

Procedimientos

Existentes en Gobstones

Procedimiento	Explicación	Ejemplo
Poner (color)	Pone una bolita del color indicado en la casilla actual	Poner (Rojo) pone una bolita roja en la casilla actual.
Sacar (color)	Saca una bolita del color indicado de la casilla actual	Sacar (Negro) saca una bolita negra de las que hay en la casilla actual.
Mover (direc)	Mueve el cabezal indicador de la casilla actual un paso hacia la dirección indicada.	Mover (Este) mueve el cabezal una vez hacia el Este.
IrAlBorde (direc)	Lleva el cabezal todo lo que se puede hacia la dirección indicada	IrAlBorde (Norte) mueve el cabezal de la celda actual a la última celda en la dirección Norte.
VaciarTablero ()	Saca todas las bolitas del tablero, dejando el cabezal en la posición en la que estaba.	En un tablero con alguna ó muchas bolitas, VaciarTablero () las saca todas.

Creados en las prácticas

Procedimiento	Explicación	Ejemplo:  (1A2N4R)
PonerN (cantBolitas, color)	Pone en la casilla actual tantas bolitas como se indique, del color que se indique.	PonerN (4, Rojo) pone 4 bolitas rojas en el casillero actual
SacarN (cantBolitas, color)	Saca de la casilla actual tantas bolitas como se indique, del color que se indique.	SacarN (2, Verde) saca del casillero actual 2 bolitas.
MoverN (cantidad, direc)	Mueve el cabezal tantas veces como se indique en la dirección que se indique.	MoverN (3, Este) mueve el cabezal 3 veces hacia el este.
Línea (longitud, direc, color)	Hace una "línea" de bolitas desde la casilla actual y en la dirección indicada, dejando el cabezal en la posición donde comenzó. La línea se representa por una bolita del color indicado en cada casilla de la línea.	Línea (5, Norte, Azul) hace una línea hacia arriba de 5 bolitas dejando 1 bolita por celda, y el cabezal al principio de la línea.
SacarTodas (color)	Saca todas las bolitas del color indicado de la celda actual.	SacarTodas (Rojo) en el casillero de ejemplo, deja el casillero con una azul y 2 negras.

Expresiones

Existentes en Gobstones

Expresión	¿Qué tipo denota?	¿Qué significa lo que denota?	Ejemplo:  (1A2N4R)
nroBolitas (color)	número	La cantidad de bolitas del color indicado que hay en la casilla actual.	nroBolitas (Rojo) denota 4
opuesto (direc)	dirección	La dirección opuesta a la provista	opuesto (Norte) denota Sur
opuesto (número)	número	El número negado	opuesto (59) denota -59
siguiente (direc)	dirección	La siguiente dirección a la provista. Es decir, la próxima en sentido horario.	siguiente (Oeste) denota Norte
previo (direc)	dirección	La dirección anterior a la provista. Es decir, la próxima en sentido anti horario.	previo (Sur) denota Este
hayBolitas (color)	booleano	Es cierto cuando en la casilla actual hay al menos una bolita del valor indicado.	En la casilla de ejemplo, hayBolitas (Rojo) denota cierto, hayBolitas (Verde) denota falso.
puedeMover (direc)	booleano	Si el cabezal puede moverse en esa dirección (o sea, no está en el borde).	Estando el cabezal en la esquina de abajo a la izquierda, puedeMover (Norte) denota cierto, mientras que puedeMover (Oeste) denota falso.
< > >= <= == /=	booleano	menor que mayor que mayor o igual que menor o igual que igual que distinto que	5 >= 3 denota cierto. 5 >= 5 denota cierto. 5 < 5 denota falso. En la casilla de ejemplo, nroBolitas (Rojo) == nroBolitas (Azul) denota falso.
not	booleano	Sencillamente <i>da vuelta</i> la respuesta de una expresión booleana.	En la casilla de ejemplo, not hayBolitas (Verde) denota cierto.
+ - div mod	número	suma resta división entera resto de la división	4 + 2 denota 6 6 div 2 denota 3 9 div 4 denota 2 9 mod 4 denota 1

