Algoritmos y Estructura de Datos 2

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

Exorcismo Extremo

TP1

Integrante	LU	Correo electrónico
Rosinov, Gaston Einan	37/18	grosinov@gmail.com
Schuster, Martin Ariel	208/18	m.a.schuster98@gmail.com
Panichelli, Manuel	72/18	panicmanu@gmail.com

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Índice

1. Extensiones y Renombres

TAD Fantasma **ES** Nat

TAD PJ **ES** Nat

TAD Posicion **ES** Tupla(Nat, Nat)

TAD Nat **extiende** Nat

otras operaciones

$\bullet \,\%\, \bullet$: nat $\times$ nat $\longrightarrow$ nat

axiomas $\forall\, n, m$: nat

n % m $\equiv$ **if** n $<$ m **then** n **else** (n - m) % m **fi**

Fin TAD

TAD Ubicacion **extiende** Tupla(Posicion, Direccion)

otras operaciones

direccion

pos : ubicacion $\longrightarrow$ posicion

dir : ubicacion $\longrightarrow$ direccion

axiomas $\forall\, u$: ubicacion

pos(u) $\equiv$ Π_1(u)

dir(u) $\equiv$ Π_2(u)

Fin TAD

TAD Secuencia **extiende** Secuencia

otras operaciones

$\bullet[\bullet]$: secu(α) s $\times$ nat i $\longrightarrow$ α $\{\text{i} < \text{long(s)}\}$

axiomas $\forall\, s$: secu(α), $\forall\, i$: nat

s[i] $\equiv$ **if** $i = 0$? **then** prim(s) **else** fin(s)[$i - 1$] **fi**

Fin TAD

2. TAD Juego

TAD Juego

géneros juego

exporta juego, observadores, generadores, puntaje, ronda

usa Secuencia, Nat, Conjunto, Habitacion, Ubicacion, PJ, Accion, Direccion, Fantasma, PJ, Diccionario, Bool

igualdad observacional

$$(\forall j, j' : \text{juego}) \left(j =_{\text{obs}} j' \iff \left(\begin{array}{l} (\text{accionesPJs(j)} =_{\text{obs}} \text{accionesPJs(j')}) \wedge \\ (\text{accionesFan(j)} =_{\text{obs}} \text{accionesFan(j')}) \wedge \\ (\text{localizarJugadores(j)} =_{\text{obs}} \text{localizarJugadores(j')}) \wedge \\ (\text{hab(j)} =_{\text{obs}} \text{hab(j')}) \wedge \\ ((\forall\ \text{p} : \text{pj})\ (\text{vivePJ?(j, p)} =_{\text{obs}} \text{vivePJ?(j', p)})) \wedge \\ ((\forall\ \text{f} : \text{fantasma})\ ((\text{viveFan?(j, p)} =_{\text{obs}} \text{viveFan?(j', p)}) \wedge \\ (\text{ubicacionInicialFan(j, f)} =_{\text{obs}} \text{ubicacionInicialFan(j', f)})) \end{array} \right) \right)$$

observadores básicos

accionesPJs	: juego	$\longrightarrow$	dicc(pj, secu(accion))	
accionesFan	: juego	$\longrightarrow$	dicc(pj, secu(accion))	
hab	: juego	$\longrightarrow$	hab	
vivePJ?	: juego j $\times$ pj p	$\longrightarrow$	bool	{p $\in$ jugadores(j)}
viveFan?	: juego j $\times$ fantasma f	$\longrightarrow$	bool	{f $\in$ fantasmas(j)}
ubicacionInicialFan	: juego j $\times$ fantasma f	$\longrightarrow$	ubicacion	{f $\in$ fantasmas(f)}
localizarJugadores	: juego	$\longrightarrow$	dicc(pj, ubicacion)	

generadores

iniciar	: conj(pj) pjs $\times$ secu(accion) as $\times$ ubicacion u $\times$ hab h	$\longrightarrow$	juego	{esConexa?(h) $\wedge$ $\neg$ $\emptyset$?(as) $\wedge$ $\neg$ $\emptyset$?(pjs) $\wedge$ esValida?(h, pos(u))}
proxPaso	: juego j $\times$ pj p $\times$ accion a	$\longrightarrow$	juego	{p $\in$ jugadores(j) $\wedge_{\text{L}}$ vivePJ?(j, p) $\wedge$ $\neg$ termino?(j) $\wedge$ $\neg$ esMirar(a)}

