Programa con TurtleArt

Resumen de la lección

Público objetivo: *10-16 años. Nivel principiante.*

Duración: *90 minutos*

Objetivos de aprendizaje:
Tus estudiantes participarán en una actividad práctica y atractiva en la que aplicarán sus habilidades de programación para dibujar figuras geométricas creadas con bloques de código. Mediante el uso de ejemplos y su modificación, tus estudiantes aprenderán a crear sus propias figuras. El objetivo de esta actividad es ampliar su comprensión de la programación básica y la generación de resultados gráficos.
El objetivo es ayudar a los alumnos a desarrollar habilidades de pensamiento computacional mediante:

- El desarrollo de habilidades de programación para resolver problemas (algoritmos y codificación).
- La descomposición del problema en partes manejables (descomposición de problemas).
- El diseño y la aplicación de algoritmos para procesar datos (diseño algorítmico y automatización de procesos).

Online u offline: online

Pensamiento Computacional:

- **Fundamentos del PC**:
    - Descomposición → dividir un problema en partes más pequeñas
    - Reconocimiento de patrones → buscar similitudes o patrones dentro de esas partes más pequeñas que puedan ayudar a resolver el problema.
    - Abstracción → distinguir los aspectos fundamentales

Materiales

- Ordenador.
- Internet.
- Enlace: playfulinvention.com/webturtleart

Preparación

1. **Divide a tus estudiantes en pequeños grupos de 2 o 3 personas.** Si los grupos son más grandes, puede resultar difícil que todos interactúen.
2. **Proporciona un ordenador a cada grupo.**

Descripción de la lección - Programa con TurtleArt

Introducción al Pensamiento Computacional (10 minutos)

Introducción al Pensamiento Computacional (10 minutos)

Pregunta a tus estudiantes:

- ¿Qué sabes sobre cómo funcionan los ordenadores y los teléfonos?
- ¿Pueden pensar por sí mismos? (¿Por qué sí o por qué no?)
- ¿Quién controla lo que hace un ordenador?
- ¿Son creativos los ordenadores?
- ¿Pueden los ordenadores resolver problemas?

Explica a tus estudiantes que van a trabajar en el Pensamiento Computacional. En pocas palabras, esto implica aprender a hacer que un ordenador resuelva un problema por ti. No se trata solo de programar, sino también, por ejemplo, de aprender a descomponer un problema en partes o a reconocer patrones para poder resolverlo mejor.

Hay cuatro pilares fundamentales del Pensamiento Computacional:

- Descomposición - dividir un problema en partes más pequeñas
- Reconocimiento de patrones - buscar similitudes o patrones dentro de esas partes más pequeñas que puedan ayudarte a resolver el problema.
- Abstracción - distinguir entre los aspectos principales y secundarios. ¿Qué es realmente importante para resolver el problema?
- Algoritmos - elaborar instrucciones paso a paso para resolver el problema. En esta lección se introducirá el reconocimiento de patrones

En esta lección te introduciremos al reconocimiento de patrones.

Actividad principal – Programación con TurtleArt

Paso 1 (20 minutos)

- Breve introducción a la interfaz de TurtleArt (ver Apéndice 1).
- Copia el código del Ejemplo 1 (ver Apéndice 2).
- Ejecuta el programa.
- ¿Qué ocurre si cambias los números de los bloques rosas?
- ¿Puedes dibujar una simple cruz en su lugar?
- ¿Puedes cambiar el color?
- Explica cómo funciona a un compañero/a.

Paso 2 (10 minutos)

- Copia el código del Ejemplo 2 (ver Apéndice 2)
- Ejecuta el programa
- ¿Qué pasa si cambias los números de los bloques rosas?
- Explica cómo funciona a una compañera/o.

Paso 3 (10 minutos)

- Copia el código del Ejemplo 3 (ver Apéndice 2).
- Ejecuta el programa.
- ¿Qué pasa si cambias los números de los bloques rosas?
- Explica cómo funciona a un compañero/a.

Paso 4 (10 minutos)

- ¿Puedes hacer un cuadrado?
- ¿Puedes hacer un triángulo?
- ¿Puedes hacer una espiral cuadrada (avanzado)?

- Explica cómo funciona a una compañera/o.

