



# APRENDE Y DIVIÉRTETE



SECTEI

CIUDAD INNOVADORA  
Y DE DERECHOS

## Introducción a Scratch

# Módulo 2

Semana	Tiempo sugerido	Temas/ Subtemas	Aprendizaje esperados	Eje -Ámbitos-Ambientes Sociales de Aprendizaje
5	70-90 minutos	¿Qué es Scratch?	Identifica el entorno de programación basada en un IDE (Entorno de Desarrollo Integrado) de un bajo nivel para desarrollar sus habilidades lógicas.	Estudio, Número, Álgebra, Variación.
6	70-90 minutos	Música en Vivo.	Aplica el entorno conocido para desarrollar su primer programa.	Estudio, Número, Álgebra, Variación.
7	90-120 minutos	¡Atrapa las Criaturas!	Interactúa con las diferentes herramientas que ofrece Scratch. Añade variables y comienza a interactuar con condiciones como if.	Número, Álgebra, Variación.
8	70-90 minutos	Conociendo el Espacio.	Comprende sentencias de programación en lenguaje sencillo; al unirlos crea una serie de animaciones simulando un espacio y añade efectos visuales.	Forma, Espacio y Medida
9	70-90 minutos	Rompecabezas de una Célula	Emplea los conocimientos aprendidos en Scratch Crea un rompecabezas en forma de célula Refuerza sus conocimientos en biología para identificar partes principales de la célula.	Estudio, Número, Ciencias, Variación.



## 5. ¿Qué es Scratch?

### Aprendizajes esperados.

Habilidades	Medio	Contenido	Finalidad
Conoce un entorno que facilita el aprendizaje de la programación.	Por medio de un IDE (entorno de desarrollo integrado): Scratch	¿Qué es Scratch?. ¿Cómo utilizar Scratch? Entorno de trabajo.	Adquirir habilidades de programación básicas para crear programas simples.



## 5. ¿Qué es Scratch?

Para empezar a hablar de Scratch es necesario conocer qué es un lenguaje de programación.

Un lenguaje de programación es un lenguaje formal mediante el que una computadora puede recibir instrucciones para controlar su comportamiento físico y lógico por medio de secuencias, con el objetivo de producir y mostrar diversas clases de datos. Scratch nos facilita la tarea de escribir una secuencia de órdenes de manera visual para programar tu Raspberry.



Realiza este pequeño ejercicio: Anota en un papel las instrucciones que le darías a una persona para llegar de tu casa a la escuela.

### Ejemplo:

Camina 2 calles y giras a la izquierda, luego tomas un pesero que dice: ruta 14. Te bajas en la esquina de avenida 1 y luego caminas dos calles hasta llegar a la cerrada “Juárez”. Luego das vuelta a la izquierda. Verás una pared blanca y una reja verde. Hay un letrero que dice: Escuela Sor Juana Inés de la Cruz. Has llegado a la escuela.

Lo que acabas de hacer es muy parecido a un algoritmo. Así de sencillo. Has generado indicaciones precisas para que una persona pueda hacer una tarea.

Y describiste tu algoritmo utilizando un lenguaje: el español. Pudiste haber ocupado cualquier otro lenguaje, por ejemplo: árabe, náhuatl o inglés.

Bueno, lo mismo pasa con las computadoras y eso es lo que hace un programador: diseñar instrucciones con un lenguaje que las computadoras entienden e interpretan para generar un resultado. Los lenguajes en las computadoras son tan variados como los lenguajes de los humanos. FORTRAN (del inglés **F**ormula **T**ranslating **S**ystem) Fue prácticamente el primer lenguaje de alto nivel para computadoras. Se entiende como lenguaje de alto nivel a aquel que se asemeja al lenguaje humano; si te sientes algo curioso busca algo de información sobre esto.

Cada lenguaje de programación tiene reglas precisas que sirven para comunicarse con la computadora. Ahora verás uno muy entretenido basado en bloques de construcción, que te permitirá construir juegos. Este lenguaje se llama: Scratch.

Scratch es un lenguaje de programación desarrollado por el MIT (Massachusetts Institute of Technology) para personas que desean iniciar en el mundo de la programación. Como ya dijimos, facilita el aprendizaje de forma intuitiva mediante bloques de construcción. En Scratch no escribimos código, se tienen diferentes piezas que se deben ir acomodando y encajando para desarrollar órdenes.

Scratch es una aplicación que se puede descargar en dispositivos móviles o en casi cualquier tipo de computadora. Está disponible en varios sistemas operativos como Windows, Linux, Mac, etc. También está disponible en Raspberry Pi.

# Algunos Lenguajes de Programación.

**1843:** Ada Lovelace crea un lenguaje de programación "manual" para calcular la serie de Bernoulli con la máquina de Babbage.

**1936:** Alan Turing introduce el concepto teórico de una máquina que mediante algoritmos puede resolver cualquier problema.

**1957:** Jonh W. Backus inventa FORTRAN, el primer lenguaje de programación de alto nivel, incluido en la IBM 704.

**1960:** Greace Murray Hopper desarrolla COBOL; Common Business Oriented Language en ingles. Muy utilizado aun en bancos.

**1964:** John G. Kemeny y Thomas E. Kurtz diseñan BASIC, un lenguaje interpretado que fue muy popular en los 80.

**1964:** LOGO fue diseñado por Seymour Papert, con el objetivo de que los niños aprendieran a programar de manera gráfica.

**1970:** PASCAL es creado por Niklaus Wirth, con el objetivo de ser un lenguaje elegante, sencillo de aprender y muy funcional.

**1972:** Dennis Ritchie desarrolla C para los laboratorios "Bell Telephone", para ser integrado en sistemas UNIX

**1972:** Robert Kowalski expone la idea de usar la Lógica como lenguaje de programación. Años después nace PROLOG.

**1979:** ADA es un lenguaje de programación poco popular, algo complejo, cuyo nombre es inspirado en la Condesa de Lovelace.

**1983:** C++ es diseñado por Bjarne Stroustrup. Es una extensión de C como lenguaje de programación orientado a objetos.

**1887:** Larry Wall crea PERL al notar que UNIX no contenía las operaciones que el necesitaba para la manipulación de Texto.

**1991:** PYTHON es creado por Guido Van Rossum, sucesor del lenguaje ABC, con énfasis en la legibilidad del código.

**1991:** JAVA es un lenguaje orientado a objetos muy popular y versátil, diseñado por SUN Microsystems con sintaxis similar a la de C.

**2013:** SCRATCH es desarrollado por el grupo Lifelong Kindergarten del MIT media Lab. En este módulo lo conocerás mejor.

# ¿Cómo utilizar Scratch?

## Pasos a seguir.

Entra a la Raspberry, da clic en el menú principal (la frambuesa), da clic en “**programación**” y selecciona Scratch 3. Al darle clic. Scratch demorará unos segundo en abrir. Es recomendable que no tengas otros programas abiertos para no saturar la memoria, recuerda que la Raspberry solo tiene 1 GB disponible.

## Selecciona: Nuevo proyecto



## Entorno de Trabajo.

Comencemos a explorar Scratch. Para cambiar el idioma haz clic en el “Mundo” y selecciona español.

Ahora haz clic en “Archivo”; ahí podrás crear un nuevo documento, cargar proyectos ya realizados y guardar tus proyectos en la Raspberry.