otras operaciones

jugadores	: juego	$\longrightarrow$	conj(pj)	
fantasmas	: juego	$\longrightarrow$	conj(fantasma)	
nombreSiguienteFan	: juego	$\longrightarrow$	fantasma	
puntaje	: juego	$\longrightarrow$	nat	
ronda	: juego	$\longrightarrow$	nat	
paso	: juego	$\longrightarrow$	nat	
cantAcciones	: juego $\times$ conj(pj)	$\longrightarrow$	nat	
terminaRonda	: juego j $\times$ pj p $\times$ accion	$\longrightarrow$	bool	{p $\in$ jugadores(j)}
fantasmaEspecial	: juego	$\longrightarrow$	fantasma	
termino?	: juego	$\longrightarrow$	bool	
estanVivos	: juego $\times$ conj(pj) pjs	$\longrightarrow$	bool	{pjs $\subseteq$ jugadores(j)}

ubicacionInicialPJ	: juego j × pj p	⟶	ubicacion	{p ∈ jugadores(j)}
ubicacionPJ	: juego j × pj p	⟶	ubicacion	{p ∈ jugadores(j)}
ubicacionFan	: juego j × fantamsa f	⟶	ubicacion	{f ∈ fantasmas(j)}
deducirUbicacion	: juego j × ubicacion u × acciones	⟶	ubicacion	{esValida?(hab(j), pos(u))}
moriraFantasma	: juego j × pj p × accion × fantasma f	⟶	bool	{p ∈ jugadores(j) ∧ f ∈ fantasmas(j) ∧ₗ viveFan?(j, f) ∧ vivePJ?(j, p)}
moriraPJ	: juego j × conj(fantasma) fs × pj p × accion	⟶	bool	{p ∈ jugadores(j) ∧ fs ⊆ fantasmas(j) ∧ₗ vivePJ?(j, p)}
moriraPJPorFan	: juego j × fantasma f × pj p × accion	⟶	bool	{p ∈ jugadores(j) ∧ f ∈ fantasmas(j) ∧ₗ vivePJ?(j, p) ∧ viveFan?(j, f)}
accionFan	: juego j × fantasma f	⟶	accion	{f ∈ fantasmas(j) ∧ₗ viveFan?(j, f)}
inicializarAcciones	: conj(pj)	⟶	dicc(pj, secu(accion))	
agregarFantasma	: juego j × ubicacion u × dicc(fantasma × secu(accion)) × fantasma × secu(accion)	⟶	dicc(fantasma, secu(accion))	{esValida?(hab(j), pos(u))}
generarAccionesFan	: juego j × ubicacion u × secu(accion)	⟶	secu(accion)	{esValida?(hab(j), pos(u))}

axiomas

∀ p: pj,
∀ pjs: conj(pj),
∀ f: fantasma,
∀ fs: conj(fantasma),
∀ j: juego,
∀ h: hab,
∀ $u, uInicialPJ$: ubicacion,
∀ a: accion,
∀ as: secu(accion)

accionesPJs(iniciar(pjs, as, u, h))	≡	inicializarAcciones(pjs)
accionesPJs(proxPaso(j, p, a))	≡	**if** ¬ terminaRonda(j, p, a) **then** definir(p, obtener(p, accionesPJs(j)) ∘ a, accionesPJs(j)) **else** inicializarAcciones(jugadores(j)) **fi**
accionesFan(iniciar(pjs, as, u, h))	≡	definir(nombreSiguienteFan(j), as, vacio)
accionesFan(proxPaso(j, p, a))	≡	**if** ¬ terminaRonda(j, p, a) **then** accionesFan(j) **else** agregarFantasma(j, ubicacionInicialPJ(j, p), accionesFan(j), nombreSiguienteFan(j), obtener(p, accionesPJs(j)) ∘ a) **fi**
hab(iniciar(pjs, as, u, h))	≡	h
hab(proxPaso(j, p, a))	≡	hab(j)
vivePJ?(iniciar(pjs, as, u, h), p')	≡	true

