Procesamiento

- Lógica
- Amolar
- Matemáticas
- Textos
- Listas
- Uso
- Variables
- Funciones

Lógica

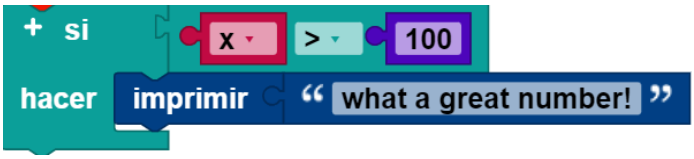
Instrucciones condicionales

Las instrucciones son fundamentales para realizar la programación. Permiten realizar distinciones entre casos para la formulación, como:

- Si hay un camino a la izquierda, gire a la izquierda.
- Si la puntuación = 100, pulse «¡Bien hecho!»

Bloques si

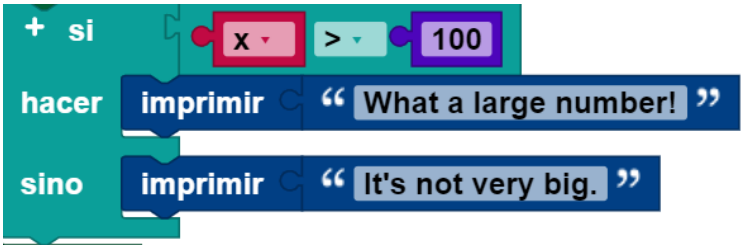
El **bloque** si incluye una condición sencilla:



Si este se implementa, el valor de la variable **x** se compara con 100. Cuando el valor es superior aparece «¡Qué cifra más alta!» En caso contrario, no se muestra nada.

Bloques si-en caso contrario

También es posible indicar lo que se desea que suceda, en caso de que no se cumpla la condición, como en este ejemplo:

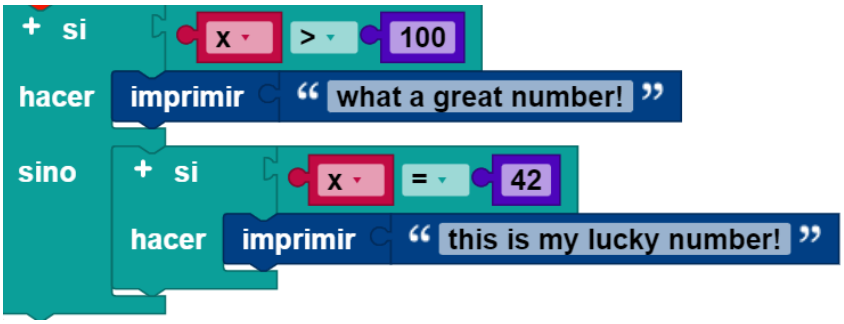


Como en el bloque anterior, se muestra «¡Qué cifra más alta!» cuando **x** > 100. De lo contrario, aparecerá «La cifra no es muy alta».

Un **bloque** si tiene un **apartado** en caso contrario, pero no más de uno.

Bloques **si-en caso contrario-si**

También es posible comprobar más condiciones con un único bloque **si**, en el que se pueden añadir cláusulas **en caso contrario-si**:

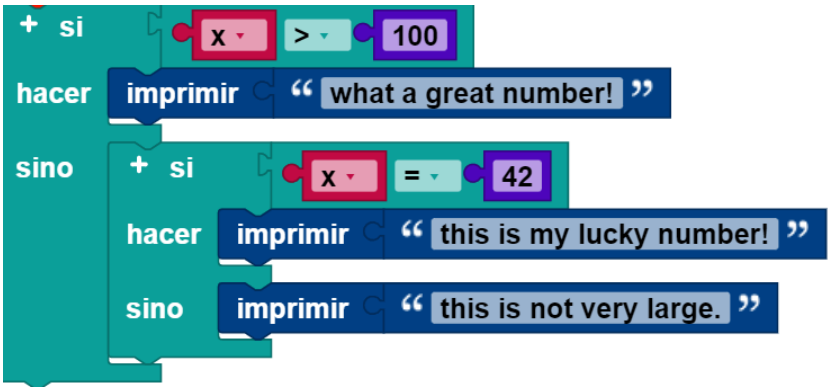


El bloque examina primero si **x** > 100, y muestra «¡Qué cifra más alta!» en caso de que corresponda. Si no se da el caso, vuelve a comprobar si **x** = 42. Si ese es el caso, aparece «¡Ese es mi número de la suerte!» En caso contrario, no se muestra nada.

Un bloque **si** puede tener un número discrecional de apartados **en caso contrario-si**. Las condiciones se evalúan en orden descendente hasta que se cumple una de ellas o hasta que no quedan más condiciones.

Bloques **si-en caso contrario-si-en caso contrario**

Los bloques **si** pueden tener apartados **en caso contrario-si** y apartados **en caso contrario**:



El apartado **en caso contrario** garantiza que se lleve a cabo una acción aunque no se cumplan las condiciones anteriores.

El apartado **en caso contrario** puede aparecer después de un número discrecional de secciones **en caso contrario-si**, incluido el cero. Después se obtiene un bloque completamente normal **si-en caso contrario**.

Modificación de bloques

En la barra de herramientas solamente aparece el bloque sencillo **si** y el bloque **si-en caso contrario**:

Para añadir las cláusulas **en caso contrario-si** - y **en caso contrario**, haga clic en el símbolo (+). Con el símbolo (-) puede quitar las cláusulas **en caso contrario-si**:

Tenga en cuenta que las formas de los bloques permiten añadir cualquier número de subbloques **en caso contrario-si**, pero únicamente un bloque **si**.

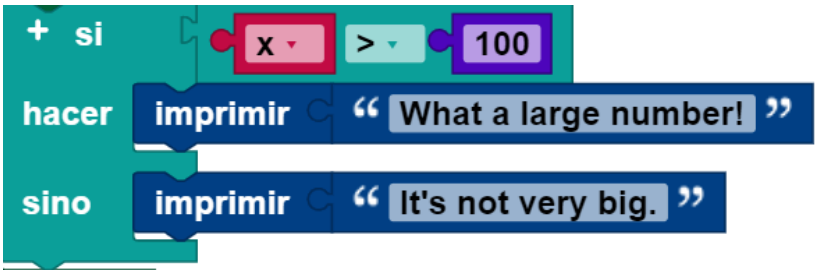
Lógica booleana

La lógica booleana es un sistema matemático simple que tiene dos valores:

- **verdadero**
- **falso**

Los bloques lógicos en ROBO Pro Coding se utilizan generalmente para controlar condiciones y bucles.