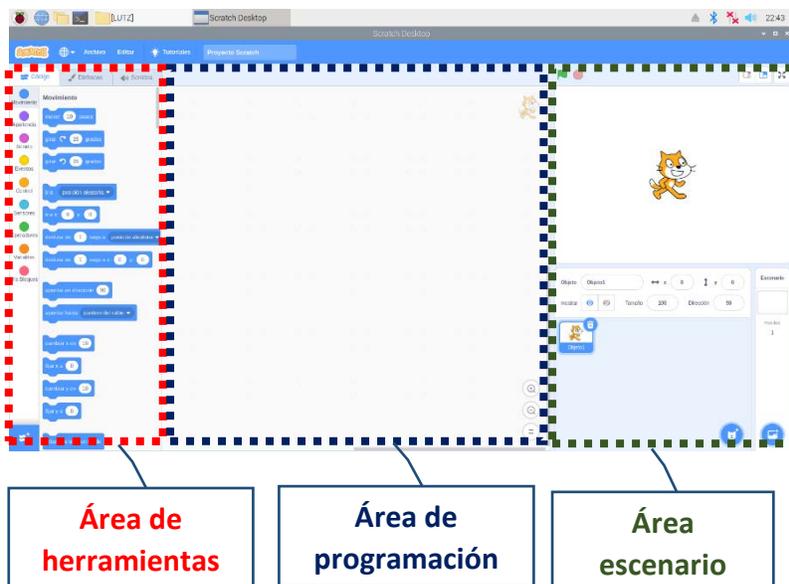
En cuanto a su entorno de desarrollo, Scratch está dividido en tres áreas de trabajo:

En la parte izquierda se encuentra el área de herramientas en donde encontrarás objetos agrupados por paneles. Estos te permitirán darle movimiento a tus imágenes o animaciones, cambiar la apariencia de los objetos, darles sonidos, etc.

También ahí se encuentran las herramientas que permiten seguir trayectorias, manipular estructuras para controlar el movimiento y las acciones de los objetos (actores o personajes), manejar operadores aritméticos, relacionales y lógicos, definir variables y asignar valores.

En la parte central colocarás las instrucciones de tu programa.

Finalmente a la derecha verás el área de escenario donde se visualizará todo lo que se va realizando y como va quedando el programa.



## Área de Herramientas.

### Panel de Movimiento.

Este panel posee bloques que permiten controlar el movimiento de los objetos, darles orientación y posicionar los objetos en la pantalla.



### Panel de Apariencia.

Con los bloques de este panel podrás agregar diálogos y mostrar mensajes en pantalla.

También podrás cambiar los “disfraces” de los personajes (la forma que adoptan en ese momento), que aparezcan o se oculten en la pantalla y modificar su tamaño.



### Panel de Sonido.

Este panel encontraras bloques que te dan la oportunidad de ponerles sonido a tus personajes y controlar su timbre, duración, tiempo, volumen, etc.



## Panel de Eventos.

Aquí encontrarás bloques que permiten iniciar la programación, detener el programa o ejecutar las acciones que quieras cuando “algo” ocurra; ese “algo” es lo que llamaremos evento.



## Panel de Control.

Los bloques que hallarás en este panel, permiten que una secuencia de sentencias se ejecute o se repitan de acuerdo al objetivo que desees.

Aquí se encuentran las estructuras para tomar decisiones como son “si” y “si-no”, estos dos bloques son condicionales para ejecutar o frenar una acción.

También hay estructuras de repetición como: “repetir”, “por siempre”, “por siempre si”, “repetir hasta”.

Con estas instrucciones, provocamos que el programa se detenga o continúe hasta que una condición ocurra.



## Panel de Variables.

Estos bloques sirven para guardar valores que cambian a medida que avanza el programa: puntajes, variables de control, niveles de juego, características de tus personajes, mundo, vidas; en fin, lo que se te ocurra.



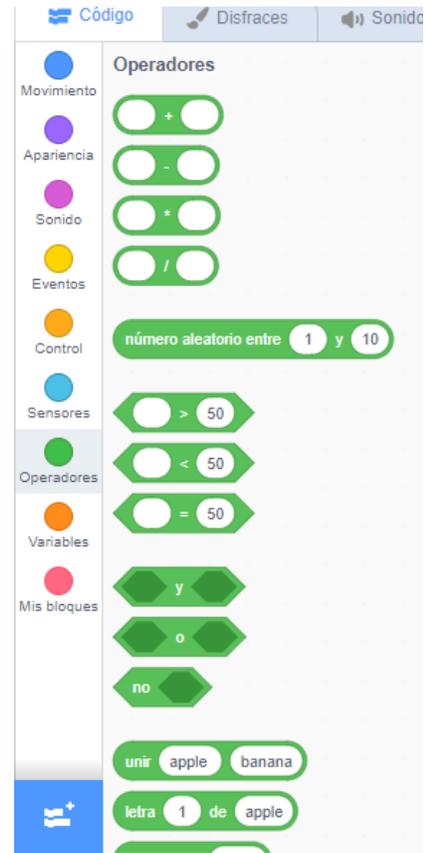
## Panel de Sensores.

Con los bloques de este panel, podrás detectar señales de lo que ocurre con tus objetos y reaccionar frente a ellas.



## Panel de Operadores.

Estos bloques permiten realizar operaciones aritméticas, lógicas y relacionales. También permiten manejar números de forma aleatoria (al azar), como cuando lanzas dados o una moneda, unir textos (concatenar cadenas), conocer su longitud, etc.



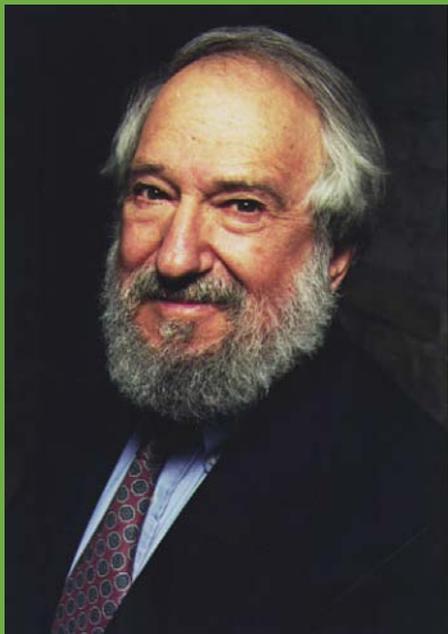
Para elaborar un programa, lo primero que debes hacer es seleccionar un escenario y tus personajes u objetos, luego colocar el puntero sobre el objeto que deseas programar su comportamiento, y después arrastrar los bloques que se encuentran en la parte izquierda superior para que haga lo que desees: seleccionar los diferentes disfraces que pueden tener, los diferentes sonidos o tonos y cómo reaccionar ante lo que ocurra. No te preocupes por ahora si no te imaginas cómo hacerlo, lo iremos viendo paso a paso con ejemplos específicos.

Grace Murray Hopper (Nueva York, USA, 1906-1992) fue una científica de la computación y militar estadounidense con grado de contraalmirante. Pionera en el mundo de las Ciencias de la Computación y la primera programadora que utilizó "Mark I", la primera computadora electromecánica, fabricada por IBM.

En 1952 desarrolló el primer compilador para un lenguaje de programación llamado A-0; poco antes de que John Backus desarrollara el primer lenguaje de programación de alto nivel "FORTRAN" en 1954.

