Universidad Internacional de La Rioja
Facultad de Educación

Máster Universitario en Tecnología Educativa
y Competencias Digitales

Desarrollo De Pensamiento Computacional En Estudiantes De Secundaria Usando Aprendizaje Basado En Proyectos

Trabajo fin de estudio presentado por:	Jaime Andrés Gómez Chingaté
Tipo de trabajo:	TFM
Director/a:	Pedro Antonio Soriano Lorenz
Fecha:	25/02/2026

Resumen

El presente Proyecto de Innovación Educativa aborda la necesidad de fortalecer el desarrollo del pensamiento computacional en la educación básica secundaria en Colombia, específicamente en el contexto de la Institución Educativa "Instituto Montenegro". El objetivo principal de este proyecto fue diseñar una propuesta de innovación pedagógica basada en el Aprendizaje Basado en Proyectos (ABP) y la integración de herramientas de programación no code, como Scratch y Micro:bit, para estudiantes de grado octavo. La metodología empleada consistió en un diseño cualitativo y descriptivo, que abarcó la contextualización del problema, la fundamentación teórica del pensamiento computacional y el ABP, la caracterización de los destinatarios, la definición de objetivos, contenidos y competencias, el diseño de una secuencia didáctica detallada en cinco actividades, la elaboración de instrumentos de evaluación formativa y sumativa, y la reflexión sobre la atención a la diversidad. Aunque la implementación real no se llevó a cabo, el análisis de la propuesta y su prospectiva permitieron identificar el potencial de la metodología y las herramientas para fomentar habilidades de resolución de problemas, creatividad y pensamiento lógico en los estudiantes. La principal conclusión radica en la viabilidad y pertinencia de integrar el ABP y la programación no code como estrategias efectivas para introducir y desarrollar el pensamiento computacional en la educación secundaria con recursos limitados, abriendo futuras líneas de investigación en la transición hacia lenguajes de programación formales y la exploración más profunda de las herramientas tecnológicas.

Palabras clave: Pensamiento Computacional, Aprendizaje Basado en Proyectos, Programación No Code, Scratch, Micro:bit.

Abstract

This Educational Innovation Project addresses the need to strengthen the development of computational thinking in basic secondary education in Colombia, specifically in the context of the Educational Institution “Instituto Montenegro”. The main objective of this project was to design an innovative pedagogical proposal based on Project-Based Learning (PBL) and the integration of no-code programming tools, such as Scratch and Micro:bit, for eighth-grade students. The methodology used consisted of a qualitative and descriptive design, which included the contextualization of the problem, the theoretical foundation of computational thinking and PBL, the characterization of the target audience, the definition of objectives, content, and competencies, the design of a detailed didactic sequence in five activities, the development of formative and summative evaluation instruments, and reflection on attention to diversity. Although real implementation was not carried out, the analysis of the proposal and its prospects allowed identifying the potential of the methodology and tools to foster problem-solving skills, creativity, and logical thinking in students. The main conclusion lies in the viability and relevance of integrating PBL and no-code programming as effective strategies to introduce and develop computational thinking in secondary education with limited resources, opening future lines of research in the transition towards formal programming languages and the deeper exploration of technological tools.

Keywords: Computational Thinking, Project-Based Learning, No-Code Programming, Scratch, Micro:bit.

Índice de contenidos

Índice de figuras

Índice de tablas

1. Introducción

Uno de los retos más grandes que se presenta actualmente en la educación en Colombia es la necesidad de un cambio en el paradigma de enseñanza a niveles de primaria y secundaria (principalmente en instituciones educativas públicas), ya que se ha encontrado una brecha en la calidad educativa entre estos niveles escolares y la educación superior; la cual se debe a factores como el entorno socio económico, la formación académica de la familia, el acceso que se tenga a internet, las características de las instituciones educativas, la inversión de recursos y el alcance de las políticas estatales, entre otros entre otros (Castro & Garzón, 2017). El Ministerio de Educación Nacional (Ministerio de Educación Nacional (MEN), 2016) ha establecido lineamientos curriculares para la integración de tecnologías e informática, lo que subraya la importancia de desarrollar competencias como el pensamiento computacional en la educación secundaria.

