

The background image shows two children, a girl with curly blonde hair and a boy with dark hair, sitting at a table in a workshop. They are focused on assembling a small robot. The table is covered with various electronic components, including a breadboard, wires, and small motors. In the foreground, there are more components like a servo motor, a small black sensor module, and several colored wires (green, red, black) connected to a circuit. The workshop has shelves in the background with various containers and materials.

# **Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM**

**Julio Antonio Encalada Cuenca**  
**Rosemary de Lourdes Samaniego Ocampo**  
**Luis Adrián Luzuriaga Pesantes**



© **Julio Antonio Encalada Cuenca**  
**Rosemary de Lourdes Samaniego Ocampo**  
**Luis Adrián Luzuriaga Pesantes**

Primera edición, 2025-11-02

**ISBN: 978-9942-53-000-4**

**DOI: <http://doi.org/10.48190/9789942530004>**

Distribución online

 Acceso abierto

#### **Cita**

Encalada, J., Samaniego, R., Luzuriaga, L. (2025) Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM. Editorial Grupo Compás

Este libro es parte de la colección de la Univesidad Técnica de Machala y ha sido debidamente examinado y valorado en la modalidad doble par ciego con fin de garantizar la calidad de la publicación. El copyright estimula la creatividad, defiende la diversidad en el ámbito de las ideas y el conocimiento, promueve la libre expresión y favorece una cultura viva. Quedan rigurosamente prohibidas, bajo las sanciones en las leyes, la producción o almacenamiento total o parcial de la presente publicación, incluyendo el diseño de la portada, así como la transmisión de la misma por cualquiera de sus medios, tanto si es electrónico, como químico, mecánico, óptico, de grabación o bien de fotocopia, sin la autorización de los titulares del copyright.

## **Prefacio**

### **Introducción**

Según Palioura y Sapounidis (2025), la robótica educativa se ha convertido en una herramienta fundamental en la enseñanza de la tecnología y la ciencia, permitiendo a los niños desarrollar habilidades de pensamiento lógico, creatividad y resolución de problemas desde una edad temprana. Este libro está diseñado para servir como una guía práctica para futuros docentes de segundo año de Educación General Básica que deseen introducir a los niños en el uso de servomotores dentro de proyectos de robótica educativa y de esta manera los niños puedan desarrollar su pensamiento computacional mientras ensamblan y programan los prototipos robóticos basados en estos actuadores rotacionales.

Garnica y Ramos (2023) señalan que en los últimos años, la enseñanza del pensamiento computacional y robótica educativa se trabaja de manera innovadora usando metodología STEAM (Ciencia, Tecnología, Ingeniería, Artes y Matemática). La robótica educativa, combinada con la metodología STEAM, se utiliza para desarrollar el pensamiento computacional al integrar actividades prácticas que involucran Ciencia, Tecnología, Ingeniería, Arte y Matemáticas (Ouyang & Xu, 2024). Los estudiantes aprenden a programar robots, resolver problemas y diseñar soluciones innovadoras, fomentando habilidades de análisis y lógica.

Esta metodología promueve la creatividad, el trabajo en equipo y el pensamiento crítico, permitiendo a los niños aplicar conceptos matemáticos y científicos de manera práctica y significativa (Correia et al., 2024). Y les permitirá a futuros docentes de segundo año de EGB integrar la robótica educativa con la metodología STEAM como una

herramienta poderosa para desarrollar el pensamiento computacional en los niños. Al combinar ciencia, tecnología, ingeniería, arte y matemáticas, los estudiantes no solo adquieren conocimientos técnicos, sino que también desarrollan habilidades críticas como la creatividad, la resolución de problemas y el trabajo en equipo, preparando a los futuros ciudadanos para enfrentar desafíos del mundo real de manera innovadora y efectiva.

## **Objetivo general**

El propósito principal de este libro es proporcionar a los futuros docentes de segundo año de EGB una metodología clara y accesible para el desarrollo del pensamiento computacional involucrando el uso de servomotores con robótica educativa. A través de explicaciones sencillas y actividades prácticas, se busca facilitar el aprendizaje de conceptos básicos de electrónica y programación, adaptados a la comprensión y capacidades cognitivas de los niños en esta etapa de desarrollo que corresponde al segundo año de EGB. Este libro se basa en la metodología STEAM (Ciencia, Tecnología, Ingeniería, Arte y Matemáticas), promoviendo un aprendizaje interdisciplinario que fomenta la creatividad, el pensamiento crítico y la resolución de problemas de manera integral.



## **Estructura**

- El capítulo 1 aborda aspectos teóricos sobre el pensamiento computacional, la programación, la metodología STEAM, y la relación que existe entre ellas. Se generaliza, pero también se hace énfasis de forma particular a la programación de un servomotor, haciendo girar la fundamentación teórica alrededor de este componente electrónico.
- El capítulo 2 aborda la organización de la programación con servomotores, la importancia y funcionamiento de las variables en programación con este tipo de motores de control angular, el rol de la programación de procedimientos y funciones en la resolución de problemas con servomotores, y las sentencias de control secuenciales con estos actuadores rotacionales. El abordaje se lo realiza considerando tres áreas específicas de la metodología STEAM: Tecnología, Ingeniería y Matemática.
- El capítulo 3 aborda aspectos relacionados a las cinco áreas específicas de la metodología STEAM: Ciencia, Tecnología Ingeniería, Arte y Matemáticas. Se desarrollan proyectos interdisciplinarios con servomotor por medio de secuencias didácticas poniendo en práctica todo el conocimiento aprendido en los capítulos anteriores.

## **Características pedagógicas del libro**

Este libro ha sido diseñado teniendo en cuenta principios pedagógicos que favorecen el aprendizaje significativo y el desarrollo de habilidades en los niños. Algunas de sus características incluyen:

- **Enfoque Constructivista:** Se parte del principio de que el aprendizaje es más significativo cuando los estudiantes participan activamente en la construcción de su propio conocimiento. Por esta razón, el libro prioriza actividades prácticas y proyectos que incentivan la exploración, la programación y la manipulación de objetos como servomotores. En esta línea, Nyamekye et al. (2023) argumentan que el constructivismo favorece el aprendizaje de los estudiantes.
- **Aprendizaje Basado en Proyectos (ABP):** Las propuestas de actividades prácticas basadas en circuitos electrónicos de servomotores que se plantean en este libro permiten a los estudiantes poner en acción los conceptos aprendidos en situaciones reales empleando metodología STEAM. A través de la resolución de retos específicos, los niños no solo fortalecen sus conocimientos técnicos, sino que también potencian habilidades clave como la creatividad, el pensamiento crítico y la cooperación en equipo (Huang et al., 2023).
- **Enfoque Educativo STEAM:** Los proyectos de robótica educativa que se elaboren a medida que se avanza en los capítulos adoptan el enfoque STEAM, lo que permitirá a los estudiantes explorar y aplicar conceptos de Ciencia, Tecnología, Ingeniería, Arte y Matemáticas de manera práctica y colaborativa (Spyropoulou & Kameas, 2024).

Este libro ofrece a los futuros docentes de segundo año de EGB los conocimientos fundamentales para enseñar y

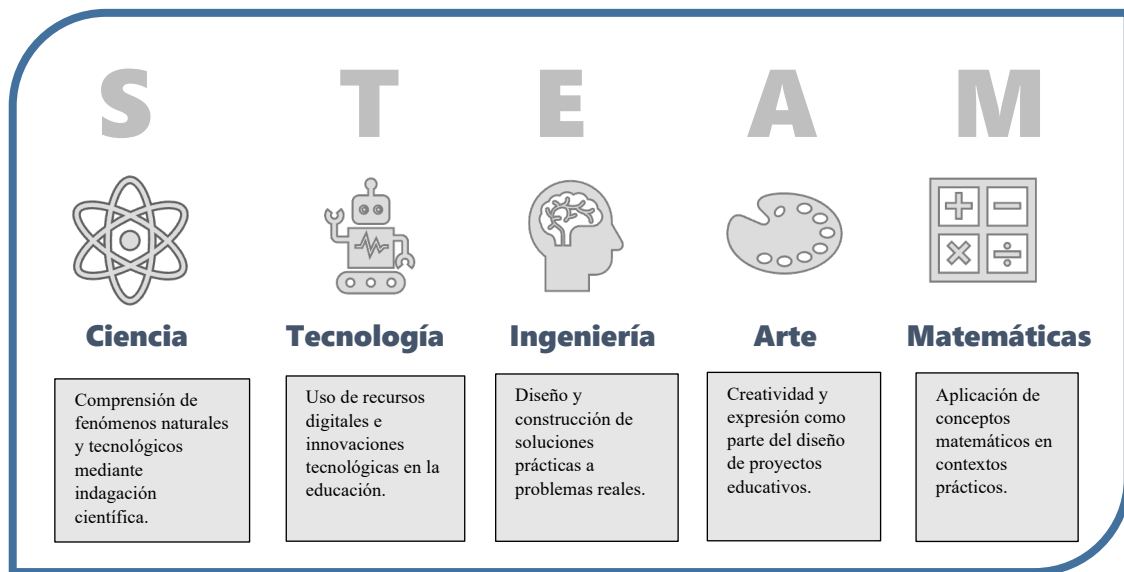
aprender pensamiento computacional a través de la robótica educativa basada en circuitos electrónicos basados en servomotores, utilizando la metodología STEAM. Proporciona información esencial, propone actividades didácticas y presenta recursos básicos para su aplicación. Asimismo, permite al futuro docente ajustar y enriquecer la metodología de acuerdo con sus necesidades y el contexto en el que se desempeñe.

Arafat et al. (2024) argumentan que la Robótica Educativa, combinada con la metodología STEAM, facilita que los niños adquieran habilidades clave como el pensamiento crítico, la creatividad y la capacidad para resolver problemas. A través de la integración de Ciencia, Tecnología, Ingeniería, Arte y Matemáticas, este enfoque promueve un aprendizaje activo, estimulando la curiosidad y motivando a los estudiantes a explorar conceptos de manera práctica. Además, los conecta con situaciones reales, fortaleciendo su comprensión y aplicación de los conocimientos.

Siguiendo la metodología STEAM, en los capítulos 2 y 3 de este libro se aborda de manera pedagógica las distintas áreas de Ciencia, Tecnología, Ingeniería, Artes y Matemáticas (Filipe et al., 2024). En la figura 1 se muestra un resumen visual de los aprendizajes que desarrollará el futuro docente en cada una de las áreas STEAM para su aplicación en la enseñanza a los niños. A lo largo de los capítulos, el gráfico se irá coloreando progresivamente, reflejando las áreas trabajadas.

**Figura 1**

*Gráfico de aprendizaje de las áreas STEAM*



*Nota:* En esta figura se observa las áreas de aprendizaje STEAM que el futuro docente debe lograr.

Este libro ofrece una guía detallada para la formación de futuros docentes de segundo año de Educación General Básica (EGB), proporcionando fundamentos teóricos y prácticos esenciales para desarrollar en los niños el pensamiento computacional. A través del uso de la robótica educativa basada en servomotores y la metodología STEAM, se presentan estrategias didácticas innovadoras que integran ciencia, tecnología, ingeniería, arte y matemáticas. El objetivo es fomentar en los estudiantes habilidades de resolución de problemas, creatividad, pensamiento lógico y colaboración, preparando a los niños para enfrentar los retos de la sociedad actual.

## Contenido

### **CAPÍTULO 1: EL PENSAMIENTO COMPUTACIONAL, LA PROGRAMACIÓN Y LA METODOLOGÍA STEAM USANDO SEVOMOTORES.....13**

- 1.1. Introducción: El pensamiento computacional en la programación de servomotores ..... 13
- 1.2. Definición de pensamiento computacional y el servomotor ..... 16
- 1.3. Definición de programación y el servomotor ..... 21
- 1.4. Definición de metodología STEAM y el servomotor ..... 22
- 1.5. Relación del pensamiento computacional, la programación y la metodología STEAM..... 24
- 1.6. Autoevaluación del capítulo ..... 26

### **CAPÍTULO 2: LA PROGRAMACIÓN EN LA RESOLUCIÓN DE PROBLEMAS SECUENCIALES CON SERVOMOTORES .....30**

- 2.1. Introducción: la educación del pensamiento computacional secuencial ..... 30
- 2.2. Organización de la programación de servomotores con mBlock..... 39
- 2.3. La importancia y funcionamiento de las variables en programación de servomotores con mBlock..... 41
- 2.4. El rol de la programación de procedimientos y funciones en la resolución de problemas con servomotores en mBlock ..... 42
- 2.5. Sentencias de control secuenciales para servomotores con mBlock..... 45
- 2.6. El hardware necesario para trabajar robótica educativa basada en servomotores con mBlock ..... 53

### **CAPÍTULO 3: PROYECTOS INTERDISCIPLINARIOS CON SERVOMOTORES .....73**

- 3.1. Introducción: la Educación con el Aprendizaje Basada en Proyectos..... 73
- 3.2. Aprendizaje Basado en Proyectos (ABP) con énfasis en servomotores ..... 78

3.3. Secuencia Didáctica 1: Un viaje al corazón del servomotor .....	81
3.4. Secuencia Didáctica 2: Circuitos con Arduino y servomotores.....	97
3.5. Secuencia Didáctica 3: Primeros pasos en programación con mBlock .....	105
3.6. Secuencia Didáctica 4: Diseño de un Graduador de Ángulos y Brazo Robótico con Servomotores .....	116
3.7 Secuencia Didáctica 5: Indicador de las Estaciones del Año .....	127
3.8. Autoevaluación del capítulo .....	137
<b>CRÉDITOS DE SOFTWARE Y RECURSOS VISUALES .....</b>	<b>142</b>
<b>REFERENCIAS BIBLIOGRÁFICAS .....</b>	<b>143</b>

## **CAPITULO 1: EL PENSAMIENTO COMPUTACIONAL, LA PROGRAMACIÓN Y LA METODOLOGÍA STEAM USANDO SEVOMOTORES**

### **1.1. Introducción: El pensamiento computacional en la programación de servomotores**

**Objetivo del capítulo:** Al finalizar este capítulo el futuro docente será capaz de fundamentar las definiciones de pensamiento computacional, programación y metodología STEAM, y la relación entre éstas considerando el servomotor como elemento central.

#### **Aprendizajes Esperados**

- Definición de pensamiento computacional y el servomotor
- Definición de programación y el servomotor
- Definición de metodología STEAM y el servomotor
- Relación del pensamiento computacional, la programación y la metodología STEAM

En la actualidad, es fundamental que los procesos de enseñanza y aprendizaje en la Educación General Básica incluyan experiencias que fomenten el desarrollo de habilidades del siglo XXI, como el pensamiento lógico, la creatividad y la resolución de problemas. El pensamiento computacional, en este sentido, se presenta como una competencia clave para que los estudiantes enfrenten desafíos de manera estructurada, utilizando estrategias como la descomposición, la identificación de patrones, la

abstracción y el diseño de algoritmos (Angeli & Giannakos, 2020). Cuando se trabaja con servomotores, esta habilidad cobra un sentido práctico, ya que los estudiantes pueden aplicar sus ideas en proyectos que implican controlar movimientos precisos y programar acciones con base en decisiones lógicas.

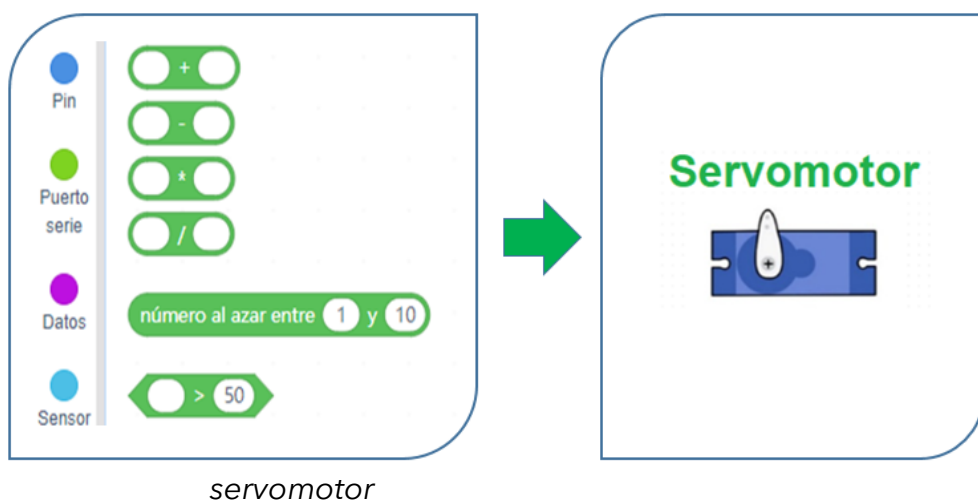
Por su parte, la programación es el medio que permite convertir esas ideas en instrucciones concretas que pueden ser comprendidas por una máquina o robot. A través de lenguajes de programación, ya sean visuales (por bloques) o textuales, los estudiantes aprenden a escribir códigos que le indican al servomotor cómo y cuándo moverse, en qué dirección y con qué intensidad. Esta interacción entre la lógica del pensamiento y la ejecución técnica fortalece no solo las competencias digitales, sino también la autonomía, la perseverancia y el pensamiento crítico de los estudiantes desde edades tempranas.

Un servomotor puede ser programado utilizando programación textual o programación por bloques, ambos enfoques relacionados con el pensamiento computacional. A través de estos métodos, los estudiantes desarrollan habilidades como la descomposición y el diseño de algoritmos, esenciales para estructurar soluciones paso a paso. Al programar un servomotor, los estudiantes deben crear secuencias lógicas de instrucciones que controlen su movimiento, aplicando así el pensamiento computacional para generar algoritmos eficientes, como se observa en la figura 2.



**Figura 2**

*Interfaz de programación con bloques de un*



*Nota.* En esta figura se observa un servomotor en un ambiente de programación con bloques. Fuente: Captura de pantalla adaptada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

Asimismo, la metodología STEAM (Ciencia, Tecnología, Ingeniería, Arte y Matemáticas) propone un enfoque educativo integral, que promueve el aprendizaje activo, interdisciplinario y basado en la resolución de problemas reales (Suslenco, 2024). La incorporación del servomotor en proyectos STEAM permite a los estudiantes experimentar con conceptos de física, matemática y diseño mientras construyen soluciones tecnológicas concretas. Por ejemplo, al desarrollar un prototipo de una puerta automática, los estudiantes no solo programan el movimiento del motor, sino que también reflexionan sobre el diseño, el

funcionamiento mecánico y su aplicación en la vida real, integrando el arte y la creatividad en el proceso.

Este capítulo explorará cómo el pensamiento computacional, la programación y la metodología STEAM no solo se relacionan entre sí, sino que se potencian mutuamente al integrarse en proyectos educativos con servomotores. A través del análisis teórico y la propuesta de actividades prácticas, se busca proporcionar a los futuros docentes una comprensión clara de estas herramientas y estrategias, para que puedan implementar experiencias de aprendizaje significativas, motivadoras e innovadoras en el aula. Así, se sientan las bases para una educación más contextualizada, colaborativa y alineada con las demandas tecnológicas y sociales del presente y del futuro.

## **1.2. Definición de pensamiento computacional y el servomotor**

Barragán Perea (2023) indica que el pensamiento computacional es una forma de pensar y resolver problemas que implica formular cuestiones de manera que una computadora o una persona pueda procesarlas de manera efectiva. Incluye conceptos como la descomposición, la abstracción, el reconocimiento de patrones y el diseño de algoritmos, y se considera una habilidad fundamental para todos, no solo para programadores o científicos informáticos.

El pensamiento computacional es la capacidad de abordar problemas de forma estructurada y lógica, aplicando métodos similares a los utilizados por los científicos de la computación (Weintrop et al., 2023). En el ámbito educativo, permite a los estudiantes desarrollar habilidades para

entender y resolver desafíos complejos mediante el uso de estrategias sistemáticas y creativas, aplicables en distintas áreas del conocimiento.

García (2020) caracteriza al pensamiento computacional como una habilidad esencial que permite resolver problemas de manera estructurada, aplicando estrategias como la descomposición, la identificación de patrones, la abstracción y el diseño de algoritmos. En el aula, esta capacidad puede desarrollarse a través de experiencias concretas, como el uso de servomotores en proyectos de robótica educativa. Por ejemplo, al enfrentar el reto de mover una barrera automática, los estudiantes deben dividir la tarea en pasos lógicos: conectar el servomotor, definir el ángulo de giro, programar la secuencia y probar el resultado. Este proceso los lleva a pensar de forma ordenada y precisa, fortaleciendo su capacidad para resolver problemas reales mediante soluciones tecnológicas.

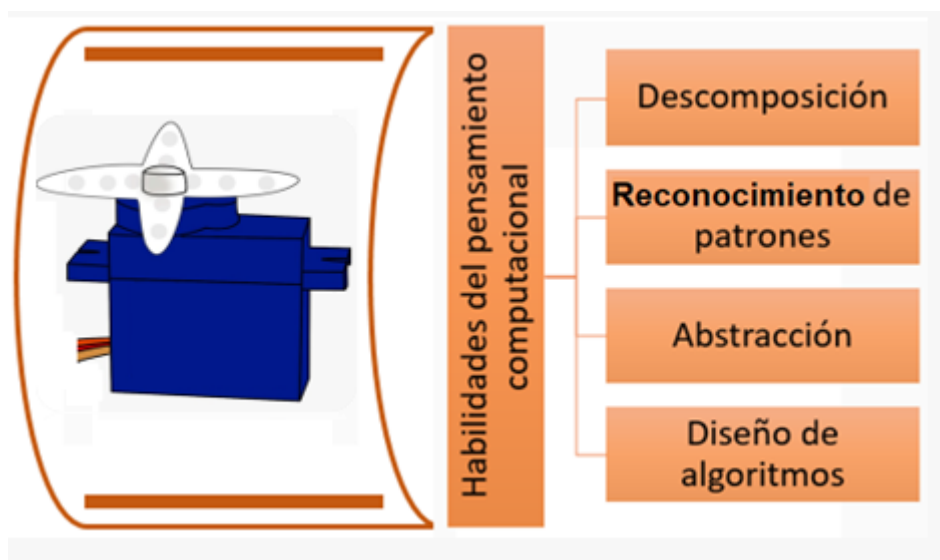
El trabajo con servomotores permite aplicar el pensamiento computacional de forma tangible, ya que los estudiantes deben crear algoritmos que controlen con exactitud los movimientos del motor: por ejemplo, girar 90° hacia un lado, esperar unos segundos, y volver a la posición inicial. Estas acciones no solo requieren lógica, sino también planificación, predicción y evaluación de resultados, lo que fomenta un aprendizaje activo y significativo. De esta manera, el pensamiento computacional se convierte en un puente entre la teoría y la práctica, ayudando a los estudiantes a comprender cómo una secuencia de instrucciones puede traducirse en un comportamiento físico observable dentro de un entorno tecnológico.

## **Habilidades que incluye el pensamiento computacional asociadas al servomotor**

El pensamiento computacional incluye y potencia el desarrollo de diversas habilidades cognitivas (ver figura 3): la descomposición de problemas, la identificación de patrones, la abstracción de información relevante, la creación de algoritmos (Çiftçi & Topçu, 2023). Estas habilidades facilitan la resolución de problemas complejos de forma estructurada, ingeniosa y efectiva, resultando aplicables tanto en el ámbito tecnológico como en diversas situaciones de la vida diaria. Estas habilidades se potencian especialmente cuando se integran servomotores en actividades de programación y robótica educativa. Al programar un servomotor para realizar un movimiento específico -como girar una compuerta, levantar un brazo robótico o desplazar un mecanismo en cierto ángulo- los estudiantes deben pensar en cada paso del proceso: desde la identificación del problema hasta la creación y evaluación del algoritmo. Esto implica descomponer el reto en tareas más pequeñas, reconocer patrones de movimiento, abstraer lo esencial para el control del motor y diseñar una secuencia lógica de instrucciones que permita ejecutar la acción con precisión. El trabajo con servomotores brinda así una experiencia tangible en la que el pensamiento computacional se transforma en una herramienta concreta para diseñar soluciones inteligentes.

### Figura 3

#### *Habilidades del pensamiento computacional*



*Nota.* En esta figura se observa las habilidades del pensamiento computacional.

**Descomposición:** La descomposición permite abordar un desafío complejo dividiéndolo en pasos más simples y comprensibles (Çoban & Korkmaz, 2021). Al trabajar con servomotores, esta habilidad se activa cuando los estudiantes deben hacer que un robot realice un movimiento específico, como girar un brazo o mover una compuerta. Para lograrlo, los niños deben identificar y separar las distintas acciones: conectar correctamente el servomotor, definir el ángulo de giro, programar el tiempo de activación y sincronizarlo con otras funciones del proyecto. Esta descomposición les ayuda a comprender cómo funciona cada parte del sistema y cómo se combinan para lograr una acción final coherente.