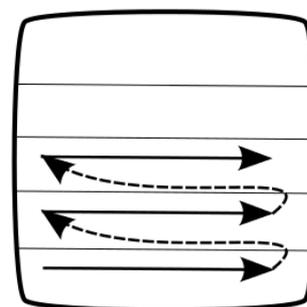
Matemática VS Gobstones

Matemática	Gobstones
$p \wedge q$	<code>p && q</code>
$p \vee q$	<code>p q</code>
$\neg p$	<code>not p</code>
$x = y$	<code>x == y</code>
$x \neq y$	<code>x /= y</code>
$x \geq y$	<code>x >= y</code>

Recorridos del tablero

Recordar que las siguientes funciones y procedimientos están hechos para un recorrido del tablero que va por filas de Oeste a Este, empezando por la fila de abajo, fila por fila hacia arriba.

Esto significa que cada paso avanza al Este, salvo cuando llega al borde, que su próximo paso es ir al inicio de la fila siguiente al Norte.



Nota importante: Si el parcial ó ejercicio requiriese un recorrido diferente, (por ejemplo empezando desde el Norte y yendo al Oeste), entonces deberíase implementar el mismo.

Procedimiento	Explicación	Ejemplo
<code>IrAlOrigen()</code> (procedimiento)	Mueve el cabezal al inicio del recorrido del tablero. El inicio es la casilla 0,0 (todo al Sur, todo al Oeste)	<code>IrAlOrigen()</code> en cualquier parte de cualquier tablero hace que el cabezal quede en la posición inicial del recorrido.
<code>haySiguiete(color)</code> (función)	Es una función que denota <i>booleano</i> . Dice si es posible dar un paso más en el recorrido del tablero.	<code>haySiguiete()</code> en la esquina Norte Este del tablero da False.
<code>IrAlSiguiete()</code> (procedimiento)	Mueve el cabezal a a la siguiente posición en el recorrido del tablero. En una celda central, mueve al Este. En un extremo Este, mueve al borde Oeste de la fila siguiente al Norte.	<code>IrAlSiguiete()</code> en el final de la fila 0, mueve al 1,0 (fila 1 columna 0).

Registros

Definición

Supongamos que tenemos un registro `Persona`, que está definido de la siguiente forma:

```
type Persona is record {
  field dni
  field colorFavorito
  field edad
}
```

Creación

Creación de una persona con dni 32.222.222, color favorito azul y edad 29 años:

```
Persona(dni <- 32222222, colorFavorito <- Azul, edad <- 29)
```

Accessors / Proyectores / Observadores

Para saber	Escribo
Si me guardo a Pedro en una variable: <code>pedro := Persona(dni <- 32222222, colorFavorito <- Azul, edad <- 29)</code>	
Cuál es el DNI de Pedro	<code>dni(pedro)</code>
Cuál es el color favorito de Pedro	<code>colorFavorito(pedro)</code>
Cuál es la edad de Pedro	<code>edad(pedro)</code>

Modificando Registros

¡No se puede! El truco es **crear uno nuevo**. Hay dos formas de hacerlo:

Primera forma: copiando todo y cambiando lo que quiero

```
pedro := Persona(dni <- 32222222, colorFavorito <- Azul, edad <- 29)
pedroNuevo := Persona(dni <- dni(pedro), colorFavorito <- colorFavorito(pedro), edad <- 30)
```

Segunda forma: usando “el truquito de la barra vertical”

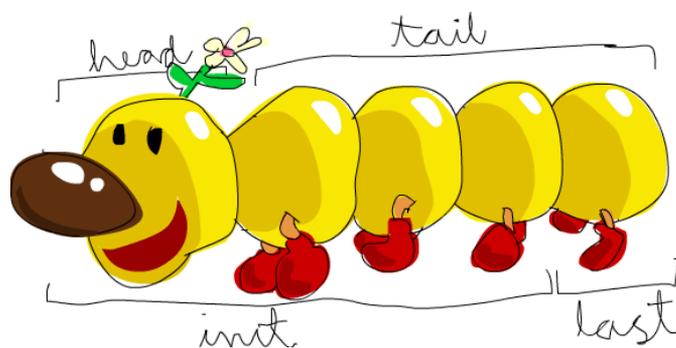
```
pedro := Persona(dni <- 32222222, colorFavorito <- Azul, edad <- 29)
pedroNuevo := Persona( pedro | edad <- 30 )
```

Básicamente esto significa “quiero una persona como Pedro, **pero** con la edad en 30”.