El siguiente es un ejemplo:



Cuando el valor de la variable x es superior a 100, cuando se cumple la condición y se muestra el texto «¡Qué cifra más alta!» Si el valor de x no es superior a 100, no se cumple la condición y se muestra «La cifra no es muy alta». Los valores booleanos también se pueden almacenar en variables y pasar a funciones, al igual que los números, el texto y los valores de lista.

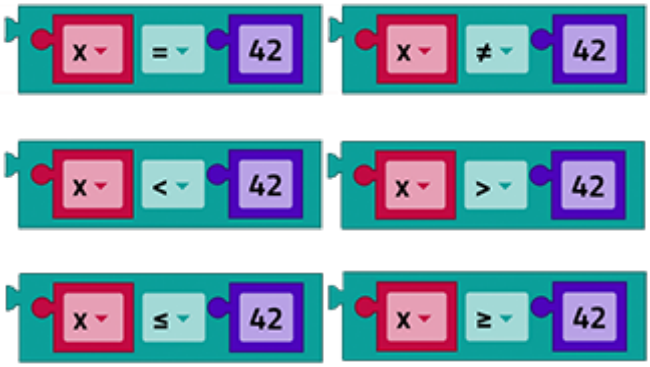
Si en un bloque se ha de introducir un valor booleano, cuando falta la entrada, esta se interpreta como **falsa**. Los valores no booleanos no se pueden insertar directamente en el lugar de los valores booleanos, aunque es posible (pero no recomendable) almacenar un valor no booleano en una variable y después insertarlo en la entrada que hace referencia a una condición concreta. Este método no se recomienda y su comportamiento puede cambiar en futuras versiones de ROBO Pro Coding.

Valores

Se puede utilizar un único bloque con una lista desplegable que muestre **verdadero** o **falso** para acceder a un valor booleano:

Operadores comparativos

Hay seis operadores comparativos. A cada uno se le otorgan dos entradas (normalmente dos cifras) y el operador comparativo muestra **verdadero** o **falso**, dependiendo de cómo se comparen las entradas.



Los seis operadores son: igual, diferente, inferior a, superior a, inferior a o igual y superior a o igual.

Operadores lógicos

El bloque **y** muestra **verdadero** únicamente cuando se cumplen ambos valores de entrada.

El bloque **o** muestra **verdadero** cuando se cumple, como mínimo, uno de sus dos valores de entrada.

no

El bloque **no** convierte una entrada booleana en su opuesto. Por ejemplo, esto es resultado de:

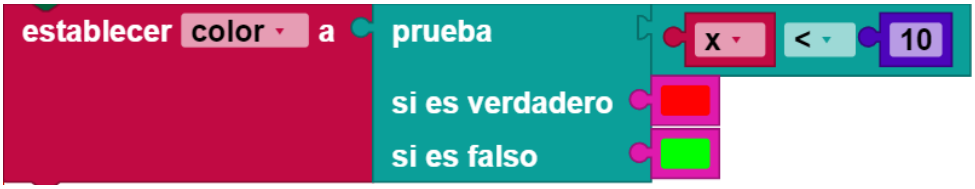
falso.

Si no hay ninguna entrada, se asume que el valor es **verdadero** por lo que el siguiente bloque genera el valor **falso**:

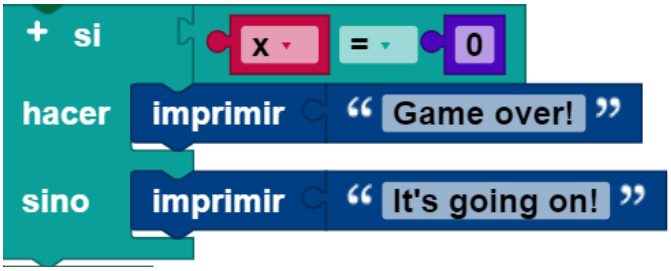
Sin embargo, no se recomienda dejar una entrada vacía.

Operador tres

El operador tres se comporta como un bloque en miniatura**si-en caso contrario**. Se necesitan tres valores de entrada: el primer valor de entrada es la condición booleana que se va a probar, el segundo valor de entrada es el valor que debe mostrarse si la prueba da como resultado **verdadero**, y el tercer valor de entrada es el valor que debe aparecer si la prueba muestra falso. En el siguiente ejemplo, la variable **color** se configura en rojo, si la variable **x** es inferior a 10. En caso contrario, la variable **color** se configura en verde.



Un bloque tres siempre se puede reemplazar por un bloque **si-en caso contrario**. Los dos ejemplos siguientes son exactamente iguales.



Amolar

El área «Control» contiene bloques que controlan si se están implementando otros bloques ubicados dentro de ellos. Existen dos tipos de bloques de control: **los bloques si-en caso contrario** (que se describen en una página independiente) y los bloques que controlan la frecuencia con la que se ejecutan sus elementos internos. Estos últimos se denominan bucles porque su interior, también conocido como cuerpo o cuerpo del bucle, se repite (probablemente) varias veces. Cada recorrido de un bucle se denomina iteración.

Bloques para la creación de bucles

repetir constantemente

El bloque **repetir constantemente** ejecuta el código en su cuerpo hasta que finaliza el programa.

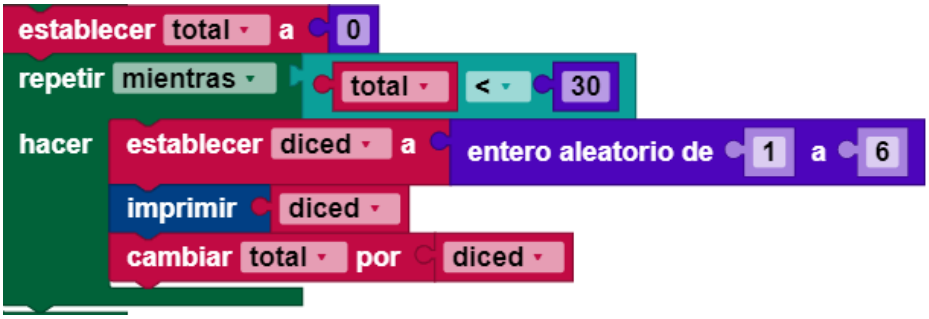
repetición

El bloque **repetición** ejecuta el código en su cuerpo según la frecuencia especificada. Por ejemplo, el siguiente bloque muestra «¡Hola!» diez veces:

repetición-mientras que

Imagine un juego en el que un jugador lanza un dado y suma todos los valores obtenidos, siempre que el total sea inferior a 30. Los siguientes bloques implementan este juego:

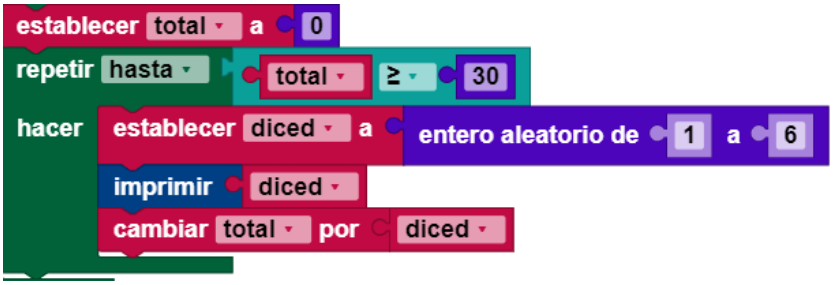
1. Una variable denominada **en total** contiene un valor inicial de 0.
2. El bucle comprueba en primer lugar si **en total** es inferior a 30. Si es inferior, los bloques se implementan en el cuerpo.
3. Se genera un número aleatorio en el intervalo de 1 a 6 (para simular una tirada de dados) y se almacena en una variable denominada **tirada de dados**.
4. Se muestra el número obtenido.
5. La variable **en total** aumenta con el número de tiradas.
6. Cuando se llega al final del bucle, el control vuelve al paso



Una vez finalizado el bucle, se recorren todos los bloques posteriores (no mostrados). En el ejemplo, el recorrido de los bucles finaliza después de haberse mostrado un número determinado de cifras aleatorias en el intervalo de 1 a 6, y el valor de la variable **en total** tiene la suma de estos números que es, como mínimo, de 30.

repetición-hasta

Los bucles **repetición-si** repiten su cuerpo **si** se cumple una condición. Los bucles **repetición hasta** son parecidos, con la diferencia de que repiten su cuerpo **hasta** que se cumple una condición determinada. Los bloques siguientes son equivalentes al ejemplo anterior, ya que el bucle se ejecuta hasta que **en total** es superior o igual a 30.



contar-desde-hasta

El bucle **contar-desde-hasta** incrementa el valor de una variable, comenzando con un primer valor, terminando con un segundo valor y en incrementos de un tercer valor, ejecutando una vez el cuerpo por cada valor de la variable. Por ejemplo, el siguiente programa genera los números 1, 3 y 5.

Como muestran los dos bucles siguientes, que generan los números 5, 3 y 1, este primer valor puede ser superior al segundo. El comportamiento es el mismo, independientemente de que el valor incremental (tercer valor) sea positivo o negativo.

para cada

El bloque **para cada** es similar al bucle **contar-desde-hasta**, solo que en lugar de usar la variable del bucle en orden numérico, utiliza los valores de una lista en orden. El siguiente programa genera cada elemento de la lista «alfa», «beta» y «gamma»:

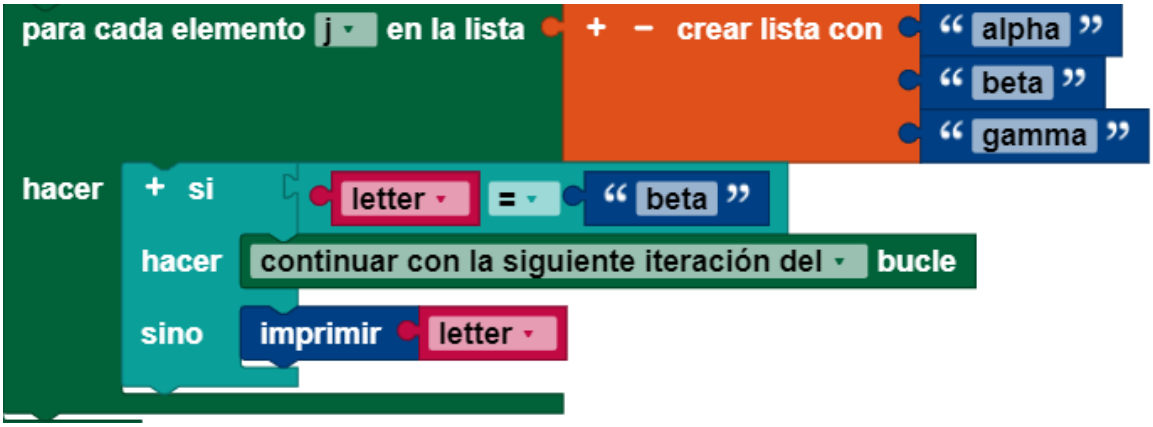
Bloques de interrupción de bucles

La mayoría de los bucles se ejecutan hasta que se cumple la condición de interrupción (para los bloques de **repetición**) o hasta que se aceptan todos los valores de la variable del bucle (en el caso de los bloques **contar con** y **para cada**). Dos bloques de uso poco frecuente, pero ocasionalmente útiles, ofrecen opciones adicionales para controlar el comportamiento del bucle. Se pueden utilizar con cualquier tipo de bucle, aunque los siguientes ejemplos muestran su uso con el bucle **para cada**.

continuar-con-la-siguiente-iteración

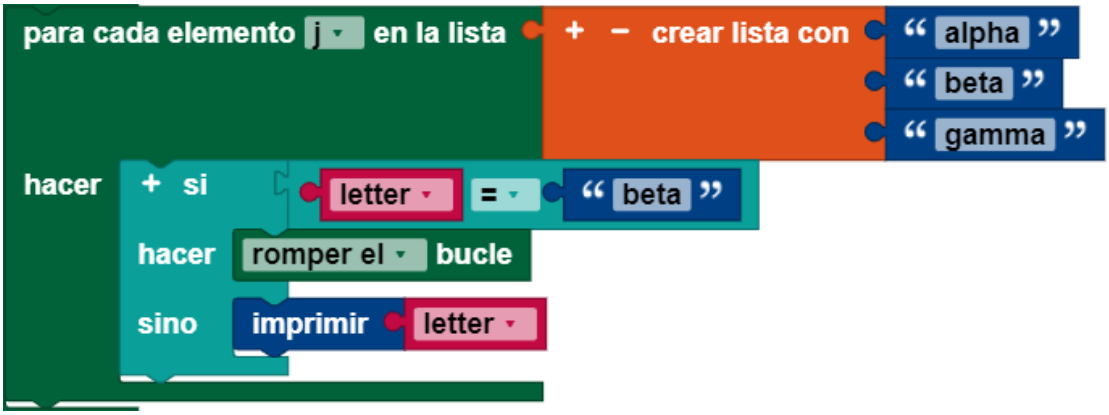
continuar-con-la-siguiente-iteración hace que se omitan los bloques restantes en el cuerpo del bucle y que comience la siguiente iteración del bucle.

El siguiente programa genera «alfa» en la primera iteración del bucle. En la segunda iteración se ejecuta el bloque **continuar con la siguiente iteración**, por lo que se omite la salida de «beta». En la última iteración se pulsa «gamma».