**Reconocimiento de patrones:** El reconocimiento de patrones consiste en detectar secuencias o comportamientos que se repiten (Kayhan et al., 2024). En el caso de los servomotores, esta habilidad se pone en práctica cuando los estudiantes observan cómo el motor se mueve de manera cíclica o repetitiva, por ejemplo, al abrir y cerrar una barrera en intervalos regulares o al realizar un mismo giro cada vez que se cumple una condición. Al identificar estas regularidades, los niños pueden programar con mayor precisión, anticipar los movimientos del robot y ajustar su funcionamiento según el patrón deseado, fortaleciendo así su capacidad de análisis y predicción.

**Abstracción:** La abstracción es la habilidad de enfocarse en los aspectos más relevantes de un problema, ignorando lo que no aporta al objetivo principal (Shin et al., 2022). En proyectos con servomotores, esta capacidad se desarrolla cuando los estudiantes comprenden que lo esencial es definir el ángulo o la dirección del movimiento, sin preocuparse por los componentes internos del motor. Por ejemplo, si un servomotor debe girar 90 grados para mover una palanca, los estudiantes abstraen el concepto de "giro preciso" y se concentran en programar únicamente lo necesario para que la acción ocurra, facilitando la toma de decisiones durante el diseño y la programación.

**Diseño de algoritmos:** El diseño de algoritmos implica construir una secuencia lógica de pasos para que un robot cumpla una tarea específica (Payne et al., 2022). Al utilizar servomotores, esta habilidad se aplica cuando los estudiantes programan instrucciones como: mover el motor a 45 grados, esperar dos segundos, luego volver a la posición inicial. Estas instrucciones deben estar ordenadas y claras para que el movimiento sea exacto. Al diseñar estos

algoritmos, los estudiantes aprenden a planificar acciones de manera lógica, conectar causa y efecto, y comprobar que los resultados coincidan con sus intenciones, fortaleciendo así su pensamiento estructurado y sistemático.

### **1.3. Definición de programación y el servomotor**

La programación es el proceso de escribir, diseñar, depurar y mantener el código fuente de programas informáticos, utilizando uno o más lenguajes de programación (Betancourt Almaguer y Peña Rodríguez, 2023). Su objetivo es crear instrucciones precisas que una computadora pueda interpretar y ejecutar para realizar tareas específicas. A través de la programación, se pueden desarrollar aplicaciones, sistemas operativos, videojuegos, inteligencia artificial y muchas otras herramientas tecnológicas que facilitan diversas actividades humanas.

La programación es el proceso de diseñar y escribir un conjunto de instrucciones que una computadora o dispositivo puede entender y ejecutar para realizar una tarea específica. En el ámbito educativo, especialmente en niveles iniciales, la programación puede abordarse a través de lenguajes visuales como los bloques o mediante código textual, permitiendo a los estudiantes construir secuencias lógicas para controlar el comportamiento de distintos elementos tecnológicos. Uno de estos elementos es el servomotor, un componente que convierte señales en movimientos precisos. Programarlo implica definir con claridad cómo debe moverse -por ejemplo, a qué ángulo, en qué momento y durante cuánto tiempo- lo cual refuerza en los estudiantes la necesidad de ser detallados y ordenados al construir instrucciones.

Al utilizar servomotores en proyectos escolares, la programación se convierte en una experiencia concreta y motivadora. Los estudiantes no solo escriben código, sino que también observan cómo sus instrucciones generan acciones físicas reales, como levantar una barrera, mover un brazo robótico o girar un mecanismo. Esta interacción directa entre lo digital y lo físico les permite comprender el valor de los algoritmos y de la precisión en la programación. Además, trabajar con servomotores fomenta el ensayo y error, el pensamiento lógico y la mejora continua, habilidades clave para el desarrollo de competencias tecnológicas desde edades tempranas.

#### **1.4. Definición de metodología STEAM y el servomotor**

Urgiles-Rodríguez et al. (2022) señalan que la metodología STEAM (por sus siglas en inglés: Science, Technology, Engineering, Arts and Mathematics) es un enfoque educativo interdisciplinario que integra la ciencia, la tecnología, la ingeniería, el arte y las matemáticas en experiencias de aprendizaje significativas. Su objetivo es fomentar en los estudiantes el pensamiento crítico, la creatividad, la resolución de problemas y el trabajo colaborativo, mediante proyectos que conectan conocimientos de distintas áreas para resolver desafíos del mundo real.

La metodología STEAM es un enfoque educativo que integra de manera interdisciplinaria cinco áreas fundamentales: Ciencia, Tecnología, Ingeniería, Arte y Matemáticas. Esta metodología propone que los estudiantes aprendan haciendo, mediante proyectos prácticos y significativos que promuevan la resolución de problemas reales, la creatividad, el pensamiento crítico y el trabajo colaborativo (Chen &



Ding, 2024). En este contexto, el uso de servomotores cobra especial relevancia, ya que permite a los estudiantes diseñar y construir mecanismos con partes móviles que requieren precisión y control, como brazos robóticos, compuertas automáticas o figuras articuladas. A través de estas experiencias, los niños exploran de forma vivencial principios científicos como el movimiento, la fuerza y la energía; aplican conceptos matemáticos relacionados con ángulos y medidas; desarrollan soluciones de ingeniería funcionales; incorporan el arte en el diseño estético de sus creaciones, y emplean la tecnología y la programación para darles vida. Así, el servomotor se convierte en un recurso didáctico que potencia el enfoque STEAM y transforma el aula en un espacio de innovación activa y significativa.

El uso de servomotores en actividades STEAM permite a los estudiantes aplicar sus conocimientos en situaciones concretas, como crear un brazo robótico que recoja objetos, un semáforo que se active automáticamente, o una maqueta con partes móviles. Al integrar el diseño artístico, el pensamiento lógico y los cálculos matemáticos con la programación y la mecánica, se genera un aprendizaje completo y conectado. Así, STEAM no solo fomenta el dominio de contenidos curriculares, sino también habilidades blandas como la comunicación, la innovación y la perseverancia, haciendo del aula un espacio dinámico donde el conocimiento cobra vida a través de la tecnología (Montés et al., 2024).

## 1.5. Relación del pensamiento computacional, la programación y la metodología STEAM.

El pensamiento computacional es clave para resolver problemas de manera lógica, usando estructuras de programación. Las estructuras secuenciales, condicionales y repetitivas permiten organizar el flujo de un programa. La estructura secuencial ejecuta pasos en orden, las condicionales toman decisiones y las repetitivas permiten ejecutar bloques de código múltiples veces. Estas estructuras ayudan a desarrollar soluciones claras y eficientes, permitiendo que los estudiantes aprendan a pensar de forma lógica y estructurada.

**Figura 4**

*Relación de STEAM, programación y pensamiento computacional*



*Nota.* En esta figura se observa la relación de STEAM, programación y pensamiento computacional.

En la metodología STEAM, las estructuras de programación permiten desarrollar el pensamiento computacional y se aplican en proyectos interdisciplinarios donde ciencia, tecnología, ingeniería, arte y matemáticas se combinan (ver figura 4). Por ejemplo, en un proyecto de robótica, se usan

estructuras secuenciales para ordenar acciones, condicionales para responder a estímulos y repetitivas para realizar tareas múltiples. Integrando estas estructuras, los estudiantes resuelven problemas de manera creativa, vinculando conceptos de distintas áreas y desarrollando habilidades de pensamiento lógico y resolución de problemas.

La programación en STEAM no solo abarca la tecnología, sino también la creatividad y la innovación. Las estructuras secuenciales, condicionales y repetitivas permiten que los estudiantes diseñen proyectos como videojuegos o simulaciones, donde la lógica controla la interacción con el usuario. Estas herramientas de programación no solo enseñan tecnología, sino que conectan áreas como el arte y la matemática, fomentando la creatividad, la toma de decisiones y el trabajo colaborativo.

El pensamiento computacional, la programación y la metodología STEAM se integran de forma natural en experiencias de aprendizaje que combinan la lógica, la creatividad y la tecnología, y los servomotores representan un punto de encuentro concreto entre estas tres dimensiones. Al diseñar proyectos STEAM que incluyan componentes móviles, los estudiantes deben aplicar el pensamiento computacional para analizar el problema, descomponerlo en partes manejables y crear algoritmos que controlen el movimiento del servomotor. La programación actúa como el medio técnico para traducir esas ideas en instrucciones precisas, mientras que STEAM enmarca todo el proceso en un contexto interdisciplinario que une Ciencia, Tecnología, Artes y Matemáticas (Decorte & and Vlieghe, 2024). Esta combinación no solo promueve un aprendizaje activo y significativo, sino que también desarrolla

habilidades esenciales para el mundo actual, como la innovación, la resolución de problemas y el trabajo en equipo.

## 1.6. Autoevaluación del capítulo

**Pregunta 1:** ¿Cuál de las siguientes opciones es una característica clave del pensamiento computacional?

**Opciones de respuesta**

- a) se basa únicamente en el uso de lenguajes de programación.
- b) implica la descomposición de un problema en partes más pequeñas y manejables.
- c) se centra exclusivamente en la resolución de problemas matemáticos.
- d) requiere conocimientos avanzados en hardware de computadoras.

**Pregunta 2:** ¿Cómo se relaciona el pensamiento computacional con la programación?

**Opciones de respuesta**

- a) el pensamiento computacional se enfoca únicamente en escribir código de programación, sin importar el proceso de solución del problema.
- b) el pensamiento computacional ayuda a estructurar el proceso de resolución de problemas, lo que facilita la creación de programas más eficientes y claros.
- c) la programación solo utiliza el pensamiento computacional para crear algoritmos complejos.
- d) el pensamiento computacional es un tipo de lenguaje de programación que se utiliza para optimizar los programas.

**Pregunta 3:** ¿Cuál es la relación entre el pensamiento computacional y el enfoque STEAM?

**Opciones de respuesta**

- a) el pensamiento computacional se aplica solo en el área de tecnología y programación dentro de STEAM.
- b) el pensamiento computacional permite a los estudiantes integrar habilidades de resolución de problemas en todas las áreas de STEAM, facilitando la creación de soluciones innovadoras.
- c) el pensamiento computacional reemplaza las ciencias y las matemáticas dentro del enfoque STEAM.
- d) el pensamiento computacional solo se utiliza en proyectos artísticos dentro de STEAM.

**Pregunta 4:** ¿Cuál de las siguientes opciones describe correctamente la programación?

**Opciones de respuesta**

- a) la programación es el proceso de diseñar y desarrollar hardware para computadoras.
- b) la programación es el proceso de escribir instrucciones en un lenguaje que las computadoras pueden entender y ejecutar.
- c) la programación se refiere únicamente a la creación de videojuegos y aplicaciones móviles.
- d) la programación es un conjunto de tareas que se realizan manualmente para hacer funcionar un sistema operativo.

**Pregunta 5:** ¿Cómo se relacionan el pensamiento computacional y la programación en el desarrollo de software?

**Opciones de respuesta**

- a) El pensamiento computacional es un proceso únicamente teórico, mientras que la programación es la ejecución práctica del código.
- b) El pensamiento computacional permite a los programadores abordar problemas de manera estructurada, descomponerlos en partes más pequeñas y crear soluciones eficaces mediante programación.

c) La programación se basa únicamente en escribir código, mientras que el pensamiento computacional es solo para resolver problemas matemáticos.

d) El pensamiento computacional solo se utiliza para escribir código en lenguajes de programación, mientras que la programación se enfoca solo en diseño gráfico.

**Pregunta 6:** ¿Qué tipo de lenguajes de programación se utilizan comúnmente en niveles educativos iniciales?

**Opciones de respuesta:**

a) Solo lenguajes de código textual.

b) Lenguajes de bloques y código textual.

c) Lenguajes avanzados de inteligencia artificial.

d) Lenguajes exclusivamente matemáticos.

**Pregunta 7:** ¿Qué beneficios aporta trabajar con servomotores en proyectos escolares, según el texto?

**Opciones de respuesta:**

a) Favorece únicamente la construcción de mecanismos físicos.

b) Permite a los estudiantes comprender el valor de los algoritmos y desarrollar habilidades como el pensamiento lógico y la mejora continua.

c) Enseña a los estudiantes a construir robots de manera automática sin necesidad de programarlos.

d) Limita la creatividad al enfocarse solo en actividades digitales.

**Pregunta 8:** ¿Qué tipo de aprendizaje se promueve al integrar servomotores en actividades STEAM?

**Opciones de respuesta:**

a) Un aprendizaje basado únicamente en programación avanzada.

b) Un aprendizaje memorístico enfocado en la teoría.

Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

c) Un aprendizaje completo y conectado que integra diseño artístico, pensamiento lógico, cálculos matemáticos, programación y mecánica.

d) Un aprendizaje limitado a proyectos de robótica competitiva.

## **CAPÍTULO 2: LA PROGRAMACIÓN EN LA RESOLUCIÓN DE PROBLEMAS SECUENCIALES CON SERVOMOTORES**

### **2.1. Introducción: la educación del pensamiento computacional secuencial**

**Objetivo del capítulo:** Al finalizar este capítulo el futuro docente será capaz de aplicar el pensamiento computacional con énfasis en estructuras secuenciales usando mBlock y circuitos electrónicos basados en servomotores.

#### **Aprendizajes Esperados**

- Organización de la programación con servomotores
- La importancia y funcionamiento de las variables en programación con servomotores
- El rol de la programación de procedimientos y funciones en la resolución de problemas con servomotores
- Sentencias de control secuenciales con servomotores

En este capítulo, el futuro docente aprenderá a programar el uso de servomotores, comprenderá y describirá los componentes electrónicos esenciales asociados a su funcionamiento, y desarrollará habilidades para ensamblar circuitos básicos con servomotores. También, se abordará la importancia de valorar y aplicar estos conocimientos en el diseño de proyectos educativos y tecnológicos, fomentando un enfoque práctico y creativo que beneficie tanto el aprendizaje de los estudiantes como la integración de la tecnología en el aula.

Además, en este capítulo exploraremos cómo programar servomotores utilizando Arduino y mBlock, integrando de



forma práctica las áreas STEAM: Ciencia, Tecnología, Ingeniería, Arte y Matemáticas. A través del control de movimientos precisos, los estudiantes comprenderán principios científicos relacionados con la mecánica, aplicarán la tecnología de programación por bloques, utilizarán conceptos de ingeniería para resolver desafíos de diseño, incorporarán elementos artísticos al crear movimientos creativos o expresivos, y aplicarán las matemáticas para ajustar ángulos, tiempos y secuencias. Esta experiencia interdisciplinaria fomenta el pensamiento crítico, la creatividad y el trabajo colaborativo, fundamentales para enfrentar los retos del mundo actual. A continuación se detalla los aportes de cada área STEAM al aprendizaje de programación con servomotores.

**En el área de la tecnología:** La combinación de la metodología STEAM con la robótica basada en servomotor ofrece a los niños la oportunidad de aprender conceptos clave en tecnología y programación. A través de la programación básica y avanzada, los estudiantes desarrollan habilidades en la creación de algoritmos, el control de robots y la resolución de problemas. Empezando con bloques visuales y avanzando hacia lenguajes de programación más complejos, los niños pueden aprender a programar robots, controlando actuadores como los servomotores para interactuar con su entorno.

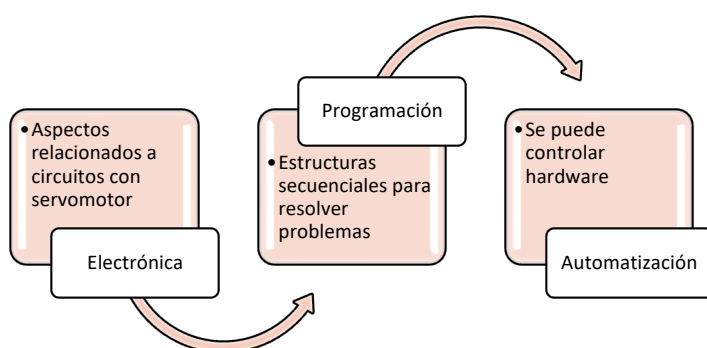
En palabras de Afecto et al. (2024), la robótica también fomenta el pensamiento lógico y secuencial, permitiendo que los niños resuelvan desafíos prácticos y aprendan sobre diseño y construcción. Al trabajar con robots, los estudiantes entienden cómo integrar diversos componentes mecánicos y electrónicos, desarrollando habilidades en ingeniería y ciencias de la computación. Además, aprenden a depurar errores en su código y a ajustar el comportamiento del robot según los resultados que obtienen.

El trabajo en equipo es otro aspecto fundamental de esta metodología. Los niños colaboran para diseñar, programar y construir robots, lo que mejora sus habilidades de comunicación y cooperación. Este enfoque interdisciplinario también fomenta la creatividad y la innovación, ya que los estudiantes tienen la libertad de crear soluciones únicas y desarrollar proyectos que combinan tecnología, arte y matemáticas, preparándolos para enfrentar desafíos futuros en diversas áreas.

En conclusión, al integrar la programación secuencial con Arduino, los niños no solo desarrollan habilidades fundamentales en electrónica, programación y lógica, sino que también obtienen una comprensión detallada de cómo el software puede controlar y automatizar el hardware (ver figura 5). Este aprendizaje les permite explorar una amplia gama de posibilidades creativas y técnicas, abriendo puertas a campos como la electrónica, la robótica y la automatización, y fomentando su capacidad para innovar y resolver problemas prácticos de manera efectiva.

### Figura 5

*Enseñanza de tecnología relacionada a circuitos basados en servomotor*



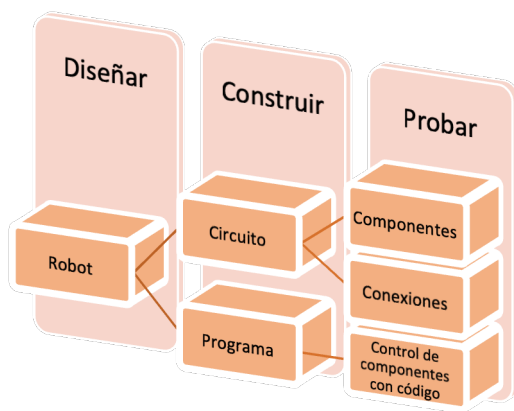
*Nota.* En esta figura se observan datos sobre la enseñanza de tecnología relacionada a circuitos con servomotor.

**En el área de la ingeniería:** Utilizando la metodología STEAM con robótica educativa, los niños aprenden las fases del proceso de ingeniería: identificación del problema, diseño, construcción, pruebas y optimización (Taranto et al., 2024). Definen el desafío, investigan soluciones, diseñan y construyen un prototipo, luego lo prueban y mejoran continuamente. Este enfoque fomenta habilidades en resolución de problemas, pensamiento crítico y la aplicación práctica de conceptos de ingeniería.

Además, a lo largo de este proceso, los niños desarrollan habilidades colaborativas al trabajar en equipo para diseñar, construir y probar sus robots (ver figura 6). La interacción con otros les permite compartir ideas, discutir soluciones y aprender de los errores de manera conjunta. Este enfoque les enseña que la ingeniería es un proceso iterativo y que los desafíos pueden ser superados a través de la creatividad, la cooperación y la mejora constante.

**Figura 6**

*Enseñanza de procesos de ingeniería a niños*



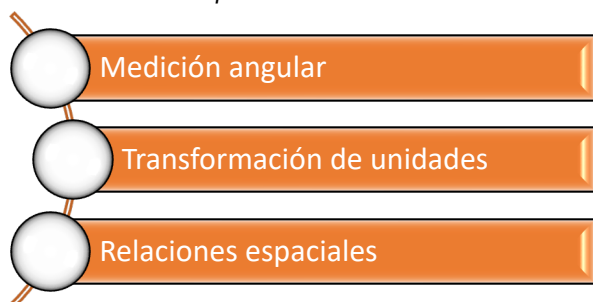
*Nota.* En esta figura se observa un proceso de ingeniería para enseñanza a niños.

**En el área de Matemática:** los niños a comprender conceptos matemáticos clave de manera práctica. Por ejemplo, al controlar el movimiento de un servomotor, los niños aprenden sobre geometría y ángulos. Los servos generalmente funcionan con una rotación de 0 a 180 grados, por lo que los niños deben entender cómo programar y calcular estos ángulos para que los servos realicen movimientos precisos. Esto les permite explorar conceptos como medición angular, transformación de unidades y relaciones espaciales (ver figura 7).

Además, al trabajar con servomotores, los niños pueden aplicar aritmética y álgebra al calcular la velocidad y la distancia recorrida por los motores en función de las rotaciones, promoviendo la comprensión de proporciones y funciones lineales. También pueden experimentar con secuencias matemáticas al programar movimientos que dependen de ciertos patrones o intervalos de tiempo, lo que fortalece su comprensión de conceptos como tiempo, ritmo y frecuencia. A través de estos proyectos, los niños no solo ven la teoría matemática aplicada, sino que también desarrollan habilidades para resolver problemas utilizando herramientas matemáticas en situaciones del mundo real.

### **Figura 7**

*Enseñanza de Matemática con pensamiento computacional*

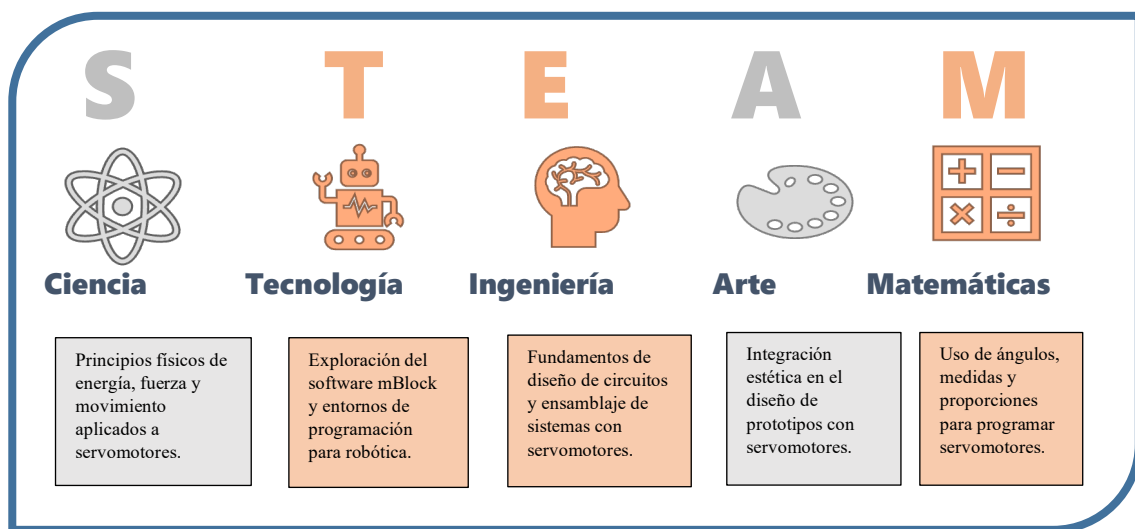


*Nota.* En esta figura se observan competencias clave de la enseñanza de Matemática con pensamiento computacional.

Los contenidos de este capítulo se centran con mayor énfasis en tres áreas clave de la metodología STEAM: Tecnología, Ingeniería y Matemáticas (ver figura 8). A través del aprendizaje del pensamiento computacional, aplicado en circuitos electrónicos que incorporan servomotores, se busca fortalecer de manera innovadora y creativa las competencias técnicas y analíticas, promoviendo habilidades esenciales para resolver problemas en entornos educativos dinámicos.

**Figura 8**

*Gráfico de aprendizaje de las áreas STEAM*



*Nota.* En esta figura se observa las áreas de aprendizaje STEAM que el futuro docente debe lograr.

Los contenidos presentados en este capítulo se enfocan, principalmente, en tres áreas esenciales de la metodología STEAM: Tecnología, Ingeniería y Matemáticas (ver figura 8). Principalmente, se trabaja el proceso de ingeniería que puede enseñarse a un niño para desarrollar un programa en mBlock utilizando estructuras secuenciales. Este proceso se puede simplificar y adaptar según el nivel de comprensión del niño, permitiendo que aprenda de manera efectiva. A continuación, se describen de forma estructurada las principales fases o pasos involucrados en este proceso, facilitando su aprendizaje paso a paso.

## **Identificación del Problema**

En esta fase, el niño debe comprender el problema que se quiere resolver con el robot. Por ejemplo, puede hacer que el robot gire la cabeza mediante un servomotor. Se trata de introducir la idea de que los robots pueden realizar tareas de manera autónoma si están correctamente programados y equipados con servomotores y otros componentes electrónicos.

## **Investigación y Planificación**

El niño investiga cómo los diferentes componentes del robot (motores y otros componentes Arduino) pueden trabajar juntos para resolver el problema. Aquí se presentan las herramientas de programación, como mBlock, y se explica cómo usar bloques para interactuar con el hardware. El niño debe planificar qué pasos seguirá para lograr que el robot haga lo que desea.