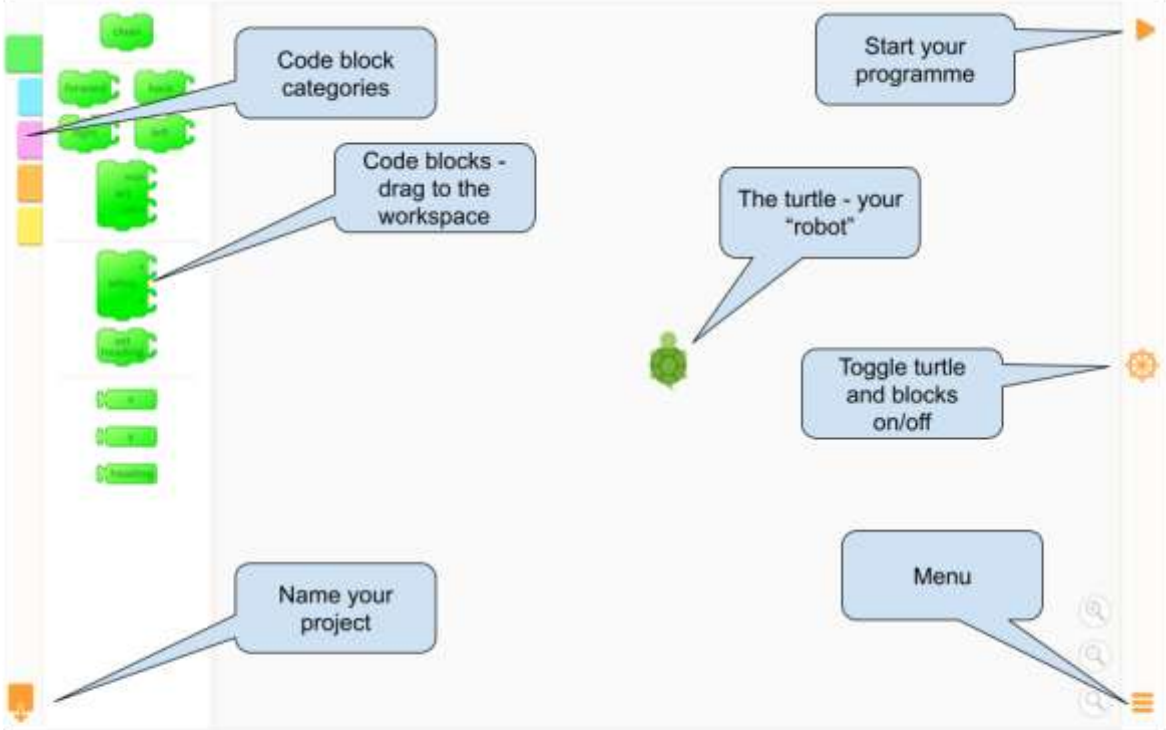
Paso 5 (10 minutos)

- Make your own work of art.
- Explain how it works for a fellow student.

Reflexión y evaluación (5 minutos)

- ¿Qué hemos aprendido?
- ¿Te ha resultado difícil entender cómo funcionaba el código?
- ¿Te ha resultado difícil crear tu propia obra de arte?
- ¿Dónde podemos aplicar lo que hemos aprendido?
- Etc.

Apéndice 1 - Explicación de la interfaz

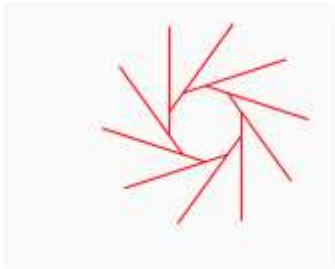


Apéndice 2 - Ejemplos básicos

Ejemplo 1

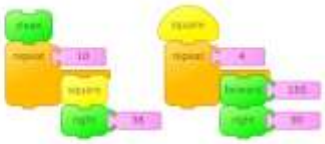
forward 200 draws a line. Then **back 200** retraces that line and brings you back to the starting point.
Repeat the line 10 times with **right 36** in between. This makes a star with the lines evenly spaced. The total amount of turning is 10*36=360 degrees, the number of degrees in a circle.

