



Gobstones

Introducción a la Programación - Práctica 6

Alternativas Condicionales

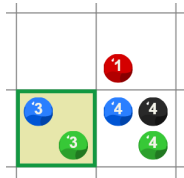
CONSEJOS:

- Leer el enunciado en su totalidad y pensar en la forma de resolverlo **ANTES** de empezar a escribir código.
- Si un ejercicio no sale, se puede dejar para después y continuar con los ejercicios que siguen.
- Los ejercicios están pensados para ser hechos después de haber mirado la teórica correspondiente, y esta después de las actividades de indagación.
- Los ejercicios están tomados de las guías prácticas utilizadas en la materia de Introducción a la Programación de la Universidad Nacional de Quilmes por Pablo Ernesto "Fidel" Martínez López y su equipo, Introducción a la Programación de la Universidad Nacional de Hurlingham de Alan Rodas Bonjour y su equipo, de ejercicios y actividades realizadas por Federico Aloï y Miguel Miloro, a su vez basada en las guías Ejercicios de Introducción a la Programación del CIU General Belgrano, elaboradas por Carlos Lombardi y Alfredo Sanzo, y Fundamentos de la Programación del Proyecto Mumuki. Agradecemos a todos los que nos ayudaron con su inspiración.
- **Realizar en papel los ejercicios que así lo indiquen.**
- **Sí un ejercicio indica [BIBLIOTECA](#) significa que será útil para la realización de futuros ejercicios tanto en esta guía como en las siguientes, pudiendo ser utilizado sin tener que volver a definirlo. Es útil mantener registro de dichos procedimientos en su carpeta.**

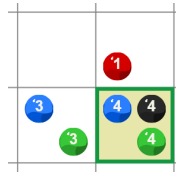
EXPRESIONES BOOLEANAS:

1. Mis primeros booleanos

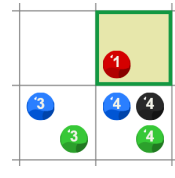
EN PAPEL Las siguientes expresiones son todas expresiones que representan booleanos. Indicar el valor de cada una evaluada para cada uno de los tableros A, B y C.



(A)



(B)



(C)

a.

```
not hayBolitas(Rojo)
```

b.

```
puedeMover(Sur) && puedeMover(Oeste)
```

c.

```
puedeMover(Sur) || puedeMover(Oeste)
```

d.

```
not puedeMover(Sur) && puedeMover(Oeste)
```

e.

```
nroBolitas(Negro) == nroBolitas(Azul)
&&
nroBolitas(Negro) == nroBolitas(Verde)
```

f.

```
puedeMover(opuesto(opuesto(dirección)))
```

suponiendo que, por alguna razón, esta expresión aparece dentro del cuerpo del procedimiento siguiente:

```
procedure Mover_SegúnColor_(dirección, color)
/*
  PROPÓSITO: Mover el cabezal en la dirección dada
             tantas celdas como el número de bolitas del
             color dado haya en la celda actual
  PRECONDICIONES: Hay al menos tantas celdas en la
             dirección dada como número de bolitas del color
             dado en la celda actual
  PARÁMETROS:
    * dirección, una dirección, hacia donde moverse
    * color, un color para saber la cantidad a moverse
*/
```

Y considerando que el mismo fue invocado como:

```
Mover_SegúnColor_(Norte, Verde)
```

- g. ¿En qué situaciones al invocar **Mover_SegúnColor_** la siguiente expresión sería verdadera, en caso de aparecer dentro del cuerpo del procedimiento?

```
not puedeMover(dirección)
&&
not puedeMover(opuesto(dirección))
```

2. Definiendo mis primeros booleanos

Definir expresiones que sean verdaderas (describen el valor de verdad Verdadero) para cada uno de los siguientes casos.

- Cuando la celda actual tiene más de 5 bolitas de color Rojo.
- Cuando la celda actual tiene al menos 9 bolitas en total entre rojas y negras.
- Cuando la celda actual es la esquina Norte-Este.
- Cuando la celda actual está vacía.
- Cuando hay una sola celda en el tablero.

ALTERNATIVA CONDICIONAL:

3. Sí se puede, sí se puede...

Escribir los siguientes procedimientos, recordando no mezclar niveles de abstracción del problema, para lo cual puede ser necesario definir otros procedimientos y/o funciones.

- SacarUnaFicha_SiSePuede(colorDeLaFicha)** que, dado el colorDeLaFicha que debe sacarse, saque una ficha siempre y cuando la misma esté en la celda. Si no hubiera fichas del color dado, el procedimiento no hace nada. Si hubiera varias fichas, solo debe sacar una.
OBSERVACIÓN: cada ficha se representa con una bolita del color correspondiente.
- DesempatarParaElLocal_Contra_(colorDelLocal, colorDelVisitante)** que, dados los colores de dos jugadores, cuyos puntos se representan mediante la cantidad de bolitas del color del jugador, otorgue un punto al jugador con color **colorDelLocal** solamente en el caso en que la celda actual contiene la misma cantidad de bolitas de ambos colores.
- ExpandirBacteriaDeLaColonia()**, que siempre que en la celda actual haya un cultivo de bacterias y haya suficientes nutrientes, agregue exactamente una bacteria más y consuma nutrientes, a razón de dos nutrientes por bacteria expandida; si no hay bacterias o no hay suficientes nutrientes, no hace nada. Las bacterias se representan con bolitas Verdes y los nutrientes con bolitas Rojas.
- PonerFlecha_AlNorteSiCorresponde(colorDeLaFlecha)**, que dado un color para representar flechas, ponga una flecha al Norte si existe espacio para moverse en esa dirección. Las flechas serán representadas con una bolita del color dado.

