



Funciones en Karel

OMIPS AGS - Primer etapa

Repaso Ciclos Anidados: Solución Rellena la habitación

8	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
	1	2	3	4	5	6	7	8

```
iniciar-programa
  inicia-ejecucion
    mientras derecha-libre hacer
      inicio
        mientras frente-libre hacer
          inicio
            deja-zumbador;
            avanza;
          fin;
        deja-zumbador;
        gira-izquierda;
        gira-izquierda;
        mientras frente-libre hacer avanza;
        gira-izquierda;
        avanza;
        gira-izquierda;
      fin;

    mientras frente-libre hacer
      inicio
        deja-zumbador;
        avanza;
      fin;
    deja-zumbador;
    gira-izquierda;
    gira-izquierda;
    mientras frente-libre hacer avanza;
    apagate;
  termina-ejecucion
finalizar-programa
```

Ciclos Anidados

La mayoría de las veces que vamos a anidar ciclos es porque tenemos un patrón cuyas repeticiones podemos imaginarlas como por filas y columnas, es decir, hacemos algo para cada columna (como avanzar al norte) y queremos repetirlo para todas las filas.

Columnas

5	1				
4	1				
3	1				
2	1				
1	1				
	1	2	3	4	5

Filas



Repetir acciones

A veces usamos ciclos para repetir acciones, pero no siempre queremos usar la misma acción en el mismo lugar de nuestro código, es para eso que usamos funciones.

Las funciones son bloques de instrucciones agrupadas que puedes llamar desde cualquier parte del código.



¿Cómo declarar una función?

En OmegaUp es necesario usar la siguiente sintaxis:

```
define-nueva-instruccion NOMBRE como  
Instrucción;
```

```
define-nueva-instruccion NOMBRE como  
inicio  
Instrucción; {una o más instrucciones}  
fin;
```



¿Cómo declarar una función?

Aunque en ReKarel las cuatro formas son correctas:

```
define-nueva-instruccion NOMBRE como  
Instrucción;
```

```
define NOMBRE como  
Instrucción;
```

```
define-nueva-instruccion NOMBRE como  
inicio  
Instrucción; {una o más instrucciones}  
fin;
```

```
define NOMBRE como  
inicio  
Instrucción; {una o más instrucciones}  
fin;
```

Ejemplo gira-derecha





Funciones

Algunas ventajas de usar funciones:

- Reutilización de código: podemos escribir un conjunto de instrucciones una sola vez y luego usarlas en diferentes lugares del programa. Esto evita la repetición y reduce la cantidad de código.
- Mantenimiento: si necesitamos cambiar o corregir algo podemos hacerlo una sola vez en lugar de en cada lugar desde donde se llama a la función.
- Claridad: dentro de la estructura de nuestro código es más fácil entender el llamado a una función que el conjunto de instrucciones que realiza la función.
- Modulación: podemos descomponer problemas complejos en partes más pequeñas y manejables, cada función cumpliendo su tarea para llegar a la solución.



A. Karel recoge sus juguetes

```
{Karel recoge sus juguetes}
iniciar-programa

  define-nueva-instruccion limpiar-cuarto como
  inicio
    mientras junto-a-zumbador hacer
      coge-zumbador;
    avanza;
  fin;

  inicia-ejecucion

    mientras frente-libre hacer
      limpiar-cuarto;
    si junto-a-zumbador entonces coge-zumbador;

  apagate;
  termina-ejecucion
finalizar-programa
```



Observaciones Importantes

En el envío en OmegaUp debes usar:

→ `define-nueva-instruccion` NOMBRE como

Las funciones se declaran después de `iniciar-programa` y antes de `inicia-ejecución`

```
iniciar-programa
  define-nueva-instruccion gira-derecha como
  inicio
    gira-izquierda;
    gira-izquierda;
    gira-izquierda;
  fin;

  inicia-ejecucion
    gira-derecha;
    apagate;
  termina-ejecucion
finalizar-programa
```

C. Karel Sparrow

```
{Karel Sparrow}
iniciar-programa

  define leer-pista como
  inicio
    mientras junto-a-zumbador hacer
    inicio
      coge-zumbador;
      gira-izquierda;
    fin;
    avanza;
  fin;

  inicia-ejecucion
  mientras junto-a-zumbador hacer
  inicio
    leer-pista;
  fin;
  ▶apagate;
  termina-ejecucion
finalizar-programa
```

Nota: en OmegaUp se debe usar define-nueva-instruccion

5		2	2	5	1
4					8
3	4	3	6		2
2	2				3
1			1	5	2
	1	2	3	4	5



Ejercicios

Problema (0.00 / 100.00)

A. Karel recoge sus juguetes

Problema (0.00 / 100.00)

B. Karel 7Segment Display

Problema (0.00 / 100.00)

C. Karel Sparrow