Interrupción del bucle

El bloque **interrupción del bucle** permite salir de manera anticipada del bucle. El siguiente programa genera «alfa» en la primera iteración e interrumpe el bucle en la segunda iteración si la variable del bucle es igual a «beta». El tercer punto de la lista nunca se alcanza.


para cada elemento j en la lista
+ − crear lista con
" alpha "
" beta "
" gamma "
hacer
+ si
letter
=
" beta "
hacer
romper el
bucle
sino
imprimir
letter

Matemáticas

Los bloques de la categoría Matemáticas se utilizan para realizar cálculos. Por ejemplo, los resultados de los cálculos se pueden utilizar como valores para variables. La mayoría de los bloques matemáticos se relacionan con cálculos matemáticos generales y deben ser autoexplicativos.

Bloques

Cifras

Utilice el bloque numérico para añadir cualquier número a su programa o para asignar este número como valor a una variable. Este programa asigna el número 12 a la variable **edad**:

Cálculos sencillos

Este bloque tiene la estructura valor-operador-valor. Los operadores aritméticos existentes son: **+**, **-**, **÷**, **×** y **^**. El operador se puede seleccionar en el menú desplegable. Se puede aplicar directamente a números o valores de variables. Ejemplo:

Este bloque genera el resultado 144 (12^2).

Cálculos especiales

Este bloque aplica el tipo de cálculo seleccionado a través del menú desplegable al número anterior o al valor de la variable previo. Las operaciones aritméticas disponibles son:

- Raíz cuadrada,
- Cantidad,
- Logaritmo natural,
- Logaritmo decádico,
- Función exponencial con base e (e^1, e^2, ...),
- Función exponencial con base 10 (10^1, 10^2,...),
- Cambio de signo (multiplicación por -1).

Aquí e es el número de Euler. Este bloque extrae la raíz cuadrada de 16 y añade la variable **i** al resultado.

Funciones trigonométricas

Este bloque funciona de manera similar al bloque descrito anteriormente, con la diferencia de que las funciones trigonométricas seno, coseno, tangente y sus funciones inversas se utilizan como operaciones aritméticas. El número o el valor de la variable especificados se utilizan en la función seleccionada en el menú desplegable y el resultado se puede procesar posteriormente en el programa. Además, también existe el bloque **arctan2 of X: ... Y: ...**, que permite generar un valor de función del arctan2 en el rango de 360° con la ayuda de dos números reales (que se insertan como X e Y).

Constantes de uso frecuente

Este bloque funciona de la misma manera que el bloque numérico, pero usted mismo no puede introducir aquí el valor numérico. En cambio, las constantes de uso frecuente (p. ej., ?) se guardan previamente. La constante se puede seleccionar en el menú desplegable.

Resto de una división

El bloque **resto de ...** se utiliza para generar el resto de una división. Este programa asigna el resto de la división de 3:2, es decir, 1 a la variable **resto**:

Redondeos

El bloque **redondeo ...** se puede utilizar para redondear a un número entero un número decimal específico o el valor de una variable concreta. Hay tres opciones para elegir en el menú desplegable:

- con «redondear» se redondea al siguiente número entero (por ejemplo, 4,5 se convierte en 5)
- con «redondear hacia arriba» se redondea hacia arriba (por ejemplo, 5,1 se convierte en 6)
- con «redondear hacia abajo» se redondea hacia abajo (por ejemplo, 5,9 se convierte en 5).

Evaluación de listas

Con el bloque **... de la lista** se puede generar

- con «suma» la suma de todos los valores de una lista,
- con «mínimo» el valor más pequeño de una lista,
- con «máximo» el valor más grande de una lista,
- con «promedio» el promedio de todos los valores de una lista,
- con «mediana» la mediana de una lista,
- con «modas» el valor que aparece con más frecuencia en una lista,
- con «desviación estándar» la desviación estándar de todos los valores de una lista,
- con «elemento aleatorio» un valor aleatorio de una lista

. Todas estas opciones se pueden seleccionar a través del menú desplegable del bloque:

Limitar los valores de entrada

El bloque **limitación ... desde ... hasta ...** permite limitar un valor de entrada en un intervalo determinado. Antes de que se siga procesando un valor de entrada, se comprueba si se encuentra dentro del intervalo especificado. Hay tres opciones para proceder con un valor introducido:

- El valor está en el intervalo, por lo que se transfiere sin cambios.
- El valor está por debajo del límite inferior del intervalo, por lo que este límite inferior se transfiere.
- El valor está por encima del límite superior del intervalo, por lo que este límite superior se transfiere.

En este ejemplo, el bloque se utiliza para limitar el valor de la variable **velocidad** al número de revoluciones admitido por el motor:

Generar valores aleatorios

Los dos bloques **número aleatorio desde ... hasta...** y **fracción aleatoria** generan un valor aleatorio. Para ello, el **bloque número aleatorio desde ... hasta...** genera un número en el intervalo especificado. Por el contrario, el bloque **fracción aleatoria** genera un valor entre 0,0 (conectado) y 1,0 (desconectado).

Textos

Estos son algunos ejemplos de textos:

«Elemento 1»
«12 de marzo de 2010»
«» (texto vacío)
El texto puede contener letras (mayúsculas o minúsculas), cifras, signos de puntuación, otros símbolos y espacios en blanco.

Bloques

Creación de texto

El siguiente bloque genera el texto «Hola» y lo guarda en la variable denominada **saludo:**

El bloque **crear texto con** combina el valor de la variable **saludo** y el texto nuevo «mundo» con el texto «HolaMundo». Tenga en cuenta que no hay espacios en blanco entre los dos textos, ya que no existían en los textos originales.

Para aumentar el número de entradas de texto, haga clic en el símbolo (+). Para eliminar las últimas ediciones, haga clic en el símbolo (-).

Modificación del texto

El bloque **añadir ... a** agrega el texto especificado a una variable concreta. En este ejemplo cambia el valor de la variable **saludo** de «Hola» a «¡Hey, hola!»:

Longitud del texto

El bloque **longitud de** cuenta el número de signos (letras, cifras, etc.) que contiene un texto. La longitud de ¡Nosotros somos #1! es 19, y la longitud del texto vacío es 0.

Comprobar si hay texto vacío

El bloque **está vacío** comprueba si el texto especificado está vacío (la longitud es de 0). En el primer ejemplo el resultado es **verdadero** y en el segundo ejemplo es **falso**.

