

Capítulo 3

Datos



Objetivos

Los problemas resueltos anteriormente sólo buscan representar un conjunto de acciones a tomar dependiendo de la situación actual. Este es el objetivo de las estructuras de control. A través de ellas puede decidirse, durante el algoritmo, cuál es el rumbo de acción a seguir.

Sin embargo, existen situaciones en las que es preciso representar información adicional específica del problema a resolver. Por ejemplo, podría resultar de interés saber la cantidad de veces que se apagó el horno mientras se estaba horneando una pizza o la cantidad de pasos realizados por el robot hasta encontrar una flor.

El objetivo de este capítulo es incorporar las herramientas necesarias para lograr representar y utilizar esta información.



Temas a tratar

- ✓ Conceptos de Control y Datos.
- ✓ Representación de los Datos.
- ✓ Variables
 - Sintaxis para la declaración de variables.
- ✓ Tipos de datos.
 - Tipo de dato numérico (numero).
 - Tipo de dato lógico (boolean).
- ✓ Esquema de un Programa en el ambiente para el robot R-info.
- ✓ Modificación de la información representada.
- ✓ Ejemplos.
- ✓ Comparaciones.
- ✓ Representación de más de un dato dentro del programa.
- ✓ Conclusiones.
- ✓ Ejercitación.

3.1 Conceptos de Control y Datos

Hasta ahora se ha insistido en las instrucciones que constituyen un algoritmo.

El orden de lectura de dichas instrucciones, en algoritmos como los del capítulo anterior, constituye el **control** del algoritmo. Normalmente la lectura del control de un algoritmo, que también puede representarse gráficamente, indica el orden en que se irán ejecutando las instrucciones y como consecuencia de ello, qué es lo que ese algoritmo hace.

Retomemos el ejemplo del capítulo 2, ejercicio 9.

Ejemplo 2.9: Escriba un programa que le permita al robot recorrer la avenida 7 hasta encontrar una esquina que no tiene flores. Al finalizar debe informar en qué calle quedó parado. Por simplicidad, suponga que esta esquina seguro existe.

```
programa Cap2Ejemplo9
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
comenzar
  Pos(7,1)
  mientras (HayFlorEnLaEsquina)
    mover
  Informar(PosCa )
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

Si se quisiera saber cuántas cuadras recorrió R-info, se necesitaría considerar dicha cantidad como un dato.

Un dato es un elemento u objeto de la realidad que los algoritmos representan y son capaces de modificar y procesar.

Cuando se resuelven problemas con computadora, muchas veces se modelizan los objetos reales mediante objetos más abstractos, representables y entendibles sobre una computadora.

Es importante entender que el mecanismo de resolución de problemas involucra generalmente una transformación o un procesamiento de los datos, manejado por el control del algoritmo.

3.2 Representación de los Datos

Como se dijo anteriormente, los algoritmos modifican objetos de la realidad. La representación de dichos objetos estará dada por sus características principales o por la información que nos interese conocer de ellos.

En el caso del último ejemplo visto, sólo nos concentramos en representar las cuadras recorridas por el robot R-info, y dejamos de lado la cantidad de flores y papeles de cada esquina visitada. Es decir, los datos a representar son aquellos de interés específico para resolver ese problema.

Si además de contar la cantidad de cuadras recorridas, necesitaríamos contar la cantidad de flores en todas las esquinas visitadas, sería necesario representar esta información dentro del algoritmo.

Por lo tanto, se comenzará extendiendo la notación empleada en el algoritmo para dar lugar a las declaraciones de los datos que resultan relevantes al problema.

3.3 Variables

Además de su capacidad de movimiento y de recoger o depositar flores y papeles, el robot posee la habilidad de manejar datos que le permiten representar ciertos atributos de los problemas que debe resolver. Por ejemplo, es capaz de calcular la cantidad de flores que lleva en su bolsa o recordar si en una esquina dada había más flores que papeles.

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

Una variable permite almacenar un valor que puede ser modificado a lo largo de la ejecución del programa. Dicho valor representa un dato relevante para el problema a resolver.

3.3.1 Sintaxis para la declaración de variables

Se denominan identificadores a los nombres descriptivos que se asocian a los datos (variables) para representar, dentro del programa (código del robot), su valor.

Como se dijo en los comienzos del capítulo 2, en el ambiente lenguaje del robot R-info, existe un área o lugar dentro de cada robot (en nuestro curso R-info) donde se declaran las variables que se puedan necesitar para resolver los diferentes programas.

Un punto importante a considerar es que en este ambiente no existe la posibilidad de declarar variables en el programa (exceptuando la declaración del robot R-info), por el contrario sólo se permite declarar variables dentro del código del robot.

Para declarar una variable dentro del ambiente del robot R-info se debe elegir un nombre que la identifique y un tipo de datos al cual pertenece dicha variable. Los tipos existentes en este ambiente se explicarán más adelante. La sintaxis dentro del robot R-info para declarar una variable es la siguiente:

```

robots
  robot robot1
  variables
    nombreVariable: tipoVariable
  comenzar
    ...
fin

```

Variable que puede ser utilizada mientras se ejecuta el código del robot.

Teniendo en cuenta esto, el programa completo quedaría de la forma:

```

programa Cap3EjemploVariables
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
  variables
    nombreVariable: tipoVariable
  comenzar
    ...
  fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin

```

Notemos que se ha incorporado una sección para la declaración de las variables dentro del código del robot. Esta sección se encuentra entre la palabra robot y la palabra comenzar.

Ahora quedan distinguidos dentro del robot dos sectores: un sector superior donde se declaran las variables de interés y un sector inferior donde se detallan las instrucciones necesarias para que el robot R-info resuelva el problema planteado.

Si en el ejemplo antes visto se quisiera contar la cantidad de cuadras recorridas hasta encontrar una esquina sin flor, entonces deberíamos modificar el código del robot R-info agregando en la zona de declaración de variables un dato que permita manejar y procesar esa información.

Tener en cuenta que la declaración de variables que se presenta a continuación en la línea indicada con (*) es aún incompleta dado que nos quedan por ver algunos temas adicionales antes de escribirla correctamente.

Veamos cómo quedaría:

```
programa Cap2Ejemplo9
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  cantidadCuadras: (*)
comenzar
  ...
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

3.4 Tipos de datos

Independientemente del lenguaje de programación que se utilice, la definición de un tipo de dato implica considerar tres aspectos fundamentales que lo caracterizan:

- Identificar cuáles son los valores que puede tomar un dato.
- Definir cuáles son las operaciones que pueden aplicarse sobre los datos de este tipo.
- Indicar cuáles son las relaciones de orden que permiten compararlos.

Se utilizará este mecanismo para definir los dos tipos de datos con los que trabaja en el ambiente de programación del robot R-info.

3.4.1 Tipo de dato numérico (número)

Los elementos de este tipo de dato son los números enteros, como por ejemplo:

-3, -2, -1, 0, 1, 2, 3,

Una computadora sólo puede representar un subconjunto finito de valores, con lo cual existirá un número máximo y un número mínimo. Dichos valores estarán determinados por la cantidad de memoria que se utilice para representarlo. En el ambiente de programación del robot R-info, el número máximo que se puede representar es $2^{31} = 2147483647$ y el mínimo es $-2^{31} = -2147483647$.

Las operaciones válidas para los números son: suma, resta, multiplicación y división entera. Estas operaciones en el ambiente de programación del robot R-info se simbolizan como: +, -, *, / respectivamente.

Si el resultado de alguna operación sobrepasara el límite superior permitido para los elementos del tipo numero, se dice que ocurre un overflowy si se encuentra por debajo del mínimo se dice que ocurre underflow.

Algunos lenguajes detectan el overflow y el underflow como error, otros lo ignoran convirtiendo el valor resultado en un valor válido dentro del rango.

Las relaciones de orden entre números son: igual, menor, mayor, menor o igual, mayor o igual y distinto. En la tabla 3.1 se indica la sintaxis de cada uno de ellos así como un ejemplo de aplicación.