## **Diseño**

El diseño implica crear un plan detallado para la construcción y programación del robot. El niño debe decidir cómo organizar los componentes, cómo se conectarán entre sí, y qué funciones debe realizar el robot. Además, debe escribir un esquema de programación con bloques, anticipando las acciones del robot ante diferentes entradas.

## **Construcción**

En esta fase, el niño se encarga de armar físicamente el robot, conectando los servomotores a la placa de Arduino de acuerdo con el plan establecido. Además, escribe el código en mBlock para programar el movimiento y las acciones de los servomotores, controlando de manera

precisa el comportamiento del robot y asegurando que siga las instrucciones programadas.

## **Pruebas**

Una vez construido el robot, el niño procede a realizar pruebas para comprobar que funcione según lo planeado. Durante esta etapa, se examinan cuidadosamente las conexiones físicas de los servomotores y la programación en mBlock, evaluando la respuesta del robot ante las instrucciones. Si el desempeño no es el esperado, se inicia un proceso de revisión y ajuste, tanto del hardware como del software, identificando errores, corrigiendo fallos y optimizando el funcionamiento general del prototipo para lograr un mejor desempeño.

## **Mejoras**

Tras las pruebas iniciales, el niño reflexiona sobre cómo mejorar el robot. Pueden agregarse nuevas funcionalidades, mejorar la eficiencia del código o modificar el diseño. Esta fase es crucial para entender que la ingeniería es un proceso iterativo, donde siempre se puede encontrar una forma mejor de hacer las cosas.

## **Presentación**

Finalmente, el niño presenta su robot y el proceso que siguió para crearlo. Explica cómo resolvió el problema y cómo el robot cumple su función. Esto no solo refuerza el aprendizaje, sino que también ayuda a desarrollar habilidades de comunicación y trabajo en equipo, ya que debe transmitir claramente su proyecto.



## **2.2. Organización de la programación de servomotores con mBlock**

La organización de la programación en mBlock es fundamental para crear proyectos eficientes y fáciles de mantener. mBlock es una plataforma visual de programación basada en bloques, lo que facilita la comprensión de conceptos de programación sin necesidad de escribir código. A continuación, te explicaré cómo se organiza la programación en mBlock, usando herramientas y prácticas que ayudan a estructurar el código de manera efectiva:

### **1. Uso de Categorías de Bloques**

mBlock organiza los bloques de programación en varias categorías que agrupan las instrucciones de acuerdo con su función. Al comenzar un proyecto, puedes acceder a estas categorías desde el menú lateral, lo que facilita la navegación y la búsqueda de bloques específicos. Algunas categorías comunes incluyen:

**Eventos:** Bloques que se ejecutan cuando ocurre un evento, como "cuando bandera verde clickeada" o "cuando se recibe una señal".

**Movimiento:** Bloques para controlar la posición de objetos en el escenario o controlar componentes como motores y servomotores.

**Control:** Bloques de control como ciclos, condiciones ("si... entonces") y espera.

**Datos:** Bloques para crear y manipular variables y listas.

**Sensores:** Bloques para leer datos de sensores físicos conectados al dispositivo.

Utilizar estas categorías ayuda a organizar el código de manera más accesible y coherente, ya que agrupa las acciones en bloques de propósito específico.

### **2. Creación de Variables y Listas**

Una parte esencial de la organización en mBlock es el uso de variables y listas para almacenar y gestionar datos. Las variables te permiten almacenar valores que pueden cambiar durante la ejecución del programa, como números, textos o posiciones. Por ejemplo, si estás controlando un servomotor, puedes usar una variable para almacenar el ángulo de posición. Las listas se utilizan para almacenar colecciones de datos y son útiles en proyectos más complejos.

Para mantener tu código limpio y comprensible, es importante dar nombres descriptivos a las variables y listas. Por ejemplo, en un proyecto de robótica, podrías crear una variable llamada "Velocidad" o "Puntos" para llevar un registro de la velocidad de un motor o del puntaje en un juego. Esto mejora la legibilidad y la organización del proyecto, especialmente cuando el programa crece en complejidad.

### 3. Uso de Procedimientos y Funciones (Bloques Personalizados)

Una de las mejores prácticas para organizar proyectos más grandes en mBlock es la creación de procedimientos. Los procedimientos te permiten agrupar bloques de código que realizan tareas específicas y luego llamarlos cuando sea necesario. En lugar de repetir el mismo conjunto de bloques varias veces, puedes crear un procedimiento y reutilizarlo.

En mBlock, los procedimientos se pueden definir mediante bloques personalizados, y dentro de estos procedimientos puedes incluir cualquier tipo de instrucción que el programa necesite. Esto hace que el código sea más modular y más fácil de leer, ya que puedes dividir un proyecto en secciones manejables.

Por ejemplo, en un proyecto de control de servomotores, podrías tener un procedimiento para mover el servomotor a un ángulo específico. Este procedimiento puede ser llamado varias veces en diferentes partes del código, eliminando la redundancia y manteniendo todo organizado.

El control del flujo de ejecución es otra clave para organizar la programación en mBlock. Los bloques de eventos, condiciones y ciclos permiten dirigir el comportamiento del programa según las interacciones con el entorno o las entradas del usuario.

Eventos: Los bloques de eventos como "cuando bandera verde clickeada" permiten definir el inicio de un programa o de una acción específica, como cuando se presiona un botón o se detecta una señal.

### **2.3. La importancia y funcionamiento de las variables en programación de servomotores con mBlock**

En mBlock, el uso de variables es fundamental para controlar de manera flexible y dinámica dispositivos como los servomotores. Al emplear variables, puedes almacenar y modificar valores como los ángulos del servomotor durante la ejecución del programa, lo que te permite ajustar su comportamiento sin necesidad de reescribir el código cada vez. Por ejemplo, puedes usar una variable llamada "Ángulo" para almacenar el valor de la posición del servomotor y actualizarla conforme el programa avanza, permitiendo que el servomotor se mueva de manera precisa y controlada. De esta manera, el pensamiento computacional aplicado en mBlock se convierte en una herramienta poderosa para automatizar y personalizar el comportamiento de los servomotores.

Además, las variables en mBlock no solo sirven para almacenar valores de manera estática, sino que permiten realizar cálculos y llevar un seguimiento del estado del sistema. Por ejemplo, si deseas contar cuántas veces el servomotor ha sido movido a una nueva posición, puedes crear una variable llamada "ContadorMovimientos" que se

incremente cada vez que el servomotor se mueva a una nueva posición. Esto es útil no solo para mostrar el número de movimientos en la pantalla, sino también para tomar decisiones dentro del programa basadas en cuántos movimientos se han realizado, como realizar un cambio de comportamiento o ejecutar una acción específica después de ciertos movimientos.

El uso de variables también facilita la creación de programas más flexibles y reutilizables. En lugar de escribir código repetido para controlar los servomotores en diferentes posiciones, puedes almacenar los valores de los ángulos en variables y utilizarlos en diversas partes del programa. Además, puedes combinar las variables con ciclos, como el ciclo "para siempre", para que el servomotor se mueva continuamente entre diferentes ángulos o realice movimientos específicos de acuerdo a los valores de la variable, lo que permite una gran versatilidad en la programación de sistemas con servomotores. Esta capacidad de modificar dinámicamente el comportamiento del servomotor con variables hace que la programación sea más eficiente y fácil de modificar según las necesidades del proyecto.

## **2.4. El rol de la programación de procedimientos y funciones en la resolución de problemas con servomotores en mBlock**

Procedimientos: En mBlock, los procedimientos se crean utilizando bloques que permiten agrupar una secuencia de acciones y luego llamarlos cuando sea necesario. Esto se hace mediante el uso de bloques como "cuando se haga clic en la bandera verde" o "al recibir una señal". Dentro de mBlock, puedes crear un conjunto de bloques que realicen una tarea específica y luego usarlo en cualquier parte del programa. Los procedimientos en mBlock ayudan a

organizar el código, hacer que sea más modular y reutilizable.

Con base en lo anterior, en un proyecto de puerta automática con servomotor, los procedimientos resultan esenciales. Se puede definir un bloque "abrir puerta" que gire el servomotor a 90° y un bloque "cerrar puerta" que lo devuelva a 0°. Cada vez que se necesite abrir o cerrar, bastará con llamar al procedimiento adecuado, sin repetir instrucciones.

En otro ejemplo de brazo mecánico, los procedimientos permiten programar movimientos básicos como "subir brazo", "bajar brazo" o "colocar en reposo". Esto facilita que los estudiantes entiendan cómo dividir una acción compleja en pequeñas rutinas que luego se combinan para realizar tareas más elaboradas, como tomar y soltar objetos.

En un sistema de barrera levadiza (como en estacionamientos), se pueden crear procedimientos para "levantar barrera" y "bajar barrera". Cada procedimiento controla directamente al servomotor, asegurando que la barrera se mueva con precisión y evitando que el estudiante deba escribir las mismas instrucciones en diferentes partes del programa.

Los procedimientos en proyectos con servomotores ayudan a estandarizar movimientos repetitivos, mejorar la claridad del código y preparar al estudiante para el uso de algoritmos más complejos en el futuro.

Funciones: Aunque mBlock no utiliza funciones en el sentido tradicional como en lenguajes de programación textuales como Python o C++, se pueden simular mediante el uso de bloques que permiten devolver un valor o realizar una

operación específica. Esto se logra con bloques como "definir" (para establecer un valor o acción) y "responder" (para devolver un valor cuando se le llama). En términos simples, se pueden crear bloques personalizados que comporten una funcionalidad similar a las funciones tradicionales.

En un brazo mecánico con servomotores, una función puede calcular el ángulo exacto que debe girar para alcanzar un punto de agarre. De esta forma, el bloque "calcular ángulo" devuelve el valor correcto y simplifica el diseño de rutinas de movimiento.

En un sistema de puerta automática, una función puede devolver un valor de apertura parcial o total, dependiendo de la necesidad. Por ejemplo, devolver 45° para una apertura parcial y 90° para una apertura completa. Así, el programa se adapta a distintos escenarios sin necesidad de cambiar la lógica en cada sección.

En un proyecto de barrera levadiza, una función puede determinar si el servomotor debe moverse a un ángulo de 70° o de 0°, en función de un parámetro previamente establecido por el usuario. Esto permite reutilizar el mismo bloque en diferentes contextos, sin necesidad de redefinir instrucciones cada vez.

De este modo, las funciones permiten a los estudiantes trabajar con la abstracción y la reutilización en proyectos con servomotores. Cada función encapsula un cálculo o decisión, y el programa principal se centra únicamente en la ejecución de movimientos, lo que facilita la comprensión y depuración del código.

Aunque mBlock no usa directamente el concepto de funciones y procedimientos como en lenguajes de texto, la plataforma facilita la creación de estructuras modulares y reutilizables a través de su sistema de bloques, lo que facilita

la comprensión de los conceptos básicos de programación, especialmente para principiantes y en entornos educativos.

## **2.5. Sentencias de control secuenciales para servomotores con mBlock**

### **Las sentencias secuenciales en el pensamiento computacional**

Las estructuras secuenciales en programación son aquellas que ejecutan las instrucciones de forma lineal y en el mismo orden en que están escritas en el código. En este tipo de estructuras, no hay decisiones ni saltos condicionales; cada paso del programa se lleva a cabo uno tras otro, sin desviaciones. Es la estructura más básica y se considera la base de cualquier algoritmo, ya que permite que el programa realice tareas de manera ordenada, como leer datos, procesar información y mostrar resultados. Este enfoque es esencial en programación, ya que garantiza una ejecución clara y fácil de comprender.

El flujo de control en una estructura secuencial sigue un orden preestablecido de instrucciones. A medida que el programa se ejecuta, cada línea de código se lleva a cabo en el mismo orden en que está escrita, asegurando que no haya saltos ni decisiones condicionales. Este enfoque es especialmente útil en situaciones donde no se necesitan bifurcaciones o repeticiones, como en tareas simples de entrada de datos, procesamiento y generación de resultados. Las estructuras secuenciales son esenciales para programas sencillos y para enseñar los fundamentos de la programación, ya que proporcionan una manera clara y accesible de construir algoritmos.

## Sintaxis básica de un programa secuencial

En un programa secuencial, el flujo se construye escribiendo cada paso o instrucción en orden secuencial. Las acciones o comandos se organizan uno después del otro, y cada uno representa una tarea específica que el programa debe ejecutar en el orden establecido.

## Ejemplos de programas secuenciales

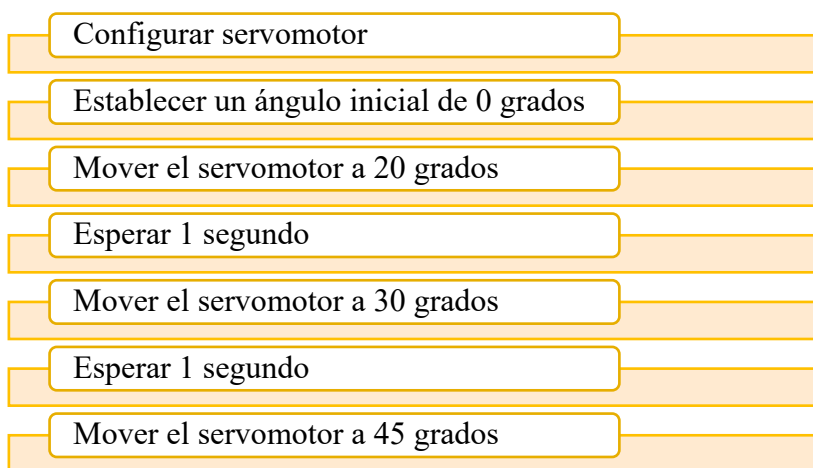
A continuación, se presentan algunos ejemplos de estructuras secuenciales basados en situaciones cotidianas que los niños pueden comprender fácilmente. Estos ejemplos ayudan a relacionar el concepto de secuencia en programación con actividades de la vida diaria.

### Ejemplo 1: Mover un servomotor

Imaginemos que estamos dando instrucciones para que un motor se mueva. Es una tarea sencilla que implica varios pasos que deben seguirse en orden (ver figura 9).

#### Figura 9

*Pasos a seguir para mover un servomotor de 0 a 45 grados*





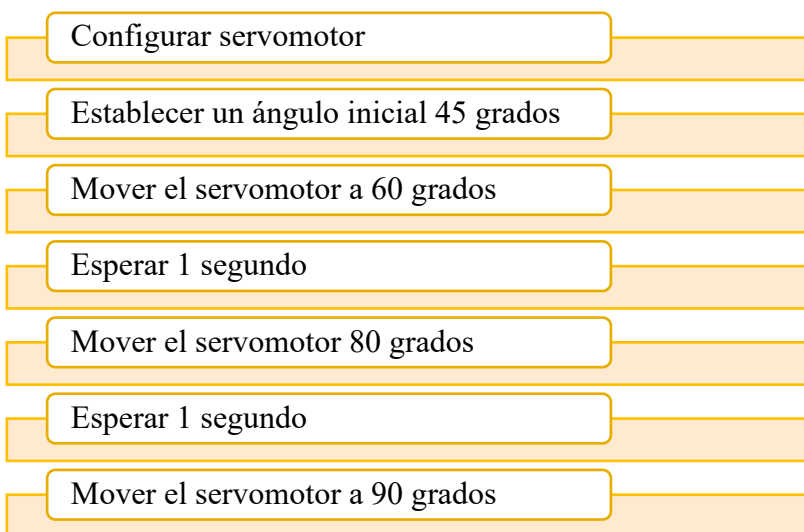
*Nota.* La figura muestra los pasos para mover un motor de 0 a 45 grados

### **Ejemplo 1:** Mover un servomotor

Imaginemos que estamos dando instrucciones para que un motor se mueva. Es una tarea sencilla que implica varios pasos que deben seguirse en orden (ver figura 10).

#### **Figura 10**

*Pasos a seguir para mover un servomotor de 45 a 90 grados*



*Nota.* La figura muestra los pasos para mover un motor de 45 a 90 grados

Las estructuras secuenciales son esenciales en programación, ya que permiten ejecutar instrucciones de forma ordenada, una después de otra. Este concepto se aplica en varios lenguajes de programación, como Python, Java, o en entornos de programación por bloques como mBlock o Scratch.

En la programación por bloques, las estructuras secuenciales implican ejecutar instrucciones en un orden específico, conectando bloques que representan acciones. Herramientas como mBlock o Scratch permiten a los niños aprender a organizar y enlazar bloques para realizar tareas de manera lógica y secuencial.

A continuación, se describirá cómo trabajar con secuencias en el lenguaje de programación por bloques mBlock. Primero, se realizarán simulaciones utilizando objetos virtuales, y luego se aplicarán estos conceptos en robótica educativa, enfocándose específicamente en circuitos electrónicos físicos que incluyan servomotores.

## **La programación secuencial por bloques con servomotores**

### **Descripción de mBlock**

**mBlock** es una plataforma de programación visual basada en bloques, creada para enseñar conceptos de programación a niños de manera sencilla. Permite desarrollar proyectos interactivos, como animaciones, juegos y robots, a través de una interfaz amigable que facilita el aprendizaje de la lógica de programación.

### **El entorno de programación de mBlock**

El entorno de programación de **mBlock** es una plataforma visual e intuitiva que emplea bloques gráficos para representar instrucciones de programación. Permite a los usuarios arrastrar y conectar bloques para construir programas, lo que simplifica el aprendizaje de la lógica y las estructuras de programación. Además, incluye herramientas para controlar robots y dispositivos interactivos. En la siguiente imagen (ver figura 11), se muestra el entorno de mBlock, destacando las áreas más importantes de esta herramienta de programación por bloques. El entorno está

compuesto principalmente por un Escenario donde los diferentes Objetos (llamados Sprites en inglés) interactúan y se desarrollan.

### Figura 11



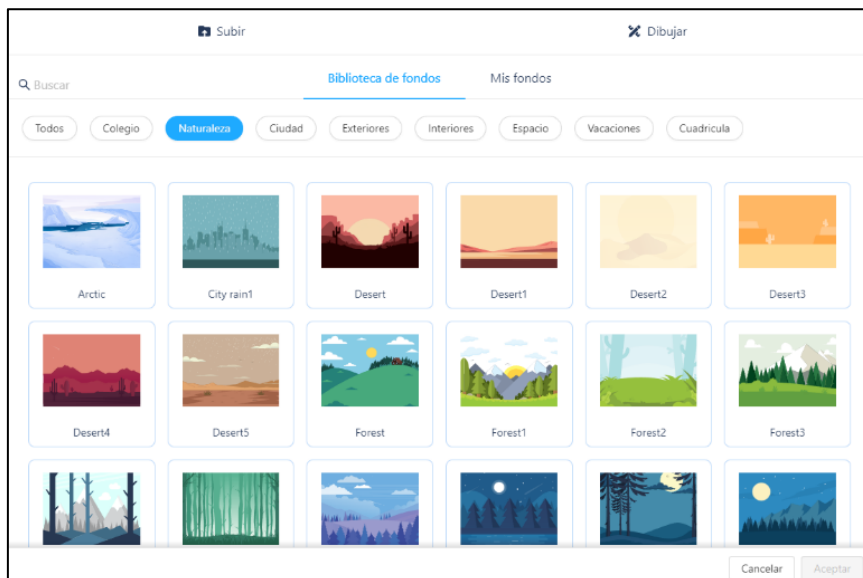
*Nota.* La figura muestra el Entorno de programación por bloques de mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

## Implementar un escenario

Para crear un escenario adecuado, primero es necesario seleccionar un Fondo (ver figura 12), que puede modificarse durante la ejecución de una animación o programa. En la siguiente imagen, se presenta la colección de fondos disponibles en mBlock. También es importante tener en cuenta que podemos diseñar nuestro propio fondo para el escenario e incluso agregar una imagen guardada en nuestra computadora.

## Figura 12

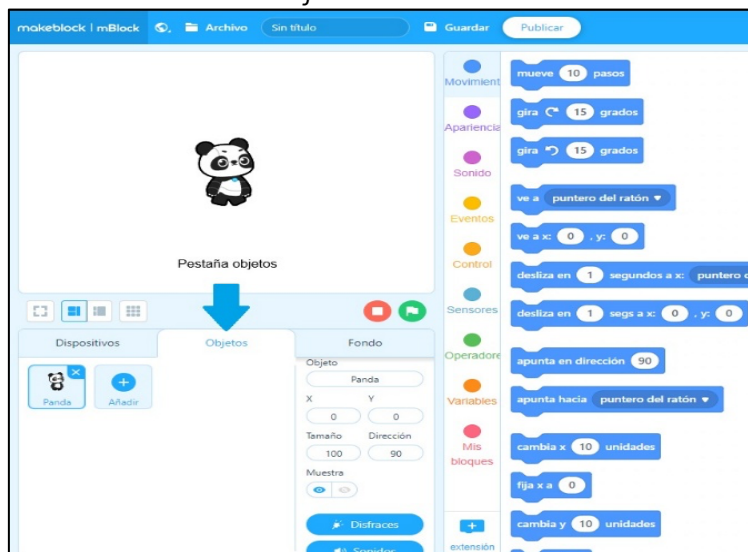
### *Fondos disponibles en mBlock*



*Nota.* La figura muestra Fondos disponibles en mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

Una parte fundamental del escenario son los Objetos, que son elementos clave para crear nuestras aplicaciones, representando a los distintos personajes en el entorno virtual de mBlock. A estos objetos se les puede asignar interacciones para desarrollar lo que conocemos como un programa (o Scripts en inglés). Los Objetos pueden presentarse en el escenario con múltiples disfraces, lo que nos permite asociar y definir acciones según las diferentes formas en que se muestran. En la pestaña Objetos (ver figura 13), podemos comenzar a trabajar con diversos personajes en el entorno de programación de mBlock.

**Figura 13**  
*Pestaña Objetos en mBlock*

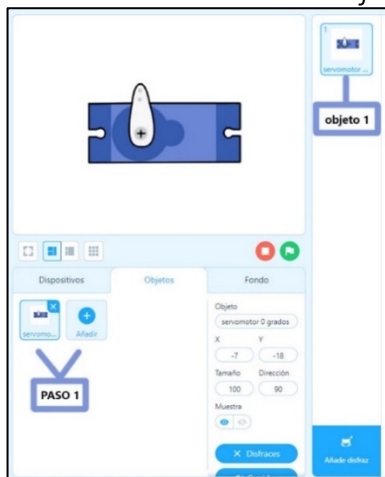


*Nota.* La figura muestra la pestaña Objetos en mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

La siguiente figura muestra la creación del objeto (1), y luego también se muestra cómo se agrega un disfraz objeto (2). A los objetos se les debe colocar un nombre apropiado para luego usar esos nombres en la programación por bloques. En este caso, a los objetos, se les ha colocado como nombres *servomotor 0 grados* y *servomotor 90 grados* (ver figura 14 y 15).

### Figura 14

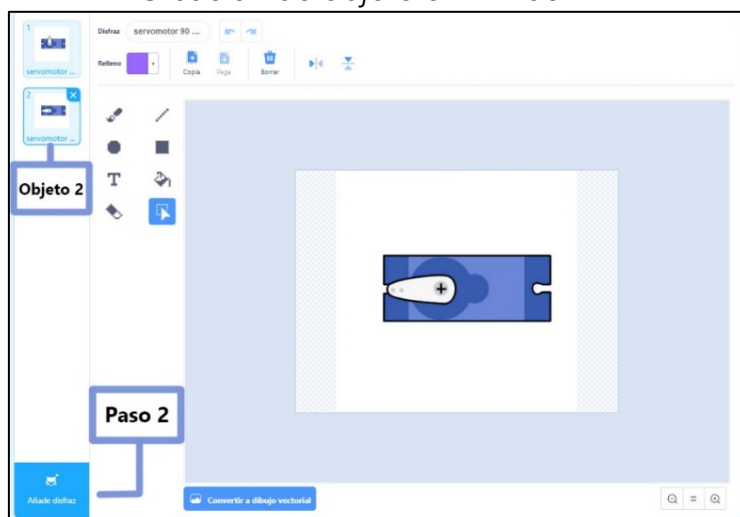
*Creación de objeto en mBlock*



*Nota. La figura muestra la creación de objetos en mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)*

### Figura 15

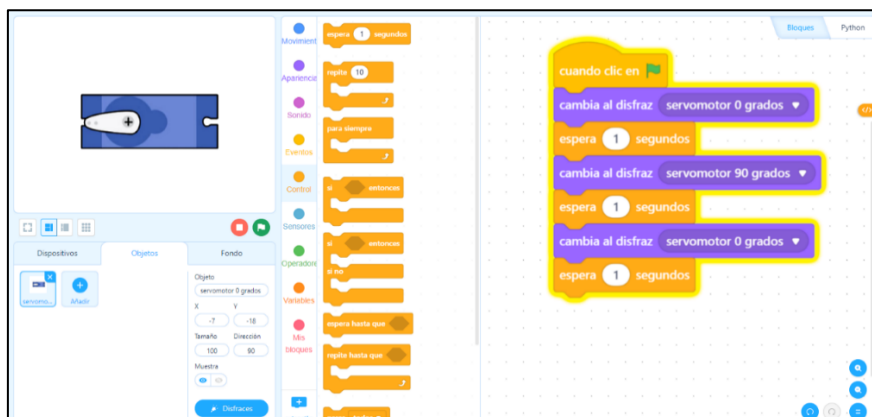
*Creación de objeto en mBlock*



*Nota. La figura muestra como agregar un disfraz a un objeto. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)*

Una vez que tenemos el fondo, el objeto y los disfraces para el objeto dentro del escenario, ya podemos empezar a programar estructuras secuenciales sobre dicho escenario. En este caso vamos a simular que el servomotor hace un giro de 0 grados a 90 grados y vuelve a la posición inicial de 0 grados (ver figura 16).

