

Capítulo 5

Repaso



Objetivos

Hasta ahora se ha definido la manera de escribir programas utilizando el lenguaje del robot. También se ha presentado la sintaxis utilizada que permite trasladar al robot, recoger y/o depositar flores y papeles y saber si hay o no flores en la esquina o en la bolsa.

Por otro lado se han analizado diferentes situaciones que requieren la posibilidad de representar información específica del problema.

El objetivo de este capítulo es presentar, analizar y resolver diferentes ejemplos que permitirán la ejercitación de los temas vistos en los capítulos anteriores.



Temas a tratar

- ✓ Presentación, análisis y resolución de ejemplos.
- ✓ Conclusiones.
- ✓ Ejercitación.

5.1 Repaso de variables

En los capítulos anteriores se ha definido la sintaxis de las acciones u órdenes que el robot puede llevar a cabo y se ha indicado como se representará y trabajará con la información relevante que presenta el problema a resolver utilizando el lenguaje del ambiente del robot Rinfo.

Como ya hemos visto:

En general, durante la ejecución de un programa es necesario manipular información que puede cambiar continuamente. Por este motivo, es necesario contar con un recurso que permita variar la información que se maneja en cada momento. Este recurso es lo que se conoce como variable.

Además, sabemos que dentro de un mismo programa pueden utilizarse tantas variables como sean necesarias para representar adecuadamente todos los datos presentes en el problema.

Sin embargo, de todos los ejemplos vistos en los capítulos 2 y 3 podríamos pensar que cada vez que un enunciado requiere informar una cantidad, es necesario recurrir a una variable. A través de un ejemplo, podemos observar que esto no siempre es así.

Analicemos el siguiente ejemplo:

Ejemplo 5.1: Programe al robot para que recorra la calle 45 deteniéndose cuando encuentre una esquina que no tiene flores, sabiendo que esa esquina seguro existe. Al terminar debe informar la cantidad de pasos dados.

Este problema admite dos soluciones. Una de ellas utiliza una variable para representar la cantidad de pasos que da el robot y la otra no.

| | |
|---|--|
| <pre> programa cap5Ejemplo1a areas ciudad: AreaC(1,1,100,100) robots robot robot1 comenzar Pos(1,45) derecha mientras (HayFlorEnLaEsquina) mover Informar (PosAv-1) fin variables Rinfo: robot1 comenzar AsignarArea(Rinfo,ciudad) Iniciar(Rinfo,1,1) fin </pre> | <pre> programa cap5Ejemplo1b areas ciudad: AreaC(1,1,100,100) robots robot robot1 variables pasos: numero comenzar Pos(1,45) derecha pasos:= 0 mientras (HayFlorEnLaEsquina) mover pasos:= pasos + 1 Informar (pasos) fin variables Rinfo: robot1 comenzar AsignarArea(Rinfo,ciudad) Iniciar(Rinfo,1,1) fin </pre> |
|---|--|

El ejemplo 5.1 muestra que antes de decidir representar nueva información dentro del programa, es importante analizar si el robot no cuenta con la posibilidad de manejar los datos pedidos y de este modo evitar la declaración de un dato.

Analicemos:



¿Por qué en el programa Cap5Ejemplo1a se informa (PosAv-1)?

¿Qué ocurriría en Cap5Ejemplo1b si la línea (1) es reemplazada por pasos:= 1 y la línea (2) es reemplazada por Informar (pasos –1) ?

¿Cuál de las formas de resolver el problema le parece más adecuada? Justificar la respuesta.

5.2 Repaso de expresiones lógicas

Recordemos que las expresiones lógicas pueden formarse con variables y expresiones relacionales utilizando los operadores lógicos de la tabla 2.3.