vivePJ?(proxPaso(j, p, a), p')	≡	terminaRonda?(j, p, a) ∨ **if** p = p' **then** ¬ moriraPJ(j, fantasmas(j), p, a) **else** vivePJ?(j, p') ∧$_L$ ¬ moriraPJ(j, fantasmas(j), p, a) **fi**
viveFan?(iniciar(pjs, as, u, h), f)	≡	true
viveFan?(proxPaso(j, p, a), f)	≡	terminaRonda?(j, p, a) ∨ (viveFan?(j, f) ∧$_L$ ¬ moriraFantasma(j, p, a, f))
ubicacionInicialFan(iniciar(pjs, as, u, h))	≡	u
ubicacionInicialFan(proxPaso(j, p, a))	≡	**if** f ∈ fantasmas(j) **then** ubicacionInicialFan(j, f) **else** ubicacionInicialPJ(j, p) **fi**
jugadores(j)	≡	claves(accionesPJs(j))
fantasmas(j)	≡	claves(accionesFan(j))
nombreSiguienteFan(j)	≡	#(claves(accionesFan(j))) + 1
puntaje(j)	≡	ronda(j) - 1
ronda(j)	≡	#(fantasmas(j))
paso(j)	≡	cantAcciones(j, jugadores(j))
cantAcciones(j, pjs)	≡	**if** ∅?(pjs) **then** 0 **else** long(obtener(dameUno(pjs), accionesPJs(j))) + cantAcciones(j, sinUno(pjs)) **fi**
termino?(j)	≡	¬ estanVivos(j, jugadores(j))
estanVivos(j, pjs)	≡	**if** ∅?(pjs) **then** true **else** vivePJ?(j, dameUno(pjs)) ∧ estanVivos(j, sinUno(pjs)) **fi**
fantasmaEspecial(j)	≡	#(claves(accionesFan(j)))
ubicacionInicialPJ(j, p)	≡	obtener(p, localizarJugadores(j))
ubicacionPJ(j, p)	≡	deducirUbicacion(j, ubicacionInicialPJ(j, p), obtener(p, accionesPJs(j)))
ubicacionFan(j, f)	≡	deducirUbicacion(j, ubicacionInicialFan(j, f), obtener(f, accionesFan(j)))
deducirUbicacion(j, u, as)	≡	**if** vacia?(as) **then** u **else** deducirUbicacion(j, ubicacionLuegoDe(prim(as), hab(j), u), fin(as)) **fi**
agregarFantasma(j, uInicialPJ, accionesFantasmas, f, as)	≡	definir(f, generarAccionesFantasma(j, uInicialPJ, as), accionesFantasmas)
generarAccionesFan(j, uInicialPJ, as)	≡	as & (nada • nada • nada • nada • nada • <>) & invertir(hab(j), uInicialPJ, as)
inicializarAcciones(pjs)	≡	**if** ∅?(pjs) **then** vacio **else** definir(dameUno(pjs), <>, inicializarAcciones(sinUno(pjs))) **fi**

terminaRonda(j, p, a)	≡	moriraFantasma(j, p, a, fantasmaEspecial(j))
moriraFantasma(j, p, a, f)	≡	pos(ubicacionFan(j, f)) ∈ posicionesAfectadasPor(a, hab(j), ubicacionPJ(j, p))
moriraPJ(j, fs, p, a)	≡	**if** ∅?(fs) **then** false **else** (viveFan?(j, dameUno(fs)) $\wedge_{\text{L}}$ moriraPJPorFan(j, dameUno(fs), p, a)) $\vee_{\text{L}}$ moriraPJ(j, sinUno(fs), p, a) **fi**
moriraPJPorFan(j, f, p, a)	≡	¬ moriraFantasma(j, p, a, f) ∧ (pos(ubicacionLuegoDe(a, hab(j), ubicacionPJ(j, p))) ∈ posicionesAfectadasPor(accionFan(j, f), hab(j), ubicacionFan(j, f)))
accionFan(j, f)	≡	obtener(accionesFan(j), f)[paso(j) % obtener(accionesFan(j), f)]

Fin TAD

3. TAD Habitacion

TAD Habitacion

géneros hab

exporta hab, observadores, generadores, esConexa?

