - 1, \pi_2(posSeg) >\}$
                                                             $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                         fi
                                                     fi
                                                 else
                                                     if $\pi_2(dameUno(hscerca)) > \pi_2(posSeg)$ then
                                                         $\{< \pi_1(posSeg), \pi_2(posSeg) + 1 >\}$
                                                         $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                     else
                                                         $\{< \pi_1(posSeg), \pi_2(posSeg) - 1 >\}$
                                                         $\cup\ proxPosiciones(sinUno(minPos), posSeg)$
                                                     fi
                                                 fi
                                             fi
                                         fi
hippiesMasCerca(a,seg)                 ≡ $minDistsPos(campus(a), \pi_2(seg), posHippies(a))$
#hippies(a)                            ≡ $contarHippies(a, conjPos(campus(a), 0, 0))$
#estudiantes(a)                        ≡ $contarEstudiantes(a, conjPos(campus(a), 0, 0))$
```

```
contarHippies(a,poss)        ≡ if $\neg(\emptyset?(poss))$ then
                                   if $posValida(campus(a), dameUno(poss))$ then
                                       if $hayHippie(a, dameUno(poss))$ then
                                           $1 + contarHippies(a, sinUno(poss))$
                                       else
                                           $contarHippies(a, sinUno(poss))$
                                       fi
                                   else
                                       $contarHippies(a, sinUno(poss))$
                                   fi
                               else
                                   $0$
                               fi
contarEstudiantes(a,poss)    ≡ if $\neg(\emptyset?(poss))$ then
                                   if $posValida(campus(a), dameUno(poss))$ then
                                       if $hayEst?(a, dameUno(poss))$ then
                                           $1 + contarEstudiantes(a, sinUno(poss))$
                                       else
                                           $contarEstudiantes(a, sinUno(poss))$
                                       fi
                                   else
                                       $contarEstudiantes(a, sinUno(poss))$
                                   fi
                               else
                                   $0$
                               fi
masVigilante(a)              ≡ $dameUno(masCapturas(a, seguridad(a)))$
masCapturas(a,segs)          ≡ if $\neg(\emptyset?(segs))$ then
                                   if $\#capturas(a, dameUno(segs)) \geq$
                                   $maxCapturas(a, segs)$ then
                                       $ag(masCapturas(a, sinUno(segs)), dameUno(segs))$
                                   else
                                       $masCapturas(a, sinUno(segs))$
                                   fi
                               else
                                   $\emptyset$
                               fi
maxCapturas(a,segs)          ≡ if $\emptyset?(segs)$ then
                                   $0$
                               else
                                   if $\#capturas(a, dameUno(segs)) \geq$
                                   $maxCapturas(a, sinUno(segs))$
                                   then
                                       $\#capturas(a, dameUno(segs))$
                                   else
                                       $maxCapturas(a, sinUno(segs))$
                                   fi
                               fi
```