Ejemplo 2

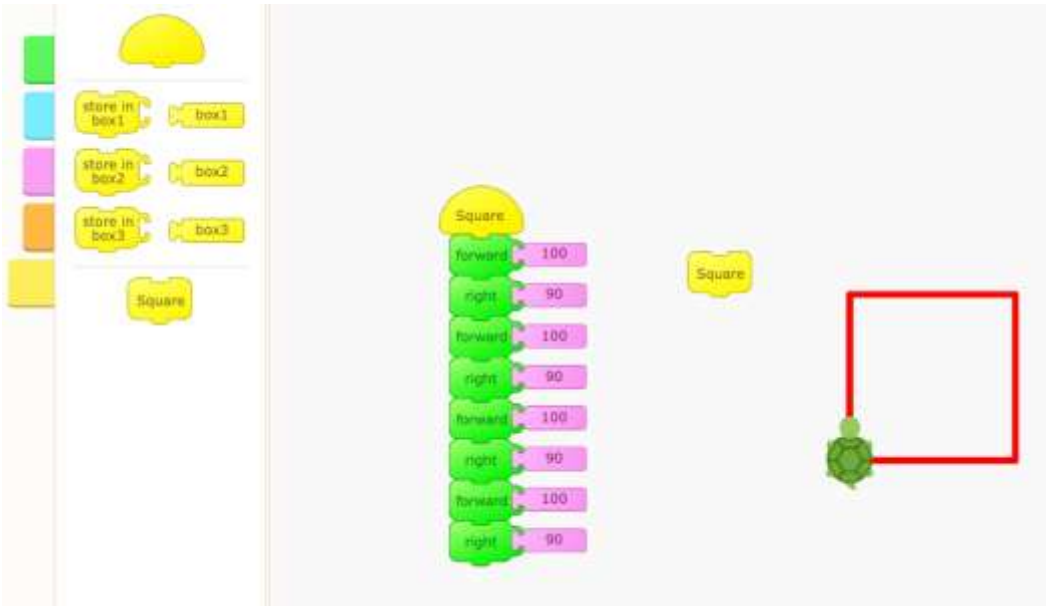


forward 250 draws a line. **back 200** retraces most of it. The Turtle doesn't quite get back to its starting point. Each line starts at a different position than the previous one.

Ejemplo 3

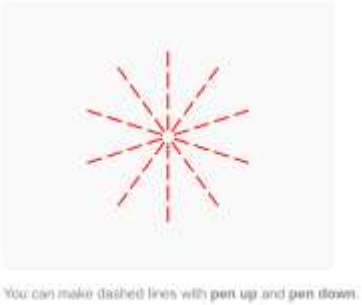


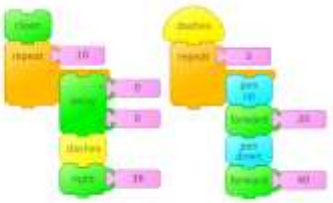
You can name stacks. Pull out an empty hat block (the round yellow one). Click on it and type in a name. A new block to use that stack will appear in the blocks palette. In this sample, the **square** block isn't there when you start. It only appears after you have pulled out a hat and named it **square**.



Puedes escribir directamente en el bloque de texto conectado al bloque amarillo Show (que puede parecerse a una seta). Esto te permite mostrar tu texto personalizado en la pantalla cuando se ejecuta el programa. Simplemente haz clic en el bloque de texto y escribe tu mensaje.

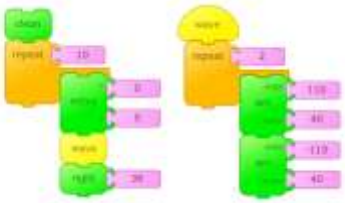
Exemplo 4





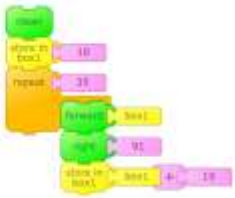
Apéndice 2: Ejemplos avanzados

An arc is a part of a circle. There are two inputs to the **arc** block: radius and angle. The radius is the size of the circle. The angle is how much of the circle to draw. An angle of 360 draws the whole circle.

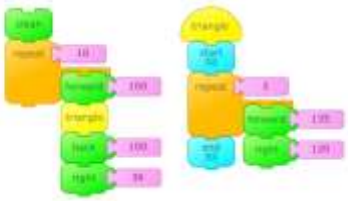


Once you have an interesting element you can "spin" the element with a **repeat** and a **right**. In this case, the element is a double wave with each of the waves made with two **arc** blocks.

store in box1 lets you save a number. **box1** recalls that number. You can use the boxes to make images that have repetition with variation.