**Figura 16**  
*Simulación en mBlock*



Nota. La figura muestra una simulación en mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

## 2.6. El hardware necesario para trabajar robótica educativa basada en servomotores con mBlock

### Descripción de un servomotor

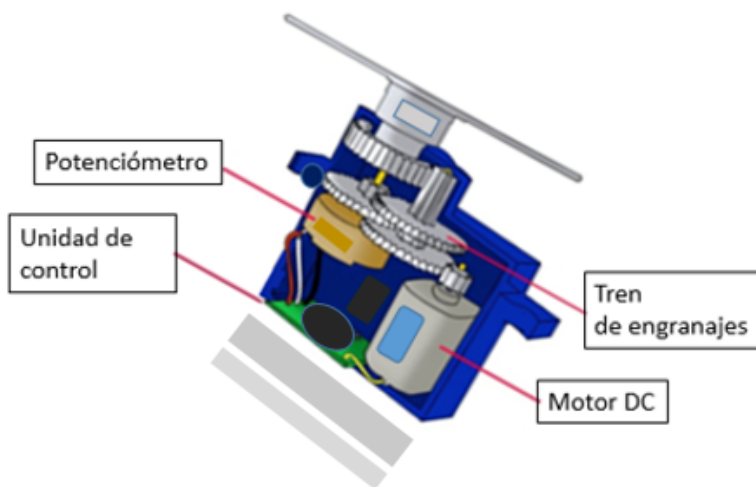
Un servomotor es un tipo de motor eléctrico diseñado para controlar con precisión la posición, velocidad y aceleración de un mecanismo. Esta combinación permite que los servomotores sean ideales para aplicaciones en robótica, automatización y sistemas que requieren movimientos controlados con alta precisión.

### Estructura de un servomotor

Un servomotor está compuesto por un motor eléctrico, que puede ser de corriente continua o alterna, un sistema de control encargado de regular su funcionamiento y un sensor de retroalimentación, como un potenciómetro, que mide la posición o velocidad del eje. Además, cuenta con un reductor de engranajes que disminuye la velocidad y aumenta el torque, permitiendo mayor precisión en el movimiento. Finalmente, dispone de un circuito de alimentación y señal que recibe la energía necesaria y procesa las instrucciones enviadas por un controlador externo (ver figura 17).

**Figura 17**

*Servomotor y sus partes*



*Nota.* La figura muestra las partes del servomotor



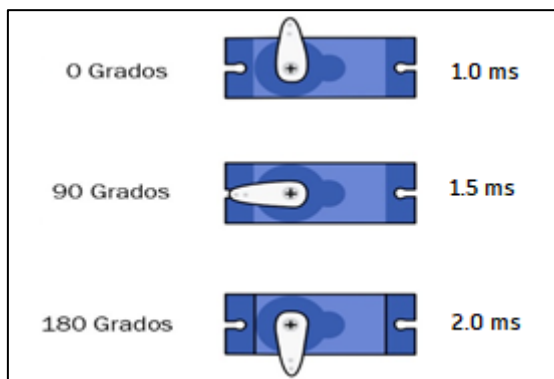
## Funcionamiento de un servomotor

Para que el servomotor funcione, el sistema de control debe recibir una señal de entrada que indique la posición deseada, enviando la energía necesaria al motor eléctrico para generar el movimiento (ver figura 18).

*Se puede controlar el servomotor enviándole una serie de pulsos. Un servomotor típico espera un pulso cada 20 milisegundos.*

**Figura 18**

*Funcionamiento del servomotor*



*Nota.* La figura muestra el tiempo que demora en avanzar el servomotor a 0, 90, 180 grados

El ángulo de giro de los servomotores (ver Tabla 1) está directamente relacionado con la señal de control que reciben, la cual se expresa en pulsos de ancho modulado (PWM). Los servomotores estándar suelen operar en un rango de 0° a 180°, donde un pulso de aproximadamente 1 ms posiciona el eje en 0°, un pulso de 1.5 ms lo sitúa en 90°, y un pulso de 2 ms lo mueve a 180°. Ajustar correctamente

la señal PWM es fundamental para garantizar movimientos precisos y evitar esfuerzos innecesarios en el motor. La selección del ancho de pulso adecuado depende de la configuración del controlador y de la aplicación específica, asegurando un funcionamiento eficiente y prolongando la vida útil del servomotor.

**Tabla 1**  
*Relación entre el ancho de pulso y el ángulo de giro en un servomotor estándar*

Ángulo de Giro (°)	Ancho de Pulso (ms)	Estado del Servomotor
0°	1.0 ms	Posición mínima
90°	1.5 ms	Posición central
180°	2.0 ms	Posición máxima

*Nota.* La tabla muestra la relación entre el ancho de pulso y el ángulo de giro en un servomotor

## Componentes electrónicos complementarios

### Tarjeta Arduino NANO

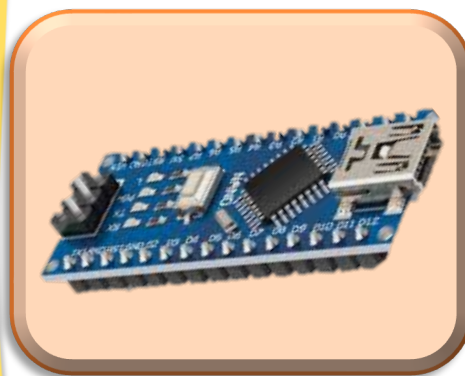
La tarjeta Arduino es una plataforma de hardware y software de código abierto que permite a los usuarios crear proyectos electrónicos interactivos de manera sencilla. Básicamente, es una pequeña computadora que puede ser programada para controlar una variedad de dispositivos electrónicos, como luces, motores, sensores y más. Está compuesta por una placa con un microcontrolador, que es el cerebro de la tarjeta, y una serie de pines de entrada y salida (I/O) que permiten conectarla con otros componentes electrónicos.

### **Figura 19**

#### *Tarjeta Arduino NANO*

#### **Tarjeta Arduino NANO**

La tarjeta Arduino es como el cerebro de un robot o una máquina. Es una pequeña computadora que puedes programar para hacer cosas como encender luces, mover motores o leer información de sensores.



*Nota.* La figura muestra una tarjeta Arduino NANO

El microcontrolador en la tarjeta Arduino se puede programar utilizando el software Arduino IDE, un entorno de desarrollo que permite escribir y cargar código en la tarjeta para que realice tareas específicas. Este código puede ser escrito en un lenguaje de programación similar a C/C++, lo que facilita la programación, incluso para principiantes. Además, la comunidad de Arduino es muy activa, lo que significa que hay muchos tutoriales, ejemplos y proyectos listos para ser utilizados.

Una de las principales ventajas de Arduino es su versatilidad y facilidad de uso. Los niños y principiantes pueden usarla para aprender conceptos básicos de electrónica y programación, mientras que los expertos pueden aprovechar su capacidad para crear proyectos más complejos. Desde proyectos básicos como mover un servomotor a una posición específica hasta la construcción de brazos robóticos y sistemas automatizados, Arduino es

una herramienta poderosa que fomenta la creatividad y el aprendizaje práctico en el campo de la tecnología.

## **Cable de datos Arduino NANO**

El cable de datos para Arduino Nano es un cable USB que se utiliza para conectar la placa Arduino Nano a una computadora o a un adaptador USB para permitir la programación y la comunicación de datos entre la placa y el ordenador. Además, este cable también suministra energía a la placa durante el proceso de desarrollo, lo que facilita realizar pruebas sin necesidad de una fuente externa de alimentación. Es recomendable utilizar un cable de buena calidad para evitar fallos de transmisión de datos o desconexiones inesperadas durante la programación.

### **Figura 20**

*Cable de datos de Arduino NANO*



### **Cable de Datos**

Este cable conecta el Arduino a la computadora. Cuando escribes un programa o código en la computadora, el cable de datos le pasa esas instrucciones a Arduino, para que haga lo que le pides, como encender luces.

*Nota.* La figura muestra un cable de datos de Arduino NANO

## **Características del cable de datos para Arduino Nano:**

### *Conector USB Tipo A a Mini-USB o Micro-USB:*

- La Arduino Nano generalmente utiliza un conector Mini-USB o Micro-USB, dependiendo del modelo, para conectar la placa a la computadora.
- El extremo Mini-USB o Micro-USB se conecta al puerto de la tarjeta Arduino Nano.
- El extremo USB Tipo A se conecta al puerto USB de la computadora.

### *Propósito:*

- Programación: El cable se utiliza para cargar programas desde el Arduino IDE a la placa Arduino Nano. Al conectar el Arduino Nano a la computadora, se puede transferir código escrito en el entorno de desarrollo Arduino.
- Comunicación de datos: Permite que el Arduino Nano se comuniquen con la computadora, lo que es útil para enviar datos de sensores, leer entradas del teclado o de otros dispositivos y procesar resultados en tiempo real.

### *Conexión y alimentación:*

- Además de la transferencia de datos, el cable USB también alimenta el Arduino Nano a través del puerto USB de la computadora. Esto significa que no necesitas una fuente de alimentación externa para que el Nano funcione cuando lo conectas por USB.

### *Compatibilidad:*

- Es importante usar un cable que sea compatible con el modelo específico de tu Arduino Nano.

Algunos modelos antiguos de Arduino Nano usan un conector Mini-USB, mientras que las versiones más recientes usan Micro-USB. Asegúrate de que el cable que adquieras tenga el conector adecuado para tu placa.

## **Tarjeta de expansión NANO**

Es una tarjeta (de la categoría de los shields) de expansión diseñada para ser compatible con la placa **Arduino Nano**. Los **shields** son placas adicionales que se pueden montar directamente sobre las placas Arduino (como el Nano) para añadirles funcionalidades específicas sin necesidad de usar cables o componentes adicionales dispersos.

Un **Arduino Nano Shield** se conecta a los pines del Arduino Nano y expande su capacidad, añadiendo más funcionalidades de manera fácil y rápida. Estos shields pueden incluir una variedad de módulos o componentes, como pantallas LCD, sensores, motores, módulos de comunicación, entre otros. Están diseñados para que encajen perfectamente con la disposición de los pines de la placa Nano, de modo que no sea necesario realizar conexiones complejas con cables.

## Figura 21

### *Tarjeta de expansión de Arduino NANO*

#### **Tarjeta de expansión NANO**

Una tarjeta de expansión para Arduino Nano es un accesorio que se conecta a la placa para facilitar la conexión de sensores, motores, pantallas y otros componentes electrónicos.



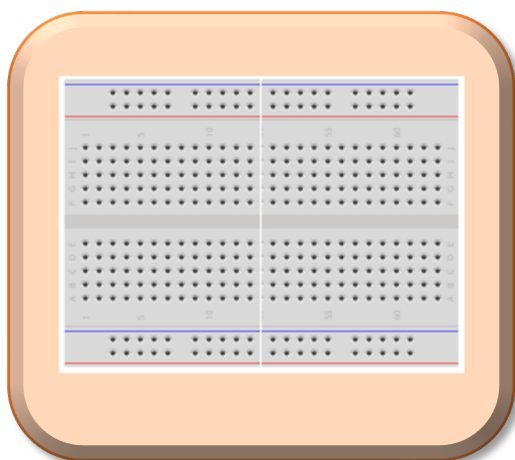
*Nota.* La figura muestra una tarjeta de expansión de Arduino NANO

### **Protoboard**

Un protoboard es una herramienta utilizada en electrónica para la construcción de circuitos de prueba de manera rápida y sin necesidad de soldadura. Está compuesta por una base de plástico con filas y columnas de agujeros conectados internamente, que permiten insertar componentes electrónicos (como resistores, transistores, chips, etc.) y cables para crear circuitos temporales. Además, su diseño modular facilita modificar, ampliar o corregir los montajes de forma sencilla, lo que lo convierte en un recurso esencial para el aprendizaje, la experimentación y el desarrollo de prototipos en proyectos educativos y profesionales.

## Figura 22

*Placa de pruebas de Arduino NANO*



### Protoboard

Sirve para modificar el circuito fácilmente sin tener que soldar y desoldar cada componente. Esto hace que sea ideal para experimentación de circuitos.

*Nota.* La figura muestra una placa de pruebas para Arduino NANO

### Cables de conexión Dupont

Los **cables Dupont** son cables de conexión versátiles y ampliamente utilizados en proyectos de electrónica y robótica, especialmente cuando se trabaja con plataformas como **Arduino**, **Raspberry Pi**, y otros sistemas de prototipado. Son conocidos por su facilidad de uso y por permitir conexiones rápidas y seguras entre la placa base (como el **Arduino Nano**) y otros componentes electrónicos, como sensores, módulos, y actuadores.



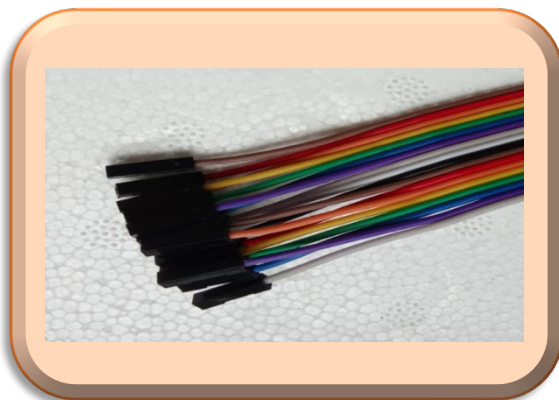
### **Figura 23**

#### *Cables de conexión Dupont*



#### **Cables de conexión Dupont**

Los cables están disponibles en versiones macho a macho, hembra a hembra, o macho a hembra, lo que los hace muy flexibles según las necesidades del proyecto.



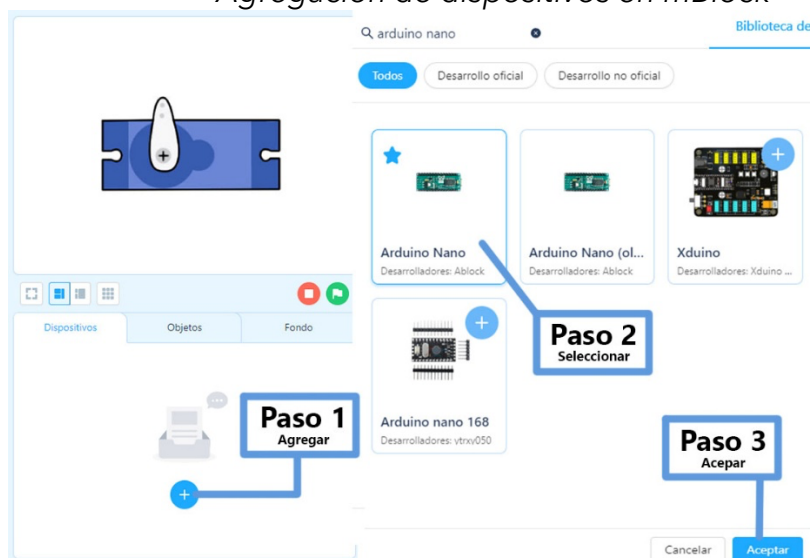
*Nota.* La figura muestra cables de conexión Dupont

## **2.7. Desafíos lógicos secuenciales con servomotores utilizando programación por bloques**

Para empezar a programar en mBlock, primero se debe agregar el tipo de placa Arduino con la que se planea trabajar en la biblioteca de dispositivos (ver figura 24), en este caso se trabaja con la tarjeta Arduino NANO.

**Figura 24**

*Agregación de dispositivos en mBlock*



Nota. La figura muestra la agregación de dispositivos en mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

Se debe conectar la tarjeta Arduino NANO a un puerto USB de la computadora. Ahora debemos conectar la tarjeta Arduino NANO al mBlock (ver figura 25) de acuerdo al puerto COM que fue detectado por mBlock, en este caso se está conectando al puerto COM34.

## Figura 25

### Conexión de tarjeta Arduino NANO a mBlock



Nota. La figura muestra la conexión de tarjeta Arduino a mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

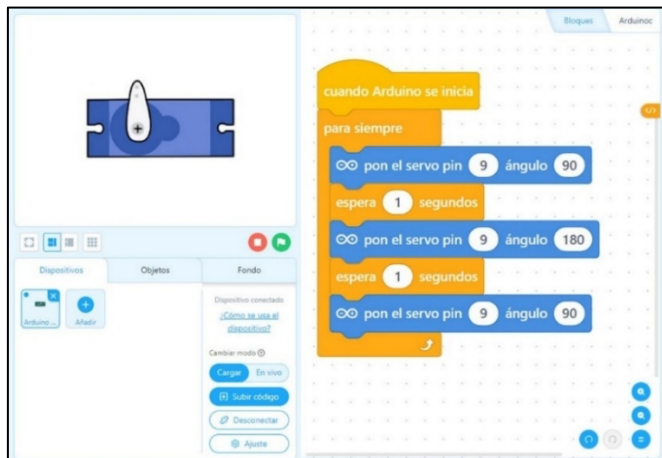
El entorno de programación de mBlock ya está listo y todo conectado correctamente para empezar a programar nuestros desafíos lógicos secuenciales con circuitos electrónicos basados en servomotor.

**Desafío 1:** Programar el movimiento del servomotor de 90 a 180 grados.

**Solución:** La figura 26 muestra la programación por bloques para que el servomotor haga el movimiento de 90 a 180 grados.

## Figura 26

### *Movimiento de servomotor en mBlock*



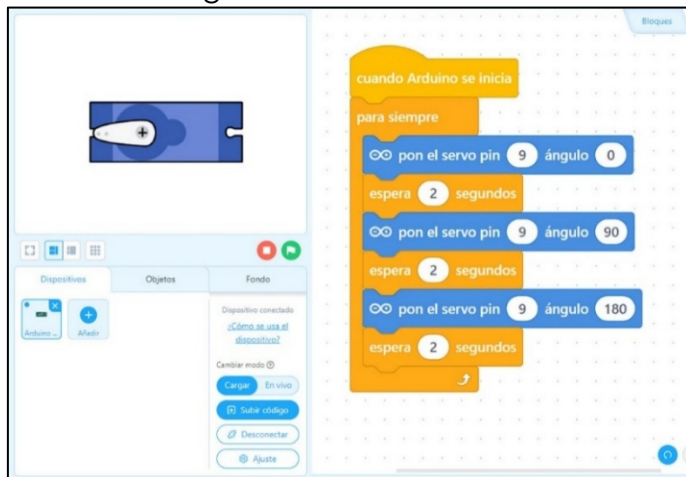
*Nota. La figura muestra la programación por bloques del movimiento en mBlock. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)*

**Desafío 2:** Programar el movimiento secuencial del servomotor de 0 a 90 y 180 grados cada 2 segundos.

**Solución:** La figura 27 muestra la programación por bloques del movimiento secuencial del servomotor.

## Figura 27

### *Programación de movimiento secuencial*



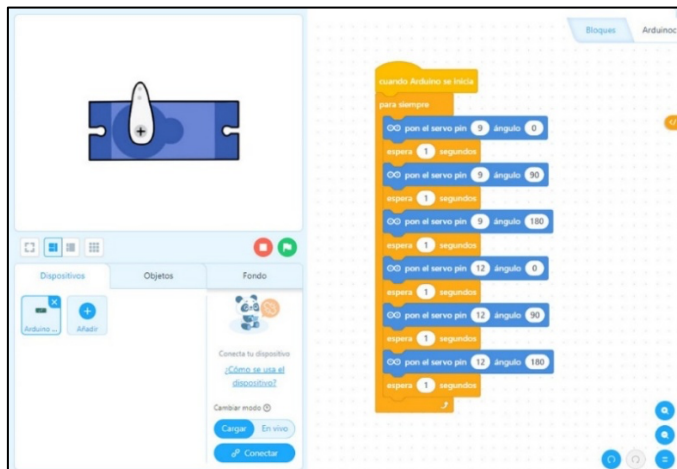
Nota. La figura muestra la secuencia del movimiento del servomotor. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

**Desafío 3:** Programar el movimiento secuencial de 2 servomotores de 0 a 90 y 180 grados.

**Solución:** La figura 28 muestra la programación por bloques del movimiento secuencial de 2 servomotores.

## Figura 28

### *Movimiento secuencial de dos servomotores*

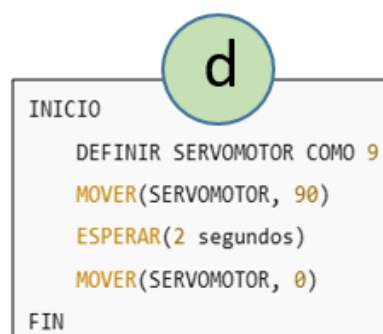
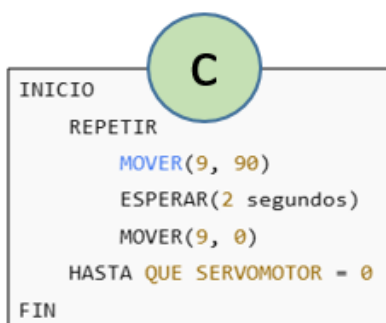
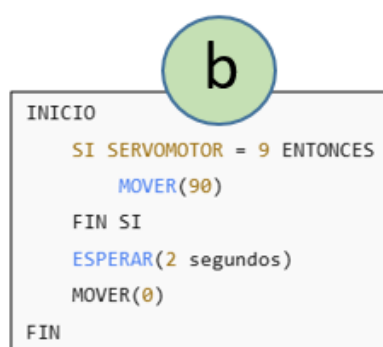


Nota. La figura muestra la programación de movimiento de dos servomotores. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

## 2.7. Autoevaluación del capítulo

**Pregunta 1:** ¿Cuál de los siguientes fragmentos de pseudocódigo representa correctamente una secuencia de instrucciones para mover un servomotor conectado al pin 9 de un Arduino Nano a 90 grados, esperar 2 segundos y luego moverlo a 0 grados?

**Opciones de respuesta:**



**Pregunta 2:** ¿Qué sección se utiliza para arrastrar y soltar los bloques de programación al área de trabajo, permitiendo crear secuencias de código para controlar dispositivos como Arduino o robots?

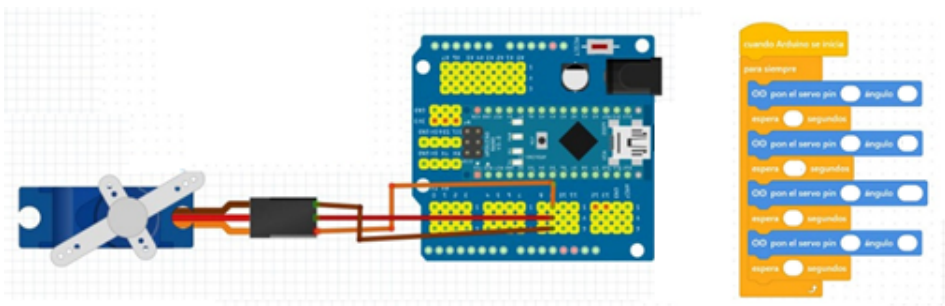
**Opciones de respuesta:**

- a) Panel de Sprites
- b) Área de Escenarios
- c) Área de Scripts (o de Programación)
- d) Barra de Herramientas

**Pregunta 3:** Considerando el siguiente gráfico rellene los círculos blancos con los números adecuados para que el servomotor avance de 0 a 180 cada segundo.

**Figura 29**

*Movimiento secuencial de servomotor*



Fuente: Captura compuesta de las interfaces de Fritzing y mBlock. Fritzing (<https://fritzing.org>) y Makeblock Co., Ltd. (<https://www.mblock.cc>)

**Pregunta 4:** En mBlock, ¿cuál es la función principal de la pestaña "Dispositivos" en la interfaz?

**Opciones de respuesta:**

- a) Seleccionar y configurar hardware como Arduino o mBot.
- b) Dibujar nuevos objetos y personajes para los proyectos.
- c) Ajustar el tamaño y la apariencia del escenario.



d) Escribir código en lenguaje Python en lugar de usar bloques.

**Pregunta 5:** Considerando los tipos de programación que se pueden realizar en mBlock, ¿qué tipo de programación permite su interfaz principal?

**Opciones de respuesta:**

- a) Solo programación en texto con Python.
- b) Programación basada en bloques y en texto (como Python o Arduino C).
- c) Exclusivamente programación en lenguaje C++.
- d) Solo edición de gráficos y animaciones sin programación.