Popularizó la idea de una máquina independiente de los lenguajes de programación y, en 1955, crea el primer lenguaje de alto nivel orientado al ámbito de gestión: "FLOW-MATIC", lo que derivó posteriormente en el desarrollo de COBOL.

COBOL fue creado en 1959 con el objetivo de tener un lenguaje de programación universal que pudiera ser usado en cualquier computadora.



Seymour Papert (Pretoria, Sudáfrica, 1928-2016): Destacado matemático y educador. Fue un pionero de la Inteligencia Artificial, inventor del Lenguaje de Programación LOGO en 1968. Seymour Papert trabajó con el psicólogo educativo Jean Piaget en la Universidad de Ginebra, desde 1959 hasta 1963.

En 1963 fue invitado a unirse al Instituto Tecnológico de Massachusetts (MIT) donde, junto con Marvin Minsky, fundó un Instituto de Inteligencia Artificial.

Basándose en los trabajos sobre constructivismo de Jean Piaget, desarrolla una visión del aprendizaje llamada construccionismo. Aplica la teoría de Piaget para desarrollar un lenguaje de programación de computadoras llamado LOGO; que es una potente herramienta para el desarrollo de los procesos de pensamiento lógico-matemáticos en niños. Para ello, ideó un "robot" virtual, llamado la "tortuga", que permitía a los alumnos resolver problemas.

## 6. Música en vivo.

### Aprendizajes esperados.

Habilidades	Medio	Contenido	Finalidad
Conoce un entorno que facilita el aprendizaje de la programación.	Por medio de un IDE: Scratch.	Introducción. Objetos. Escenario. Crea un instrumento. Crea una cantante.	Adquirir habilidades de programación básicas para crear programas simples.

## 6. Música en vivo.

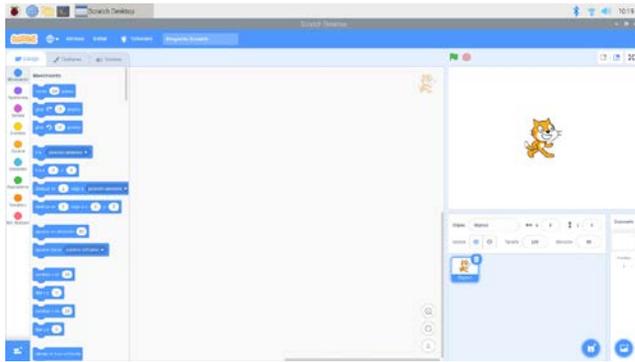
### Introducción

¡En esta práctica aprenderás a codificar tus propios instrumentos musicales y sonidos!

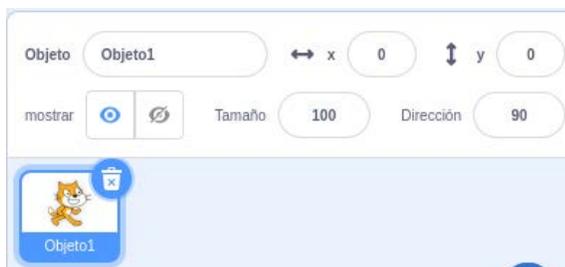
Bueno no solo eso, comenzaras por elegir escenarios y objetos y darles apariencia y comportamiento. Comencemos.

### Elige un objeto.

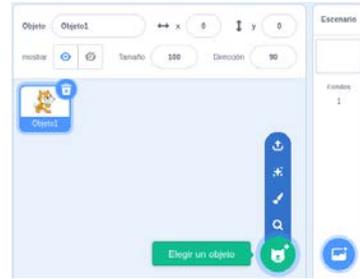
Para empezar elige un “objeto” para animarlo. Abre el editor de Scratch. Observa la parte inferior derecha de tu espacio de trabajo.



El gato que aparece en el área de escenario es la mascota de Scratch. Deshazte de él haciendo clic en el botón derecho, luego clic en “borrar” o también puedes hacer clic en el bote de “basura” que se activa al seleccionar el objeto.

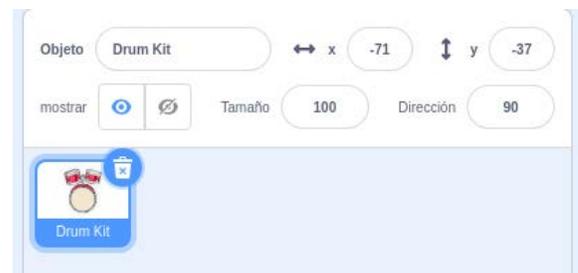


Ahora, haz clic en “Elige un objeto”.



Explora hasta encontrar el objeto de un tambor y selecciónalo.

Haz clic en “tamaño”, y cambia el tamaño del tambor para hacerlo más pequeño, a tu gusto.



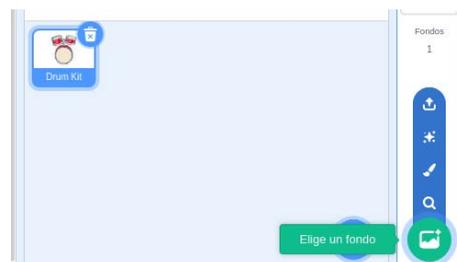
### Selecciona un escenario.

Ahora es momento de crear un escenario. Un escenario es donde tu proyecto cobra vida. ¡Como en un escenario real!

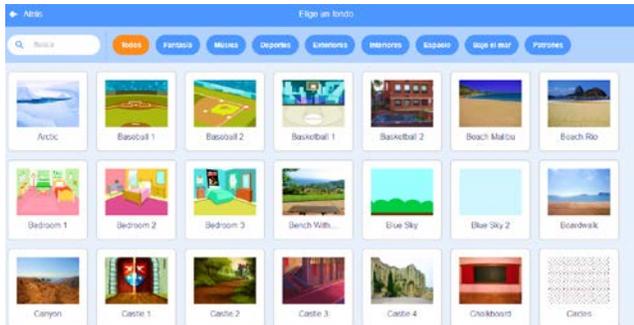
A la derecha de los objetos, en esa misma área de trabajo, encontrarás la herramienta “Escenario”.



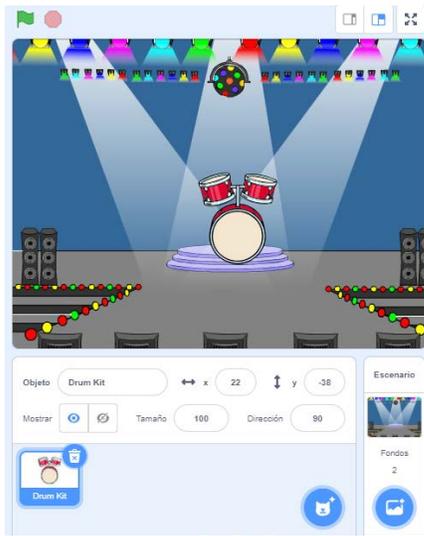
Ve a “Escenario” y selecciona un fondo para tu programa.