Relación	Sintaxis	Ejemplo
Igualdad	=	$A = B$
Menor	<	$3 * C < D$
Mayor	>	$C > 2 * B$
Menor o igual	<=	$A < (2 * C + 4)$
Mayor o igual	>=	$(B + C) > D$
Distinto	<>	$A <> B$

Tabla 3.1: Relaciones de Orden para números

Tener en cuenta que en la tabla 3.1 A, B, C y D son variables numéricas.

Para declarar una variable numérica deberá utilizarse el sector de declaraciones dentro del robot.

En el ejemplo de contar las cuadras (programa cap2Ejemplo9), para completar la declaración del dato cantidadCuadras que nos había quedado pendiente, ahora realizamos lo siguiente:

```

programa Cap2Ejemplo9
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
  variables
    cantidadCuadras: numero
  comenzar
    ...
  fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin

```

Donde numero indica en el ambiente de programación del robot que la variable será de tipo numérico, y esto implica que sus valores entrarán en el rango especificado

anteriormente y que podrá utilizar cualquiera de las operaciones válidas para el tipo número.

Dado que el robot es capaz de realizar cálculos con sus variables numéricas, analicemos otro ejemplo donde se declaran dos variables numéricas, prestando especial atención a las líneas numeradas:

```

programa numerico
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  nro1: numero
  nro2: numero
comenzar
  nro1 := 23                                {1}
  nro2 := 30                                {2}
  Informar ( nro1 * nro2 )                  {3}
  Informar ( 25 / 3 )                       {4}
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin

```

La línea marcada con (1) utiliza el operador de asignación que veremos en una sección posterior. El operador de asignación en el ambiente de programación del robot R-info se representa con un dos puntos igual (:=) y permite dar valor a una variable. En el caso de (1) le está dando el valor 23, y en el caso de (2) le está dando el valor 30. En la línea marcada con (3) se abrirá en pantalla una ventana mostrando el valor 690 (resultante de la instrucción Informar) y la línea (2) visualizará el valor 8. Notemos que la división es entera por lo que la parte fraccionaria, sin importar su valor, será ignorada.

En la mayoría de los programas que se resolverán a lo largo de este curso, el uso de las variables numéricas se verá restringido a operaciones sobre números enteros positivos como por ejemplo: informar cantidad de pasos dados, cantidad de flores recogidas, cantidad de veces que ocurrió un determinado hecho, etc.

3.4.2 Tipo de dato lógico (boolean)

Este tipo de dato lógico puede tomar uno de los dos siguientes valores: Verdadero o Falso. En el ambiente de programación del robot R-info están denotados por V y F. Si

se utilizan variables de tipo booleano se puede asignar en ellas el resultado de cualquier expresión lógica o relacional.

En el ejemplo que se presenta a continuación, se utiliza una variable *identicos* que guardará el valor V (verdadero) si el robot se encuentra ubicado en una esquina en la cual el valor de la avenida coincide con el valor de la calle y guardará F (falso) para cualquier otra situación. Del mismo modo, la variable *esPositivo* guardará el valor F si está ubicado sobre la calle 1 y en cualquier otro caso, guardará V.

Analicemos la solución presentada para este ejemplo:

```
programa numvariablesLogicas
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  esPositivo: boolean
  identicos: boolean
comenzar
  identicos := (PosCa = PosAv)
  esPositivo := (PosCa > 1)
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

Veamos el otro problema que permite ejemplificar el uso del tipo de dato booleano:

Ejemplo 3.1: El robot debe ir de (1,1) a (1,2) y debe informar si en (1,1) hay flor o no.

```
programa cap3Ejemplo1
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  habiaFlor: boolean
comenzar
  habiaFlor := HayFlorEnLaEsquina
  mover
  Informar(habiaFlor)
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

En este caso la variable lógica permite registrar V o F según si en (1,1) hay flor o no. Luego, al posicionar el robot en (1,2) el valor almacenado es informado.

El objetivo de este sencillo ejemplo es mostrar la posibilidad de la variable lógica de guardar el resultado de la evaluación de una condición (en este caso HayFlorEnLaEsquina) para usarlo con posterioridad. Esto también puede aplicarse a cualquier proposición ya sea atómica o molecular.