4. Hacer solo si...

La combinación de parámetros y expresiones booleanas es interesante.

- BIBLIOTECA** Escribir un procedimiento **Poner_Si_(color, condición)** que dado un color y un valor de verdad llamado condición, ponga en la celda actual una bolita del color dado si el valor de verdad de la condición es verdadero, y no lo ponga si no.
EJEMPLO: **Poner_Si_(Rojo, nroBolitas(Rojo) == 2)** solamente pone una bolita roja cuando hay exactamente dos rojas en la celda actual.
- BIBLIOTECA** Escribir el procedimiento **Sacar_Si_(color, condición)** que actúa de forma similar al anterior, pero ahora sacando bolitas si la condición se cumple.

- c. **BIBLIOTECA** Escribir el procedimiento **Mover_Si_(dirección, condición)** que actúa de forma similar a los anteriores, pero ahora moviendo solo si se cumple la condición dada..
- d. ¿Que beneficios trae tener los procedimientos **Sacar_Si_** y **Poner_Si_** contra utilizar if en cada caso?

5. ¡Nuevamente Nova tiene problemas!

- a. **EN PAPEL** Como Nova sigue confundido con las buenas prácticas de programación, nos consultó sobre cuál de las siguientes dos soluciones que se le ocurrieron es la correcta. El problema es sacar exactamente 8 bolitas de color Azul y la precondition, como es de esperar, es que haya al menos 8 bolitas azules en la celda actual. Explicarle a Nova cuál es la correcta, y por qué la otra no es una buena opción.

```
procedure SacarExactamente8BolitasAzulesOpciónA() {  
  /*  
    PROPÓSITO: Sacar exactamente 8 bolitas de color Azul  
    PRECONDICIONES: Hay al menos 8 bolitas de color azul  
  */  
  repeat (8) {  
    Sacar(Azul)  
  }  
}  
procedure SacarExactamente8BolitasAzulesOpciónB() {  
  /*  
    PROPÓSITO: Sacar exactamente 8 bolitas de color Azul  
    PRECONDICIONES: Hay al menos 8 bolitas de color azul  
  */  
  repeat (8) {  
    Sacar_Si_(Azul, hayBolitas(Azul))  
  }  
}
```

- b. **EN PAPEL** Ayudar a Nova a generalizar este procedimiento, escribiendo **SacarExactamente_BolitasDeColor_(cantidadASacar, colorASacar)**

6. ¿Vamos al banco? - Parte 1

En este ejercicio utilizaremos el tablero de Gobstones para representar cuentas bancarias. Cada celda representará a una cuenta bancaria, y en cada una de ellas puede haber dinero en distintas monedas, que representaremos con distintos colores:

- bolitas negras para pesos argentinos.
- bolitas verdes para dólares estadounidenses.
- bolitas azules para euros.
- bolitas rojas para yuanes chinos.

Se pueden hacer tres operaciones: depósitos, extracciones y conversiones a divisa extranjera. Las extracciones pueden hacerse en cualquier moneda, pero los depósitos siempre serán en pesos.

En el caso en que se quiera depositar un monto en una moneda extranjera, se aplicará automáticamente la conversión a pesos según el precio de venta dado en la siguiente tabla:

Precios de venta	
1 dólar	80 pesos
1 euro	90 pesos
1 yuan	12 pesos




En cuanto a la conversión a divisa extranjera, el banco actualmente aplica las siguientes tarifas para la compra de divisa:

Precios de compra	
100 pesos	1 dólar
115 pesos	1 euro
17 pesos	1 yuan




Realizar los siguientes procedimientos para poder manipular la cuenta:

- Depositar_EnMoneda_ComoPesos(cantidadADepositar, moneda)**, que dada una cantidad de dinero a depositar y un color que representa la moneda en la que está representado ese monto, agrega a la cuenta la cantidad de pesos equivalente a lo indicado para depositar. En este procedimiento hay que aplicar la conversión indicada para el precio de venta.

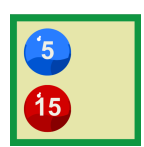
Ej.

		
Celda inicial	Depositar_EnMoneda_ComoPesos(2, Rojo)	Depositar_EnMoneda_ComoPesos(5, Verde)

- b. **ExtraerHasta_EnMoneda_(cantidadAExtraer, moneda)**, que dada una cantidad de dinero a extraer y un color que representa la moneda en la que se va a extraer, remueve de la cuenta la cantidad que se indica. Si no hubiera tanto dinero como el solicitado, se extrae todo lo que haya.

		
Celda inicial	ExtraerHasta_EnMoneda_(5, Rojo)	ExtraerHasta_EnMoneda_(10, Azul)

- c. **ConvertirHasta_PesosA_(pesosAConvertir, moneda)**, que dada una cantidad de pesos a convertir y un color que representa la moneda en la cual se quiere convertir, remueve los pesos de la cuenta y agrega la moneda solicitada. Si en la cuenta hubiera menos pesos de lo solicitado, se convierte todo lo que haya.

		
Celda inicial	ConvertirHasta_PesosA_(68, Rojo)	ConvertirHasta_PesosA_(100, Verde)

El último ejemplo es interesante: se piden convertir 100 pesos a dólares pero no hay 10 pesos en la cuenta, por lo que se va a intentar convertir el total de pesos que haya, 90. Con 90 pesos, no se llega a comprar ningún dólar, y como Gobstones solo trabaja con números enteros, no es posible tener medio dólar, por lo que queda en cero dólares.

- d. **RealizarCorridaCambiaria()**, que dado un tablero de 1 única fila y 10 columnas, donde cada celda representa una cuenta bancaria, se realiza una corrida cambiaria, donde en cada cuenta se cambia la totalidad de los pesos a dólares.