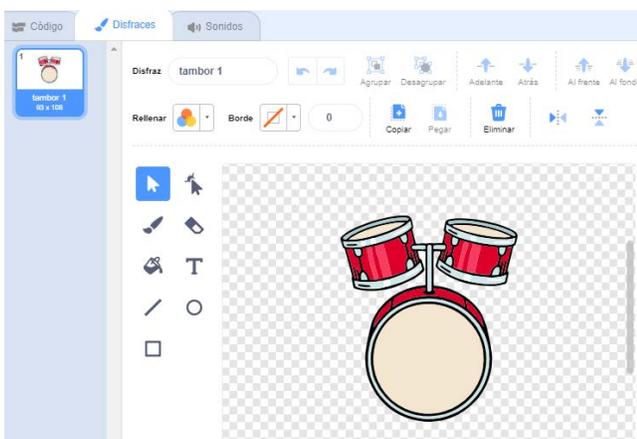
Para añadir un fondo al escenario, haz clic en **Elige un fondo** y explora hasta encontrar el de tu agrado. Por ejemplo: una sala de teatro, de conciertos o una discoteca. Usa tu creatividad para elegir.



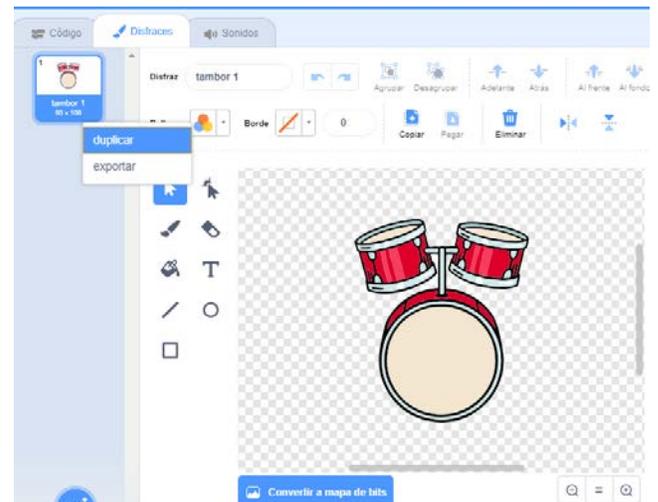
El escenario debería de verse como la siguiente figura.



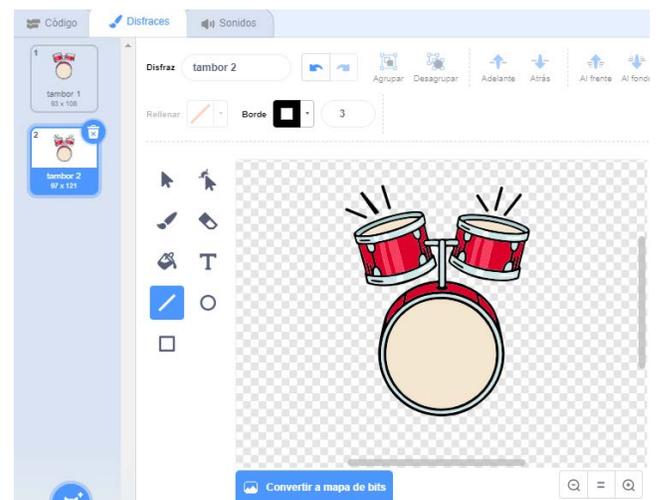
También puedes cambiar el aspecto de tu tambor cada que le des clic; esto se logra generando un nuevo disfraz. Haz clic en la pestaña **Disfraces** y verás la imagen del tambor.



Para crear una nueva animación da clic derecho en el tambor y selecciona **duplicar**.



Haz clic en el nuevo disfraz (que se llama "tambor 2"). Selecciona la herramienta de línea y dibuja líneas para que parezca que el tambor está sonando.

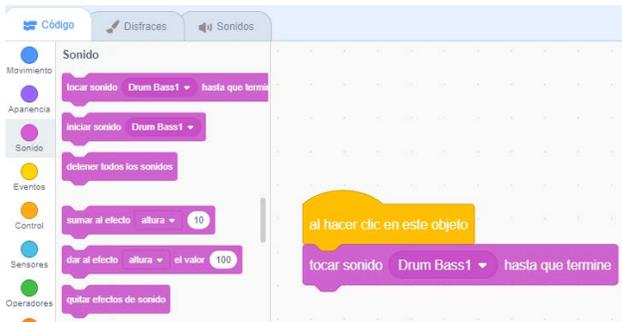


Ahora que ya tienes 2 disfraces diferentes para el tambor, puedes elegir qué disfraz mostrar.

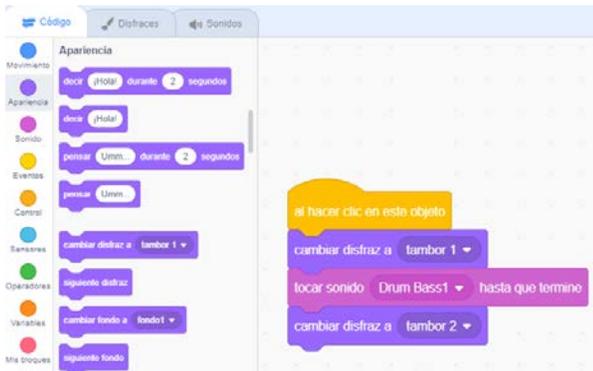
### Añade sonido

Cada que presiones el tambor debe emitir un sonido; para eso utiliza los bloques de código que se encuentran en la sección de "Sonido".

Haz clic en el objeto tambor y luego arrastra los dos bloques que aparecen en la imagen siguiente. Asegúrate de que estén conectados juntos (como los bloques de lego):



Cada que toques tu tambor deberá cambiar de apariencia también. Para esto, añade los siguientes bloques a tu tambor.



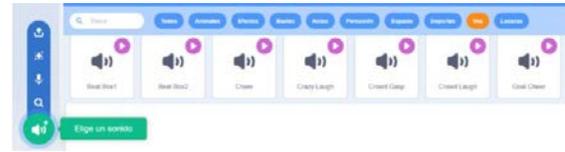
El bloque de código para cambiar el disfraz se encuentra en la sección "Apariencia".

## Añade una cantante

Ahora añade otros dos objetos a tu escenario: una cantante y un micrófono.



Para que tu personaje cante, necesitarás asignarle un sonido a tu objeto. Haz clic en la pestaña "Sonidos" y haz clic en "Selecciona un sonido de la biblioteca":



Ahora que has añadido sonido, puedes agregar este código tu cantante.



Prueba que tu personaje cante cuando le des clic y habrás terminado esta práctica.

## ¡Reto!

¿Puedes cambiar el sonido que hace tu tambor cuando haces clic sobre él?

¿Puedes generar un sonido en el tambor cuando se presiona la barra espaciadora?

En caso de que no puedas solucionarlo, utiliza el siguiente código para cumplir el reto.



## ¡Reto!

¿Puedes animar a tu cantante?



¿Quieres crear tu propia banda?

Para ello, puedes usar los objetos existentes o dibujar los tuyos para colocar distintos instrumentos con diferentes sonidos.

También, si tienes un micrófono puedes grabar tus propios sonidos.



El límite es tu imaginación, usa lo que has aprendido en este proyecto para crear tu propio conjunto musical, colócale todos los instrumentos que te imagines. Explora los diferentes sonidos e instrumentos disponibles y agrega los que se te ocurran.

John George Kemeny (Budapest, Hungría, 1926-1992) fue un informático y profesor nacido en Budapest (Hungría) y Thomas Eugene Kurtz es un científico de la computación estadounidense nacido en 1928.