```
captura?$(a, p)$  $\equiv$ if $(posValida(campus(a), < \pi_1(p) + 1, \pi_2(p) >)$ then
        $(hayObstaculo?(campus(a), < \pi_1(p) + 1, \pi_2(p) >) \vee$
        $haySeg?(a, < \pi_1(p) + 1, \pi_2(p) >))$
    else
        $\neg(hayEst?(a, < \pi_1(p), \pi_2(p) >))$
    fi
    $\wedge$
    if $(posValida(campus(a), < \pi_1(p) - 1, \pi_2(p) >)$ then
        $(hayObstaculo?(campus(a), < \pi_1(p) - 1, \pi_2(p) >) \vee$
        $haySeg?(a, < \pi_1(p) - 1, \pi_2(p) >))$
    else
        $\neg(hayEst?(a, < \pi_1(p), \pi_2(p) >))$
    fi
    $\wedge$
    if $(posValida(campus(a), < \pi_1(p), \pi_2(p) + 1 >)$ then
        $(hayObstaculo?(campus(a), < \pi_1(p), \pi_2(p) + 1 >) \vee$
        $haySeg?(a, < \pi_1(p), \pi_2(p) + 1 >))$
    else
        $True$
    fi
    $\wedge$
    if $(posValida(campus(a), < \pi_1(p), \pi_2(p) - 1 >)$ then
        $(hayObstaculo?(campus(a), < \pi_1(p), \pi_2(p) - 1 >) \vee$
        $haySeg?(a, < \pi_1(p), \pi_2(p) - 1 >))$
    else
        $True$
    fi
hippiesVecinos$(a, p)$  $\equiv$ if hayHippie?(a,p) then 1 else 0 fi
hippieNatural$(a, p)$  $\equiv$ if posValida (campus(a),<$\pi_1(p) + 1, \pi_2(p)$>) then
        hippiesVecinos(a,<$\pi_1(p) + 1, \pi_2(p)$>)
    else
        0
    fi+if posValida (campus(a),<$\pi_1(p) - 1, \pi_2(p)$>) then
        hippiesVecinos(a,<$\pi_1(p) - 1, \pi_2(p)$>)
    else
        0
    fi+if posValida (campus(a),<$\pi_1(p), \pi_2(p) + 1$>) then
        hippiesVecinos(a,<$\pi_1(p), \pi_2(p) + 1$>)
    else
        0
    fi+if posValida (campus(a),<$\pi_1(p), \pi_2(p) - 1$>) then
        hippiesVecinos(a,<$\pi_1(p), \pi_2(p) - 1$>)
    else
        0
    fi
posValidaPersona$(a, p)$  $\equiv$ if $\neg(hayObstaculo?(campus(a), p))$ then
        $(\pi_2(p) = 0 \vee \pi_2$=alto(campus(a)))
    else
        $False$
    fi
```

Fin TAD

2. TAD campus

TAD CAMPUS

géneros campus

usa BOOL,NAT,TUPLA

exporta CAMPUS, observadores, generadores, posValida, posIngreso,minDistPos,adyacente,

observadores básicos

alto : campus $\longrightarrow$ nat

ancho : campus $\longrightarrow$ nat

obstaculos : campus $\longrightarrow$ conj(pos)

generadores

nuevo : nat *ancho* × nat *alto* × conj(pos) *obst* $\longrightarrow$ campus
$\{1 \leq ancho \wedge 1 \leq alto \wedge (\forall\, p\text{:pos})\ p \in obst \Rightarrow_{\text{L}} posValida(c, p)\}$

otras operaciones

adyacente : campus *c* × pos *pe* × pos *pd* $\longrightarrow$ bool $\{posValida(c, pe) \wedge posValida(c, pd)\}$

posValida : campus *c* × pos *p* $\longrightarrow$ bool

posIngreso : campus *c* × pos *p* $\longrightarrow$ bool {posValida(c,p)}

minDistsPos : campus *c* × pos *p* × conj(pos) *posiciones* $\longrightarrow$ conj(pos)
$\{posValida(c, p) \wedge \neg(\emptyset?(posiciones))\}$

minDist : campus *c* × pos *p* × conj(posiciones) *posiciones* $\longrightarrow$ nat
$\{posValida(c, p) \wedge \neg(\emptyset?(posiciones))\}$

distMan : campus *c* × pos *p*1 × pos *p*2 $\longrightarrow$ nat $\{posValida(c, p1) \wedge posValida(c, p2)\}$

restaAbs : nat × nat $\longrightarrow$ nat

conjPos : campus × nat × nat $\longrightarrow$ conj(pos)

hayObstaculo? : campus *c* × pos *p* $\longrightarrow$ bool {posValida(c,p)}

axiomas ∀ *alto*:nat, ∀ *ancho*:nat, ∀ *obst*:conj (pos)
∀ p_1:pos ∀ p_2:pos

alto(nuevo(*ancho*,*alto*,*obst*)) $\equiv$ *alto*

ancho(nuevo(*ancho*,*alto*,*obst*)) $\equiv$ *ancho*

obstaculos(nuevo(*ancho*,*alto*,*obst*)) $\equiv$ *obst*

posValida(nuevo(*ancho*,*alto*,*obst*),p_1) $\equiv$ $\pi_1(p_1) < ancho \wedge \pi_2(p_1) < alto$

adyacente(nuevo(*ancho*,*alto*,*obst*),p_1,p_2) $\equiv$ $(\pi_1(p_1) = \pi_1(p_2) - 1 \vee \pi_1(p_1) = \pi_1(p_2) + 1) \wedge$
$(\pi_2(p_1) = \pi_2(p_2) - 1 \vee \pi_2(p_1) = \pi_2(p_2) + 1)$

minDistsPos(c,p,posiciones) $\equiv$

```
if $\emptyset?(sinUno(posiciones))$ then
    $dameUno(posiciones)$
else
    if $distMan(c, p, dameUno(posiciones)) \leq$
    $minDist(c, p, posiciones)$ then
        $Ag(minDistsPos(c, sinUno(posiciones)),$
        $dameUno(posiciones))$
    else
        $minDistsPos(c, seg, sinUno(posiciones))$
    fi
fi
```

```
minDist(c,p,posiciones)      ≡ if $\emptyset?(sinUno(posiciones))$ then
                                   $distMan(c, p, dameUno(posiciones))$
                               else
                                   if $distMan(c, p, dameUno(posiciones)) \leq minDist(c, pos/p, sinUno(posiciones))$
                                   then
                                       $distMan(c, p, dameUno(posiciones))$
                                   else
                                       $minDist(c, p, sinUno(posiciones))$
                                   fi
                               fi
distMan(c,$p_1$,$p_2$)           ≡ $restaAbs(\pi_2(p_1), \pi_2(p_2)) + restaAbs(\pi_1(p_1), \pi_1(p_2))$
restaAbs(n1,n2)              ≡ if $n2 > n1$ then $n2 - n1$ else $n1 - n2$ fi
conjPos(c,x,y)               ≡ if $x \geq ancho(c)$ then
                                   $\emptyset$
                               else
                                   if $y \geq alto(c)$ then
                                       $conjPos(c, x + 1, 0)$
                                   else
                                       $ag(conjPos(c, x, y + 1), < x, y >)$
                                   fi
                               fi
hayObstaculo?(c,p)           ≡ $p \in obstaculos(c)$
```

Fin TAD