**Pregunta 6:** Durante la fase de investigación y planificación, ¿qué debe hacer el niño según el proceso descrito?

**Opciones de respuesta:**

- a) Memorizar códigos de programación sin aplicarlos.
- b) Investigar cómo funcionan los componentes y planificar los pasos necesarios para que el robot cumpla su objetivo.
- c) Construir el robot sin necesidad de planificación previa.
- d) Usar solo teoría para resolver el problema, sin interactuar con el hardware.

**Pregunta 7:** ¿Qué acciones realiza el niño durante la fase de pruebas del robot?

**Opciones de respuesta:**

- a) Solo observa el robot sin intervenir en su funcionamiento.
- b) Ajusta y corrige errores en las conexiones y en la programación si el robot no funciona como se espera.
- c) Desarma el robot para construir uno nuevo.

d) Cambia los servomotores por otros componentes sin probar primero.

**Pregunta 8:** ¿Cuál es la principal ventaja de usar variables en mBlock para controlar servomotores?

**Opciones de respuesta:**

a) Permite mover el servomotor de forma aleatoria sin necesidad de programación.

b) Facilita almacenar y modificar valores como los ángulos del servomotor durante la ejecución del programa, sin reescribir el código.

c) Hace que el servomotor se detenga automáticamente en cualquier posición.

d) Evita el uso del pensamiento computacional en la programación.

## **CAPÍTULO 3: PROYECTOS INTERDISCIPLINARIOS CON SERVOMOTORES**

### **3.1. Introducción: la Educación con el Aprendizaje Basada en Proyectos**

**Objetivo del capítulo:** Al finalizar este capítulo el futuro docente será capaz de elaborar propuestas de Aprendizaje Basado en Proyectos para enseñanza de Pensamiento Computacional con énfasis en estructuras secuenciales usando mBlock y circuitos electrónicos basados en servomotor.

#### **Aprendizajes Esperados**

- Caracteriza el Aprendizaje Basado en Proyectos
- Elabora propuestas de secuencias didácticas combinadas con Aprendizaje Basada en Proyectos con enfoque interdisciplinar para enseñanza de pensamiento computacional secuencial utilizando programación por bloques usando circuitos electrónicos basados en servomotor.

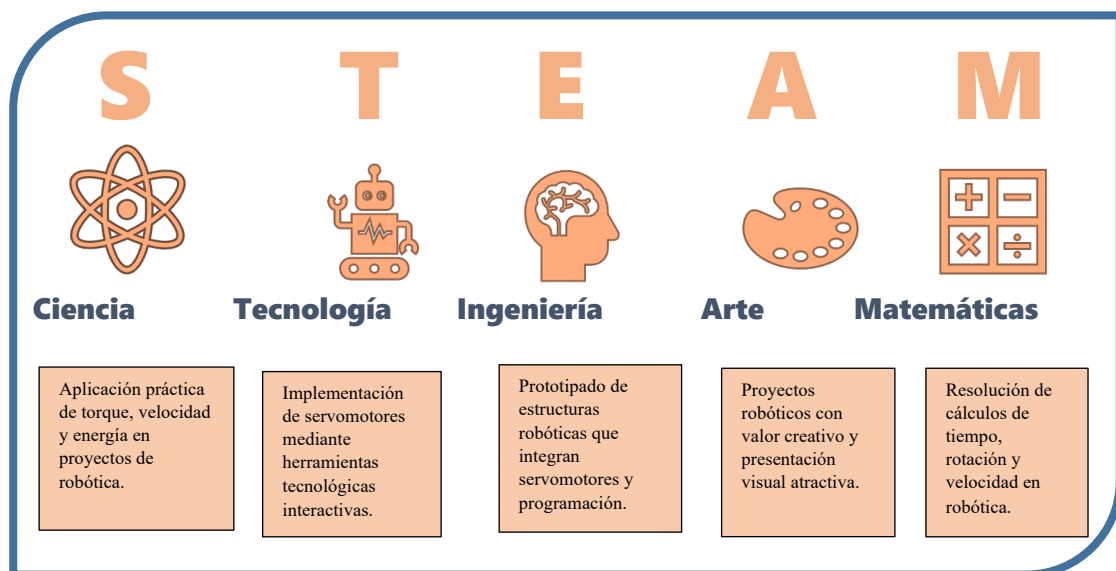
Los temas tratados en este capítulo abarcan de manera integral todas las áreas que componen la metodología STEAM: Ciencia, Tecnología, Ingeniería, Arte y Matemáticas. A través de la creación de proyectos interdisciplinarios, se promueve un enfoque educativo que fusiona estas cinco disciplinas de forma práctica y colaborativa (Correa-Delgado, 2024). Cada una de estas áreas se aborda en conjunto, favoreciendo un aprendizaje más completo y enriquecido, tal como se explica en los siguientes apartados.

Estos proyectos interdisciplinarios abarcan todas las áreas STEAM (ver figura 30), lo que no solo ofrece a los niños la oportunidad de aprender conceptos técnicos y científicos, sino que también promueve el desarrollo de habilidades clave como la creatividad, la colaboración y la capacidad para resolver problemas (Tapullima-Mori et al., 2024). Al combinar teoría con práctica de una manera lúdica, los niños pueden aplicar lo aprendido a situaciones reales, mejorando su comprensión y habilidades en un entorno dinámico y participativo. Además, estos proyectos estimulan su curiosidad, permiten que experimenten y se enfrenten a retos que fomentan el pensamiento crítico, todo mientras disfrutan de un proceso de aprendizaje activo y colaborativo.

Según los aprendizajes esperados, este capítulo tiene como eje central el Aprendizaje Basado en Proyectos con un enfoque interdisciplinario, aplicando metodologías que favorezcan el desarrollo del pensamiento computacional secuencial (Wang, 2023). A través de la utilización de circuitos electrónicos y servomotores, los estudiantes aprenderán a diseñar y programar, adquiriendo habilidades técnicas y cognitivas clave. Este enfoque promueve la integración de conceptos de distintas disciplinas, permitiendo a los niños resolver problemas prácticos y desarrollar competencias de manera dinámica y aplicada.

**Figura 30**

*Gráfico de aprendizaje de las áreas STEAM*



*Nota.* En esta figura se observa las áreas de aprendizaje STEAM que el futuro docente debe lograr.

El aprendizaje basado en proyectos (ABP) es un enfoque pedagógico centrado en el estudiante que implica la realización de proyectos como una forma de adquirir conocimientos y habilidades (Nannim et al., 2024). A través de este enfoque, los estudiantes trabajan de manera activa y colaborativa para investigar, resolver problemas y crear productos que demuestren su aprendizaje. El ABP no se basa solo en la recepción pasiva de información, sino que fomenta la exploración, el análisis crítico y la creatividad.

Algunas características clave del aprendizaje basado en proyectos son:

**Enfoque centrado en el estudiante:** En el ABP, los estudiantes asumen un rol activo en su aprendizaje, dejando de ser receptores pasivos para convertirse en protagonistas

(Rosa et al., 2023). Tienen autonomía para elegir, diseñar, desarrollar y gestionar sus proyectos, con el apoyo del docente. Esto promueve la motivación intrínseca y la responsabilidad sobre su propio aprendizaje. Además, les permite explorar temas de su interés, lo que les da un sentido de pertenencia y relevancia. Este enfoque también fomenta habilidades metacognitivas, como la planificación, la reflexión y la autorregulación, ya que los estudiantes deben ajustar su proceso de trabajo según sea necesario.

**Interdisciplinariedad:** El ABP integra conocimientos y habilidades de diversas áreas, permitiendo a los estudiantes ver conexiones entre disciplinas (Čavić et al., 2023). Esto facilita una comprensión más global y profunda de los temas, promoviendo la transferencia de aprendizajes y abordando problemas complejos desde múltiples perspectivas. Por ejemplo, un proyecto sobre sostenibilidad ambiental puede combinar ciencias, matemáticas, arte y ética, reflejando la naturaleza interconectada de los desafíos del mundo real.

**Aprendizaje activo:** Esta metodología involucra a los estudiantes en tareas prácticas que requieren esfuerzo cognitivo, fomentando la construcción activa de conocimientos a través de la exploración, investigación y resolución de problemas (Team et al., 2023). Los estudiantes no solo reciben información, sino que la aplican, la cuestionan y la transforman mediante la experiencia. Este enfoque dinámico estimula la curiosidad, el pensamiento crítico y la creatividad, además de mejorar la retención del conocimiento al involucrar aspectos cognitivos y emocionales.

**Colaboración:** El ABP promueve el trabajo en equipo, donde los estudiantes aprenden a colaborar, compartir ideas, escuchar a otros y tomar decisiones conjuntas (Jaleniauskiene & Lisaite, 2023). Esto desarrolla habilidades sociales, de comunicación y liderazgo, esenciales tanto en el ámbito laboral como en la vida cotidiana. La colaboración también enseña a negociar, resolver conflictos y trabajar

Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

eficientemente bajo presión, preparando a los estudiantes para entornos profesionales donde el trabajo en equipo es fundamental.

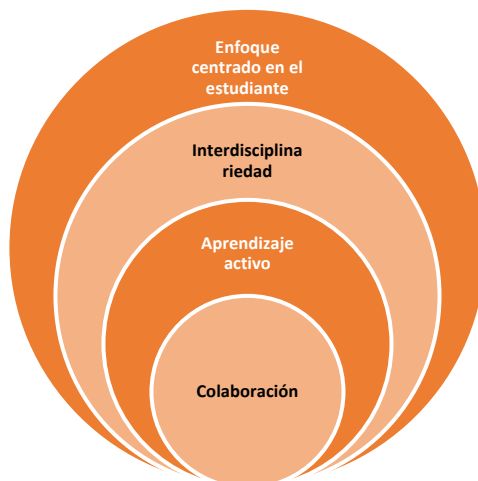
**Evaluación continua:** En el ABP, la evaluación es un proceso constante y formativo. Los estudiantes reciben retroalimentación regular del docente, sus compañeros y de sí mismos, lo que les permite ajustar y mejorar su trabajo (Sari et al., 2023). La evaluación no se limita al resultado final, sino que también considera el proceso: cómo investigaron, colaboraron, resolvieron problemas y reflexionaron sobre su aprendizaje. Este enfoque fomenta una mentalidad de crecimiento y permite personalizar la enseñanza según las necesidades individuales.

**Reflexión:** La reflexión es un componente esencial del ABP, ya que ayuda a los estudiantes a analizar su proceso de aprendizaje, identificar aciertos y áreas de mejora, y aplicar lo aprendido a otras situaciones (DeLisi et al., 2025). Esta práctica no solo ocurre al final del proyecto, sino durante todo el proceso, permitiendo a los estudiantes aprender de sus errores y fortalecer sus habilidades. La reflexión también desarrolla la metacognición, es decir, la capacidad de pensar sobre el propio pensamiento, lo que resulta valioso tanto en el ámbito educativo como en la vida personal y profesional.

A continuación, se presenta un resumen gráfico de las características más importantes del Aprendizaje Basado en Proyectos (ver figura 31).

### **Figura 31**

#### *Características de Aprendizaje Basado en Proyectos*



*Nota.* La figura muestra las características de Aprendizaje Basado en Proyectos.

### **3.2. Aprendizaje Basado en Proyectos (ABP) con énfasis en servomotores**

#### **Conceptualización del ABP relacionada a servomotores**

El Aprendizaje Basado en Proyectos (ABP) es una metodología educativa en la que los estudiantes participan de manera activa y colaborativa en proyectos extensos y complejos (You, 2024). A lo largo de este proceso, investigan, abordan problemas reales y proponen soluciones prácticas, lo que les permite integrar conocimientos de diversas áreas y desarrollar competencias como la creatividad, la investigación y la resolución de problemas. En el contexto del uso de servomotores, el ABP permite integrar conocimientos de electrónica,



Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

programación y mecánica, fomentando un enfoque interdisciplinario alineado con la educación STEAM.

El ABP implica que los estudiantes trabajen en proyectos durante un período específico, siendo evaluados no solo al final, sino de manera continua durante todo el proceso (Tierney et al., 2025). Esta evaluación formativa proporciona retroalimentación constante sobre su progreso, las habilidades adquiridas y el producto final, lo que fomenta la autorreflexión y la mejora continua durante la ejecución del proyecto.

Una característica distintiva del Aprendizaje Basado en Proyectos es su enfoque en la creación de un producto o solución tangible al concluir el proyecto. Este resultado final puede adoptar diversas formas, como un informe, una presentación, un prototipo, entre otros, y representa la aplicación práctica de los conocimientos obtenidos. A lo largo del desarrollo del proyecto, se establece una conexión entre los conceptos teóricos y situaciones o problemas del mundo real, lo que favorece un aprendizaje significativo y relevante.

### **Fases del Aprendizaje Basado en Proyectos relacionadas a servomotores**

El Aprendizaje Basado en Proyectos se estructura en varias fases que guían el proceso desde la planificación hasta la presentación final, Pupik Dean et al. (2023) propone las siguientes fases de Aprendizaje Basado en Proyectos:

*Planteamiento del problema:* Se presenta una situación real o contextualizada que requiere una solución mediante un sistema de movimiento controlado, donde los servomotores juegan un papel clave. Por ejemplo: construir un brazo robótico para mover objetos, diseñar una puerta automática o crear una maqueta interactiva. El objetivo es motivar al

estudiante a investigar, crear y resolver, partiendo de una necesidad significativa.

*Investigación y planificación:* En esta fase, los estudiantes investigan cómo funcionan los servomotores, su estructura interna, el tipo de señales que reciben (PWM), cómo se programan y cómo se integran con plataformas como Arduino o mBlock. También se planifican los recursos necesarios, se distribuyen roles dentro del equipo, y se elabora un cronograma de actividades. Se definen los objetivos del proyecto, sus etapas y posibles productos finales.

*Diseño e implementación del prototipo:* Los estudiantes diseñan la solución física y lógica del proyecto de servomotores. Se conectan los servomotores al sistema de control, se diseña la estructura (puede incluir materiales reciclados, impresiones 3D o piezas mecánicas simples), y se programa el movimiento del servomotor. Esta fase es clave para aplicar conceptos de programación como bucles, condicionales y variables, integrando así habilidades técnicas y pensamiento computacional.

*Prueba, ajuste y mejora:* Con el prototipo armado de servomotores, se realizan pruebas para verificar su funcionamiento. Si el servomotor no se mueve correctamente o no responde como se esperaba, se revisan las conexiones, la programación y la mecánica del sistema. Esta etapa promueve el análisis de errores, la mejora continua y el aprendizaje por descubrimiento, permitiendo múltiples iteraciones hasta lograr un funcionamiento óptimo.

*Presentación del proyecto:* Una vez finalizado el prototipo de servomotores, los estudiantes presentan su proyecto ante sus compañeros, docentes o comunidad. Explican el

Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

propósito del sistema, cómo utilizaron el servomotor, los retos que enfrentaron y cómo los resolvieron. Esta fase fortalece la comunicación, la argumentación técnica y el pensamiento crítico, además de dar valor al esfuerzo realizado.

*Evaluación y retroalimentación:* Como parte final del proyecto de servomotores, se evalúa tanto el proceso como el producto final. Se consideran aspectos técnicos, el nivel de innovación, el cumplimiento de objetivos, la cooperación en el grupo y el grado de aprendizaje logrado. La retroalimentación puede ser dada por el docente, los compañeros o incluso mediante autoevaluación, fomentando la reflexión y la mejora personal y colectiva.

### **3.3. Secuencia Didáctica 1: Un viaje al corazón del servomotor**

#### **Secuencia Didáctica 1**



**"Un Viaje al Corazón  
del servomotor"**

#### **Objetivo General**

El niño comprenderá qué es un servomotor, su estructura y funcionamiento a través de técnicas activas.

## Presentación

Esta **secuencia didáctica** tiene como objetivo guiar al niño en el descubrimiento de los **servomotores**, indicándoles su definición, estructura y funcionamiento, mientras realizan actividades prácticas y creativas para consolidar el conocimiento.

## Estructura de la secuencia didáctica

**Sesión de introducción**

**Sesión 1: Descripción del servomotor**

**Sesión 2: Las partes servomotor**

**Sesión 3: ¿Cómo funciona un servomotor?**

**Sesión de cierre**

### Sesión de Introducción

#### Objetivo específico:

Familiarizar al niño con el servomotor de manera lúdica a través de actividades de pintura y una canción, despertando su curiosidad y creatividad.

#### Materiales:

Pizarra o cartel.

Papel, pinceles y pinturas de colores.

Cartulina o hojas grandes.

Marcadores, lápices y crayones.

Canción relacionada con el movimiento (puedes usar una canción inventada por ti o una canción popular modificada).

Reproductor de música o espacio para cantar.

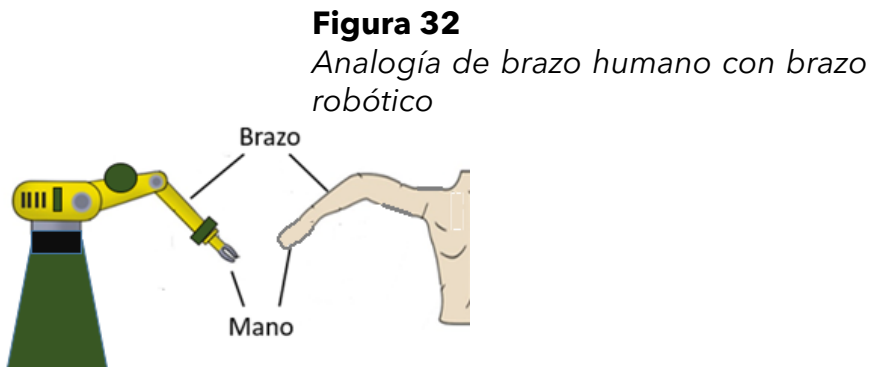
## Actividades:

### Actividad 1: Bienvenida y Presentación:

Da la bienvenida a los niños y empieza indicando que en esta sesión se familiarizarán con un componente importante para el funcionamiento de un robot denominado servomotor. Realiza una pregunta curiosa a los niños: "¿Cómo se mueven las partes de un robot?". Dale pistas: ¿será que algún mecanismo los mueve?, un brazo robótico, por ejemplo, ¿se moverá solo? o ¿alguien por dentro le ayuda a moverse? Hoy lo descubriremos.

Explica brevemente que los robots y algunos juguetes tienen un motor especial llamado **servomotor**, que les permite moverse de manera precisa, y que hoy van a aprender sobre él de una forma muy divertida.

Seguidamente se les puede mostrar imágenes con el movimiento de un brazo robótico haciendo analogía con un brazo humano (ver figura 32).



*Nota.* La figura muestra los movimientos de un brazo robótico semejantes a un humano.



**Canción del Servomotor:** Inventa o adapta una **canción** divertida sobre el servomotor, algo como:

*"El servomotor, el servomotor,  
mueve y detiene, ¡sin error!  
Con su motor, eje y más,  
mueve un robot ¡de un lugar al  
compás!"*

Puedes usar una melodía conocida para que los niños la sigan fácilmente. La idea es que la canción hable sobre cómo el servomotor hace que las cosas se muevan de forma precisa y se detengan dónde deben.

Los niños pueden cantar juntos la canción, haciendo movimientos con las manos que imiten cómo se mueve un servomotor (simulando el giro o el movimiento de un brazo robot, por ejemplo). Esto ayudará a visualizar cómo se mueve un servomotor.

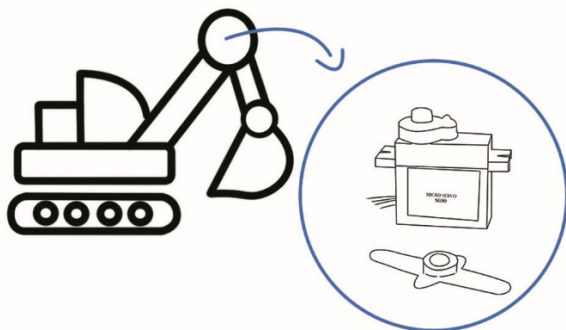
## **Actividad 2: Pinta tu Servomotor**

Ahora, los niños van a pintar su propia versión de un servomotor. Se tienen que usar los colores de fábrica del servomotor para que el niño se vaya familiarizando con los mismos. Los colores de los cables tienen un significado muy particular que se verá en próximos temas.

En una hoja impresa los niños colorean el servomotor, familiarizándose en que parte del robot iría el servomotor (ver figura 33):

### **Figura 33**

*Imagen de robot y un servomotor para colorear*



*Nota.* La figura muestra las partes del servomotor

### **Actividad 3: Socialización y reflexión de lo aprendido**

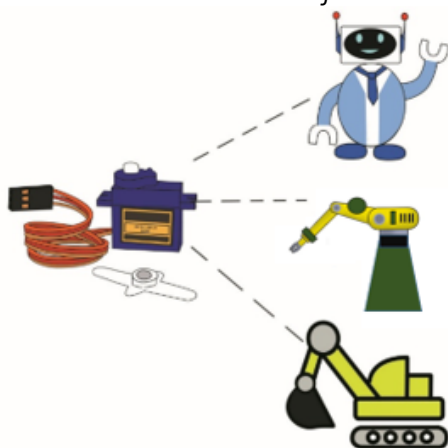
Los niños pueden compartir sus robots con los demás, explicando qué partes pintaron y qué representaba. Durante esta presentación, pueden describir, aunque de manera sencilla, como "este es el motor que ayuda al robot a moverse" o "este motor hace que la palanca del robot se mueva".

## Evaluación de la sesión

Presenta en una hoja que los niños unan con líneas en que parte del robot podría estar colocado (ver figura 34).

**Figura 34**

*Hoja con actividades de unir con líneas*



*Nota.* La imagen muestra una actividad de unir con líneas

Realiza preguntas informales para verificar si comprenden que el servomotor se mueve solo hasta un lugar determinado: ¿Recuerdas cómo se mueve un servomotor? ¿El servomotor podría mover las llantas de un carro?

## Sesión 1: Descripción del servomotor

### **Objetivo específico:**

Introducir el concepto de servomotor y generación de interés sobre su funcionamiento y aplicaciones.



## **Materiales:**

- Robots que usan servomotores.
- Papel y lápices para hacer dibujos.
- Pizarra o cartel para escribir o dibujar.
- Un servomotor real (opcional para mostrar físicamente).

## **Actividades:**

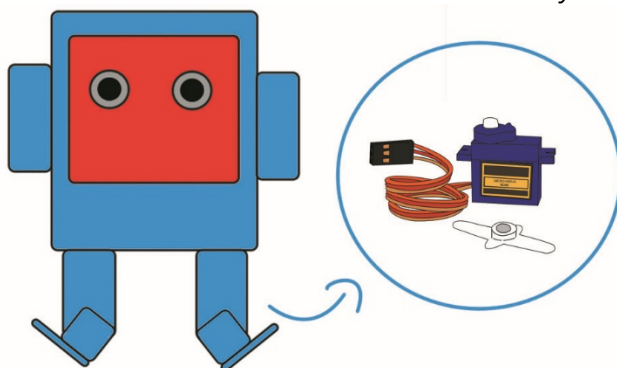
### **Actividad 1:** Bienvenida y Presentación:

Da la bienvenida a los niños y empieza indicando que en esta sesión aprenderán sobre el concepto del servomotor de una manera entretenida y lúdica. con un componente importante para el funcionamiento de un robot denominado servomotor.

Para la explicación de qué es un servomotor, se puede comenzar mostrando un robot real que utilice servomotor o a través de forma gráfica (ver figura 35).

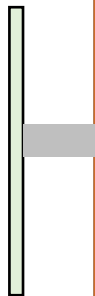
**Figura 35**

*Escenario real de robot y servomotor*



*Nota.* La figura muestra un escenario real de robot y servomotor

Un servomotor es un motor especial que ayuda a mover cosas con mucha precisión. Es como un pequeño motor que no gira todo el tiempo, sino que se detiene en un lugar exacto. Los movimientos son precisos.



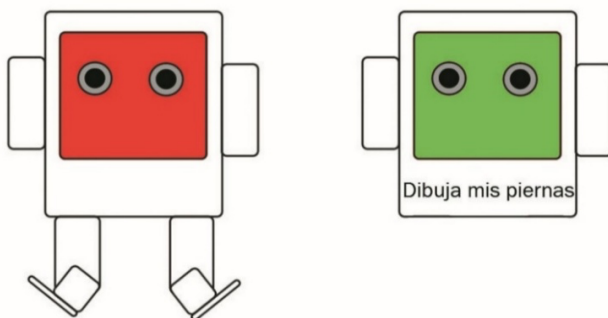
*En el robot humanoide de la figura 5, el servomotor es el motor que le permite mover sus piernas, para caminar o bailar. La precisión de los movimientos evita que el robot caiga o se tropiece.*

## **Actividad 2:** Completa el dibujo de las piernas de un robot

Se les puede hacer que los niños dibujen creativamente las piernas de un robot para que asocien la actividad anterior con el movimiento de las piernas en el dibujo (ver figura 36).