usa posicion, bool, nat

igualdad observacional

$$(\forall h, h' : \text{hab}) \left(h =_{\text{obs}} h' \iff \begin{pmatrix} (\forall p : \text{posicion})(\text{esValida?}(p, h) =_{\text{obs}} \text{esValida?}(p, h') \wedge_{\text{L}} \\ (\text{esValida?}(p, h) \Rightarrow_{\text{L}} \\ (\text{estaOcupada?}(p, h) =_{\text{obs}} \text{estaOcupada?}(p, h')))) \end{pmatrix} \right)$$

observadores básicos

esValida?	: hab × posicion	⟶ bool	
estaOcupada?	: hab h × posicion p	⟶ bool	{esValida?(h, p)}

generadores

nueva	: nat n	⟶ hab	{n>1}
ocupar	: hab h × posicion p	⟶ hab	{esValida?(h, p) $\wedge_{\text{L}}$¬ estaOcupada?(h, p)}

otras operaciones

esConexa?	: hab	⟶ bool	
tamano	: hab	⟶ nat	
posiciones	: hab	⟶ conj(posicion)	
darPosiciones	: hab × nat × nat × nat	⟶ conj(posicion)	
posicionesLibres	: hab h × conj(posicion) ps	⟶ conj(posicion)	{ps ⊆ posiciones(h)}
verificarAlcance	: hab h × conj(posicion) ps	⟶ bool	{ps ⊆ posiciones(h)}
verificarAlcancePos	: hab h × conj(posicion) ps × posicion p	⟶ bool	{ps ⊆ posiciones(h) ∧ p ∈ posiciones(h)}
esAlcanzable	: hab h × pos p × pos p'	⟶ bool	

$$\left\{\begin{array}{l}\text{esValida?(h, p)} \land \text{esValida?(h, p')} \land_{\text{L}} \\ (\neg\ \text{estaOcupada?(h, p)} \land \neg\ \text{estaOcupada?(h, p))}\end{array}\right\}$$

axiomas $\forall\ h$: hab $\forall\ ps$: conj(posicion) $\forall\ p$: posicion $\forall\ n, k, tam$: nat

esValida?(nueva(n),p)	$\equiv$	$0 \leq \Pi_1(\text{p}) < \text{n} \land 0 \leq \Pi_2(\text{p}) < \text{n}$
esValida?(ocupar(h,p'),p)	$\equiv$	$\text{p} = \text{p'} \lor_{\text{L}}$ esValida?(h, p)
estaOcupada?(nueva(n),p)	$\equiv$	false
estaOcupada?(ocupar(h,p'),p)	$\equiv$	$\text{p} = \text{p'} \lor$ estaOcupada?(h, p)
tamano(nueva(n))	$\equiv$	n
tamano(ocupar(h, p))	$\equiv$	tamano(h)
esConexa?(h)	$\equiv$	verificarAlcance(h, posicionesLibres(posiciones(h)))
posicionesLibres(h, ps)	$\equiv$	**if** $\emptyset?(ps)$ **then** $\emptyset$ **else** (**if** estaOcupada?(h, dameUno(ps)) **then** $\emptyset$ **else** {dameUno(ps)} **fi**) $\cup$ posicionesLibres(h, sinUno(ps)) **fi**
verificarAlcance(h, ps)	$\equiv$	**if** $\emptyset?(ps)$ **then** true **else** verificarAlancePos(h, ps, dameUno(ps)) $\land$ verificarAlcance(h, p) **fi**
verificarAlcancePos(h, ps, p)	$\equiv$	**if** $\emptyset?(ps)$ **then** true **else** esAlcanzable(h, p, dameUno(ps)) $\land$ verificarAlcancePos(h, p, sinUno(ps)) **fi**
posiciones(h)	$\equiv$	darPosiciones(h, tamano(h) - 1, tamano(h) - 1, tamano(h) - 1)
darPosiciones(h, n, k, tam)	$\equiv$	**if** $n = 0? \land k = 0?$ **then** $\emptyset$ **else if** $k = 0?$ **then** Ag((n,k), darPosiciones(h, n - 1, tam, tam)) **else** Ag((n,k), darPosiciones(h, n, k - 1, tam)) **fi** **fi**