Búsqueda de texto

Estos bloques se pueden utilizar para comprobar si un texto aparece en otro texto y, en caso de ser así, para saber en qué lugar aparece. Por ejemplo, aquí se pregunta acerca del primer lugar donde aparece la «a» en la palabra «Hola», y el resultado es 4:

En este caso se pregunta acerca del último lugar donde aparece la «a» en la palabra «Hola», y el resultado también es 4:

Independientemente de si se selecciona el primer o el último lugar, este bloque muestra el resultado 0, ya que «Hola» no contiene ninguna «z».

Extracción de texto

Extracción de un solo carácter

Esto da como resultado «b», la segunda letra en «abcde»:

Esto da como resultado «d», la penúltima letra en «abcde»:

Esto da como resultado «a», la primera letra en «abcde»:

Esto da como resultado «e», la última letra en «abcde»:

Esto da lugar a cada una de las 5 letras de «abcde» con la misma probabilidad:

Ninguno de los dos cambia el texto del que se extrae.

Extraer un área de texto

Con el bloque **introducir secuencia de caracteres en el texto...** se puede extraer un área de texto que comience con:

- Letra #
- Letra # del final
- Primera letra

y finalice con:

- Letra #
- Letra # del final
- Última letra

En el siguiente ejemplo se extrae «abc»:

Ajuste de la letra mayúscula/minúscula del texto

Este bloque crea una versión del texto de entrada que puede corresponderse con uno de los siguientes casos

- MAYÚSCULA (todas las letras en mayúscula) o
- minúscula (todas las letras en minúscula) o
- sustantivos (primeras letras en mayúscula y otras letras en minúscula).

El resultado del bloque siguiente es «HOLA»:

Los caracteres no alfabéticos no se ven afectados. Tenga en cuenta que este bloque no funciona con texto en idiomas que no distinguen entre mayúsculas y minúsculas, como el chino.

Quitar (eliminar) espacios en blanco

El siguiente bloque elimina espacios en blanco, dependiendo de lo que se configure en el menú desplegable (triángulo pequeño):

- al principio del texto
- al final del texto
- a ambos lados del texto

El resultado del bloque siguiente es «Hey, tú»:

No se encuentran espacios en blanco en la parte central del texto.

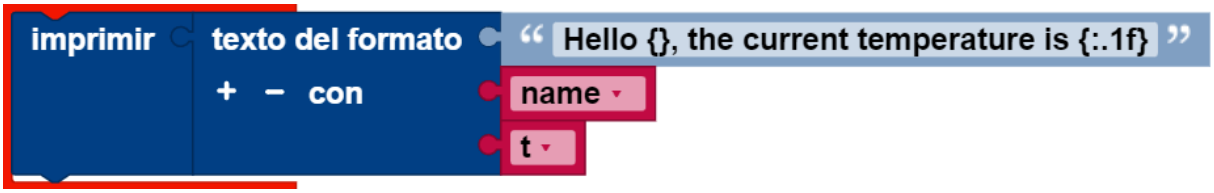
Generar texto

El bloque **generar** hace que el valor de entrada se muestre en la ventana de la consola:

En ningún caso se enviará a la impresora, como sugiere el nombre.

Editar texto con formato

Con el bloque **formatear texto** se puede editar texto con diferente contenido. Todos los marcadores de posición **{}** del texto se reemplazan por el contenido de las variables adjuntas después del texto. El formato se puede especificar entre corchetes. Por ejemplo, el formato **{:.1f}** solo genera la primera cifra decimal en la variable **t**.



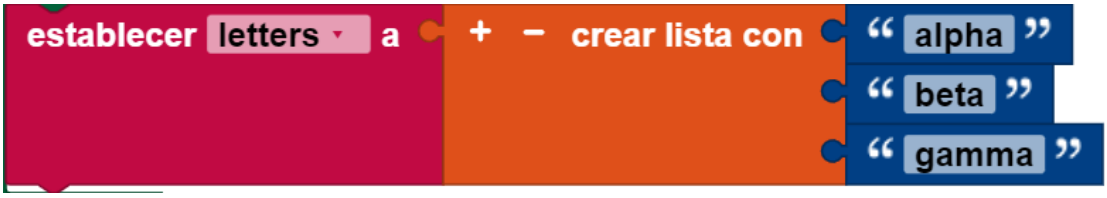
Listas

Como en el lenguaje cotidiano, una lista en ROBO Pro Coding también es un conjunto ordenado de elementos como, por ejemplo, una lista «To-Do» (de cosas por hacer) o una lista de la compra. Los elementos de una lista pueden ser de cualquier tipo y el mismo valor puede aparecer más de una vez en una lista.

Crear una lista

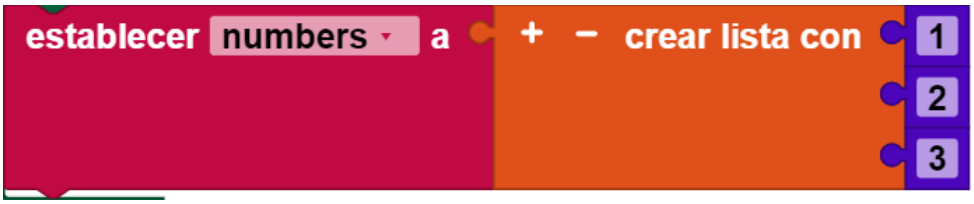
crear lista con

Con el bloque **crear lista con** se pueden insertar los valores iniciales en una nueva lista. En este ejemplo se crea una lista de palabras y se coloca en una variable denominada **letras**:



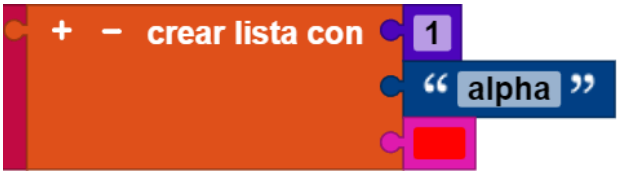
A esta lista la denominamos [«alpha», «beta», «gamma»].

Aquí se muestra la creación de una lista de **números**:



Así se crea una lista de **colores**:

Es menos común, aunque posible, crear una lista a partir de valores de diferentes tipos:



Cambiar el número de entradas

Para modificar el número de entradas, haga clic o toque el símbolo del engranaje. Con esta acción se abrirá una nueva ventana. Puede arrastrar subbloques de elementos desde la parte izquierda de la ventana hasta el bloque

de lista en la parte derecha para añadir una nueva entrada:

Aunque el nuevo elemento se ha añadido abajo en este ejemplo, se puede agregar en cualquier lugar. Del mismo modo, los subbloques de elementos no deseados se pueden arrastrar hacia la izquierda desde el bloque de lista.

Crear lista con el elemento

Con el bloque **crear lista con el elemento** puede generar una lista que contenga el número especificado de copias de un elemento. Por ejemplo, los siguientes bloques colocan la variable **palabras** en la lista [«muy», «muy», «muy»].