Analicemos los siguientes ejemplos para ejercitar la resolución de expresiones lógicas que combinan varias proposiciones:

Ejemplo 5.2: Programe al robot para que informe si en la esquina (7,4) hay sólo flor o sólo papel (pero no ambos).

```
programa cap5Ejemplo2
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
comenzar
  Pos(7,4)
  si ((HayFlorEnLaEsquina & ~ HayPapelEnLaEsquina) |      (1)
      (~ HayFlorEnLaEsquina & HayPapelEnLaEsquina))      (2)
    Informar(V)
  sino
    Informar(F)
fin
variables
  Rinfo: robot1
comenzar
  AsignarArea(Rinfo,ciudad)
  Iniciar(Rinfo,1,1)
fin
```

Como puede verse en este ejemplo, en la selección se ha utilizado una disyunción de conjunciones. Es decir que basta con que una de las dos conjunciones sea verdadera para que toda la proposición lo sea.

Cada una de las conjunciones requiere que haya uno sólo de los dos elementos: la primera pide que haya flor y no papel (1) y la segunda que haya papel y no flor (2). Obviamente, no pueden ser verdaderas al mismo tiempo. Pero basta con que sólo una de ellas lo sea para que se informe V.

Ejemplo 5.3: Programe al robot para que recorra la avenida 16 buscando una flor que puede no existir. Al finalizar informar donde está (si la encontró) o F (falso) en caso contrario.

```
programa cap5Ejemplo3
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
  comenzar
    Pos(16,1)
    {recorre la Av.16 buscando la flor }
    mientras ~HayFlorEnLaEsquina & (PosCa < 100)      (1)
      mover
      {ver si encontró la flor o no }
      si HayFlorEnLaEsquina                             (2)
        Informar(PosCa)
      sino
        Informar(F)
    fin
  variables
    Rinfo: robot1
  comenzar
    AsignarArea(Rinfo,ciudad)
    Iniciar(Rinfo,1,1)
  fin
```

Como podemos observar, en la línea (1) se utiliza una proposición molecular para controlar la iteración. Ahora no alcanza con verificar solamente que la flor no exista (~HayFlorEnLaEsquina) sino que además es necesario tener en cuenta que no se termine la avenida (PosCa < 100).

Dado que ambas condiciones deben cumplirse simultáneamente, se las ha unido por una conjunción. Esta proposición molecular será verdadera cuando ambas proposiciones lo sean. La iteración puede leerse como: "mientras no encuentre la flor y a la vez, el robot no llegue a la calle 100, debe seguir avanzando".

La iteración termina cuando la conjunción es falsa. Esto ocurre por tres motivos:

1. Encontró la flor durante el recorrido de la avenida. Es decir que la condición (PosCa < 100) es verdadera pero la proposición (~HayFlorEnLaEsquina) es falsa.

2. No encontró la flor pero llegó a la calle 100. Es decir que (\sim HayFlorEnLaEsquina) es verdadera y ($\text{PosCa} < 100$) es falsa.
3. Encontró la flor sobre la calle 100. En este caso ambas condiciones son falsas.

Por lo tanto, la iteración no necesariamente termina cuando la flor ha sido hallada y para poder informar lo solicitado en el enunciado del problema, será necesario distinguir lo que pasó. Esa es la función de la selección que aparece en la línea (2).

Analicemos:



¿Se logra el mismo resultado reemplazando la condición que aparece en la línea (2) por $\text{PosCa}=100$?. Justificar la respuesta.

Pensar en otra proposición que permita dar a la selección de (2) el mismo funcionamiento.

5.3 Ejemplos

Habiendo repasado los aspectos más importantes para la ejercitación propuesta para este capítulo, a continuación se presentan diferentes ejemplos que combinan los temas vistos hasta aquí. Es recomendable que prestemos especial atención a la definición y evaluación de proposiciones.

Ejemplo 5.4: Programe al robot para que recorra la calle 29 hasta encontrar una esquina vacía que puede no existir. En caso de encontrarla depositar en ella una flor. Si no pudo depositar (porque no tenía) informar F (falso).