Fin TAD

4. TAD Accion

TAD Accion

géneros accion

exporta accion, observadores, generadores, genero, otras operaciones

usa Direccion, Posicion, Ubicacion, Bool, Conjunto, Habitacion, Secuencia

igualdad observacional

$$(\forall a, a' : \text{accion}) \left(a =_{\text{obs}} a' \iff \begin{pmatrix} \text{esNada(a)} =_{\text{obs}} \text{esNada(a')} \wedge \\ \text{esDisparar(a)} =_{\text{obs}} \text{esDisparar(a')} \wedge \\ \text{esMover(a)} =_{\text{obs}} \text{esMover(a')} \wedge \\ \text{esMirar(a)} =_{\text{obs}} \text{esMirar(a')} \wedge \\ ((\text{esMover(a)} \vee \text{esMirar(a)}) \Rightarrow_{\text{L}} \\ \text{direccion(a)} =_{\text{obs}} \text{direccion(a')}) \end{pmatrix} \right)$$

observadores básicos

esMover	: accion	$\longrightarrow$	bool	
esMirar	: accion	$\longrightarrow$	bool	
esDisparar	: accion	$\longrightarrow$	bool	
esNada	: accion	$\longrightarrow$	bool	
direccion	: accion a	$\longrightarrow$	direccion	{esMirar(a) $\vee$ esMover(a)}

generadores

mover	: direccion	$\longrightarrow$	accion	
mirar	: direccion	$\longrightarrow$	accion	
disparar	:	$\longrightarrow$	accion	
nada	:	$\longrightarrow$	accion	

otras operaciones

ubicacionLuegoDe	: accion $\times$ hab h $\times$ ubicacion u	$\longrightarrow$	conj(posicion)	{esValida?(h, pos(u))}
posicionesAfectadasPor	: accion $\times$ hab h $\times$ ubicacion u	$\longrightarrow$	conj(posicion)	{esValida?(h, pos(u))}
$\neg \bullet$	: accion	$\longrightarrow$	accion	
invertir	: hab h $\times$ ubicacion u $\times$ secu(accion)	$\longrightarrow$	secu(accion)	{esValida?(h, pos(u))}

axiomas $\forall$ d: direccion $\forall$ u: ubicacion, $\forall$ a: habitacion, $\forall$ as: secu(accion)

posicionesAfectadasPor(mover(d), h, u) $\equiv \emptyset$

posicionesAfectadasPor(mirar(d), h, u) $\equiv \emptyset$

posicionesAfectadasPor(nada, h, u) $\equiv \emptyset$

posicionesAfectadasPor(disparar, h, u) $\equiv$ **if** esValida?(h, proxPosEnDir(dir(u), pos(u)) $\wedge_{\text{L}}$
$\neg$ estaOcupada?(h, proxPosEnDir(dir(u), pos(u)))
then Ag(proxPosEnDir(dir(u), pos(u)),
posicionesAfectadasPor(disparar, h,
$\langle$ proxPosEnDir(dir(u), pos(u)), dir(u) $\rangle$))
else $\emptyset$
fi

invertir(h, u, as)	≡	**if** vacia?(as) **then** <> **else** invertir(h, ubicacionLuegoDe(prim(as), h, u), fin(as)) ∘ ¬(prim(as), h, u) **fi**
¬(mover(d), h, u)	≡	**if** pos(ubicacionLuegoDe(mover(d), h, u)) = pos(u) **then** mirar(opuesta(d)) **else** mover(opuesta(d)) **fi**
¬(mirar(d), h, u)	≡	mirar(opuesta(d))
¬(disparar, h, u)	≡	disparar
¬(nada, h, u)	≡	nada
ubicacionLuegoDe(nada, h, u)	≡	u
ubicacionLuegoDe(disparar, h, u)	≡	u
ubicacionLuegoDe(mirar(d), h, u)	≡	⟨ pos(u), d ⟩
ubicacionLuegoDe(mover(d), h, u)	≡	⟨ (**if** esValida?(h, proxPosEnDir(d, pos(u))) $\wedge_{\text{L}}$ ¬estaOcupada?(h, proxPosEnDir(d, pos(u))) **then** proxPosEnDir(d, pos(u)) **else** pos(u) **fi**), d ⟩
esMirar(mirar(d))	≡	**true**
esMirar(mover(d))	≡	false
esMirar(disparar)	≡	false
esMirar(nada)	≡	false
esMover(mirar(d))	≡	false
esMover(mover(d))	≡	**true**
esMover(disparar)	≡	false
esMover(nada)	≡	false
esDisparar(mirar(d))	≡	false
esDisparar(mover(d))	≡	false
esDisparar(disparar)	≡	**true**
esDisparar(nada)	≡	false
esNada(mirar(d))	≡	false
esNada(mover(d))	≡	false
esNada(disparar)	≡	false
esNada(nada)	≡	**true**
direccion(mirar(d))	≡	d
direccion(mover(d))	≡	d