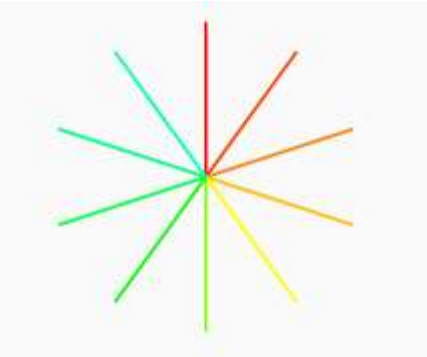
box1 can be used to create a succession of lines of increasing length. Put an angle between these lines and you get a square spiral.

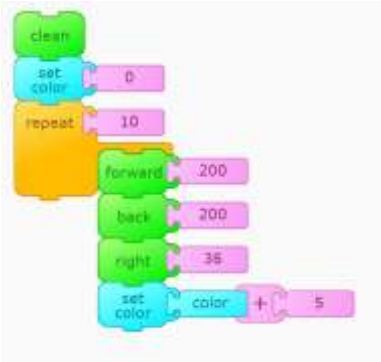


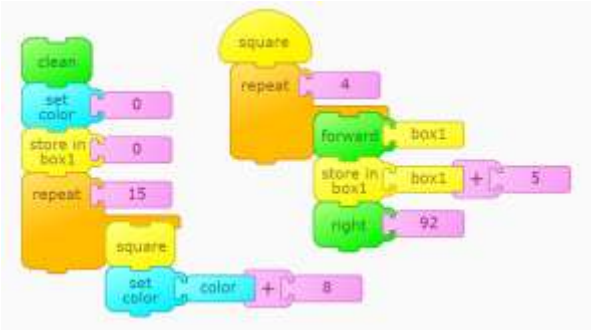
You can fill areas with **start fill** and **end fill**. The blocks between **start fill** and **end fill** define the area to be filled.

Cada vez que se ejecuta el programa, utiliza un **número aleatorio** para decidir dónde deben colocarse algunos de los puntos. Por eso, la imagen **tendrá un aspecto ligeramente diferente cada vez,** aunque el código sea el mismo.

Es importante saber esto, porque tus estudiantes podrían pensar que han cometido un error si su resultado no coincide exactamente con el ejemplo. Pero la variación es **intencionada**: forma parte del funcionamiento de la aleatoriedad en la programación.