Kemeny y Kurtz fueron pioneros en el uso de los ordenadores para la gente fuera del área de ciencias de la computación o con conocimientos básicos. Después de experimentos con el LGP-30, inventaron el conocido lenguaje de programación BASIC en 1964, con el fin de evitar la complejidad inherente de otros lenguajes y diseñado para usuarios con pocos conocimientos. También crearon uno de los primeros sistemas de tiempo compartido del mundo, el Sistema de Tiempo-Compartido de Dartmouth.

El BASIC original, el Dartmouth BASIC, fue diseñado como un medio para facilitar la programación en ordenadores a estudiantes y profesores que no fueran de Ciencias de la Computación.



## 7. ¡Atrapa las Criaturas!

### Aprendizajes esperados.

Habilidades	Medio	Contenido	Finalidad
Comprende la necesidad de pausas entre las acciones. Utiliza bloques para generar números aleatorios en Scratch. Añade una variable para almacenar una puntuación de juego en Scratch.	Por medio de un IDE: Scratch.	Añade criaturas y escenario. Criaturas cambiantes (animar varias criaturas). Atrapa una criatura. Añadir puntuación. Añadir un cronómetro.	Conocer la programación básica para crear y guardar variables en un programa

# 7. ¡Atrapa las Criaturas!

## Introducción

Este será tu primer juego de video. Las reglas serán muy simples: cada vez que atrapes una criatura ganarás puntos.



## Añade criaturas y escenario.

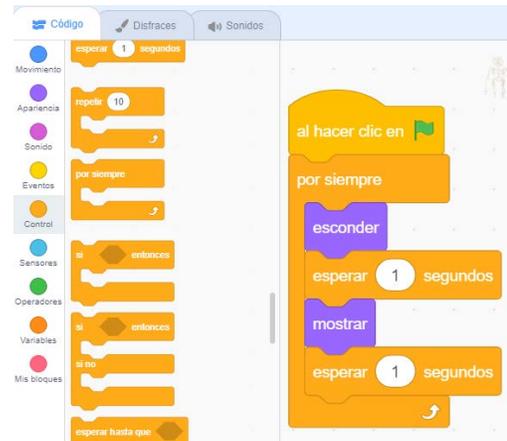
Crema un nuevo proyecto de Scratch. Añade varias criaturas y un escenario acorde.



## ¡Reto!

Intenta hacer que tu criatura aparezca y desaparezca

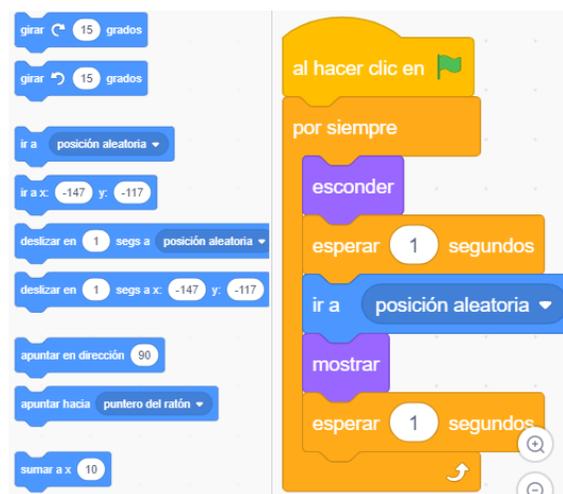
Si no puedes cumplir el reto copia el siguiente código.



## Criaturas cambiantes.

No se ve muy interesante el juego todavía ¿verdad? No te preocupes, apenas es el inicio. En este punto, tus criaturas serian muy fáciles de atrapar porque no se mueven. En lugar de quedarse en la misma posición, puedes hacer que Scratch elija sus coordenadas al azar y así dar la ilusión de que se están moviendo.

Para lograrlo, busca los bloques necesarios y cambia el código de una de tus criaturas, para que se vea como éste.



Prueba con tu criatura. Debes ver cómo cambia de posición. Repite este proceso con todas las demás criaturas.

### ¡Reto!

Haz que tu criatura espere un intervalo de tiempo al azar antes de aparecer.

¿Crees que se pueda usar el bloque “fijar tamaño” para hacer que el tamaño de tu criatura también cambie al azar cada vez que aparezca? ¡Inténtalo!

### Atrapa una criatura

Ahora, haz que el jugador intente atrapar a la criatura. Para tu juego, atrapar a una criatura significa que logras dar clic con el cursor del ratón sobre la criatura.

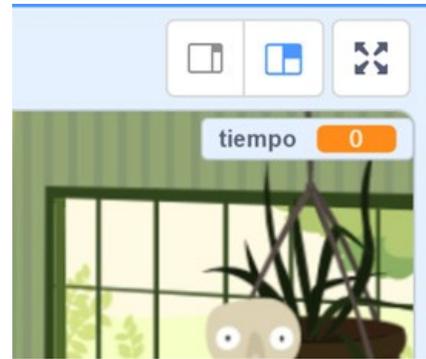
### ¡Reto!

¿Puedes hacer que tu criatura desaparezca cuando es atrapada?

Si no te imaginas como hacerlo, te lo mostramos en el siguiente código, el cual debe ser agregado a cada una de las criaturas.



Eso debe ser suficiente para que las criaturas desaparezcan al momento de dar clic sobre ellas de forma individual. Si por el tamaño de las criaturas te resulta difícil capturarlas, puedes cambiar su tamaño o jugar tu juego en modo de pantalla completa haciendo clic en el botón que se muestra a continuación:



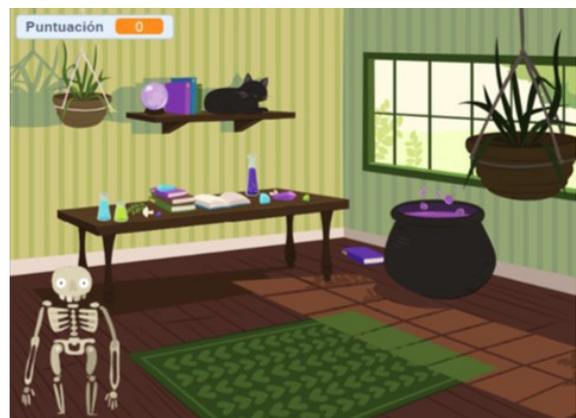
### Añade puntuación

Para añadir una puntuación, es necesario crear una variable a la que llamarás “puntuación”.

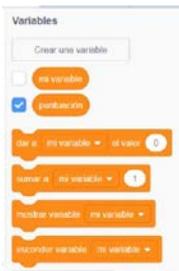
Para ello, en el área de “Variables” presiona “Crear una variable” y captura su nombre en el recuadro que está debajo de “Nombre de la variable” en el diálogo “Nueva variable”.



Después de creada, podrás ver la variable Puntuación y su valor en la parte superior izquierda del escenario.



Con tu variable creada, aparecerán nuevos bloques que podrás utilizar para cambiar el valor de la variable:



Aprovecha estos bloques para definir el comportamiento de tu variable “puntuación”, de la siguiente manera:

- 1.- cada vez que hagas clic en una criatura, la puntuación debe aumentar. Es decir, aumentar en “1” el valor de la variable
- 2.- Cuando des clic la bandera, la puntuación debe reiniciarse. Es decir, regresar a 0 el valor de la variable.