### **Figura 36**

*Escenario para dibujar las piernas de un robot*



*Nota.* La figura muestra un escenario donde el niño debe dibujar las piernas con servomotor.

### Actividad 3: Socialización y reflexión de lo aprendido

Los niños pueden compartir sus dibujos y explicar cómo imagina que el servomotor ayuda a mover las piernas del robot. Pueden describir el funcionamiento básico de las piernas, como "el servomotor mueve las piernas del robot para que pueda caminar o saltar", y señalar en sus dibujos dónde colocaron el servomotor.

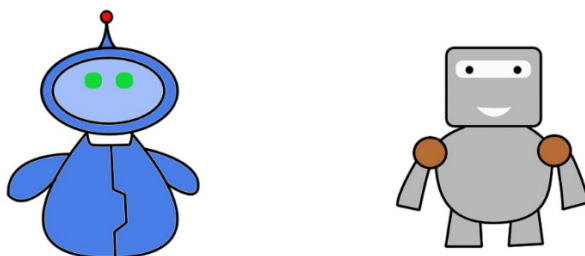
#### Evaluación de la sesión

Valora el dibujo de los niños resaltando las características de los dibujos y la creatividad reflejada, para motivar a los niños por el esfuerzo realizado.

Plantea en una hoja una pregunta relacionada a si se podría colocar un servomotor en la cabeza de un robot. Dale opciones de marcar los robots que podrían tener esta característica (ver figura 37).

**Figura 37**

*Robots con servomotor en la cabeza*



*Nota.* La figura muestra un escenario donde el niño debe seleccionar lo correcto

Anima a los niños a compartir sus dibujos y explicar cómo creen que funciona el servomotor.

## **Sesión 2: Las partes del servomotor**

### **Objetivo específico:**

Identificar las tres partes principales de un servomotor: motor, eje y potenciómetro.

### **Materiales:**

- Pizarra o cartel.
- Papel y lápices.
- Regla o varilla (para simular el eje).
- Clip o cualquier objeto pequeño para representar el eje.
- Un servomotor desarmado (opcional para mostrar las partes físicamente).

### **Actividades:**

#### **Actividad 1: Revisión rápida de la sesión anterior**

En esta sesión empezamos recordando lo que se vio en la sesión anterior, podemos hacer participar a los niños haciendo preguntas como "¿Recuerdan qué es un servomotor? ¿Qué hace?", de esta forma fomentamos la participación activa dentro del aula.

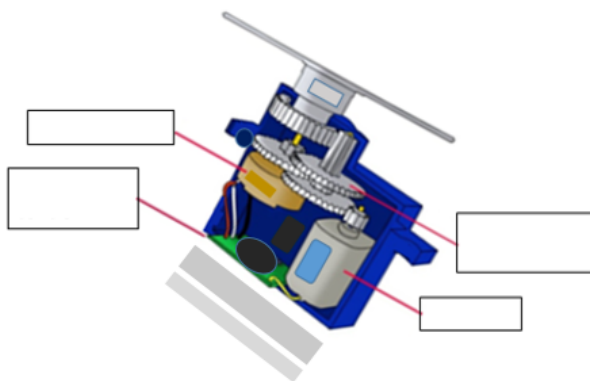
Podemos hacer uso de los dibujos de la sesión anterior para recordar y fortalecer el concepto de lo que es un servomotor.

#### **Actividad 2: Explicación de las partes de un servomotor**

Hacemos uso de la pizarra para hacer un dibujo de un servomotor y sus partes principales: motor, eje, potenciómetro y a continuación explicamos la función de cada uno de sus componentes (ver figura 38).

### Figura 38

#### *Servomotor y sus partes*



*Nota.* La figura muestra el dibujo del servomotor para colocar sus partes en la pizarra

Si tienes un servomotor desarmado, muéstralo a los niños y señala cada parte.

**Motor:** Hace que el servomotor se mueva.

**Eje:** Es la parte que rota cuando el motor se activa.

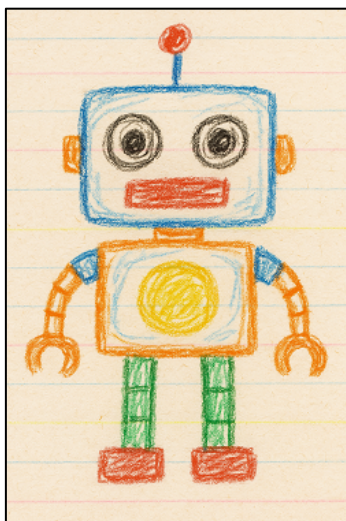
**Potenciómetro:** Controla hasta donde debe girar el motor.

### **Actividad 3: Dibujo de un servomotor y sus partes:**

Pide a los niños que dibujen un robot que use servomotor e indiquen la función de cada una de sus partes (ver figura 39)

**Figura 39**

*Robot que usa servomotor*



*Nota.* La figura muestra un dibujo realizado por un niño

### **Evaluación de la sesión**

Revisa los dibujos y asegúrate que los niños hayan identificado correctamente las partes del servomotor, luego realiza las siguientes preguntas: "¿Qué hace el motor?, ¿Qué es el eje?".

De esta forma podemos identificar si los conceptos explicados en esta sesión fueron aprendidos correctamente.

### **Sesión 3: ¿Cómo funciona el servomotor?**

#### **Objetivo específico:**

Comprender el funcionamiento de un servomotor, como se mueve a posiciones precisas y como se utiliza.

#### **Materiales:**

- Pizarra o cartel.
- Papel y lápices.
- Regla o cuerda para simular el eje del servomotor.
- Clip o similar para representar el eje.
- Un servomotor real (opcional)

#### **Actividades:**

##### **Actividad 1: Revisión rápida de la sesión anterior**

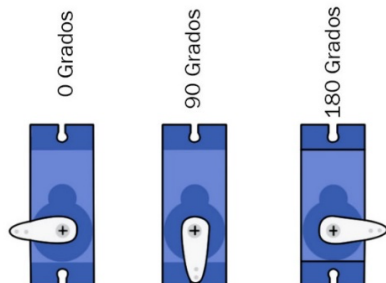
Hacemos una revisión rápida sobre los temas aprendidos en la sesión anterior, repasamos brevemente los conceptos básicos del servomotor y sus partes.

##### **Actividad 2:** Explicación del movimiento del servomotor

Empezamos la clase explicando como el servomotor se mueve hacia un punto específico y como se detiene allí, podemos hacer uso de un servomotor real para demostrar su funcionamiento (ver figura 40).

### Figura 40

#### *Movimiento del servomotor*



*Nota.* La figura muestra el movimiento que realiza el servomotor en grados

### Actividad 3: Simulación de movimiento

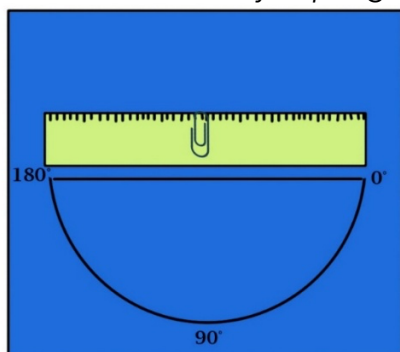
Mediante el uso de materiales sencillos como regla, clips y cartulina explicamos a los niños que el servomotor no se mueve libremente, sino que funciona a través de movimientos controlados.

Para realizar esta actividad pide a los niños que dibujen en la cartulina los grados desde 0 a 180, luego introduce un clip en medio de la regla representando el eje del servomotor, da instrucciones para que lo muevan en varias posiciones según los grados marcados en la cartulina. Explica que el eje (en este caso el clip) gira hasta donde se le indique (ver figura 41).



**Figura 41**

*Ejemplo gráfico de la actividad*



*Nota.* En la figura se muestra la simulación del eje representado por el clip.

## **Evaluación de la sesión**

Para finalizar realiza preguntas a los niños acerca de si tuvieron facilidad para entender el movimiento del servomotor y como es su funcionamiento.

## **Sesión de cierre**

### **Objetivo específico:**

Los niños resumirán lo aprendido sobre los servomotores.

### **Actividades:**

#### **Actividad 1: Juego utilizando tarjetas**

Realiza tarjetas con descripciones de las partes del servomotor, elige alumnos voluntarios para tomar las tarjetas, luego lee en voz alta la descripción y permite que

Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

los demás niños participen indicando el nombre de la parte a la que le corresponde (ver figura 42).

### **Figura 42**

*Ejemplo gráfico de la actividad*



*Nota.* La figura muestra las cartas con las descripciones de las partes del servomotor

### **Actividad 2: Reflexión grupal**

Reúne a los niños y pídeles que compartan lo que aprendieron sobre los servomotores. Interactúa con ellos y hazles preguntas sobre el servomotor, sus partes y su funcionamiento.

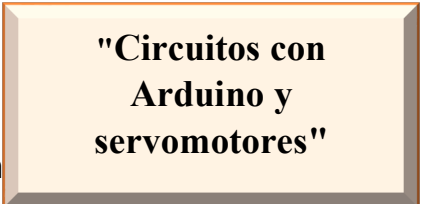
Utiliza el material didáctico utilizado anteriormente para reforzar los conceptos sobre servomotores.

### **Evaluación de la sesión**

Para evaluar los conocimientos impartidos, identifica si los niños son capaces de responder preguntas relacionadas a los servomotores y explicar su utilidad, además si pueden describir su función e identificar adecuadamente las partes que lo componen.

### **3.4. Secuencia Didáctica 2: Circuitos con Arduino y servomotores**

#### **Secuencia Didáctica 2**



##### **"Circuitos con Arduino y servomotores"**

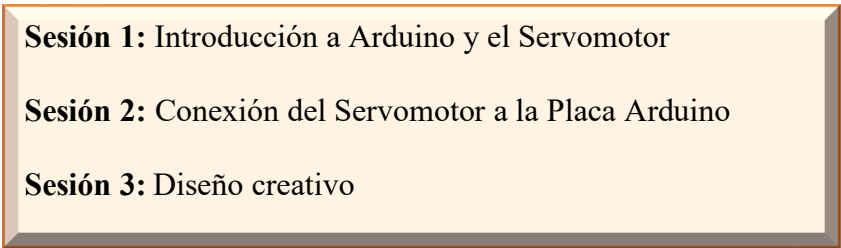
#### **Objetivo General**

Introducir a los niños en el mundo de la robótica educativa mediante el uso de Arduino y servomotores, fomentando la exploración, la creatividad y el pensamiento lógico a través de actividades lúdicas y prácticas.

#### **Presentación**

En esta secuencia didáctica, los niños explorarán el funcionamiento de Arduino y los servomotores mediante actividades prácticas y juegos de roles que faciliten su comprensión.

#### **Estructura de la secuencia didáctica**



**Sesión 1:** Introducción a Arduino y el Servomotor

**Sesión 2:** Conexión del Servomotor a la Placa Arduino

**Sesión 3:** Diseño creativo

## **Sesión 1: Introducción a Arduino y el Servomotor**

### **Objetivos:**

Conocer de una forma más profunda qué es un Arduino y cómo se usa con servomotores.

### **Materiales:**

- Arduino Nano
- Servomotor SG90
- Robot de juguete

### **Actividades:**

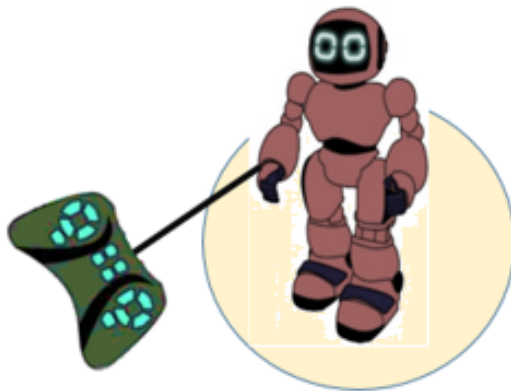
#### **Actividad 1: Exploración de Arduino**

Puedes empezar describiendo las características principales de la placa Arduino como por ejemplo su tamaño, voltaje de operación, conectividad, etc.

Explica que Arduino es como el "cerebro" de un robot, para realizar esta actividad de manera práctica utiliza como ejemplo un robot de juguete y describe como un robot necesita control para realizar movimientos (ver figura 43).

**Figura 43**

*Robot de juguete*



*Nota.* En esta figura se demuestra un robot con movimientos controlados

## **Actividad 2: Juego didáctico**

Para esta actividad le pediremos a dos niños que participen realizando el papel del Arduino y de un robot, en donde el niño que cumple la función del Arduino será el encargado de dar las órdenes como, por ejemplo: avanzar, girar, detenerse y el niño que hace el papel de robot deberá cumplir con las órdenes dadas.

Puedes pedirles que cambien de roles con los demás niños para fomentar la participación (ver figura 44).

### **Figura 44**

*Demostración gráfica de la actividad*



Nota. En esta figura se observa el juego de roles entre los niños.

### **Evaluación de la sesión:**

Pide a los niños que describan con sus propias palabras que es el Arduino y que función cumple, en base a las actividades realizadas.

## **Sesión 2: Conexión del Servomotor a la Placa Arduino**

### **Objetivo:**

Reforzar el conocimiento en los niños sobre como conectar un servomotor a la placa Arduino y observar cómo controla el movimiento del servomotor.

### **Materiales:**

- Servomotor
- Arduino Nano

## Actividades:

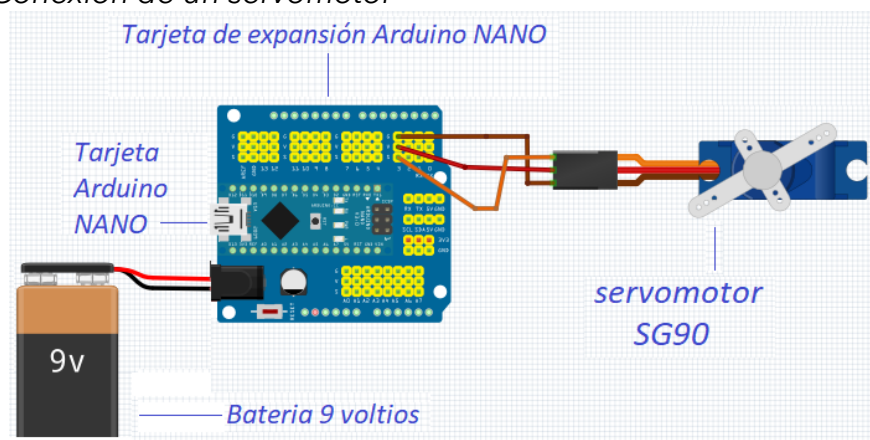
### Actividad 1: Conexión del Servomotor a Arduino

Explica cómo conectar el servomotor a los pines de Arduino, utilizando cables de colores para que los niños asocien cada pin con su función (por ejemplo, rojo = energía, amarillo = señal).

Mientras los niños observan y tienes su atención, realiza la conexión e intenta animarlos para que formulen preguntas acerca del proceso que acabas de realizar (ver figura 45).

**Figura 45**

*Conexión de un servomotor*



Nota. En esta figura se observa el servomotor conectado a Arduino. Fuente: Captura de pantalla de la interfaz de Fritzing. Fritzing (<https://fritzing.org>)

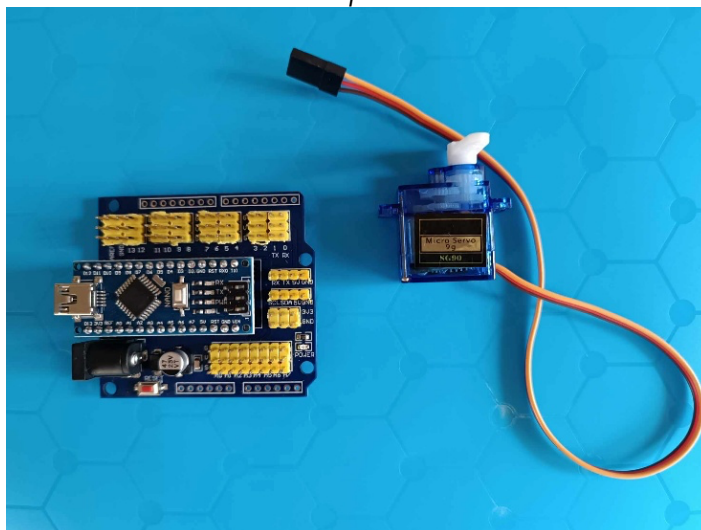
## Actividad 2: Actividad Práctica

En esta actividad puedes hacer que los niños formen grupos para que cada grupo pueda conectar su propio servomotor al Arduino.

Bajo tu supervisión dale a cada grupo un Arduino y un servomotor y observa que realicen la conexión correctamente (ver figura 46).

**Figura 46**

*Recursos para realizar la actividad*



*Nota.* La figura muestra el servomotor y Arduino para realizar la actividad

## Evaluación de la sesión

Por medio de la observación directa, analiza si los niños siguen las instrucciones correctamente y si logran conectar el servomotor al Arduino.



## **Sesión 3: Diseño creativo**

### **Objetivo:**

Incentivar la creatividad de los niños de tal manera que puedan hacer uso de su imaginación para realizar un diseño que simule un brazo robótico o accesorio similar para colocar en el servomotor y posteriormente conectarlo al Arduino de forma independiente.

### **Materiales:**

- Servomotor SG90
- Arduino Nano
- Cartulinas, Tijeras
- Lápices, Marcadores
- Pegamento, Cinta
- Paletas

### **Actividades:**

#### **Actividad 1: Taller práctico**

Indica a los niños que para la siguiente actividad van a realizar un trabajo práctico en el que deberán hacer uso de su ingenio y creatividad para crear un accesorio y colocarlo en el servomotor, dales todo el material necesario para que puedan explorar su imaginación (ver figura 47).

### **Figura 47**

*Accesorios realizados para la actividad*



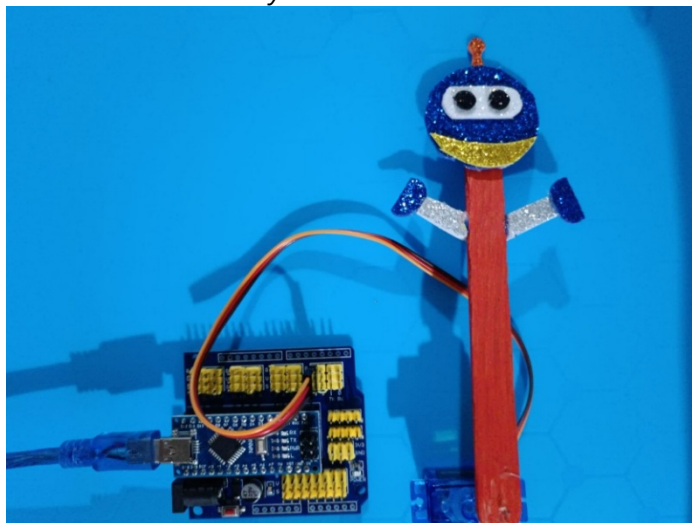
*Nota. La figura muestra los accesorios realizados por niños*

### **Actividad 2: Conexión y Prueba**

Una vez que los niños hayan terminado de realizar su diseño, procederán a realizar el siguiente paso de esta sesión, unir su diseño al servomotor para luego conectarlo de forma independiente al Arduino y ver que funcione correctamente (ver figura 48).

### **Figura 48**

*Unión y conexión de accesorio*



*Nota.* La figura muestra el ensamble completo del accesorio, servomotor y Arduino

### **Evaluación de la sesión**

Deberás evaluar el accesorio realizado por los niños teniendo en cuenta la creatividad y originalidad que han tenido al hacerlo.

Tendrás en cuenta que hayan aplicado todo lo aprendido en las sesiones anteriores como saber que es el Arduino, cuál es su función y como realizar la conexión con un servomotor correctamente.

### **3.5. Secuencia Didáctica 3: Primeros pasos en programación con mBlock**

## Secuencia Didáctica 3

### "Primeros pasos en programación con mBlock"

## Objetivo General

Desarrollar en los niños habilidades básicas de programación mediante la experimentación con servomotores y el uso de bloques en mBlock, fomentando la comprensión de conceptos como secuencias, tiempos de espera y sincronización en robótica.

## Presentación

En esta secuencia didáctica, los niños aprenderán a programar servomotores utilizando bloques en mBlock, facilitando la comprensión de la relación entre la programación y el movimiento en dispositivos físicos.

## Estruc

A través de actividades prácticas y desafíos progresivos, los niños explorarán el funcionamiento del servomotor, aprendiendo a controlar su movimiento con precisión y ajustando parámetros como ángulos y tiempos de espera.

**Sesión 1:** Programando el movimiento del servomotor

**Sesión 2:** Programando el servomotor considerando el tiempo

**Sesión 3:** Programando el movimiento secuencial de dos servomotores

## **Sesión 1: Programando el movimiento del servomotor**

### **Objetivos:**

Comprender cómo programar un servomotor para que se mueva a diferentes posiciones utilizando bloques en mBlock.

### **Materiales:**

- Arduino Nano
- Servomotor SG90
- Ordenador con mBlock
- Fuente de alimentación (cable USB)
- Pizarra

### **Actividades:**

#### **Actividad 1: ¿Cómo se mueve el servomotor?**

En esta primera actividad explicamos que el servomotor puede moverse en diferentes ángulos, lo podemos comparar con partes del cuerpo (ejemplo: girar la cabeza de un lado a otro).

En mBlock los niños explorarán el bloque "pon el servo en pin (x) ángulo (x)"(ver figura 49).

### Figura 49

*Código para el movimiento del servomotor*



Nota. La figura muestra la programación por bloques para el movimiento. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

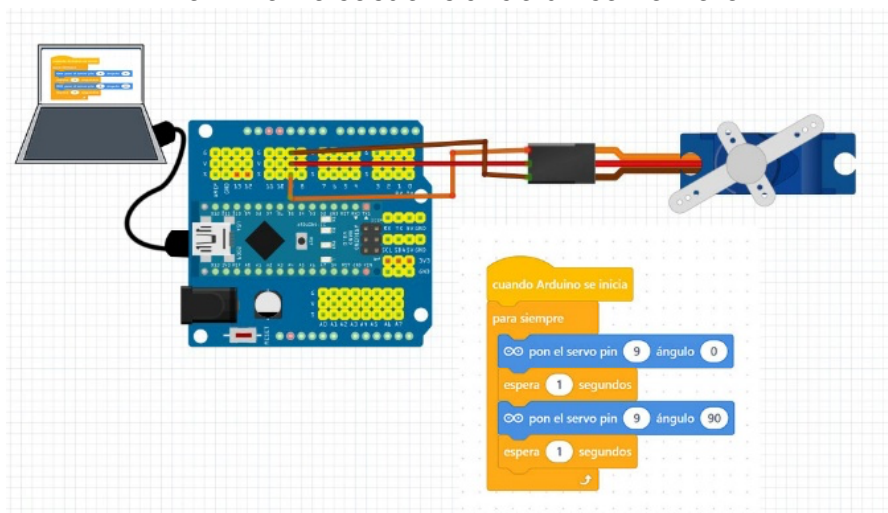
### Actividad 2: Programando mi primer movimiento

Les proporcionamos instrucciones claras a los niños y les planteamos el siguiente desafío: lograr que el servomotor se mueva de un lado a otro. Para ello, deben seguir una serie de indicaciones que les ayudarán a cumplir con el propósito de la actividad. Una de las reglas es programar el servomotor para que se desplace entre 0° a 90°, utilizando la programación por bloques en mBlock.

Si los niños presentan alguna dificultad al momento de realizar esta actividad puedes hacer uso de la pizarra para escribir el orden de los bloques de deben usar en esta actividad (ver figura 50).

### Figura 50

#### *Movimiento secuencial de un servomotor*



Nota. La figura muestra la conexión de un servomotor y programación de movimiento. Fuente: Captura compuesta de las interfaces de Fritzing y mBlock. Fritzing (<https://fritzing.org>) y Makeblock Co., Ltd. (<https://www.mblock.cc>)

### Evaluación de la sesión

Podemos hacer preguntas a los niños sobre como hicieron para que el servomotor se mueva y que bloques usaron en su programación.

## **Sesión 2: Programando el servomotor considerando el tiempo**

### **Objetivo:**

Explorar cómo el tiempo afecta el movimiento del servomotor en un programa de mBlock.