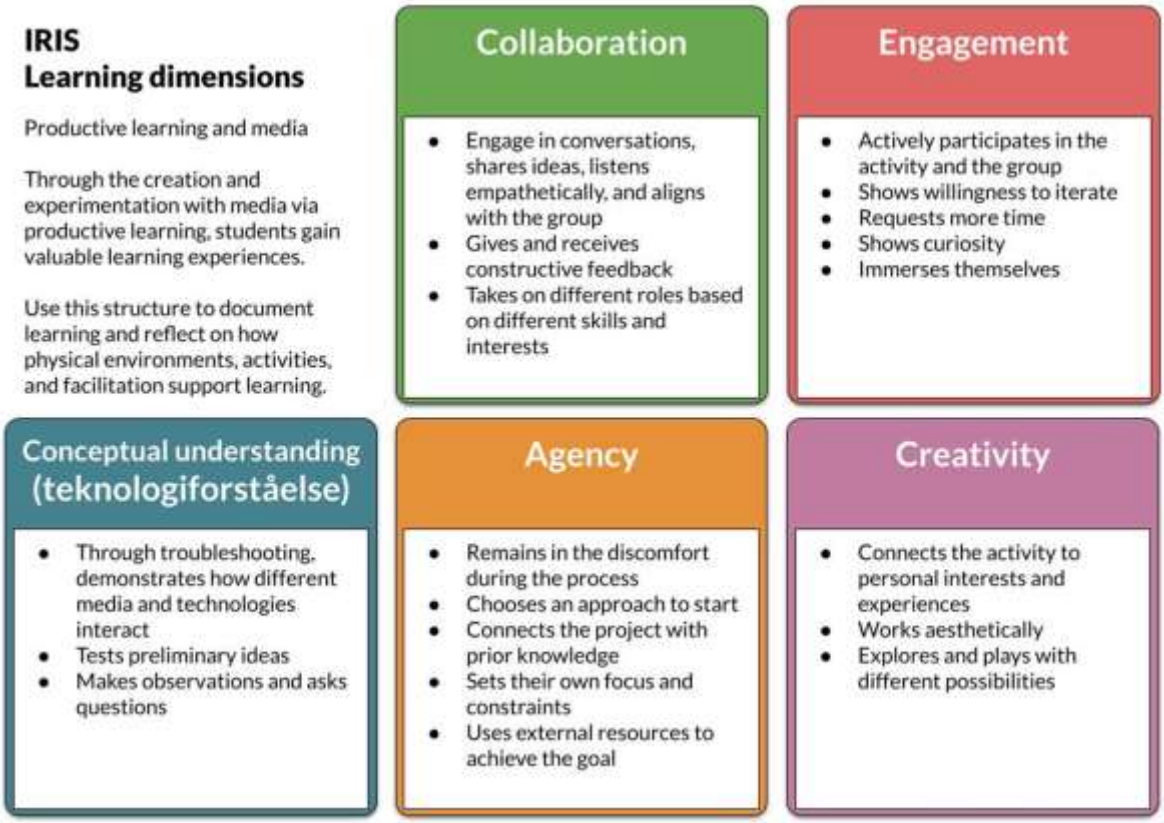






Apéndice 4 - Dimensiones del Aprendizaje

Las Dimensiones del Aprendizaje se desarrollaron en The Thinkering Studio del Exploratorium de San Francisco, EE. UU., y nuestro trabajo en torno a las reflexiones se inspira profundamente en su labor.



¿Qué queremos que los participantes obtengan de nuestras actividades, y cómo nos aseguramos de que así sea?

La calidad en la enseñanza es un tema complejo y, en ocasiones, controvertido. Por ello, utilizamos las **Dimensiones de Aprendizaje,** una herramienta dinámica desarrollada por el Exploratorium de San Francisco.

Utilizamos las Dimensiones de Aprendizaje para documentar el aprendizaje y evaluar si logramos proporcionar a los participantes una experiencia valiosa y significativa a través de nuestras actividades.

Nuestro punto de partida

Cuando desarrollamos y evaluamos una actividad, siempre nos preguntamos: **¿Qué queremos aprender o descubrir a través de esta actividad?**

Para garantizar un proceso exhaustivo y matizado, trabajamos en tres fases:

• **Reunión previa**
Antes de la actividad, celebramos una reunión para determinar en qué dimensiones de aprendizaje y en qué (dos)

• **Verde, amarillo y rojo**
Durante la actividad, hay un observador presente que toma notas basándose en las dimensiones e indicadores acordados.
Inmediatamente después de la actividad, nos reunimos para una breve sesión de reflexión, en la que evaluamos la actividad utilizando un modelo de rojo, amarillo y verde. ¿Qué funcionó bien, qué funcionó menos bien y qué no funcionó en absoluto?

• **Reunión posterior**
La semana siguiente, nos reunimos con la persona o personas que observaron la actividad para realizar un análisis más profundo de cómo se reflejaron las dimensiones de aprendizaje en la actividad.

Un marco de evaluación flexible

Evaluamos nuestras actividades utilizando una serie de dimensiones de aprendizaje e indicadores asociados. Las áreas de enfoque dependen de la actividad concreta y, por lo tanto, varían de vez en cuando. Sin embargo, también comparamos las dimensiones de aprendizaje entre diferentes actividades para identificar patrones: si ciertas dimensiones se están descuidando, necesitan actualizarse o podrían mejorarse.

Las dimensiones de aprendizaje son cruciales para evaluar los resultados de los participantes en el taller, pero también para obtener información sobre la amplitud, la profundidad y el nivel de nuestras actividades en general.

Una herramienta para la mejora continua

El objetivo de trabajar con las dimensiones de aprendizaje no es alcanzar una respuesta definitiva, sino garantizar un desarrollo continuo. Nuestro enfoque consiste en **«estar en el camino correcto»**, más que en alcanzar un objetivo fijo y predefinido. Por lo tanto, la herramienta **no** es un dispositivo de medición, sino que debe seguir siendo una ayuda dinámica que apoye a los educadores en la creación de experiencias de aprendizaje inspiradoras y atractivas.