Los siguientes grupos de bloques deberían ser suficientes para lograrlo



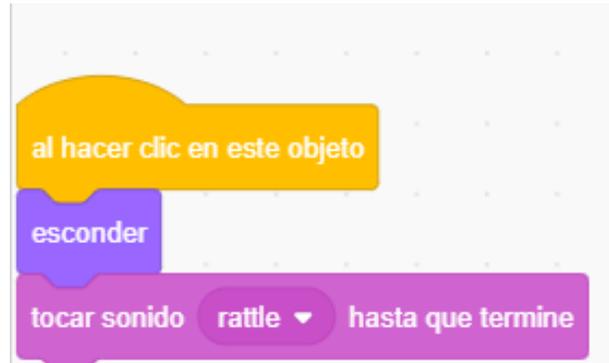
El último grupo de bloques debe repetirse en todas las criaturas.

Ejecuta tu programa de nuevo y atrapa algunas criaturas. ¿Cambió tu puntuación?

## ¡Reto!

¿Puedes hacer que aparezca un sonido cada vez que atrapas una criatura? ¡Inténtalo!

Si no lo lograste, intenta utilizar el código que se muestra a continuación:



## Añade un cronómetro

Ahora intenta añadir un temporizador a tu juego para que tu jugador sólo tenga 10 segundos para atrapar tantas criaturas como le sea posible.

Tu temporizador debería:

Comenzar en 10 segundos y contar hacia atrás cada segundo.

El juego debe detenerse cuando el temporizador llegue a 0.

Para lograrlo, crea una nueva variable llamada “tiempo” y agrégale el siguiente código:



Pídele a un/a amigo/a que pruebe tu juego.

¿Cuántos puntos puede conseguir?

Si tu juego es demasiado fácil, puedes:

- Darle menos tiempo al jugador
- Hacer que las criaturas desaparezcan con mayor frecuencia
- Hacer las criaturas más pequeñas

Modifica y prueba tu juego varias veces hasta que creas que tiene el nivel de dificultad adecuado

### ¡Reto!

Prueba con añadir otros objetos a tu juego

Cuando agregues un objeto nuevo, necesitarás tener en cuenta lo siguiente:

- ¿Cuál será su tamaño?
- ¿Qué tan frecuentemente debe aparecer?
- ¿Qué aspecto tendrá y qué sonido emitirá cuando lo atrapen?

¡Sigue explorando las posibilidades de tu juego y diviértete!





Joan Elisabeth Lowther Murray, nacida como Joan Elisabeth Lowther Clarke (Londres, 1917-1996) fue una criptoanalista y numismática británica que trabajó en Bletchley Park durante la Segunda Guerra Mundial.

Fue la única mujer que trabajó en el equipo del matemático Alan Turing en el proyecto Enigma, que descifró las comunicaciones secretas de la Alemania nazi. Su papel en este proceso le valió premios y reconocimientos, como el nombramiento de miembro de la Orden del Imperio Británico (MBE), en 1946.

En 1936, ganó una beca para estudiar en el Newnham College, Cambridge, donde obtuvo una doble titulación en matemáticas. Se le negó un título completo, ya que Cambridge sólo se lo otorgó a los hombres hasta 1948.

Las habilidades matemáticas de Clarke fueron descubiertas por primera vez por Gordon Welchman, en una clase de Geometría en Cambridge. Después de darse cuenta de las habilidades matemáticas de Clarke, la reclutó para que se uniera a él y formara parte de la Government Code and Cypher School (GC&CS).

La GC&CS comenzó en 1939 con un único propósito, romper el Código Enigma alemán. Enigma era una máquina que los alemanes inventaron para ocultar sus mensajes; y se creía firmemente que su cifrado era irrompible.

## 8. Conociendo el espacio

### Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Usa un ciclo de repetición para animar un objeto en Scratch. Usa un ciclo infinito para repetir la animación indefinidamente. Entiende que los ciclos pueden estar dentro de otros ciclos.	Por medio de un IDE: Scratch.	Anima una imagen. Utiliza ciclos. Gato flotante. Un asteroide que rebota Una estrella brillante Cambio de disfraz.	Escribir programas cortos que contengan: secuenciación, repetición, variables y selección.

# Conociendo el Espacio

## Introducción

En esta práctica vas a aprender a programar una animación. Para ello aprenderás a utilizar y combinar ciclos de repetición par manipular a tu personaje.

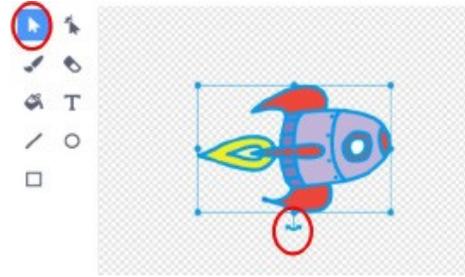
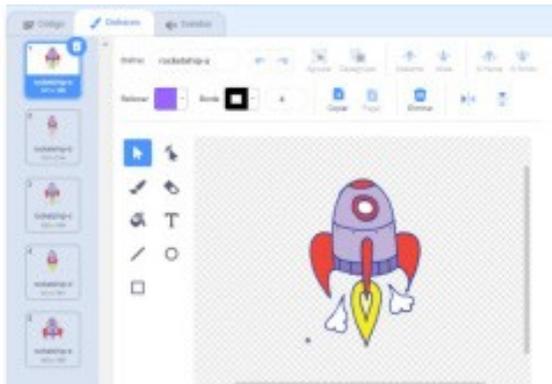
### Anima una nave espacial

En esta sección construirás una nave espacial que vuele hacia un planeta.

Crema un nuevo proyecto en Scratch. Borra el objeto gato para que tu proyecto esté vacío. Añade los objetos "Rocketship" (Nave espacial) y "Button1" (botón) al escenario. También tendrás que añadir el fondo "Stars" (Estrellas) a tu escenario.



Haz clic en el nuevo objeto de nave espacial, y selecciona la pestaña "Disfraces".



Para comenzar, agrega el siguiente código a tu nave espacial.



Cambia los números en los bloques para que el código sea exactamente el mismo que en la imagen de arriba.

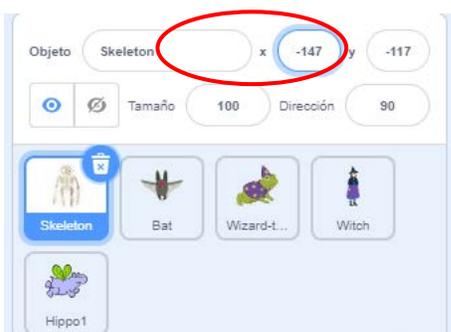
Si haces clic en la bandera verde, la nave espacial, girará y se deslizará hacia el centro del escenario.



Explora modificando los números del código para ver qué efecto tiene en tu programa y cómo se comporta tu nave espacial.

**Nota:** La posición de la pantalla x:(0) y:(0) es el centro del escenario. Una posición como x:(-150) y:(-150) está hacia la parte inferior izquierda del escenario, y una posición como x:(150) y: (150) está cerca de la parte superior derecha.

Si necesitas saber las coordenadas de una posición en el escenario, mueve el ratón a la posición que deseas saber y apunta las coordenadas que aparecen por debajo del escenario.



### ¡Reto!

Cambia los números en el código de tu animación, para que la nave espacial se mueva hasta tocar el planeta.

### ¡Reto!

Haz que la nave espacial se mueva más lentamente hacia el centro del escenario.

