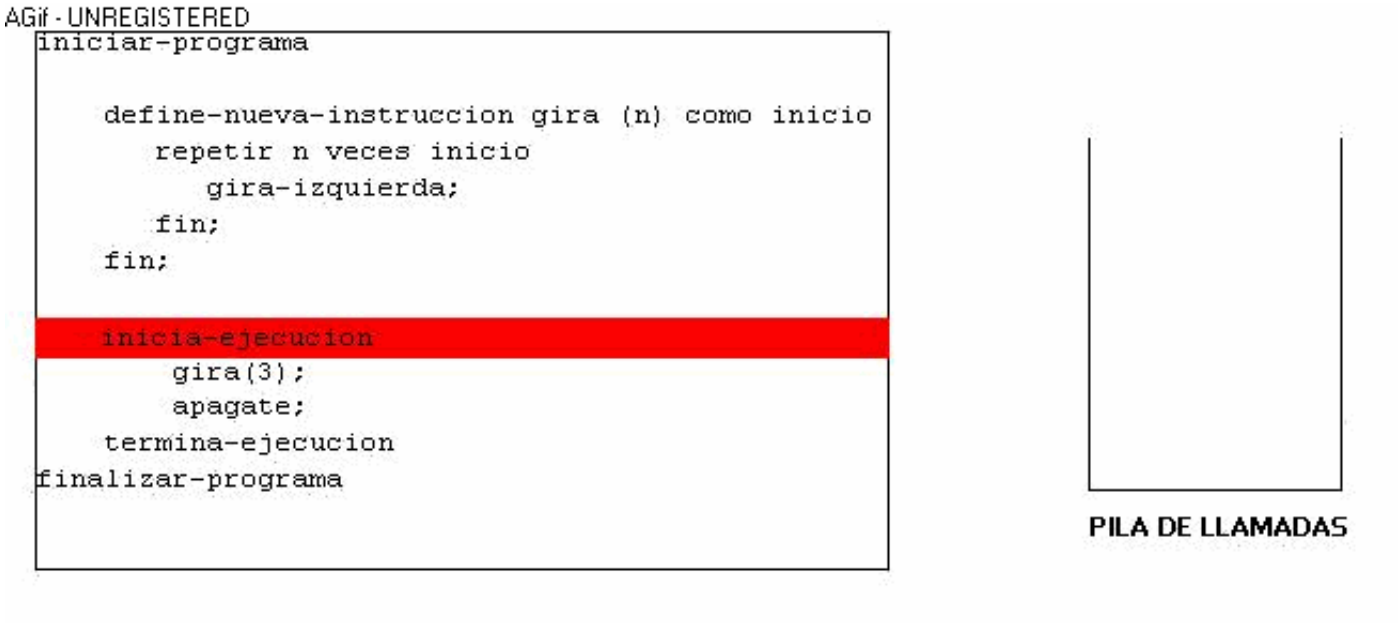
Versión de JAVA

Fundamentos

Empecemos. ¿Te has preguntado alguna vez qué sucede cuándo llamas a una instrucción que tú creaste? Como te habrás dado cuenta, se ejecuta la instrucción que le pediste y luego continua en el lugar en donde se quedó. Pero en realidad, ¿cómo hace eso la computadora? Pues bien, la computadora tiene una cosa que se llama "pila de llamadas", cuando un programa se ejecuta y encuentra una instrucción, se guarda en la pila en que lugar se quedó, entonces ejecuta la instrcción que le pediste y al terminar la instrucción, revisa en la pila en donde estaba anteriormente, para continuar en ese lugar. ¿Muy enredado? Mira este dibujo:



Espero haya quedado claro, ya que esto es el punto crucial de la recursividad. Ahora que sabemos como

funciona el llamado a procedimientos o instrucciones. Definamos lo que es recursividad.

Subir

Definición

Una instrucción recursiva, es aquella que en algún momento, se llama así misma. Por ejemplo, puedes revisar en libros y en muchos otros lados, la definición recursiva del factorial:

factorial(1) = 1

factorial(n) = n * factorial(n - 1)

Como puedes ver, para saber el factorial de un número, necesitas saber el factorial del número anterior. Allí está la recursividad.

Toda instrucción recursiva, consta de dos partes:

1.- Base

> Es un valor fijo al que eventualmente, la recursión debe de llegar, normalmente, es un valor pequeño y no demostrable. Por ejemplo, factorial de 1, es 1, y no se puede demostrar, además, si quieres calcular el factorial de cualquier número, eventualmente, debes de saber el factorial de 1.

2.- Recursión

> Consiste en la parte en donde la instrucción se llama así misma, además de efectuar algunas operaciones con el resultado. Por ejemplo, para el factorial de 3, se calcula el factorial de 2 y eso se multiplica por 3.

Ahora que sabes que es recursividad, intenta hacer la definición recursiva de la serie de Fibonacci, la cual dice que sus primeros dos elementos (1 y 2) son 1, y para cualquier otro número de Fibonacci, se debe sumar el Fibonacci de los dos números anteriores, Por ejemplo: Fibonacci(3) = 2, Fibonacci(5) = 5.

Pasemos de lleno a la recursividad en Karel.

Subir

Recursividad simple

Intentemos ahora llamar a una instrucción desde sí misma, recordemos que debe de tener una base y una definición recursiva, ¿te parece bien que la base sea si el frente esta bloqueado termina?, ¿y te parece bien que si no avance una casilla y entonces se llame asi misma?, ¡Hagámoslo!

...

```
define-nueva-instruccion recursiva como inicio
  si frente-libre entonces inicio
    avanza;
    recursiva;
  fin;
fin;
```

...

Nota que en Karel es imposible decirle "termina-instruccion", así que mejor invertimos la condición y avanzamos si el frente está libre, ¡es otra forma de poner la base!, por que, en otras palabras, la base es lo que hace que la recursión termine y no sea infinita. Nota también que en este caso las operaciones que hacemos son solamente "avanza;". Algo diferente al factorial ¿no?

¿Qué crees que haga esté código? Ponlo en el Simulador Karel y adivina...

¡Así es! Karel avanzará mientras el frente este libre. Entonces te preguntarás ¿no sería mejor un ciclo mientras? Y la respuesta es: si solamente quieres hacer eso, sí, sin embargo, ¿que tal si quieres contar con zumbadores cuantos pasos avanzaste? ¡A verdad! Si queremos hacer eso, debemos agregarle una línea a nuestro código:

...

```
define-nueva-instruccion recursiva como inicio
```

```
  si frente-libre entonces inicio
    avanza;
    recursiva;
    deja-zumbador;
  fin;
fin;
```

...

Ahora pruébalo.

¡Wow! ¡Sorprendente! ¿Por qué funciona?

Si recuerdas la sección de fundamentos, cada vez que se llama una instrucción, la siguiente instrucción se guarda en la pila de llamadas, como se va a llamar exactamente el número de veces que avanzaste, cuando la recursividad termine al llegar a la base, va a sacar una por una las instrucciones que siguen, y las que siguen siempre serán "deja-zumbador". En este ejemplo, solo hay una instrucción antes y después de las llamadas recursivas, pero puede haber muchísimas más.

Si aún no lo entiendes, trata de simular la pila de llamadas de la misma forma que está en la imagen.

Ejercicio 10: ¿Recuerdas el problema del periódico del Ejercicio 2? Ahora imagina que el periódico está en la posición 1,1 y que la casa de Karel, puede estar ubicada en cualquier posición del mundo, eso sí, con la misma orientación y forma. Haz un programa que haga que Karel lleve el periódico a su sala. ¡Usa recursividad! Recuerda, piensa en que momento la recursividad termina (BASE) y si hay que hacer algo en ese momento, luego piensa en la llamada recursiva, y las operaciones que van a ir antes, y las que van a ir después (las que se van a guardar en la pila).

Subir

Ahora ya debes de estar listo para comenzar con la sección re recursión.

Origen del curso: Curso de Karel OMI

Creador del módulo: Félix Rafael Horta Cuadrilla