### **Materiales:**

- Servomotor
- Arduino Nano
- Ordenador con mBlock
- Fuente de alimentación (cable USB)

### **Actividades:**

#### **Actividad 1: Explorando los tiempos**

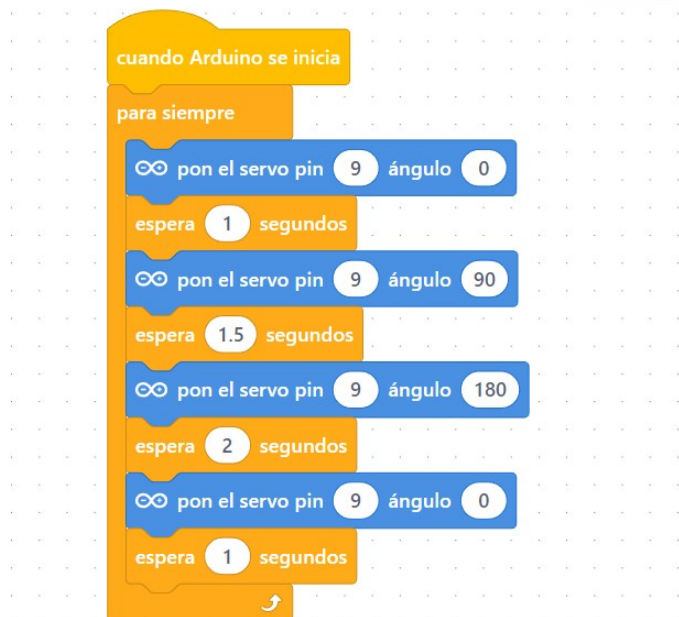
Explicamos que el servomotor puede moverse rápido o lento dependiendo del tiempo que se le asigne en los bloques de espera, explicamos que en la programación y la robótica el tiempo de espera es fundamental para controlar el movimiento de los servomotores.

Realizamos un ejemplo del movimiento del servomotor considerando los tiempos de espera (ver figura 51).



**Figura 51**

*Código para movimiento por tiempo*



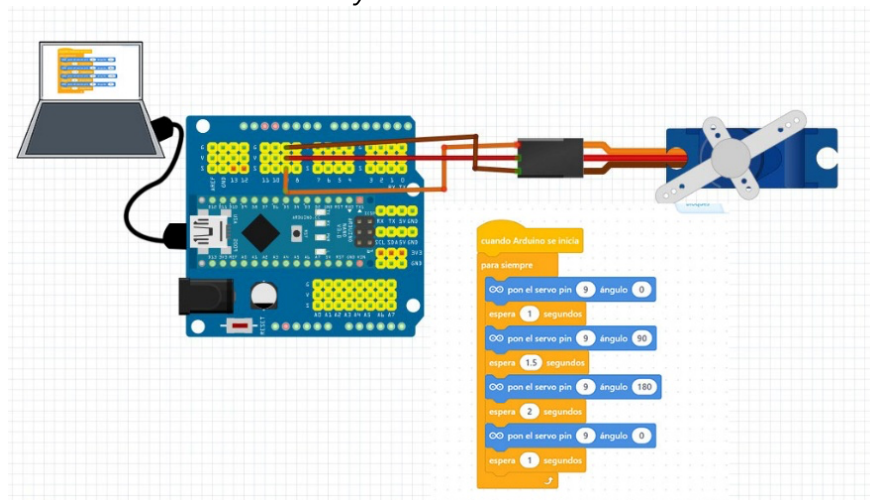
Nota. La figura muestra la programación de movimiento por tiempo. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

## **Actividad 2: Reto del tiempo exacto**

Invitamos a los niños a programar movimientos específicos para el servomotor, diseñando sus propios movimientos. Para ello, completan una tabla en la que anotan los ángulos y el tiempo de espera de cada posición. Luego, utilizan mBlock para ajustar los bloques según su secuencia y prueban su código, observando cómo el servomotor ejecuta la rutina programada. (ver figura 52).

## Figura 52

### Conexión de servomotor y duración de movimiento



Nota. La figura muestra la conexión de un servomotor considerando la duración del movimiento. Fuente: Captura compuesta de las interfaces de Fritzing y mBlock. Fritzing (<https://fritzing.org>) y Makeblock Co., Ltd. (<https://www.mblock.cc>)

## Evaluación de la sesión

Realizamos preguntas de reflexión:

¿Cómo podemos hacer que se mueva más rápido o más lento?

¿Qué bloque de programación hacemos uso para controlar el tiempo?

## Sesión 3: Programando el movimiento secuencial de dos servomotores

### Objetivo:

Aprender a programar dos servomotores para que se muevan en secuencia.

### **Materiales:**

- Servomotor SG90
- Arduino Nano
- Ordenador con mBlock
- Fuente de alimentación (cable USB)

### **Actividades:**

#### **Actividad 1: Coordinación de los servomotores**

Se presenta la idea de que es posible programar dos servomotores para que se muevan simultáneamente o en distintos momentos. Para ilustrarlo, se realiza un ejemplo visual en el que dos niños giran los brazos a diferentes ritmos, demostrando cómo pueden coordinarse o moverse de forma independiente, al igual que los servomotores en un programa.

Mostramos un ejemplo de dos servomotores moviéndose simultáneamente (ver figura 53).

### Figura 53

*Código para movimiento secuencial de dos servomotores*



*Nota.* La figura muestra la programación de movimiento secuencial de dos servomotores. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

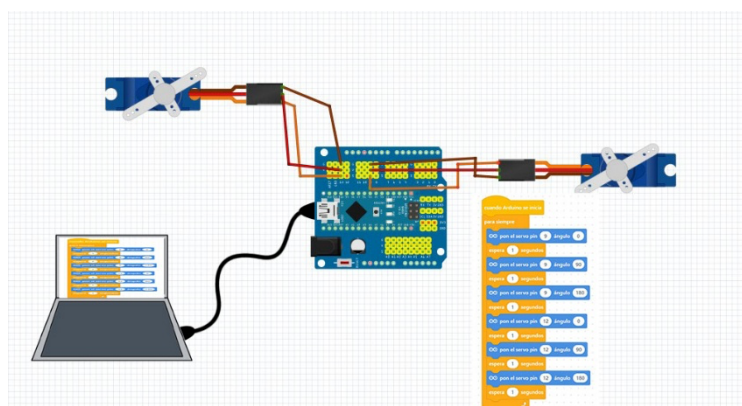
### Actividad 2: Sincronizando servomotores

Para esta actividad podemos formar grupos en donde los niños deberán diseñar una secuencia personalizada de movimientos para dos servomotores. Para ello, primero crean un esquema en una hoja, anotando los ángulos correspondientes. Luego, trasladan su diseño a mBlock, programando los bloques según su secuencia y ejecutando el código para observar el resultado. Finalmente, cada grupo comparte su trabajo con los demás, explicando cómo programaron sus servomotores.

Hay que tener en cuenta las siguientes observaciones, los dos servomotores deben moverse de manera coordinada o en secuencia según lo que se haya planteado en el diseño de la actividad. Esto significa que los movimientos de ambos

servomotores pueden ocurrir simultáneamente o de forma alternada, dependiendo de la secuencia que los niños hayan creado. Si se opta por un movimiento coordinado, ambos servomotores deben ejecutarse al mismo tiempo, asegurando que se muevan juntos en los mismos tiempos o ángulos. Si se elige una secuencia, el movimiento de un servomotor debe iniciarse solo después de que el primer servomotor haya completado su acción, lo que les permite a los niños explorar la idea de sincronización y cómo gestionar el orden y el ritmo de las acciones dentro de un programa (ver figura 54).

**Figura 54**



*Conexión de dos servomotores*

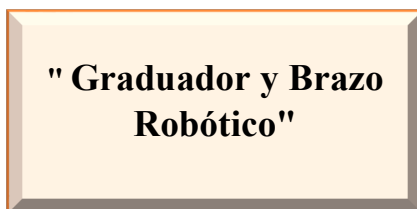
*Nota.* La figura muestra la conexión y movimiento secuencial de dos servomotores. Fuente: Captura compuesta de las interfaces de Fritzing y mBlock. Fritzing (<https://fritzing.org>) y Makeblock Co., Ltd. (<https://www.mblock.cc>)

## **Evaluación de la sesión**

Cada grupo tendrá la oportunidad de explicar su secuencia de movimientos y compartir el proceso que siguió para crearla. Durante esta explicación, podrán detallar las decisiones que tomaron respecto a los ángulos, el tiempo de espera y el orden de los movimientos.

### **3.6. Secuencia Didáctica 4: Diseño de un Graduador de Ángulos y Brazo Robótico con Servomotores**

#### **Secuencia Didáctica 4**



#### **Objetivo General**

Los estudiantes aprenderán a calcular y programar ángulos específicos utilizando servomotores y Arduino, integrando conceptos de matemáticas, y programación. Este proyecto les permitirá comprender cómo los servomotores se mueven con precisión para alcanzar ángulos determinados y

#### **Presentación**

Esta secuencia didáctica propone el desarrollo de un proyecto práctico en el que los alumnos diseñarán y programarán un graduador de ángulos controlado por servomotores y Arduino. Durante el transcurso de cinco sesiones, los estudiantes explorarán tanto la teoría como la práctica detrás de la medición de ángulos y el control preciso de movimientos.

## Estructura de la secuencia didáctica

**Sesión 1:** Introducción a los ángulos y servomotores

**Sesión 2:** Diseño del graduador de ángulos y brazo robótico

**Sesión 3:** Programación de ángulos específicos

**Sesión 4:** Simulación de aplicaciones prácticas

**Sesión 5:** Presentación final y reflexión

- Estructura para el graduador (puede ser de cartón, madera o impresión 3D)
- Transportador o regla para medir ángulos
- Computadoras con acceso a mBlock
- Paletas

**Duración:** 5 sesiones de aproximadamente 2 horas cada una.

### **Sesión 1: Introducción a los ángulos y servomotores**

#### **Objetivos:**

- Familiarizar a los estudiantes con el concepto de ángulos y su medición.
- Introducir el funcionamiento de los servomotores y su capacidad para moverse en ángulos específicos.

### Actividades:

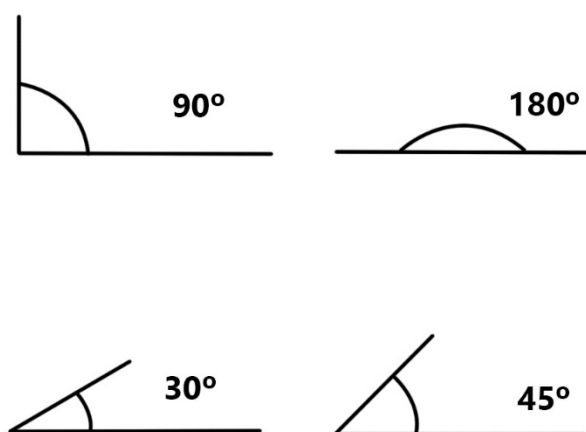
Explicar qué es un ángulo y cómo se mide, en la figura 55 se observan los diferentes ángulos.

Mostrar ejemplos de aplicaciones de ángulos en la vida real, como puentes, grúas o brazos robóticos.

Conectar un servomotor al Arduino Nano y programarlo para que gire a un ángulo específico (por ejemplo, 90 grados).

**Figura 55**

*La figura muestra ángulos comunes*



*Nota.* La figura muestra la medición de distintos ángulos.



## **Evaluación de la sesión**

Verificar que los estudiantes comprenden cómo medir ángulos y pueden programar un servomotor para moverse a un ángulo determinado.

## **Sesión 2: Diseño del graduador de ángulos y brazo robótico**

### **Objetivos:**

- Diseñar la estructura del graduador de ángulos, brazo robótico y planificar cómo se integrará el servomotor.
- Conectar el servomotor al Arduino Nano y probar su funcionamiento.

### **Actividades:**

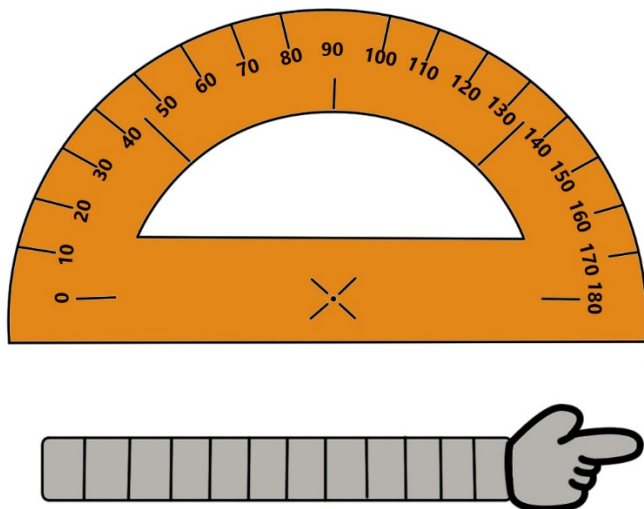
Los estudiantes diseñan la estructura del graduador, que incluirá una base y un brazo móvil controlado por el servomotor (ver figura 56).

Conectar el servomotor al Arduino Nano y programarlo para que el brazo se mueva a diferentes ángulos (por ejemplo, 45°, 90°, 180°).

Probar los movimientos del brazo y ajustar la estructura si es necesario.

### **Figura 56**

*La figura muestra el graduador de ángulos y brazo robótico*



*Nota. La figura muestra el diseño de un graduador y brazo robótico para el proyecto*

### **Evaluación de la sesión**

Asegurarse de que los estudiantes puedan conectar y controlar el servomotor correctamente.

## **Sesión 3: Programación de ángulos específicos**

### **Objetivos:**

- Programar el servomotor para que el brazo del graduador se mueva a ángulos específicos indicados por el docente.
- Introducir conceptos de precisión y calibración en el movimiento del servomotor.

### **Actividades:**

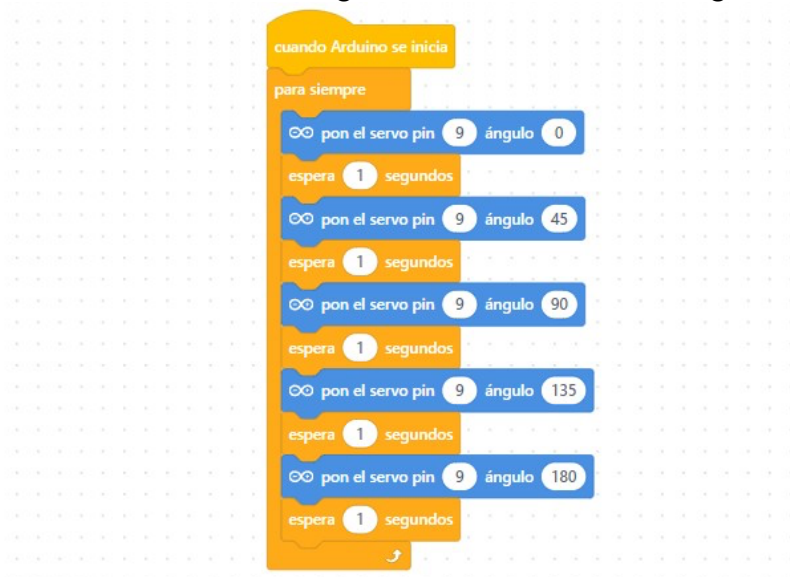
El docente proporciona una lista de ángulos (por ejemplo, 30°, 60°, 120°) que los estudiantes deben programar en el servomotor.

Los estudiantes ajustan el código para que el brazo se mueva exactamente a los ángulos indicados (ver figura 57).

Utilizar una regla para verificar la precisión del movimiento del brazo.

**Figura 57**

*Programación de distintos ángulos*



Nota. La figura muestra la programación por bloques de varios ángulos para la actividad. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

## **Evaluación de la sesión**

Observar si los estudiantes logran programar el servomotor para alcanzar los ángulos indicados con precisión.

## **Sesión 4: Simulación de aplicaciones prácticas**

### **Objetivos:**

- Programar el graduador para simular aplicaciones prácticas, como medir ángulos en estructuras o seguir trayectorias específicas.
- Integrar conceptos de física y matemáticas en el proyecto.

### **Actividades:**

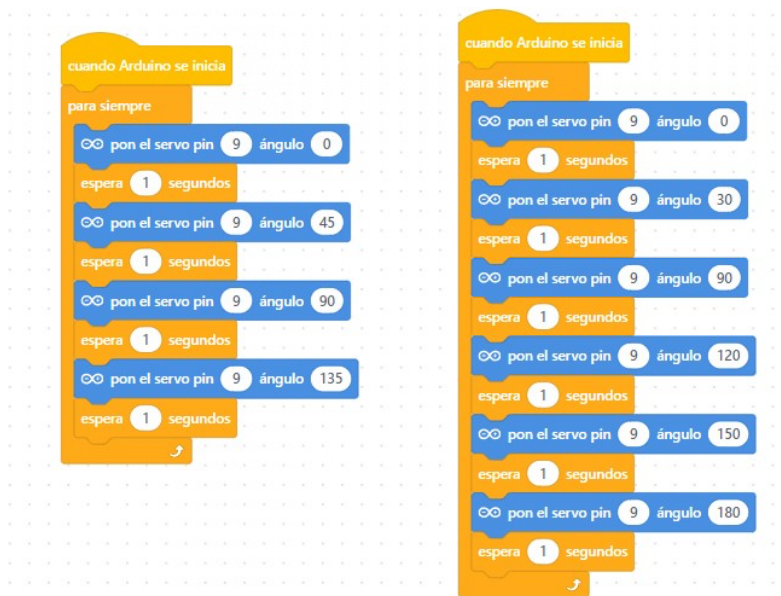
Los estudiantes programan el graduador para simular el movimiento de una grúa (ver figura 58), siguiendo una secuencia de ángulos (por ejemplo,  $0^\circ \rightarrow 45^\circ \rightarrow 90^\circ \rightarrow 135^\circ$ ).

Añadir condiciones en el código para que el brazo regrese a su posición inicial después de completar la secuencia.

Probar y ajustar el programa para mejorar la precisión y fluidez del movimiento.

### Figura 58

*Secuencia de movimientos de grúa y ventilador*



Nota. La figura muestra la programación por bloques del movimiento del brazo de una grúa(izquierda) y un ventilador(derecha). Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

### Evaluación de la sesión

Verificar que los estudiantes puedan programar secuencias de ángulos y simular aplicaciones prácticas.

## **Sesión 5: Presentación final y reflexión**

### **Objetivos:**

- Presentar el graduador de ángulos funcionando y explicar su diseño y programación.
- Reflexionar sobre los aprendizajes adquiridos en matemáticas, física y programación.

### **Actividades:**

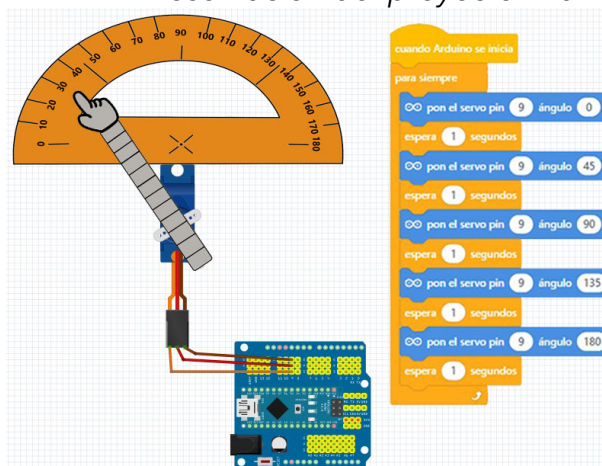
Los estudiantes presentan su graduador de ángulos, demostrando cómo puede moverse a ángulos específicos y seguir secuencias programadas (ver figura 59).

Explican cómo integraron el servomotor y cómo resolvieron los desafíos técnicos.

Reflexionan sobre cómo este proyecto les ayudó a comprender conceptos de ángulos, movimiento y programación.

**Figura 59**

*Presentación del proyecto final*



*Nota.* La figura muestra el diseño final del proyecto. Fuente: Captura compuesta de las interfaces de Fritzing y mBlock. Fritzing (<https://fritzing.org>) y Makeblock Co., Ltd. (<https://www.mblock.cc>)

## Evaluación de la sesión

Evaluar la capacidad de los estudiantes para presentar su proyecto y explicar los conceptos aprendidos.

## Conclusión:

Este proyecto permite a los estudiantes explorar el mundo de los ángulos y el movimiento de manera práctica, integrando conceptos de matemáticas, física y tecnología. Al diseñar y programar un graduador de ángulos con servomotores, los estudiantes desarrollan habilidades de programación, precisión y resolución de problemas, mientras comprenden cómo los conceptos teóricos se aplican en contextos reales.



### **3.7 Secuencia Didáctica 5: Indicador de las Estaciones del Año**

#### **Secuencia Didáctica 5**



**"Indicador de las  
Estaciones del Año"**

#### **Objetivo General**

Que los estudiantes comprendan las características de las estaciones del año y se familiaricen con el funcionamiento del servomotor SG90, integrando conceptos de lógica de programación y ángulos para diseñar y construir un indicador visual interactivo.

#### **Presentación**

Esta secuencia didáctica invita a los estudiantes a explorar el mundo de las estaciones del año de manera práctica y lúdica, fusionando conocimientos de ciencias y tecnología. A lo largo de cinco sesiones, los alumnos descubrirán las particularidades de cada estación (primavera, verano, otoño e invierno) y aprenderán a programar un servomotor SG90 para crear un indicador visual que represente estos cambios.

## Estructura de la secuencia didáctica

**Sesión 1:** Introducción a las estaciones del año y al servomotor SG90

**Sesión 2:** Diseño del indicador de estaciones

**Sesión 3:** Programación del indicador de estaciones

**Sesión 4:** Personalización y Mejora del Indicador de Estaciones

**Sesión 5:** Presentación final y reflexión

### Recursos necesarios:

Arduino Nano

Servomotor SG90

Estructura para el indicador (puede ser de cartón, madera o materiales reciclados)

Imágenes o dibujos que representen las estaciones (primavera, verano, otoño, invierno)

Computadoras con acceso a mBlock (programación por bloques)

Herramientas básicas (tijeras, pegamento, cinta)

**Duración:** 5 sesiones de aproximadamente 2 horas cada una.

## Sesión 1: Introducción a las estaciones del año y al servomotor SG90

### Objetivo:

- Familiarizar a los estudiantes con las estaciones del año y sus características.

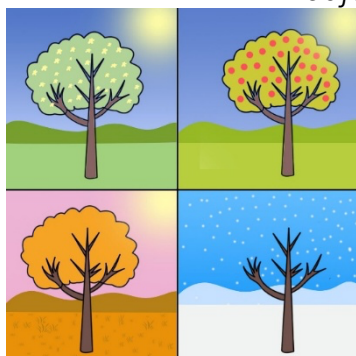
### Actividades:

Explicación de las estaciones del año (ver figura 60).

Explicar las características de cada estación (clima, colores, actividades típicas).

**Figura 60**

*Dibujos de las cuatro estaciones del año*



*Nota.* La figura muestra las cuatro estaciones del año: primavera, verano, otoño, invierno.

## **Evaluación de la sesión**

Verificar que los estudiantes comprenden las estaciones del año y pueden conectar y programar el servomotor correctamente.

## **Sesión 2: Diseño del indicador de estaciones**

### **Objetivos:**

- Diseñar y construir un indicador visual de las estaciones del año.
- Conectar el servomotor al indicador y probar su movimiento.

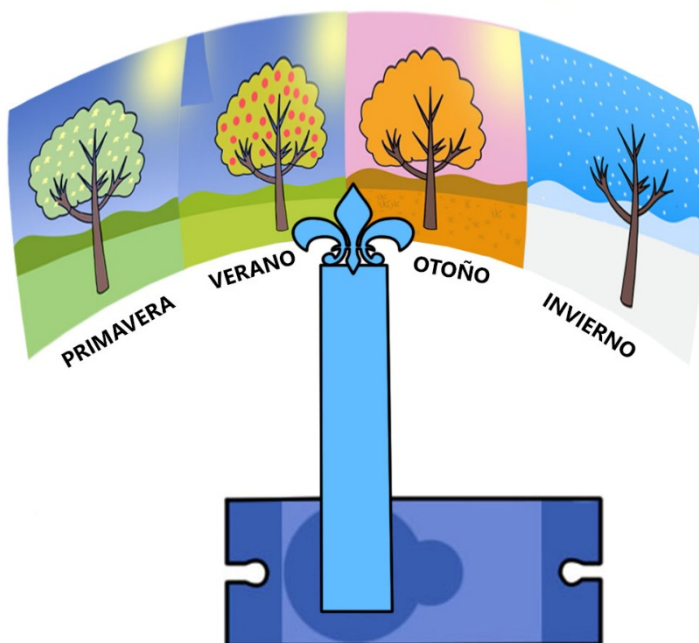
### **Actividades:**

Los estudiantes diseñan un panel dividido en cuatro secciones (ver figura 61), cada una representando una estación del año (primavera, verano, otoño, invierno).

Colocar imágenes o dibujos de cada estación en su sección correspondiente.

### **Figura 61**

*Diseño del indicador de estaciones*



*Nota.* La figura muestra un indicador de estaciones usando un servomotor

### **Evaluación de la sesión**

Asegurarse de que los estudiantes puedan conectar y controlar el servomotor correctamente.

## **Sesión 3: Programación del indicador de estaciones**

### **Objetivos:**

- Programar el servomotor para que el brazo indicador apunte a cada estación del año.
- Introducir conceptos de lógica básica en programación.