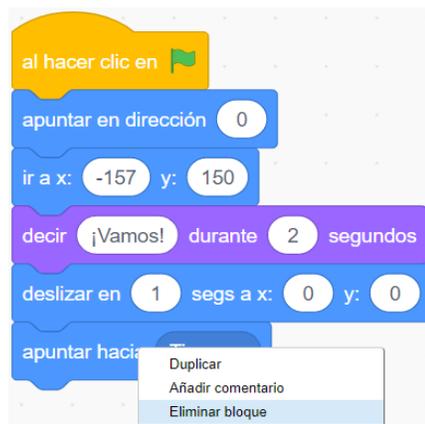
**Pista:** Observa los números en este bloque:



### Anima utilizando ciclos

Otra forma de animar la nave espacial es ordenarle que se mueva una pequeña cantidad muchas veces.

Veamos cómo hacerlo. Elimina el bloque "deslizar" de tu código. Haz clic derecho sobre él y clic en eliminar. También puedes eliminar el código arrastrándolo fuera del área de programas y devolviéndolo al área de bloques de código.



### ¡Reto!

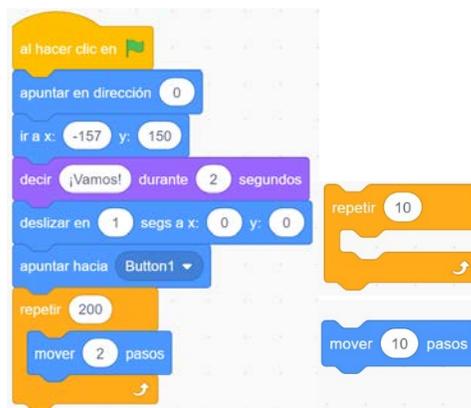
¿Puedes usar un bloque de repetición para mover tu nave espacial hacia el planeta?

¡Inténtalo! Tu nave espacial debería moverse hacia el planeta exactamente como antes; pero esta vez usando un bloque de repetición.

En lugar de deslizarse, tu nave espacial se debería mover repetidamente unos pocos pasos a la vez.

Aquí están los bloques que necesitas.

Este código te servirá para animar tu nave espacial.



Tu nave espacial podría cambiar de color mientras se mueve. Este es el código que deberás utilizar:

```
dar al efecto color el valor 25
```

### Un asteroide que rebota

Agrega una roca espacial flotante a tu animación. Para ello, agrega la figura “rock” (roca) a tu animación.



Cuando hagas clic en la bandera verde, la figura de la roca debería moverse y rebotar alrededor del escenario. Usa el siguiente código para que lo haga:

```
girar 15 grados
apuntar hacia Button1
```

```
al hacer clic en
  apuntar hacia Button1
  por siempre
    mover 2 pasos
    si toca un borde, rebotar
```

### ¡Reto!

¿Puedes hacer que tu nave espacial se haga más pequeña a medida que llega a la Tierra?

Prueba y guarda. Tu nave espacial debería hacerse más pequeña mientras se mueve.

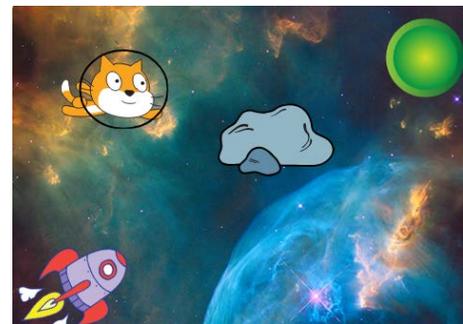
Tu nave espacial debería comenzar con un tamaño del 100%

Aquí tienes el código que incluye cambiar el tamaño de tu nave espacial mientras se mueve:

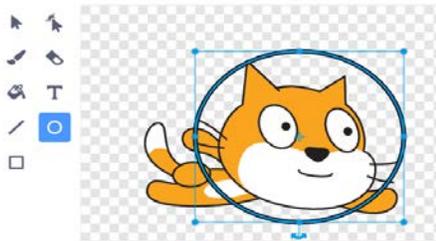
```
al hacer clic en
  fijar tamaño al 100 %
  apuntar en dirección 0
  ir a x: -157 y: 150
  decir ¡Vamos! durante 2 segundos
  deslizar en 1 segs a x: 0 y: 0
  apuntar hacia Button1
  repetir 200
    mover 2 pasos
    dar al efecto color el valor 25
    cambiar tamaño por -5
```

### Gato flotante

Ahora haz que la mascota de Scratch aparezca flotando.



Prueba añadiendo un gato. Si haces clic en tu nueva figura y luego haces clic en Disfraces, puedes editar su aspecto. Haz clic en la herramienta de elipse y dibuja un casco espacial alrededor de la cabeza del gato.



Cuando se hace clic en la bandera verde, la figura debería girar en círculos para siempre.

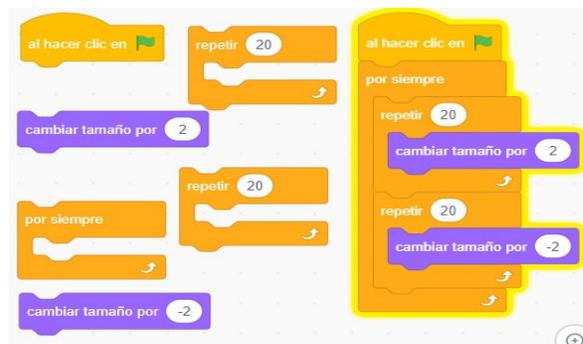
Aquí tienes el código.



## Una estrella brillante.

Finalmente, combina dos ciclos dentro de otro para hacer una estrella brillante. Agrega una figura "Star" (estrella) a tu animación.

Añade este código a tu estrella.



## ¡Reto!

Detén tu animación espacial, guárdala e inicia un nuevo proyecto en Scratch. Usa lo que has aprendido para hacer tu propia animación.



Tōru Iwatani (Meguro, Tokio; 25 de enero de 1955) es un diseñador de videojuegos japonés en la década de 1980. Creó uno de los videojuegos más populares de todos los tiempos: "Pac-Man".

Se unió a la compañía de software Namco en 1977, donde comenzó su carrera en el negocio de los videojuegos. En ese entonces, trabajó con la idea de crear un juego llamado "Puck-Man" y en 1980, junto con el programador Shigeo Funaki y otros tres empleados de la compañía Namco, terminó de dar forma al juego. Fue mostrado al mercado japonés el 10 de mayo de ese año, donde se convirtió en un verdadero éxito. Este suceso acaparó la atención del fabricante de videojuegos Midway, quién compró los derechos para comercializar el juego en los Estados Unidos con el nombre de "Pac-Man".

Debido a su innovador concepto y a la continuidad y aceptación que el juego tuvo en todo el mundo, el "Pac-Man" es considerado uno de los videojuegos más clásicos y conocidos de todos los tiempos.





## 9. Rompecabezas de una Célula

### Aprendizajes esperados

Habilidades	Medio	Contenido	Finalidad
Emplea los conocimientos aprendidos en Scratch. Crea un rompecabezas en forma de célula. Refuerza sus conocimientos en biología para identificar partes principales de la célula.	Por medio del IDE: Scratch.	Aprende a programar mediante un juego con conceptos científicos y las herramientas previamente adquiridas en las lecciones anteriores.	Identificar las partes de la célula y sus funciones principales.



## 9. Rompecabezas de una célula

### Introducción

En esta lección vas a crear un juego, con el cual aprenderás a ubicar las principales partes de una célula.