Comprobación de la extensión de una lista

está vacía

El valor del bloque **está vacía** es **verdadero** cuando la entrada se corresponde con una lista vacía, y **falso** cuando se da un caso distinto. ¿Es **verdadera** esta entrada? El valor del bloque siguiente sería **falso**, porque la variable Colores no está vacía: contiene tres elementos.

Tenga en cuenta la similitud del texto con el bloque **está vacía**.

Longitud de

El valor del bloque **longitud de** se corresponde con el número de elementos de la lista que se utilizan como entrada. Por ejemplo, el valor del siguiente bloque sería 3, ya que el **Color** contiene tres elementos:

Tenga en cuenta que el bloque **longitud de** indica cuántos elementos contiene la lista y no cuántos elementos diferentes hay en ella. Por ejemplo, a continuación se muestra 3 aunque las**palabras** se componen de tres copias del mismo texto:

Tenga en cuenta la similitud del texto con el bloque **longitud de**.

Buscar elementos en una lista

Estos bloques encuentran la posición de un elemento en una lista. El siguiente ejemplo tiene el valor 1 porque «muy» aparece por primera vez al principio de la lista de palabras ([«muy», «muy», «muy»]).

El resultado siguiente es 3, ya que «muy» aparece por última vez en **palabras** en la posición 3.

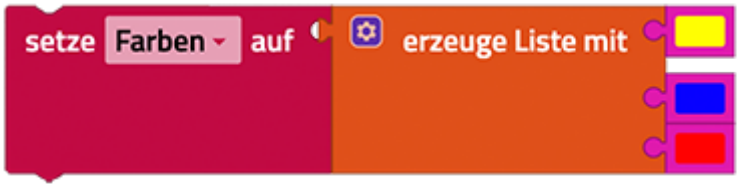
Si el elemento no aparece en ninguna parte de la lista, el resultado es el valor 0, como se muestra en este ejemplo:

Estos bloques se comportan de la misma manera que los bloques para buscar letras en un texto.

Acceder a elementos de una lista

Acceder a un único elemento

Recuerde la definición de la lista **Colores**:



El siguiente bloque se pone de color azul, ya que es el segundo elemento de la lista (comenzando a contar por la izquierda):

Este se pone verde, ya que es el segundo elemento (contando desde el extremo derecho):

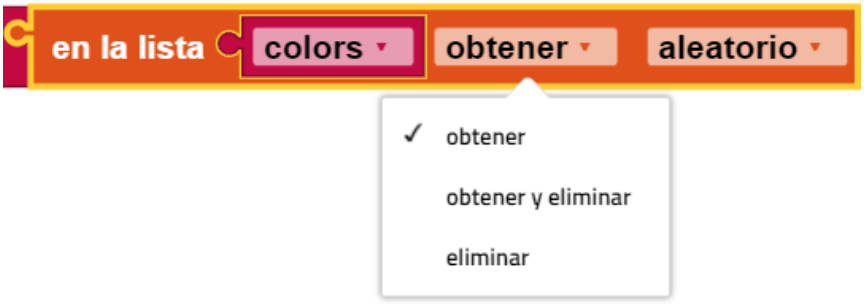
Este obtiene el primer elemento, rojo

Este obtiene el último elemento, amarillo:

Esto selecciona aleatoriamente un elemento de la lista, con la misma probabilidad de mostrar uno de los elementos rojo, azul, verde o amarillo.

Acceder a un elemento y eliminarlo

En el menú desplegable se cambia el bloque **acceder a... de la lista** por el bloque **acceder a ... de la lista y eliminar** , que da el mismo resultado, pero también cambia la lista:



Este ejemplo establece la variable **primera letra** en «alpha» y deja las letras restantes ([«beta», «gamma»]) en la lista.

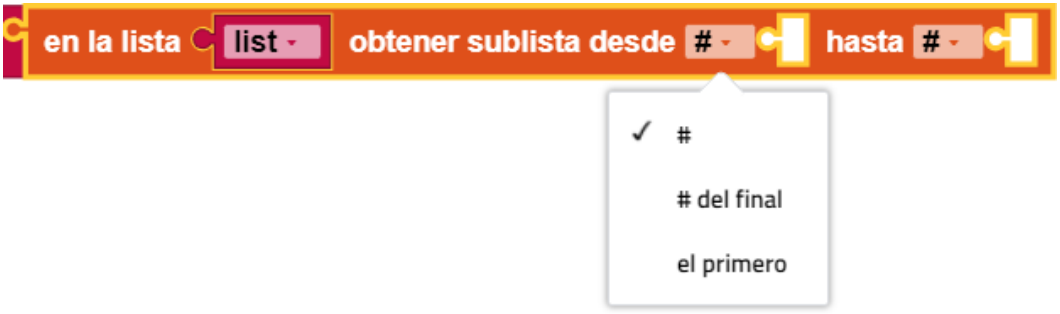
Eliminar una entrada

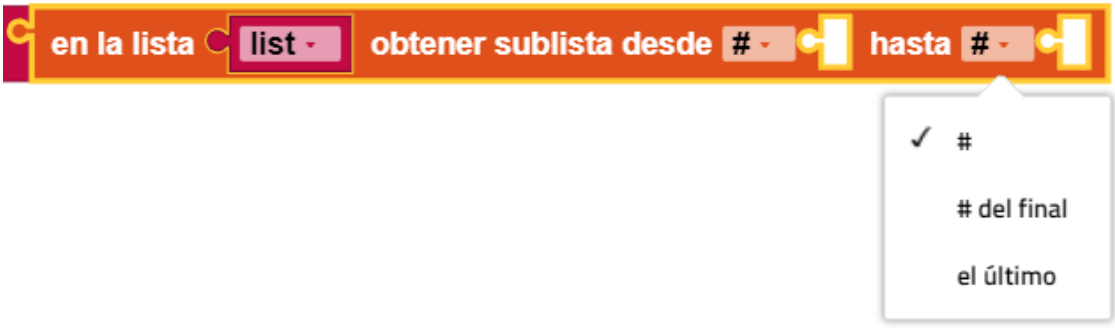
Si selecciona **eliminar** en el menú desplegable, desaparece el surco a la izquierda del bloque:

Así, se elimina el primer elemento de **letras**.