Analicemos el siguiente ejemplo:

Ejemplo 3.2: Recoger todas las flores de la esquina (11,1). Si logra recoger al menos una flor, ir a (10,10) y vaciar de flores la bolsa; sino informar que no se ha recogido ninguna flor en (11,1).

En este caso no se requiere conocer la cantidad de flores recogidas. Sólo se necesita saber si se ha recogido alguna flor en (11,1) o no. Por lo tanto, en lugar de utilizar una variable numérica, se utilizará una variable booleana para representar esta situación. Realizaremos un esquema del algoritmo que se deberá incluir en el robot R-info:

```
programa Cap3Ejemplo2
comenzar
{antesde empezar analiza y recuerda si en (11,1) hay flor o no}
{tomar todas las flores de (11,1)}
si (originalmente en (11,1) había flor)
  {ir a (10,10) y vaciar la bolsa}
sino
  {informar que no se recogió ninguna flor}
fin
```

En el lenguaje del robot esto se escribe de la siguiente forma:

```
programa cap3Ejemplo2
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  florEn11: boolean
comenzar
  Pos(11,1)
  {antesde empezar analizar y recordar si en (11,1) hay flor o no}
  florEn11 := HayFlorEnLaEsquina
  {tomar todas las flores de (11,1)}
  mientras HayFlorEnLaEsquina
    tomarFlor
  si florEn11
    Pos(10,10) {ir a (10,10) y vaciar la bolsa}
    mientras HayFlorEnLaBolsa
      depositarFlor
  sino
    {informar F, se indica que no se recogió ninguna}
    Informar (florEn11)
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
```

3.5 Modificación de la información representada

Hasta ahora sólo se ha manifestado la necesidad de conocer expresamente cierta información. Para tal fin se ha asociado un nombre, también llamado identificador, a cada dato que se desee representar.

Cada uno de estos identificadores será utilizado como un lugar para almacenar la información correspondiente. Por lo tanto, será necesario contar con la posibilidad de guardar un valor determinado y posteriormente modificarlo.

Se utilizará como lo indicamos en los ejemplos de la sección anterior la siguiente notación:

Identificador := valor a guardar

El operador := se denomina operador de asignación y permite registrar o “guardar” el valor que aparece a derecha del símbolo en el nombre que aparece a izquierda de dicho símbolo.

En el ejemplo Cap2Ejemplo9 al comenzar el recorrido debe indicarse que no se ha caminado ninguna cuadra. Para esto se utilizará la siguiente notación:

cantidadCuadras:= 0

A partir de esta asignación el valor del dato cantidadCuadrasrepresentará (o contendrá) el valor 0.

En ocasiones será necesario recuperar el valor almacenado para ello se utilizará directamente el identificador correspondiente.

Por ejemplo, la instrucción

Informar (cantidadCuadras)

permitirá informar el último valor asignado a cantidadCuadras.

También, puede utilizarse el operador de asignación para modificar el valor de un identificador tomando como base su valor anterior (el que fue almacenado previamente). Por ejemplo:

cantidadCuadras := cantidadCuadras + 1

Como se explicó anteriormente, el operador := permite asignar el valor que aparece a la derecha del símbolo al identificador que está a la izquierda del mismo. Sin embargo, la expresión que ahora aparece a derecha no es un valor constante sino que debe ser evaluado ANTES de realizar la asignación. El lado derecho indica que debe recuperarse el valor almacenado en cantidadCuadras y luego incrementarlo en 1. El resultado de la suma será almacenado nuevamente en cantidadCuadras.

Utilizando lo antes expuesto el ejemplo del Cap2Ejemplo9 se reescribe de la siguiente manera:

programa cap2Ejemplo9B

```

areas
    ciudad: AreaC(1,1,100,100)
robots
    robot robot1
variables
    cantidadCuadras: numero
comenzar
    {Hasta ahora no se camina ninguna cuadra }
    cantidadCuadras:=0 {1}
    mientras (hayFlorEnLaEsquina)
    {Incrementar en 1 la cantidad de cuadras dadas }
        cantidadCuadras:= cantidadCuadras + 1 {2}
        mover
    Informar(cantidadCuadras)
fin
variables
    R-info: robot1
comenzar
    AsignarArea(R-info,ciudad)
    Iniciar(R-info,1,1)
fin

```

Notemos que la instrucción (1) se encuentra fuera de la iteración por lo que se ejecutará una única vez al comienzo del algoritmo. Esta asignación inicial también se denomina **inicialización** del identificador.

La instrucción (2) se ejecuta cada vez que el robot avanza una cuadra. Su efecto es recuperar el último valor de cantidadCuadras, incrementarlo en 1 y volver a guardar este nuevo valor en cantidadCuadras. De esta forma, el último valor almacenado será una unidad mayor que el valor anterior.

Observemos la importancia que tiene para (2) la existencia de (1). La primera vez que (2) se ejecuta, necesita que cantidadCuadras tenga un valor asignado previamente a fin de poder realizar el incremento correctamente. Finalmente, (3) permitirá conocer la cantidad de cuadras recorridas una vez que el robot se detuvo.

3.6 Ejemplos

Ejemplo 3.3: El robot debe recorrer la avenida 1 hasta encontrar una esquina con flor y papel. Al finalizar el recorrido se debe informar la cantidad de cuadras recorridas hasta encontrar dicha esquina. Suponga que la esquina seguro existe.

Para poder resolverlo es necesario identificar los datos u objetos que se desean representar a través del algoritmo.

En este caso interesa conocer la cantidad de cuadras hechas, y por lo tanto, es preciso registrar la cantidad de cuadras que se van avanzando hasta encontrar la esquina buscada. Una solución posible en el ambiente de programación del robot R-info:

```

programa cap3Ejemplo3
areas
    ciudad: AreaC(1,1,100,100)
robots
    robot robot1

```

```

variables
    cuadras: numero

comenzar
    cuadras:=0 {1}
    mientras ~(HayFlorEnLaEsquina) | ~(HayPapelEnLaEsquina) {2}
        {anotar que se caminó una cuadra mas}
        cuadras:=cuadras+1 {3}
        mover
    Informar (cuadras) {4}

fin

variables
    R-info: robot1

comenzar
    AsignarArea (R-info, ciudad)
    Iniciar (R-info, 1, 1)

fin

```

En (1) indicamos que no se ha recorrido ninguna cuadra hasta el momento (por eso se le asigna el valor cero a la variable cuadras). En (2), se indica que mientras no haya flor ó no haya papel en la esquina, el recorrido debe continuar, esto es, la iteración terminará cuando encuentre una esquina con flor y papel. En (3) indicamos que se ha recorrido una cuadra más. En (4) informamos el último valor que quedó almacenado en cuadras, lo cual representa la cantidad de cuadras hechas en el recorrido.



¿Cómo modifico el algoritmo anterior si la esquina puede no existir?

Ejemplo 3.4: Recoger e informar la cantidad de flores de la esquina (1,1).

Para poder resolverlo es necesario identificar los datos u objetos que se desean representar en el algoritmo.

En este caso interesa conocer la cantidad de flores de la esquina (1,1) y por lo tanto es preciso registrar la cantidad de flores que se van tomando hasta que la esquina queda sin flores. El algoritmo tendría la siguiente forma:

```

programa cap3Ejemplo4
areas
    ciudad: AreaC(1,1,100,100)
robots
    robot robot1
variables
    flores: numero
comenzar
    flores:=0
    mientras (HayFlorEnLaEsquina)
        tomarFlor {registramos que se tomó una flor}
        flores:= flores+1
    Informar(flores)
fin

```

```
variables
  R-info: robot1
comenzar
  AsignarArea(R-info, ciudad)
  Iniciar(R-info, 1, 1)
fin
```

En (1) indicamos que no se ha recogido ninguna flor hasta el momento. En (2) indicamos que se ha recogido una nueva flor de la esquina. En (3) informamos el último valor que quedó almacenado en flores. Hay que recordar que el punto (2) y la instrucción tomarFlor están dentro de una estructura de control de iteración (mientras) y esto se repetirá hasta que no haya más flores en la esquina, con lo cual en cada vuelta de esta iteración se incrementará en uno el valor de la variable flores.

Ejemplo 3.5: Contar e informar la cantidad de flores de la esquina (1,1).

En este caso sólo nos interesa saber la cantidad de flores de la esquina, a diferencia del ejercicio anterior en donde debíamos además recoger las flores de la esquina (es decir dejar la esquina sin flores). Por lo tanto, se debe tener en cuenta que para poder contar las flores vamos a tener que tomarlas, y una vez que tomamos todas las flores de la esquina, debemos volver a depositarlas para que no se modifique la cantidad original de flores de la esquina.

Inicialmente el algoritmo implementado en el ambiente del robot R-info tendría la siguiente forma:

```
programa cap3Ejemplo5
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  flores: numero
comenzar
  flores:=0
  mientras (HayFlorEnLaEsquina)
    tomarFlor {registramos que se tomó una flor}
    flores:= flores+1
    {Debemos depositar las flores juntadas de la esquina} (1)
  Informar (flores)
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info, ciudad)
  Iniciar(R-info, 1, 1)
fin
```

En (1) debemos encontrar la forma de poder depositar todas las flores que juntamos de la esquina. Para esto podemos ver que si en la esquina, por ejemplo, hubo 5 flores, entonces la variable flores tiene el valor 5, por lo tanto la variable flores contiene la cantidad de flores que debo depositar en la esquina.

Resumiendo, sabemos cuántas flores hay que depositar en la esquina, esta cantidad es la que ha quedado almacenada en la variable flores. El programa se reescribirá de la siguiente manera:

```
programa cap3Ejemplo4B
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  flores: numero
comenzar
  flores:=0 {1}
  mientras (HayFlorEnLaEsquina)
    tomarFlor {registramos que se tomó una flor}
    flores:= flores+1 {2}
    {depositamos las flores juntadas de la esquina}
  repetir flores {3}
    depositarFlor
    Informar (flores) {4}
fin

variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

En (1) indicamos que no se ha recogido ninguna flor hasta el momento. En (2) indicamos que se ha recogido una nueva flor de la esquina.

Para que puedas comprender la instrucción (3), tenemos que tener en cuenta dos cuestiones. La primera es que en (3) se utiliza una estructura de control repetitiva para depositar la cantidad de flores. Esto se puede hacer ya que conocemos el valor que queremos usar en la estructura repetir, ese valor será el almacenado en flores. La segunda es que esta instrucción repetitiva debe ubicarse fuera del mientras, ya que de lo contrario, por cada flor que recoge el robot volvería a depositar y por lo tanto la estructura de control mientras nunca terminaría. Otro efecto negativo de poner el repetir dentro del mientras es que la variable flores quedaría finalmente con un valor incorrecto. En (4) informamos el valor que contiene la variable flores. En este punto debemos prestar atención que haber utilizado la variable flores en el repetir no implica que la misma se haya modificado, esto es así, porque la única forma de modificar el contenido de una variable es por medio del operador:=.



- ¿Puede ocurrir que el valor de flores permanezca en cero?
- ¿Si el valor de flores es cero, que ocurre con el repetir?
- ¿Se puede reemplazar la estructura de repetición “repetir flores”, por “mientrasHayFlorEnLaBolsadepositarFlor”?

Ejemplo 3.6: Recoger todos los papeles de la esquina (1,1) e informar la cantidad de papeles recogidos sólo si se han recogido al menos 5 papeles.

En este caso nos interesa saber la cantidad de papeles de la esquina, a diferencia del ejercicio anterior. Una vez que el robot ha juntado todos los papeles, si la cantidad de papeles obtenida es mayor o igual a 5 (al menos 5), debemos informar la cantidad recogida.

En el ambiente de programación del robot R-info lo podríamos escribir de la siguiente manera:

```
programa cap3Ejemplo6
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  papeles: numero
comenzar
  papeles:=0
  mientras (hayPapelEnLaEsquina)
    tomarPapel
    {registramos que se tomo un papel}
    papeles:=papeles+1
  si (papeles >= 5)
    Informar (papeles) {1}
  fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

Debemos tener en cuenta que la instrucción (1) está dentro de una instrucción de selección y por lo tanto sólo se ejecuta si la condición es verdadera, es decir, si la variable papeles quedó en un valor ≥ 5 .