### Crea tu proyecto

Crea un proyecto nuevo en Scratch, cámbiale el nombre y borra el objeto que aparece

### Nota:

Debes tener es tu Raspberry una carpeta llamada célula, son los elementos que te servirán para que realices tu juego. Si no es así, solicita la carpeta con los objetos necesarios para realizar la práctica.

### Agrega los objetos

Agrega los 19 objetos de la carpeta “Célula”.



### Agrega el fondo

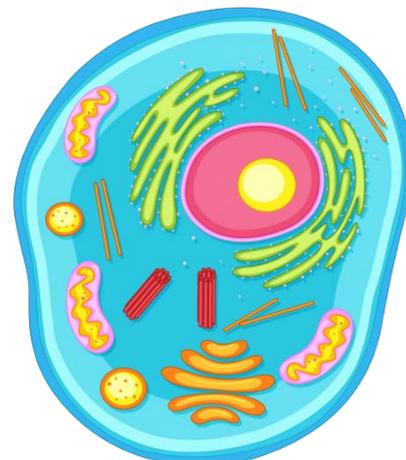
Agrega el fondo que viene en la carpeta, como se muestra en la figura.



Agrega un objeto más. Este objeto será “Abby”, la encargada de dar la bienvenida y las instrucciones del juego.



Deberás acomodar todos los objetos encima del fondo, de modo que todos queden en su lugar. Al final, la célula lucirá de esta manera.



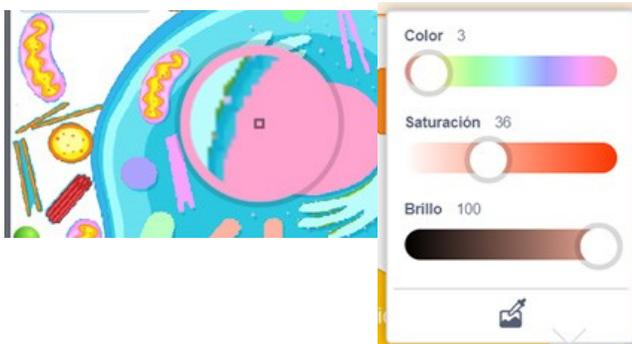
## Aprende a programar la célula

Selecciona el primer objeto (el que quieras). Debes hacer que cuando lo coloques en su lugar, se quede fijo.

Para ello, el color del fondo nos dirá si el objeto está sobre el lugar correcto. Se necesitan los siguientes bloques para conseguirlo:



Dentro del condicional “si” está el sensor que permitirá saber qué color del fondo está tocando tu objeto. Si observas con cuidado la imagen del fondo, notarás que para cada parte hay una zona con un color distinto. Al dar clic en el color de tu sensor, se abrirá una nueva ventana para cambiar los colores. Utiliza la opción del gotero, y aparecerá una lupa. Selecciona el color de la parte de la célula que corresponda.



## Programa la posición

Una vez que has elegido el color que le corresponde, tienes que programar cómo se comporta su posición.

Cuando el objeto toque el color que le asignaste, tienes que decirle que se coloque en la posición exacta que le toca. Esto es para que no quede desalineado. Para saber la posición de la figura debes ir al área de trabajo



Busca y copia las coordenadas que desees que ocupe y cambia el bloque de posición.

Agrega un objeto que de la bienvenida y las instrucciones del juego. Esto ya lo aprendiste a hacer, hazlo a tu gusto.

Al iniciar tu programa, debes ocultar las partes de la célula, hasta que comience el juego.

Usa algo como en los siguientes bloques:



También necesitarás un marcador. Para ello, crea una variable “Puntos”, y súmale uno cada vez que coloques correctamente un objeto en su lugar. Eso también ya lo aprendiste a hacer; pero como ejemplo, tu código podría verse de esta manera:



## Inicio del juego

Al comenzar todo, tendrás que hacer varias cosas:

Cada que inicie el juego, debes hacer que se esconda el marcador ("Puntos") y además regresarlo a cero.

Después debes mostrar a la presentadora con un fondo distinto al de la célula.

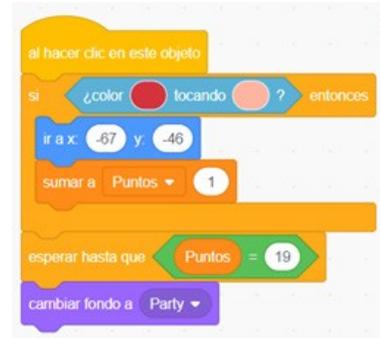
La presentadora debe dar un mensaje de bienvenida a los participantes y después debe desaparecer para que inicie el juego con un fondo nuevo y con el puntaje en cero.



## Fin del juego

Cuando el rompecabezas esté listo, y hayas completado los 19 puntos, el fondo debe cambiar para felicitar al jugador.

Para lograrlo debes agregar un fondo nuevo.



Cuando aparezca el mensaje de felicitación, será necesario esconder los demás objetos. Para eso agrega el siguiente código:



## Reto

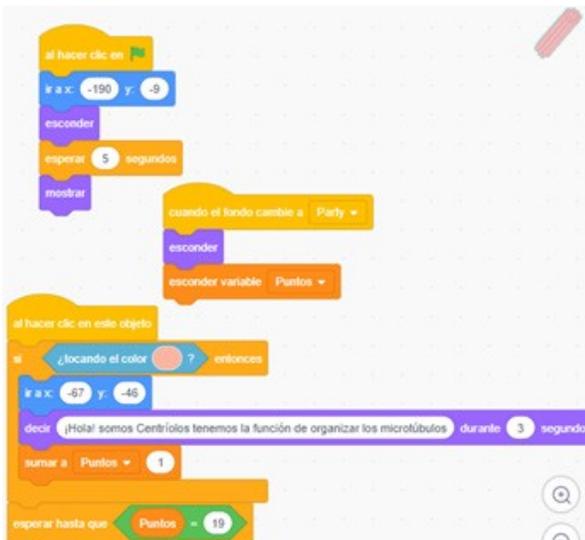
¿Puedes hacer que las partes de la célula te digan cuál es su función principal una vez que las hayas acomodado?

¡Inténtalo! Tu código podría incluir algo como lo siguiente para conseguirlo:



Ahora sólo copia el código que generaste para tu objeto, en las otras partes de la célula. Cambiando la posición y el color que debe tocar cada parte de la célula.

Tu código se debe ver muy similar al siguiente:





GOBIERNO DE LA  
CIUDAD DE MÉXICO

SECRETARÍA DE EDUCACIÓN, CIENCIA,  
TECNOLOGÍA E INNOVACIÓN



Raspberry Pi

**AEFCM**

AUTORIDAD EDUCATIVA  
FEDERAL EN LA CIUDAD DE MÉXICO



Raspbian



python™

scratch

Derechos Reservados: Secretaría de Educación, Ciencia, Tecnología e Innovación de la Ciudad de México.

Este material forma parte de una iniciativa que pretende generar en niños y jóvenes, de una manera lúdica, el gusto por la programación y el desarrollo de tecnología con base en herramientas abiertas y de bajo costo. Estamos convencidos de que esta estrategia les brindará, como agradable efecto secundario, una estructura de pensamiento lógico que les preparará para desarrollarse en el campo de las ciencias, la matemática y la ingeniería.

Algunas de las prácticas de este documento fueron inspiradas en la red de clubes de código de la “Fundación Raspberry Pi” y en la iniciativa “Programo Ergo Sum”.

<https://projects.raspberrypi.org/en>

<https://www.programoergosum.es>