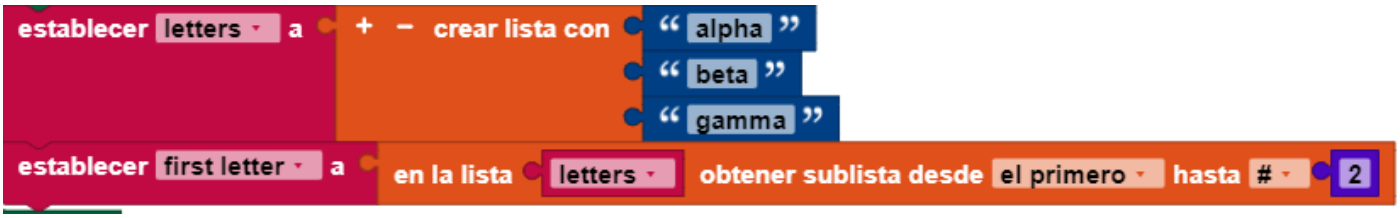
Obtener una sublista

El bloque **en la lista ... Obtener sublista** se asemeja al bloque **acceder a... de la lista** con la diferencia de que extrae una sublista en lugar de un solo elemento. Existen varias opciones para especificar el principio y el final de la sublista:





En este ejemplo se crea una nueva lista de **primera letra**. Esta nueva lista contiene dos elementos: [«alpha», «beta»].



Tenga en cuenta que este bloque no modifica la lista original.

Añadir elementos a una lista

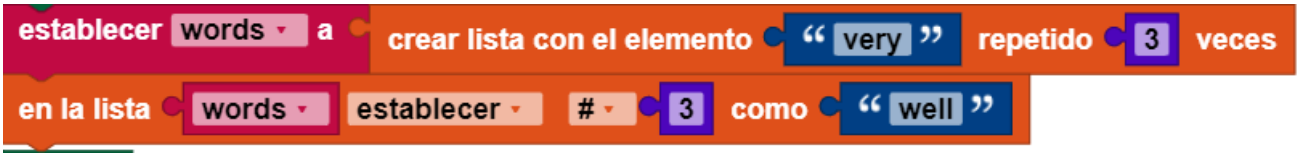
Crear elementos en una lista

El bloque **reemplazar en la lista...** sustituye el elemento que se encuentra en una determinada posición de una lista por otro elemento.

Puede consultar el significado de las opciones desplegables individuales en la sección anterior.

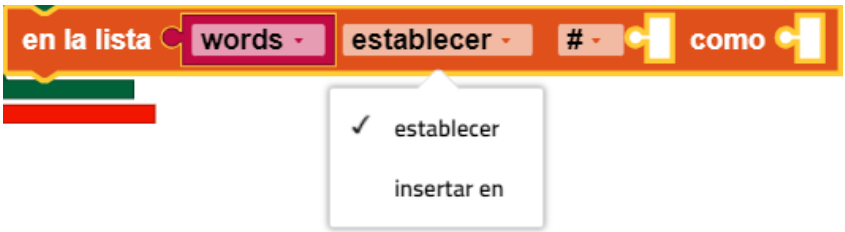
El siguiente ejemplo destaca dos cosas:

1. La lista **palabras** se crea con 3 elementos: [«muy», «muy», «muy»].
2. El tercer elemento de la lista se reemplaza por «bueno». El nuevo valor de **palabras** es [«muy», «muy», «bueno»]



Añadir elementos en una posición determinada de la lista

Se accede al bloque **insertar en la lista ...** a través del menú desplegable del bloque **reemplazar en la lista...**:



Inserta un nuevo elemento en la posición especificada de la lista, antes del elemento que estaba previamente en esa posición. El siguiente ejemplo (basado en un ejemplo anterior) hace referencia a tres cosas:

1. La lista **palabras** se crea con 3 elementos: [«muy», «muy», «muy»].
2. El tercer elemento de la lista se reemplaza por «bueno». Por tanto, el nuevo valor de **palabras** es [«muy», «muy», «bueno»].
3. La palabra «Ser» se inserta al principio de la lista. Por tanto, el valor definitivo de **palabras** es [«ser», «muy», «muy», «bueno»].

Dividir secuencias de caracteres y unir listas

Hacer lista a partir de texto

El bloque **hacer lista a partir de texto** divide el texto especificado en partes con ayuda de un delimitador:

En el ejemplo anterior se muestra una nueva lista que contiene tres fragmentos de texto: «311», «555» y «2368».

Hacer texto a partir de lista

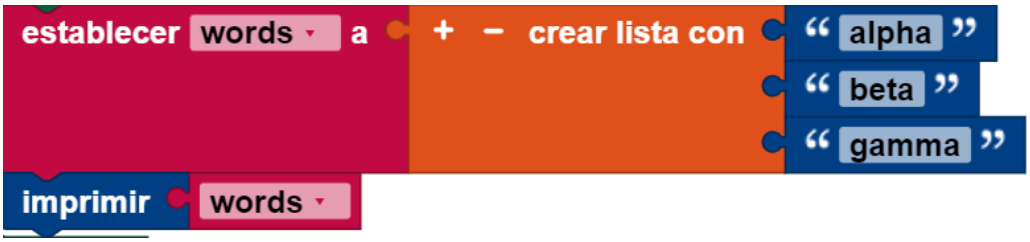
El bloque **hacer texto a partir de lista** fusiona una lista en un solo texto con la ayuda de un delimitador:

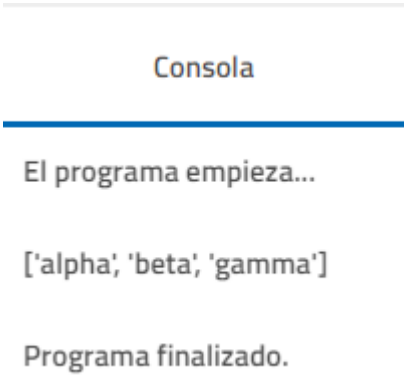
En el ejemplo anterior se muestra un nuevo texto con el valor: «311-555-2368».

Bloques relacionados

Imprimir una lista

El bloque **imprimir** de la categoría Texto puede generar listas. El resultado del siguiente programa es la salida de la consola mostrada:





Implementar algo en cada elemento de una lista

El bloque **para cada** de la categoría Control realiza una operación en cada elemento de una lista. Por ejemplo, este bloque imprime cada elemento de la lista individualmente:

Esto no elimina los elementos de la lista original.

Consulte también los ejemplos referentes a Bloques de interrupción de bucles.

Uso

Con ROBO Pro Coding, la categoría de uso incluye bloques de los siguientes tipos:

- Selección de color
- Esperar
- Código Python
- Comenzar
- Ejecución de funciones

Selección de color

Este bloque se utiliza como valor de entrada cuando se hace una pregunta sobre un color (por ejemplo, cuando la cámara hace coincidir el color). Al hacer clic o tocar el color, se puede seleccionar uno de los 70 colores de la paleta de colores.