El siguiente programa resuelve este problema:

```
programa cap5Ejemplo4
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
  comenzar
    {ubicar el robot al comienzo de la calle 29}
    Pos(1,29)
    derecha
    {recorrer la calle hasta encontrar una esquina vacía o hasta terminar}
  mientras (HayFlorEnLaEsquina | HayPapelEnLaEsquina) & (PosAv < 100)
    mover
    {si la encontró depositar en ella una flor}
  si ~HayFlorEnLaEsquina & ~HayPapelEnLaEsquina
    si HayFlorEnLaBolsa
      depositarFlor
    sino
      Informar(F)
  fin
```

variables

Rinfo: robot1

comenzar

AsignarArea(Rinfo, ciudad)

Iniciar(Rinfo, 1, 1)

fin

Analicemos:



¿Cuales son los casos en los que la evaluación de la proposición molecular que maneja la iteración da como resultado falso?

¿Puede reemplazarse la selección anterior por la siguiente?:

```
si ~HayFlorEnLaEsquina & ~HayPapelEnLaEsquina & HayFlorEnLaBolsa
    depositarFlor
sino
    Informar(F)
```

Ejemplo 5.5: Programe al robot para que informe la cantidad de papeles que hay en la esquina (67,23) SIN modificar el contenido de la esquina.

Este problema es una variante del ejemplo 3.5, donde no se pide que se recojan los papeles sino sólo que informe la cantidad. Sabemos que para poder resolver esto será necesario juntar los papeles contando y luego depositar exactamente la cantidad de papeles recogidos. Notemos que no es lo mismo vaciar los papeles de la bolsa porque ella podría contener papeles ANTES de comenzar a recoger. El programa es el siguiente:

```
programa cap5Ejemplo5
areas
    ciudad: AreaC(1,1,100,100)
robots
    robot robot1
variables
    cantP: numero
comenzar
    Pos(67,23)
    {Indicar que aun no se ha recogido nada}
    cantP := 0
    mientras HayPapelEnLaEsquina
        tomarPapel
        cantP := cantP + 1
    {Ahora la esquina ya no tiene papeles}
    Informar(cantP)
    {Volver a dejar los papeles en la esquina}
    repetir cantP
        depositarPapel
fin
```

```
variables
  Rinfo: robot1
comenzar
  AsignarArea(Rinfo,ciudad)
  Iniciar(Rinfo,1,1)
fin
```

Se ha utilizado una repetición para volver a poner los papeles en la esquina porque, luego de haberlos recogido, se conoce exactamente la cantidad de papeles que se quiere depositar. Además, para depositar no es necesario preguntar si hay papeles en la bolsa para satisfacer esta demanda, porque se ha recogido la misma cantidad de papeles a través de la iteración. Es más, suponiendo que originalmente no hubiera habido papeles en (67,23), *cantP* valdrá cero en cuyo caso el repetir no ejecutará ninguna instrucción.



- ¿Tendría el mismo efecto el código si se corre la instrucción Informar(*cantP*) luego del repetir que deposita papeles?

Ejemplo 5.6: Programe al robot para que junte las flores de la avenida 1 e informe la cantidad de flores que hay en cada una de las esquinas.

Para resolver este problema alcanzará con una única variable que represente la cantidad de flores de la esquina actual. Cada vez que llega a una esquina, el robot inicializará la variable en cero, anotará en ella cada vez que logre recoger una flor y finalmente informará su valor. Esto se debe repetir para todas las esquinas de la avenida 1. El programa será el siguiente:

```
programa cap5Ejemplo6
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  flores: numero
comenzar
  {Se recorrerán las primeras 99 esquinas}
  repetir 99
    {Indicar que aun no se ha recogido nada en esta esquina}
    flores := 0 (1)
    mientras HayFlorEnLaEsquina
      tomarFlor
      flores := flores + 1
      {Ahora la esquina ya no tiene flores}
      Informar(flores)
      {Pasar a la esquina siguiente}
      mover
    {Falta la esquina (1,100)} (2)
    {Indicar que aún no se ha recogido nada en esta esquina}
    flores := 0
    mientras HayFlorEnLaEsquina
      tomarFlor
      flores := flores + 1
      {Ahora la esquina ya no tiene flores}
```

```
        Informar(flores)
    fin
variables
    Rinfo: robot1
comenzar
    AsignarArea(Rinfo, ciudad)
    Iniciar(Rinfo, 1, 1)
fin
```

Analicemos:



- ¿Qué ocurriría si la línea (1) fuera trasladada *antes* del repetir, es decir antes de comenzar la repetición? ¿Qué valores informaría?
- ¿Por qué es necesario procesar por separado la esquina (1,100)? Vea que aparece fuera de la repetición en la línea (2).
- ¿Cómo modificaría el programa para que el robot *también* pueda informar la cantidad de flores total que había en la avenida?

5.4 Conclusiones

Se han presentado varios ejemplos que muestran el uso de los dos tipos de datos que puede manejar el robot: valores numéricos y valores booleanos. A través de ellos se ha mostrado la forma de mejorar la potencia de las soluciones ofrecidas, permitiendo que el robot registre valores para un procesamiento posterior.

También se ha definido y ejemplificado el uso de los conectivos lógicos permitiendo manejar las estructuras de control selección e iteración a través de proposiciones moleculares.



Ejercitación

1. Escribir un programa que permita al robot recorrer la calle 50 limpiando de flores y papeles cada esquina del recorrido. Al finalizar el recorrido, informar la cantidad de esquinas que estaban *originalmente* vacías.
2. Escribir un programa que le permita al robot recorrer la avenida 30 dejando en las calles pares sólo una flor y en las impares sólo un papel. Suponga que el robot cuenta con suficiente cantidad de flores y papeles en su bolsa. Al finalizar el recorrido, informar la cantidad de esquinas que *originalmente* tenían *flores y papeles*.
3. Programar al robot para que recorra la calle 3 depositando un papel en cada esquina. Si durante el recorrido se queda sin papeles para depositar, debe detenerse e informar la cantidad de esquinas en las que pudo depositar.
4. Programar al robot para que recorra la calle 10 e informe cuántas esquinas tienen sólo flores y cuántas esquinas tienen sólo papeles. *No debe modificarse la cantidad de flores y papeles de cada esquina.*
5. Programar al robot para que recorra el perímetro de la ciudad, juntando todas las flores y papeles del recorrido y depositándolos en los extremos de la ciudad. Esto es, todas las flores y papeles de la avenida 1 debe depositarlos en (1,100), todas las flores y papeles de la calle 100 debe depositarlos en (100,100) y así siguiendo. Al llegar a cada extremo, debe informar verdadero si la cantidad de flores recogidas es mayor que la cantidad de papeles.
6. Programar al robot para que recorra el perímetro de la ciudad buscando una esquina con exactamente 3 flores y 3 papeles, suponiendo que esta esquina existe. Debe informar cuál es la esquina encontrada.
7. Idem (6) pero no se puede asegurar que tal esquina existe. En caso de encontrarla, informar cuál es esa esquina.
8. Programar al robot para que recorra la ciudad por avenidas y junte lo necesario para que:
 - En las esquinas que sólo tienen papeles, quede *únicamente* un papel.
 - En las esquinas que sólo tienen flores, quede *únicamente* una flor.
 - En las esquinas que tienen flores y papeles, quede *una* flor y *un* papel.

Las esquinas vacías, deben permanecer sin cambios. Al finalizar el recorrido, informe el total de flores y el total de papeles que juntó.
9. Programar al robot para que recorra la ciudad hasta encontrar una esquina vacía (seguro existe). Al encontrarla debe detenerse e informar la cantidad de pasos dados.
10. Idem (9) pero teniendo en cuenta que la esquina puede no existir. Al finalizar el recorrido, debe informar V si encontró la esquina y F en caso contrario.



11. Cuál de estos números es menor:

a. $A51E_{16}$

b. 10111010_2

c. 0101011101_2

d. $1CF_{16}$

12. Cuál operación da mayor resultado:

a. $A5EE_{16} + BA_{16}$

b. $01111010_2 + 1A6_{16}$

c. $011101110001_2 + 56_{10}$

13. Programar el robot para que recorra la calle 4 hasta encontrar una esquina con el doble de flores que papeles. Esta esquina podría no existir y la cantidad de flores y papeles de las esquinas no debe modificarse.