Ejemplo 3.7: Recoger e informar todos los papeles de la avenida 1.

En este caso nos interesa saber la cantidad de papeles de la avenida. Una vez que hemos juntado todos los papeles de la avenida, debemos informar la cantidad recogida. El algoritmo debería tener en cuenta:

- 1) Cuál es la estructura de control para recorrer toda una avenida
- 2) Cómo contar todos los papeles de la esquina dónde se encuentra el robot.
- 3) Cómo contar los papeles de la última esquina, es decir, la (1,100).
- 4) Informar el valor de papeles recogidos en el recorrido.

La solución podría plantearse como sigue:

```

programa cap3Ejemplo7
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  papeles: numero
comenzar
  papeles:=0
  repetir 99                                {1}
    mientras (HayPapelEnLaEsquina)          {2}
      tomarPapel
      {registramos que se tomo un papel}
      papeles:=papeles+1
    mover
    {procesamos la última esquina}
    mientras (HayPapelEnLaEsquina)          {3}
      tomarPapel
      {registramos que se tomo un papel}
      papeles:=papeles+1
    Informar (papeles)                        {4}
  fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin

```

En (1) indicamos la estructura de control para recorrer toda una avenida. Como se sabe la cantidad de cuadras de una avenida completa es 99.

En (2) contamos todos los papeles de la esquina en donde se encuentra parado el robot.

En (3) contamos los papeles de la última esquina, es decir, la (1,100).

En (4) informamos el valor de todos los papeles recogidos en el recorrido, el cual se encuentra almacenado en papeles.



¿Qué modificaría en el algoritmo si se quisiera informar los papeles de cada esquina?

3.7 Representación de más de un dato dentro del algoritmo

En las secciones anteriores se presentó la necesidad de almacenar información adicional para responder a los requerimientos del problema. Es importante reconocer que los algoritmos desarrollados en los capítulos 1 y 2, sólo se referían al funcionamiento de las estructuras de control y no al manejo de los datos.

Contar con la posibilidad de asociar un identificador a un valor es equivalente a poder “registrar” un dato para recuperarlo o modificarlo posteriormente.

Como resumen de todo lo visto hasta el momento se presentará un ejemplo más complejo, repitiendo la explicación del proceso de modificación de la información y las posibles comparaciones a utilizar.

Ejemplo 3.8: El robot debe recoger e informar la cantidad de flores y papeles de la esquina (1,1).

En este ejercicio debemos representar dos datos, la cantidad de flores de la esquina y la cantidad de papeles de la esquina. Para esto, vamos a necesitar dos variables una que nos permita contar la cantidad de flores de la esquina y otra que nos permita contar la cantidad de papeles. El algoritmo quedaría de la siguiente forma:

```
programa cap3Ejemplo8
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  papeles: numero
  flores: numero
comenzar
  flores:=0    {1}
  papeles:=0   {2}
  mientras (HayPapelEnLaEsquina)  {3}
    tomarPapel
    {registramos que se tomó un papel}
    papeles:=papeles+1
  mientras (HayFlorEnLaEsquina)    {4}
    tomarFlor
    {registramos que se tomó una flor}
    flores := flores + 1
  Informar (flores)                  {5}
  Informar (papeles)                 {6}
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

En (1) indicamos que no se ha recogido ningún papel todavía. Lo mismo ocurre con las flores en (2).

En (3) contamos todos los papeles que existen en la esquina. Lo mismo hacemos en (4) pero para las flores. En este punto podemos ver que la cantidad de papeles de la esquina va almacenándose en la variable papeles y la cantidad de flores en la variable flores.

En (5) y (6) se informa el valor de flores y papeles de la esquina respectivamente.

Ejemplo 3.9: Modifique el ejercicio anterior para que el robot evalúe cual de las cantidades (flores y papeles) resultó mayor e informe dicha cantidad.

Siguiendo el razonamiento anterior, luego de recoger y contar las flores y los papeles, deberemos analizar cuál de los dos valores obtenidos es el mayor e informar dicho valor.

El algoritmo quedaría de la siguiente forma:

```
programa cap3Ejemplo9
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  papeles: numero
  flores: numero
comenzar
  papeles:=0
  mientras (HayPapelEnLaEsquina)
    tomarPapel
    {registramos que se tomó un papel}
    papeles:=papeles+1
  mientras (HayFlorEnLaEsquina)
    tomarFlor
    {registramos que se tomó una flor}
    flores := flores + 1
  si (flores>papeles) {1}
    Informar(flores)
  sino
    Informar(papeles)
  fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
fin
```

En (1) se verifica si la cantidad de flores es mayor a la de papeles, por medio de una estructura de selección, en caso que la condición resulte verdadera se informa la cantidad de flores, en cambio, si la condición es falsa se informa la cantidad de papeles.



¿Qué informa si ambas variables contienen el mismo valor?

3.8 Conclusiones

Hasta aquí se ha presentado la sintaxis de un algoritmo que utiliza datos para la solución de un problema. También se ha mostrado cómo crear, inicializar y modificar estos datos en una solución.

Contar con la posibilidad de representar información adicional al problema mejora la potencia de expresión del algoritmo, ya que los atributos de los objetos con los que opera podrán estar reflejados en su interior.

También vimos que para representar esta información el ambiente de programación del robot R-info nos provee dos tipos de datos: el tipo número y el tipo lógico.

Hasta aquí se han presentado los elementos que componen un algoritmo: el control y los datos. De todo lo visto, entonces, podemos concluir que un algoritmo es una secuencia de instrucciones que utiliza y modifica la información disponible con el objetivo de resolver un problema.



Ejercitación

1. Indique que hacen los siguientes programas considerando las diferentes situaciones que podrían presentarse:

- (a) i. todas las esquinas de la avenida 6 tienen al menos 1 flor
- ii. sólo la esquina (6,20) tiene flor.
- iii. ninguna esquina de la avenida 6 tiene flor

```
programa queHace1
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
comenzar
  Pos (6,1)
  mientras ((HayFlorEnLaEsquina) & (PosCa< 100))
    mover
    tomarFlor
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
Fin
```

- (b) i. todas las esquinas de la avenida 1 tienen al menos 1 flor y 1 papel.
- ii. sólo la esquina (6,20) tiene flor y ningún papel, las demás están vacías.
- iii. sólo la esquina (6,20) tiene papel y no tiene ninguna flor, las demás están vacías.
- iv. ninguna esquina de la avenida 1 tiene flor ni papel

```
programa queHace2
areas
  ciudad: AreaC(1,1,100,100)
robots
  robot robot1
variables
  nro : numero
comenzar
  nro := 0
  repetir 10
    si~((HayFlorEnLaEsquina) | (HayPapelEnLaEsquina))
      mover
      nro := nro + 1
  Informar (nro)
fin
variables
  R-info: robot1
comenzar
  AsignarArea(R-info,ciudad)
  Iniciar(R-info,1,1)
Fin
```

2. Programe al robot para que informe la cantidad de flores que hay en la calle 44.
 - I. Recogiendo todas las flores.
 - II. Sin modificar el contenido de cada esquina.
3. Programe al robot para que informe la cantidad de esquinas vacías que hay en la ciudad.
4. Escriba un programa que le permita al robot caminar por la calle 7 hasta encontrar 20 flores. Hay como máximo una flor por esquina. Seguro existen 20 flores.
5. Escriba un programa que le permita al robot caminar por la calle 7 hasta encontrar 20 flores. Hay como máximo una flor por esquina. Pueden no haber 20 flores.
6. Escriba un programa que le permita al robot caminar por la calle 7 hasta encontrar 20 flores. Puede haber más de una flor por esquina. Seguro existen 20 flores.
7. El robot debe limpiar de papeles la calle 34. Al terminar el recorrido debe informar cuantas esquinas tenían originalmente exactamente 6 papeles.
8. Programe al robot para que recorra la calle 2 hasta encontrar al menos 10 papeles. Pueden no haber 10 papeles.
9. Programe al robot para que recorra la calle 2 hasta encontrar 10 papeles y 4 flores. Seguro existen dichas cantidades.
10. Programe al robot para que recorra el perímetro de la ciudad e informe la cantidad de papeles recogidos en cada lado.