Esperar

Esperar hasta que se acabe el tiempo

El bloque **espere [] ...** evita que el programa continúe durante el tiempo de espera especificado. La unidad de tiempo se puede seleccionar en el menú desplegable (triángulo pequeño) y la extensión deseada de la pausa en el campo de entrada detrás de ella.

Esperar con condición

Con el bloque **esperar hasta**, la pausa no está vinculada al tiempo sino al cumplimiento de una condición (por ejemplo, si se pulsa un botón). La condición se añade al bloque **esperar hasta**.

Código Python

Si desea integrar el código Python existente en ROBO Pro Coding, puede insertarlo en el bloque **código Python** . Posteriormente, el programa ejecuta todo lo que se especificó en el bloque de Python.

Comenzar

El bloque **comenzar si** también está vinculado a una condición. Solo cuando se cumple esta condición, se inicia el programa en el cuerpo del bloque.

Ejecución de funciones

Con **ejecutar función ... en un hilo** , la función seleccionada se puede ejecutar en un hilo independiente. En algunos casos, esta medida puede permitir que un programa continúe respondiendo a las entradas y se ejecute de manera más rápida.

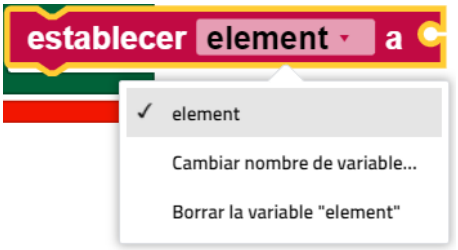
Variables

Utilizamos el término variable tal como se usa en matemáticas y en otros lenguajes de programación: un valor con nombre que se puede cambiar (variar). Las variables se pueden crear de diferentes formas.

- Algunos bloques como **contar con** y **para cada** usan una variable y definen sus valores. Un término tradicional en informática para hacer referencia a dichas variables es variables de bucle.
- Las funciones definidas por el usuario (también conocidas como «procedimientos») pueden definir entradas, creando variables que solo pueden usarse dentro de esa función. Estas variables se denominan tradicionalmente «parámetros» o «argumentos».
- Los usuarios pueden cambiar las variables en cualquier momento utilizando el bloque **configurar**. Estas se conocen tradicionalmente como «variables globales». Se pueden utilizar en cualquier lugar del código de ROBO Pro Coding.

Menú desplegable

Si hace clic en el símbolo desplegable (triángulo pequeño) de una variable, aparecerá el siguiente menú:



El menú ofrece las siguientes opciones.

- la visualización de los nombres de todas las variables existentes definidas en el programa.
- «Cambiar nombre de variable...», es decir, modificar el nombre de esta variable dondequiera que aparezca en el programa (al seleccionar esta opción, se abre una consulta sobre el nuevo nombre)
- «Eliminar variable...», es decir, la eliminación de todos los bloques que hacen referencia a esta variable, dondequiera que aparezca en el programa.

Bloques

Determinar

El bloque **configurar** asigna un valor a una variable y crea la variable si no existe aún. Por ejemplo, el valor de la variable **edad** se establece en 12:

Acceso

El bloque **acceder**proporciona el valor almacenado en una variable sin cambiarlo:

Es posible, pero no una buena idea, escribir un programa en el que aparezca un bloque **acceder** sin que exista el correspondiente bloque de configuración previa.

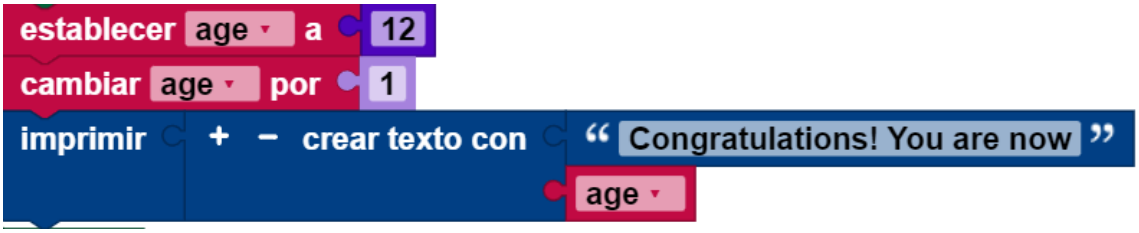
Modificar

El bloque **modificar** añade una cifra a una variable.

El bloque **modificar** es una abreviatura del siguiente constructo:

Ejemplo

Tenga en cuenta el siguiente código de muestra:



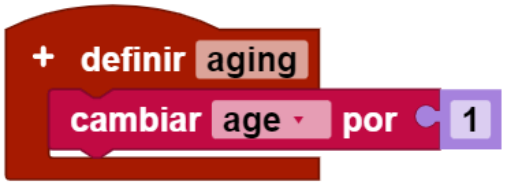
El primer conjunto de bloques crea una variable llamada **edad** y **establece** su valor inicial en el número 12. El segundo conjunto de bloques **obtiene** el valor 12, le suma 1 y almacena la suma (13) en la variable. El mensaje se emite en la última línea: «¡Enhorabuena! Ahora tiene 13».

Funciones

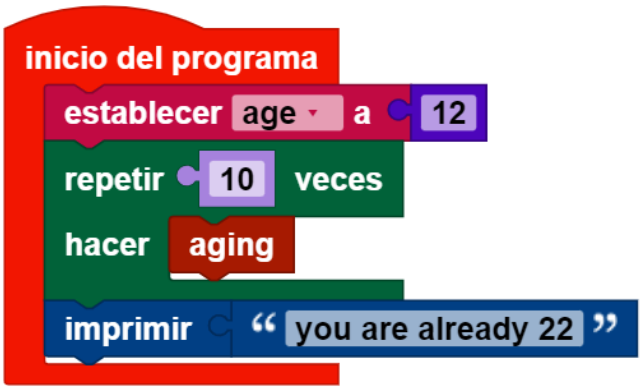
Las funciones se utilizan para saber qué partes del código son reutilizables y, por tanto, para estructurar el código a nivel global. Si llena un bloque de funciones, aparece un nuevo bloque en el menú de funciones con el mismo nombre que este bloque de funciones. Ahora es posible insertar solo el bloque con el nombre de la función en el programa principal. Cuando se ejecuta el programa, este bloque reenvía el código a la función del mismo nombre y lo procesa.

Función simple

El bloque de función simple se puede utilizar para crear una función que tenga el nombre especificado en el campo de texto. La función puede contener cualquier número de variables que se pueden añadir usando el símbolo de engranaje. Esta función **edad** añade 1 a la variable **edad**:



Posteriormente, la función se puede utilizar en el programa principal:



Función con valor de retorno

Este bloque le permite crear una función con un valor de retorno. Posteriormente, este valor de retorno se puede volver a utilizar en el programa principal. El siguiente es un ejemplo: