

Tutorial de Karel

Autor: Dra. Karina Mariela Figueroa Mora

Introducción

El robot Karel es una herramienta de aprendizaje que presenta los conceptos de programación de una forma visual además de tener un nivel de abstracción bajo. El robot Karel fue introducido por Richard Pattis en 1981, en el libro *"Karel the Robot: A Gentle Introduction to the Art of Programming with Pascal, John Wiley & Sons, Inc."*

El mundo de Karel

Karel puede orientarse en una de las cuatro direcciones: Este, Oeste, Norte y Sur. Sólo gira 90° cada vez. En el mundo de Karel, las calles van de sur a norte, y son numeradas comenzando por 1. No hay números de calle igual a 0 o negativos. Las avenidas van de oeste a este, y también están numeradas empezando por 1. Tampoco hay números de avenida igual a 0 o negativos. Se le llama esquina a la intersección de una calle con una avenida. Karel va de una esquina a la siguiente en un solo movimiento.

En el mundo de Karel sólo existen los muros (entre una esquina y otra, vertical u horizontal), y los zumbadores. Un zumbador es una marca que Karel puede escuchar sólo cuando se encuentra en la misma esquina que el zumbador. Karel tiene una mochila que puede utilizar para poner los zumbadores que va recolectando o dejando. Es posible ajustar el número inicial de zumbadores en la mochila.

Comandos básicos de Karel

Hay cinco comandos básicos para Karel, estos son:

<instrucciones>

1. avanza (avanza una esquina)
2. gira-izquierda (gira a la izquierda)
3. coge-zumbador (recoge un zumbador)
4. deja-zumbador (deja un zumbador)
5. apagate (desconecta al robot)

También se consideran instrucciones a las sentencias de control.

Karel no realizará una acción que no sea exitosa, es decir, el robot no avanzará si tiene una pared enfrente o no recogerá un zumbador si no lo hay en esa esquina. Cuando el robot no realiza una acción envía una notificación al programador y se detiene la ejecución del programa.

```
Iniciar-programa
  Inicia-ejecucion
    <instrucciones>
  apagate;
  termina-ejecucion
finalizar-programa
```

```
Programa Muestra
iniciar-programa
  inicia-ejecucion
    gira-izquierda;
  apagate;
  termina-ejecucion
finalizar-programa
```

Sentencias de Control de KAREL

Las sentencias de control se usan para elegir qué hacer, y/o cuantas veces hacerlo. Sin embargo, por si solos no causan que ocurra algo. Simplemente controlan la ejecución de otras sentencias o fragmentos de código. A continuación se lista una serie de sentencias de control de Karel:

- Si/Entonces
- repite/No de veces
- Si/Entonces/Sino
- Mientras/Hacer

Toma de decisiones

Muchas veces Karel necesita tomar decisiones, de acuerdo a las situaciones en las que se encuentre. Las situaciones que Karel puede detectar se listan a continuación:

<condiciones>

frente-libre
junto-a-zumbador
orientado-al-este
frente-bloqueado
no-junto-a-zumbador
orientado-al-oeste
izquierda-libre
algun-zumbador-en-la-mochila
no-orientado-al-norte
izquierda-bloqueada
ningun-zumbador-en-la-mochila
no-orientado-al-sur
derecha-libre
orientado-al-norte
no-orientado-al-este
derecha-bloqueada
orientado-al-sur
no-orientado-al-oeste

"libre" significa que no hay ningún muro, mientras que "bloqueado" significa que hay un muro en esa dirección. Karel puede detectar si hay o no algún zumbador en la esquina en la que se encuentra actualmente, así como detectar si tiene algún zumbador en la mochila o no. También tiene una brújula para detectar hacia qué dirección está orientado.

Decisiones simples

Karel puede hacer una serie de instrucciones, si es que se cumple con la condición establecida, que puede ser cualquiera de las que se mencionaron anteriormente. Por ejemplo

```
Programa Muestra
iniciar-programa
    inicia-ejecucion
        si junto-a-zumbador entonces inicio
            coge-zumbador;
        fin;
    apagate;
termina-ejecucion
```

finalizar-programa

Este ejemplo escenifica que para recoger un zumbador sin obtener un error de Karel, es necesario revisar si existe un zumbador en el suelo utilizando la condición **junto-a-zumbador**, al ejecutar esta instrucción.

```
Si <condición> entonces
inicio
    <instrucciones>
fin;
```

Decisiones en otro caso

En este tipo de decisiones, Karel, puede hacer una serie de instrucciones si se cumple la condición y si no se cumple, puede llevar a cabo otra serie de instrucciones diferentes.

```
Si <condición> entonces
inicio
    <instrucciones>
fin
Sino
inicio
    <instrucciones>
fin;
```

Definiciones de funciones

En este lenguaje es posible definir nuevas funciones por ejemplo, girar a la derecha es una tarea muy habitual para el robot. A continuación se muestra un ejemplo de esta nueva instrucción. La definición de cualquier función debe ir entre iniciar-programa e inicia-ejecución.

```
define-nueva-instrucción gira-derecha como
inicio
    gira-izquierda;
    gira-izquierda;
    gira-izquierda;
fin;
```

```
Define-nueva-instrucción <nombre> como
Inicio
    <instrucciones>
fin;
```

Repeticiones

Muchas veces, Karel necesita repetir una serie de instrucciones varias veces. A una repetición se le llama ciclo. Si se desea que Karel ejecute la misma instrucción varias veces se puede utilizar la instrucción **repetir x veces**

```
Repetir # veces
Inicio
    <instrucciones>
fin;
```

```
Programa Muestra
iniciar-programa
    inicia-ejecucion
        repetir 3 veces
            gira-izquierda;
        apagate;
    termina-ejecucion
finalizar-programa
```

Repeticiones condicionadas

Si necesitas que Karel repita una serie de instrucciones mientras se cumpla una condición se usa la sentencia mientras.... hacer. Esto es:

```
Mientras <condición> hacer
Inicio
    <instrucciones>
fin;
```