Fin TAD

5. TAD Direccion

TAD Direccion

géneros direccion

exporta direccion, observadores, generadores, otras operaciones

usa Bool, Posicion, Nat

igualdad observacional

$$(\forall d, d' : \text{direccion}) \left(d =_{\text{obs}} d' \iff \begin{pmatrix} \text{esArriba(d)} =_{\text{obs}} \text{esArriba(d')} \wedge \\ \text{esAbajo(d)} =_{\text{obs}} \text{esAbajo(d')} \wedge \\ \text{esIzquierda(d)} =_{\text{obs}} \text{esIzquierda(d')} \wedge \\ \text{esDerecha(d)} =_{\text{obs}} \text{esDerecha(d')} \end{pmatrix} \right)$$

observadores básicos

esArriba	: direccion	$\longrightarrow$	bool
esAbajo	: direccion	$\longrightarrow$	bool
esIzquierda	: direccion	$\longrightarrow$	bool
esDerecha	: direccion	$\longrightarrow$	bool

generadores

arriba	:	$\longrightarrow$	direccion
abajo	:	$\longrightarrow$	direccion
izquierda	:	$\longrightarrow$	direccion
derecha	:	$\longrightarrow$	direccion

otras operaciones

opuesta	: direccion	$\longrightarrow$	direccion
proxPosEnDir	: direccion $\times$ posicion	$\longrightarrow$	posicion

axiomas $\forall$ p: posicion

opuesta(arriba)	$\equiv$	abajo
opuesta(abajo)	$\equiv$	arriba
opuesta(izquierda)	$\equiv$	derecha
opuesta(derecha)	$\equiv$	izquierda
proxPosEnDir(arriba, p)	$\equiv$	$\langle\ \Pi_1(\text{p}), \Pi_2(\text{p}) + 1\ \rangle$
proxPosEnDir(abajo, p)	$\equiv$	$\langle\ \Pi_1(\text{p}), \Pi_2(\text{p}) - 1\ \rangle$
proxPosEnDir(izquierda, p)	$\equiv$	$\langle\ \Pi_1(\text{p}) - 1, \Pi_2(\text{p})\ \rangle$
proxPosEnDir(derecha, p)	$\equiv$	$\langle\ \Pi_1(\text{p}) + 1, \Pi_2(\text{p}\ \rangle$
esArriba(arriba)	$\equiv$	**true**
esArriba(abajo)	$\equiv$	false
esArriba(izquierda)	$\equiv$	false
esArriba(derecha)	$\equiv$	false
esAbajo(arriba)	$\equiv$	false
esAbajo(abajo)	$\equiv$	**true**
esAbajo(izquierda)	$\equiv$	false
esAbajo(derecha)	$\equiv$	false

esIzquierda(arriba)	$\equiv$	false
esIzquierda(abajo)	$\equiv$	false
esIzquierda(izquierda)	$\equiv$	**true**
esIzquierda(derecha)	$\equiv$	false
esDerecha(arriba)	$\equiv$	false
esDerecha(abajo)	$\equiv$	false
esDerecha(izquierda)	$\equiv$	false
esDerecha(derecha)	$\equiv$	**true**

Fin TAD

6. Decisiones tomadas

1. Asumimos la existencia de las operaciones localizarJugadores y esAlcanzable, entonces no las axiomatizamos.
2. Tomamos como validas dentro del juego habitaciones de 2x2 o mas, ya que sino nadie moriria.
3. Como los nombres de los fantasmas son Nat, el nombre del nuevo es el cardinal del el conjunto de fantasmas + 1. Asi tambien se puede facilmente saber cual es el fantasma especial.
4. La accion mirar fue agregada para considerar el caso patologico del fantasma.
   Esta fue reestringida para que solo se pueda usar en ese caso.
   Esto llevo a la creacion del TAD Direccion como un desprendimiento de Accion.