Listas

Definición

Lista con tres elementos	Lista con un elemento	Lista vacía
[Azul, Azul, Verde]	[45]	[]



Expresiones existentes en Gobstones

Expresión	¿Qué tipo denota?	¿Qué significa lo que denota?	Ejemplo: dirs := [Norte, Este, Este, Sur] vacía := []
head(lista)	elemento	El primer elemento de la lista.	head(dirs) denota Norte head(vacía) hace Boom .
tail(lista)	lista	La cola de la lista. Puede ser una lista vacía, tener un elemento ó muchos, pero siempre denota una nueva lista.	tail(dirs) denota [Este, Este, Sur] tail(vacía) hace Boom .
init(lista)	lista	El principio de la lista, sin el último elemento. Puede ser una lista vacía, tener un elemento ó muchos, pero siempre denota una nueva lista.	init(dirs) denota [Norte, Este, Este] init(vacía) hace Boom .
last(lista)	lista	El último elemento de la lista.	last(dirs) denota Sur last(vacía) hace Boom .
lista ++ otraLista	lista	Una nueva lista que tiene todos los elementos de ambas listas. O sea, concatena ambas listas en orden.	dirs ++ [Este, Sur] denota [Norte, Este, Este, Sur, Este, Sur] dirs ++ vacía denota [Norte, Este, Este, Sur] dirs ++ Oeste hace Boom (Oeste no es una lista)
[x..y]	lista	Una nueva lista ordenada con todos los elementos entre x e y, incluyendo a x y a y. ¡Esto se llama rango !	[5..11] denota [5, 6, 7, 8, 9, 10, 11] [Norte..Oeste] denota [Norte, Este, Sur, Oeste]

Funciones hechas en las prácticas

Procedimiento	Tipo de lista que admite	Explicación	Ejemplo
<code>tieneAlgo(lista)</code>	Cualquiera	Indica si en la lista hay algún elemento. Es idéntico a decir <code>not isEmpty(lista)</code>	<code>tieneAlgo([3,4])</code> denota <code>True</code> . <code>tieneAlgo([])</code> denota <code>False</code> .
<code>pertenece(e, lista)</code>	Cualquiera	Denota <code>True</code> si el elemento <code>e</code> está en la lista, y <code>False</code> si no.	<code>pertenece(4, [3,4])</code> denota <code>True</code> .
<code>longitud(lista)</code>	Cualquiera	Denota la cantidad de elementos que tiene la lista.	<code>longitud([5,7,8])</code> denota 3.
<code>sinElem(e, lista)</code>	Cualquiera	Denota la lista resultante de sacar <code>e</code> de esa lista. Elimina todas las ocurrencias de <code>e</code> .	<code>sinElem(2, [3,2,7,2])</code> denota la lista <code>[3,7]</code>
<code>sinRepetidos(lista)</code>	Cualquiera	Denota una nueva lista resultado de haber eliminado los repetidos de la lista original.	<code>sinRepetidos([1,2,1,3,3,1])</code> denota la lista <code>[1,2,3]</code> .
<code>tienenLosMismosElementos(lista1, lista2)</code>	Cualquiera	Denota si ambas listas tienen los mismos elementos, sin importar el orden	<code>tienenLosMismosElementos([1,2,3], [3,1,2])</code> denota <code>True</code>
<code>maximo(lista)</code> <code>minimo(lista)</code>	Sólo con listas numéricas	Denotan el número máximo y mínimo de la lista provista, respectivamente	<code>maximo([3,2,7])</code> denota 7 <code>minimo([3,2,7])</code> denota 2