### **Actividades:**

Programación por bloques en mBlock:

Usar mBlock para programar el servomotor y mover el brazo indicador a las posiciones correspondientes a cada estación:

0°: Primavera.

45°: Verano.

90°: Otoño.

135°: Invierno.

Simulación de cambio de estaciones:

Los estudiantes programan el brazo de acuerdo a figura 62 para que se mueva entre las estaciones según una secuencia (por ejemplo, primavera → verano → otoño → invierno).

**Figura 62**

*Secuencia de estaciones del año programadas en mBlock*



*Nota.* La figura muestra la programación por bloques las cuatro estaciones del año. Fuente: Captura de pantalla tomada de la interfaz de mBlock v5.4.0. Makeblock Co., Ltd. (<https://www.mblock.cc>)

## Evaluación de la sesión

Observar si los estudiantes logran programar el servomotor para apuntar a las estaciones correctamente.

## Sesión 4: Personalización y Mejora del Indicador de Estaciones

### Objetivos:

- Permitir que los estudiantes personalicen su proyecto y agreguen mejoras creativas.

- Introducir conceptos de retroalimentación y optimización en programación.

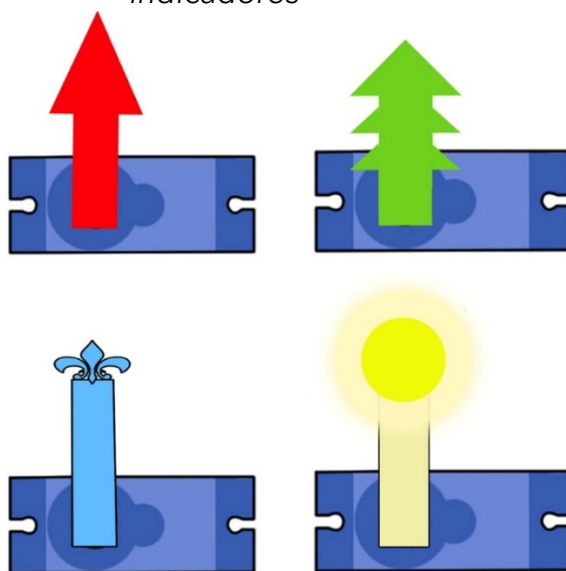
### Actividades:

Los estudiantes decoran su panel de estaciones con colores (ver figura 63), dibujos o elementos adicionales que representen cada estación (por ejemplo, flores para primavera, sol para verano, hojas secas para otoño, nieve para invierno).

Pueden agregar un brazo indicador más elaborado, como una flecha decorada o una figura que represente algo relacionado con las estaciones (un árbol, un sol, etc.).

**Figura 63**

*Diseño personalizado de distintos indicadores*



*Nota.* La figura muestra indicadores personalizados por los estudiantes



## **Evaluación de la sesión**

Evaluar la creatividad y funcionalidad de las mejoras realizadas por los estudiantes.

Verificar que los estudiantes puedan explicar cómo implementaron las nuevas funcionalidades.

## **Sesión 5: Presentación final y reflexión**

### **Objetivos:**

Presentar el indicador de estaciones funcionando.

Reflexionar sobre los aprendizajes adquiridos.

### **Actividades:**

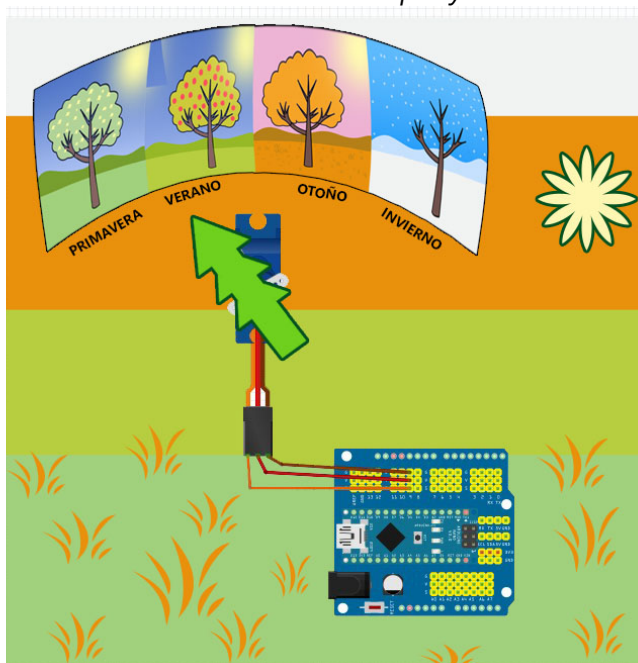
Los estudiantes presentan su indicador de estaciones, demostrando cómo el brazo se mueve para apuntar a cada estación (ver figura 64).

Preguntar a los estudiantes qué aprendieron sobre las estaciones del año, los ángulos y la programación.

Discutir cómo este proyecto podría aplicarse en la vida real (por ejemplo, en un calendario visual o un sistema educativo).

**Figura 64**

*Presentación del proyecto final*



Nota. La figura muestra la maqueta del indicador de estaciones del año. Fuente: Adaptación a partir de captura de pantalla de la interfaz de Fritzing. Fritzing (<https://fritzing.org>)

## **Evaluación de la sesión**

Evaluar la capacidad de los estudiantes para presentar su proyecto y explicar los conceptos aprendidos.

### **Conclusión:**

Este proyecto permite a los estudiantes explorar las estaciones del año, los ángulos y la programación de manera práctica y divertida. Al diseñar y programar un indicador de estaciones con un servomotor, los niños desarrollan habilidades de programación, lógica y resolución de problemas, mientras comprenden cómo los conceptos teóricos se aplican en contextos reales.

## **3.8. Autoevaluación del capítulo**

**Pregunta 1:** En el aprendizaje basado en proyectos dentro de la robótica educativa, ¿cuál es una de las principales ventajas de utilizar servomotores?

### **Opciones de respuesta:**

- a) Permiten a los estudiantes comprender principios de mecánica, electrónica y programación a través de aplicaciones prácticas.
- b) Son dispositivos que solo sirven para girar en una sola dirección sin posibilidad de control.
- c) No requieren programación ni conocimientos previos para su uso en proyectos educativos.
- d) Solo tienen aplicaciones en juguetes y no en sistemas robóticos avanzados.

**Pregunta 2:** En un proyecto de robótica educativa, ¿qué ventaja tiene el uso de servomotores frente a motores de corriente continua (DC)?

**Opciones de respuesta:**

- a) Los servomotores permiten un control preciso del ángulo de giro, mientras que los motores DC giran de manera continua sin precisión angular.
- b) Los servomotores no necesitan alimentación eléctrica, mientras que los motores DC sí.
- c) Los servomotores solo funcionan en robots industriales y no en proyectos educativos.
- d) Los servomotores siempre giran en una sola dirección y no pueden detenerse en posiciones específicas.

**Pregunta 3:** En el Aprendizaje Basado en Proyectos (ABP), ¿qué ocurre en la fase de Pruebas y ajustes al trabajar con un servomotor en un proyecto de robótica educativa?

**Opciones de respuesta:**

- a) Se verifica si el servomotor se mueve correctamente y se hacen modificaciones en la programación o el diseño si es necesario.
- b) Se elige el color del servomotor para que combine con el proyecto.
- c) Se decide qué materiales usar antes de empezar la construcción del proyecto.
- d) Se presenta el proyecto sin importar si el servomotor funciona bien o no.

**Pregunta 4:** En el Aprendizaje Basado en Proyectos (ABP), ¿qué se hace en la fase de Diseño y planificación al trabajar con un servomotor en un proyecto de robótica educativa?

**Opciones de respuesta:**

- a) Se define cómo y dónde se usará el servomotor, se crea un esquema del circuito y se planifica la programación.
- b) Se prueba directamente el servomotor sin haber pensado en su función dentro del proyecto.
- c) Se presenta el proyecto sin haberlo construido ni programado.
- d) Se investiga sobre servomotores sin necesidad de aplicarlo en un proyecto real.

**Pregunta 5:** En el Aprendizaje Basado en Proyectos (ABP), ¿qué se hace en la fase de Presentación del proyecto cuando se trabaja con un servomotor en robótica educativa?

**Opciones de respuesta:**

- a) Se explica cómo funciona el servomotor en el proyecto, se muestran los resultados obtenidos y se comparten aprendizajes.
- b) Se programa el servomotor por primera vez, sin haberlo probado antes.
- c) Se elige qué servomotor usar sin haber construido el proyecto.
- d) Se investiga sobre servomotores sin necesidad de mostrar el proyecto terminado.

**Pregunta 6:** ¿Cuál es el propósito del planteamiento del problema en un proyecto basado en ABP relacionado con servomotores?

**Opciones de respuesta:**

- a) Memorizar conceptos teóricos sobre robótica.
- b) Imitar soluciones sin analizar el problema.
- c) Motivar al estudiante a investigar, crear y resolver una necesidad significativa usando un sistema de movimiento controlado.
- d) Completar una tarea siguiendo instrucciones sin proponer soluciones.

**Pregunta 7:** ¿Qué actividades realizan los estudiantes durante la fase de diseño e implementación del prototipo en un proyecto con servomotores?

**Opciones de respuesta:**

- a) Diseñan la estructura y programan el movimiento del servomotor, aplicando conceptos como bucles, condicionales y variables.
- b) Memorizar listas de materiales para la construcción del robot.
- c) Solo prueban prototipos contruidos previamente sin modificarlos.
- d) Escriben teorías sobre robótica sin crear prototipos físicos.

**Pregunta 8:** ¿Qué es una secuencia didáctica basada en el uso de servomotores?

**Opciones de respuesta:**

- a) Un conjunto de actividades improvisadas para usar servomotores sin objetivos claros.
- b) Una serie organizada de actividades pedagógicas que guía a los estudiantes en el diseño, programación y control de servomotores para lograr un aprendizaje significativo.
- c) Una lista de componentes electrónicos necesarios para construir un robot.
- d) Una rutina de encendido y apagado de servomotores sin conexión con el aprendizaje.

## CRÉDITOS DE SOFTWARE Y RECURSOS VISUALES

Las imágenes contenidas en este libro fueron generadas mediante herramientas de simulación, diseño y programación educativa. Se agradece a las siguientes plataformas:

- **mBlock** (Makeblock Co., Ltd.): Utilizado para crear simulaciones de programación por bloques. Algunas imágenes muestran sprites y componentes visuales propiedad de Makeblock. Sitio oficial: <https://www.mblock.cc>
- **Tinkercad** (Autodesk, Inc.): Utilizado para diseñar y capturar modelos 3D y circuitos electrónicos. Las imágenes incluyen componentes prefabricados proporcionados por la biblioteca gráfica de Tinkercad. Sitio oficial: <https://www.tinkercad.com>
- **Fritzing**: Utilizado para esquemas eléctricos en vistas de protoboard, esquemático y PCB. Las capturas de pantalla fueron generadas por el autor a partir de sus propios diseños. Sitio oficial: <https://fritzing.org>

Todas las imágenes se usan con fines ilustrativos y educativos. Se respetan los términos de uso y propiedad intelectual de cada plataforma. En caso de reutilización, consulte las licencias individuales de cada software.



## REFERENCIAS BIBLIOGRÁFICAS

- Afecto, R., Moretti, A. A. da S., & Teixeira, L. de S. (2024). Robótica educacional, avanços e desafios para o ensino médio integrado ao técnico. *Dialogia*, 50, Article 50. <https://doi.org/10.5585/50.2024.27415>
- Angeli, C., & Giannakos, M. (2020). Computational thinking education: Issues and challenges. *Computers in Human Behavior*, 105, 106185. <https://doi.org/10.1016/j.chb.2019.106185>
- Arafat, M. H., Budiyanto, C. W., Yuana, R. A., & Fenyvesi, K. (2024). Implementation of Integrated STEM Learning in Educational Robotics towards 21st Century Skills: A Systematic Review. *International Journal of Education in Mathematics, Science and Technology*, 12(5), Article 5. <https://doi.org/10.46328/ijemst.4271>
- Autodesk, Inc. (2025). Interfaz de diseño en Tinkercad [Captura de pantalla]. Tinkercad. <https://www.tinkercad.com>
- Barragán Perea, E. A. (2023). Pensamiento computacional y programación en la formación de estudiantes desde edades tempranas. *Revista Educación*, 1-18. <https://doi.org/10.15517/revedu.v47i2.53645>

Betancourt Almaguer, A., & Peña Rodríguez, Y. (2023).  
Importancia de la disciplina Lenguajes y Técnicas de  
Programación en la carrera Educación Informática.  
*Didáctica y Educación ISSN 2224-2643*, 14(6), Article  
6.

[http://revistas.ult.edu.cu/index.php/didascalía/gatew  
ay/plugin/publdResolver/ark:/54724/DE.v14i6.1536](http://revistas.ult.edu.cu/index.php/didascalía/gateway/plugin/publdResolver/ark:/54724/DE.v14i6.1536)

Čavić, M., Beljin-Čavić, M., Horvat, S., Bogdanović, I., &  
Stanisavljević, J. (2023). The impact of project-based  
learning in physics education on university students'  
motivation to learn and metacognition. *Zbornik  
Instituta za pedagoška istraživanja*, 55(2), 275-299.  
[https://doiserbia.nb.rs/Article.aspx?ID=0579-  
64312302275C](https://doiserbia.nb.rs/Article.aspx?ID=0579-64312302275C)

Chen, S., & Ding, Y. (2024). Advancing STEAM education: A  
comprehensive assessment of competence. *Journal of  
Computers in Education*.  
<https://doi.org/10.1007/s40692-024-00322-1>

Çiftçi, A., & Topçu, M. S. (2023). Improving early childhood  
pre-service teachers' computational thinking skills  
through the unplugged computational thinking  
integrated STEM approach. *Thinking Skills and  
Creativity*, 49, 101337.  
<https://doi.org/10.1016/j.tsc.2023.101337>

Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

Çoban, E., & Korkmaz, Ö. (2021). An alternative approach for measuring computational thinking: Performance-based platform. *Thinking Skills and Creativity*, 42, 100929. <https://doi.org/10.1016/j.tsc.2021.100929>

Correa-Delgado, J. S. (2024). La investigación como estrategia pedagógica: Una alternativa latinoamericana a la educación STEAM. *PROSPECTIVA. Revista de Trabajo Social e Intervención Social*, e21213065-e21213065. <https://doi.org/10.25100/prts.v0i37.13065>

Correia, M., Ribeirinha, T., Beirante, D., Santos, R., Ramos, L., Dias, I. S., Luís, H., Catela, D., Galinha, S., Arrais, A., Portelada, A., Pinto, P., Simões, V., Ferreira, R., Franco, S., & Martins, M. C. (2024). Outdoor STEAM Education: Opportunities and Challenges. *Education Sciences*, 14(7), Article 7. <https://doi.org/10.3390/educsci14070688>

Decorte, B., & and Vlieghe, J. (2024). Towards a pedagogical conception of imagination in STEAM education. *Ethics and Education*, 19(2), 218-232. <https://doi.org/10.1080/17449642.2024.2361560>

DeLisi, J., Liu, E., & Fields, E. (2025). Implementing Project-Based Learning in Urban High School STEM Career

- Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM Pathways. *Urban Education*, 60(5), 1361-1384.  
<https://doi.org/10.1177/00420859231214174>
- Filipe, J., Baptista, M., & Conceição, T. (2024). Integrated STEAM Education for Students' Creativity Development. *Education Sciences*, 14(6), Article 6.  
<https://doi.org/10.3390/educsci14060676>
- Fritzing. (2025). Vista de protoboard en Fritzing [Captura de pantalla]. Fritzing. <https://fritzing.org>
- García, J. M. (2020). La expansión del Pensamiento Computacional en Uruguay. *Revista de Educación a Distancia (RED)*, 20(63), Article 63.  
<https://doi.org/10.6018/red.410441>
- Garnica, A. G., & Ramos Rivadeneira, D. X. (2023). Pensamiento computacional y enfoque STEAM como estrategia para fortalecer las competencias en matemáticas. *Revista Científica Ciencia y Tecnología*, 23(39), 16-31.  
<https://doi.org/10.47189/rcct.v23i39.595>
- Huang, W., Li, X., & Shang, J. (2023). Gamified Project-Based Learning: A Systematic Review of the Research Landscape. *Sustainability*, 15(2), Article 2.  
<https://doi.org/10.3390/su15020940>

Jaleniauskiene, E., & Lisaite, D. (2023). Scoping review methodology and its use to review online project-based learning in higher education, 2020-2023. *Interdisciplinary Journal of Problem-Based Learning*, 17(2), Article 2. <https://doi.org/10.14434/ijpbl.v17i2.36659>

Kayhan, O., Korkmaz, Ö., Özgen, & Çakır, R. (2024). How Do Computational Thinking and Logical and Math Thinking Skills Predict Programming Self-Efficacy? *Computers in the Schools*, 41(1), 51-73. <https://doi.org/10.1080/07380569.2023.2220696>

Makeblock. (s.f.). *mBlock* [Software de Computadora]. <https://www.mblock.cc>

Montés, N., Barquero, S., Martínez-Carbonell, A., Aloy, P., Ferrer, T., Romero, P. D., Millan, M., & Salazar, A. del S. (2024). Redefining STEAM to STEAM  $\forall$ H (STEAM for All Humanity) in Higher Education. *Education Sciences*, 14(8), Article 8. <https://doi.org/10.3390/educsci14080888>

Nannim, F. A., Ibezim, N. E., Oguguo, B. C. E., & Nwangwu, E. C. (2024). Effect of project-based Arduino robot application on trainee teachers computational thinking in robotics programming course. *Education*

- Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM *and Information Technologies*, 29(10), 13155-13170.  
<https://doi.org/10.1007/s10639-023-12380-6>
- Nyamekye, E., Zengulaaru, John, & Nana Frimpong, A. C. (2023). Junior high schools teachers' perceptions and practice of constructivism in Ghana: The paradox. *Cogent Education*, 10(2), 2281195.  
<https://doi.org/10.1080/2331186X.2023.2281195>
- Ouyang, F., & Xu, W. (2024). The effects of educational robotics in STEM education: A multilevel meta-analysis. *International Journal of STEM Education*, 11(1), 7. <https://doi.org/10.1186/s40594-024-00469-4>
- Palioura, M., & Sapounidis, T. (2025). "Storytelling and educational robotics: A scoping review (2004-2024)". *Computers & Education*, 225, 105186.  
<https://doi.org/10.1016/j.compedu.2024.105186>
- Payne, L., Tawfik, A., & Olney, A. M. (2022). Computational Thinking in Education: Past and Present. *TechTrends*, 66(5), 745-747. <https://doi.org/10.1007/s11528-022-00766-1>
- Pupik Dean, C. G., Grossman, P., Enumah, L., Herrmann, Z., & Kavanagh, S. S. (2023). Core practices for project-based learning: Learning from experienced

Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

practitioners in the United States. *Teaching and Teacher Education*, 133, 104275. <https://doi.org/10.1016/j.tate.2023.104275>

Rosa, T. de A., Terçariol, A. A. de L., & Ikeshoji, E. A. B. (2023). Steam, projetos e o pensamento computacional nos anos iniciais do ensino fundamental: Contribuições para uma educação disruptiva. *EccoS - Revista Científica*, 65, Article 65. <https://doi.org/10.5585/eccos.n65.24626>

Sari, E. D. P., Trisnawati, R. K., Agustina, M. F., Adiarti, D., & Noorashid, N. (2023). Assessment of Students' Creative Thinking Skill on the Implementation of Project-Based Learning. *International Journal of Language Education*, 7(3), Article 3. <https://doi.org/10.26858/ijole.v7i3.38462>

Shin, N., Bowers, J., Roderick, S., McIntyre, C., Stephens, A. L., Eidin, E., Krajcik, J., & Damelin, D. (2022). A framework for supporting systems thinking and computational thinking through constructing models. *Instructional Science*, 50(6), 933-960. <https://doi.org/10.1007/s11251-022-09590-9>

Spyropoulou, N., & Kameas, A. (2024). Augmenting the Impact of STEAM Education by Developing a Competence Framework for STEAM Educators for

Enseñanza y aprendizaje del pensamiento computacional: robótica básica con servomotores. Guía práctica con metodología STEAM

Effective Teaching and Learning. *Education Sciences*, 14(1), Article 1.  
<https://doi.org/10.3390/educsci14010025>

Suslenco, A. (2024). STEAM Education—An Effective Approach to Achieving Sustainability in Higher Education. *Revista Romaneasca Pentru Educatie Multidimensionala*, 16(1), Article 1.  
<https://doi.org/10.18662/rrem/16.1/813>

Tapullima-Mori, C., Pizzán-Tomanguillo, S. L., Pizzán-Tomanguillo, N. del P., Sangama, L. R. G., Sánchez, M. V., & Cachay, M. I. (2024). Una revisión bibliométrica del enfoque STEAM en educación universitaria 2010-2022. *Revista Digital de Investigación en Docencia Universitaria*, 18(1), Article 1.  
<https://doi.org/10.19083/ridu.2024.1790>

Taranto, C. M. B., Sampaio, R. da S., & Rodrigues-Silva, J. (2024). Percepção dos professores brasileiros sobre a educação STEAM - ciências, tecnologia, engenharia, artes/humanidades e matemáticas. *Revista Tecnologia e Sociedade*, 20(60), Article 60.  
<https://doi.org/10.3895/rts.v20n60.16494>

Team, I. E., Stojanović, B. J., Ristanović, D., Živković, P., & Džaferović, M. (2023). International Journal of Cognitive Research in Science, Engineering and



Education (IJCRSEE). *International Journal of Cognitive Research in Science, Engineering and Education*, 10(2), Article 2.  
<https://doi.org/10.23947/2334-8496-2023-11-2-213-220>

Tierney, G., Adams ,Carol, & and Ward, S. (2025). In the Service of Student Engagement: Project-Based Learning Classrooms and Teacher Practices. *The Journal of Experimental Education*, 93(1), 38-53.  
<https://doi.org/10.1080/00220973.2023.2287446>

Urgiles-Rodríguez, B. E., Tixi-Gallegos, K. G., & Allauca-Peñafiel, M. E. (2022). Metodología Steam en Ambientes Académicos. *Dominio de las Ciencias*, 8(1), Article 1. <https://doi.org/10.23857/dc.v8i1.2482>

Wang, Y. (2023). The role of computer supported project-based learning in students' computational thinking and engagement in robotics courses. *Thinking Skills and Creativity*, 48, 101269.  
<https://doi.org/10.1016/j.tsc.2023.101269>

Weintrop, D., Subramaniam, M., Morehouse, S., & Koren, N. (2023). The State of Computational Thinking in Libraries. *Technology, Knowledge and Learning*, 28(3), 1301-1324. <https://doi.org/10.1007/s10758-022-09606-w>

Enseñanza y aprendizaje del pensamiento computacional: robótica básica  
con servomotores. Guía práctica con metodología STEAM

You, J. W. (2024). Relationship between team learning  
profiles and outcomes in team project-based learning:  
A cluster analysis. *Studies in Higher Education*, 49(1),  
16-32.  
<https://doi.org/10.1080/03075079.2023.2219298>

**Julio Antonio Encalada Cuenca**  
**jencalada@utmachala.edu.ec**

Universidad Técnica de Machala, Facultad de Ciencias Sociales, Carrera de Pedagogía de las Ciencias Experimentales <https://orcid.org/0000-0002-8120-2047> Julio Encalada es docente e investigador en la Universidad Técnica de Machala. Magíster en Educación Informática, cuenta con amplia experiencia en docencia e investigación en el campo de la robótica educativa, el pensamiento computacional y la metodología STEAM. Su trabajo promueve la integración de tecnologías en entornos pedagógicos innovadores y orientados al aprendizaje activo.

**Rosemary de Lourdes Samaniego Ocampo**  
**rsamaniego@utmachala.edu.ec**

Universidad Técnica de Machala, Facultad de Ciencias Sociales, Carrera de Pedagogía de las Ciencias Experimentales <https://orcid.org/0000-0001-8042-8434> Rosemary Samaniego es doctora en Tecnología Educativa, es docente titular, investigadora y vicerrectora académica de la Universidad Técnica de Machala. Su trabajo se centra en la innovación educativa y el uso de tecnologías para transformar la enseñanza. Lidera proyectos que impulsan la calidad académica y el desarrollo institucional.

**Luis Adrián Luzuriaga Pesantes**  
**lluzuriag3@utmachala.edu.ec**

Universidad Técnica de Machala, Facultad de Ciencias Sociales, Carrera de Pedagogía de las Ciencias Experimentales <https://orcid.org/0009-0005-7711-1993> Luis Luzuriaga es investigador independiente, especializado en robótica educativa y pensamiento computacional. Su trabajo promueve el desarrollo de habilidades tecnológicas en entornos de aprendizaje. Apasionado por la innovación, impulsa proyectos que integran la tecnología en la educación, para la formación de nuevas generaciones preparadas para los desafíos del mundo digital.



ISBN: 978-9942-53-000-4