Tras analizar el contexto general de la educación en Colombia y la necesidad de integrar el pensamiento computacional en el currículo, es fundamental considerar los desafíos específicos que dificultan la implementación de prácticas pedagógicas innovadoras con el uso de tecnologías en las instituciones educativas. En este contexto de desafíos, los más importantes son:

- La persistencia de modelos pedagógicos tradicionales, como el conductismo, en numerosas instituciones educativas colombianas, representa un obstáculo significativo para la innovación en la práctica docente y la integración efectiva de las nuevas tecnologías (Tekman Education, 2021). Este enfoque limita la capacidad de los educadores para adoptar metodologías más dinámicas y centradas en el estudiante.
- Existe una marcada deficiencia en el desarrollo de habilidades y competencias digitales entre los estudiantes de secundaria en Colombia (ANDI, 2021). Esta situación genera una brecha considerable entre las habilidades que poseen al egresar de la educación secundaria y las exigencias del entorno de la educación superior y el mercado laboral.
- La disparidad entre la formación docente y la infraestructura tecnológica en las instituciones educativas colombianas constituye un desafío complejo (Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC), 2020). En algunos casos, docentes capacitados en el uso de tecnologías carecen de la infraestructura adecuada

para implementarlas, mientras que, en otros, instituciones con buena infraestructura no cuentan con docentes debidamente formados para aprovechar estos recursos.

- La falta de apropiación de las tecnologías por parte de docentes de diversas áreas es otro factor limitante (Ministerio de Educación Nacional (MEN), 2013). La percepción de que la formación en el uso de nuevas tecnologías es responsabilidad exclusiva de los docentes de tecnología e informática dificulta la implementación de enfoques interdisciplinarios que enriquezcan el proceso de aprendizaje.

La concepción del currículo para el área de Tecnología e Informática en Colombia presenta una problemática significativa. Si bien la Ley General de Educación (Ley 115 de 1994) establece su obligatoriedad y su carácter fundamental desde la educación primaria hasta la secundaria (Ministerio de Educación Nacional (MEN), 1994), la implementación se caracteriza por una falta de uniformidad a nivel nacional. En lugar de un currículo definido y de aplicación general, existen orientaciones curriculares que, aunque buscan promover el desarrollo de competencias tecnológicas e informáticas, ofrecen lineamientos amplios sin especificar contenidos y competencias de manera detallada (Ministerio de Educación Nacional (MEN), 2016). Esta falta de especificidad puede generar variaciones significativas en la formación tecnológica ofrecida en las diferentes instituciones educativas del país.

Adicionalmente, la actualización de estas orientaciones curriculares no ha sido un proceso constante. La Guía 30 (Ministerio de Educacion Nacional (MEN), 2008) fue el referente principal durante aproximadamente quince años, y solo recientemente se ha implementado una nueva versión (Ministerio de Educación Nacional (MEN), 2022) , lo que ha generado un periodo de transición en la adopción de los nuevos lineamientos. Esta discontinuidad en la actualización curricular puede impactar la pertinencia de los contenidos y las estrategias pedagógicas implementadas en el aula.

Considerando la creciente relevancia de competencias tecnológicas e informáticas como el pensamiento computacional, la robótica y la programación en el contexto del mundo contemporáneo (ISTE (International Society for Technology in Education), 2016; Wing, 2006), surge la necesidad de explorar estrategias de enseñanza que fomenten su desarrollo en estudiantes de secundaria. En este sentido, la metodología de Aprendizaje Basado en Proyectos (ABP) se presenta como un enfoque pedagógico potencialmente efectivo para cultivar el pensamiento computacional en este nivel educativo (Blumenfeld, y otros, 2011)

Por lo tanto, este proyecto de innovación tecnológica se propone explorar el potencial del Aprendizaje Basado en Proyectos como estrategia para fomentar el desarrollo del pensamiento computacional en estudiantes de secundaria en el contexto colombiano.

1.1.Justificación

El rápido avance de las Tecnologías de la Información y la Comunicación (TIC) en las últimas décadas ha posicionado la enseñanza del pensamiento computacional, la robótica y la programación como una necesidad emergente en el ámbito educativo para el desarrollo de competencias digitales esenciales en la sociedad contemporánea (Bers, 2018; Grover & Pea, 2013). No obstante, existe una dificultad significativa relacionada con la formación y actualización docente, especialmente en los niveles de educación primaria y secundaria, para abordar los nuevos paradigmas pedagógicos inherentes a estas disciplinas. Esta situación se agudiza en Latinoamérica, y particularmente en Colombia, donde la evolución e innovación educativa históricamente han sido procesos lentos, lo cual se evidencia en la limitada integración de metodologías activas y recursos tecnológicos en el aula hasta los últimos años. (Rojas-Ospina, y otros, 2023; Organización para la Cooperación y el Desarrollo Económicos (OCDE), 2020)

La pandemia de COVID-19 actuó como un catalizador que evidenció la necesidad de una reevaluación del paradigma educativo en Colombia. La transición abrupta hacia modalidades de educación a distancia y alternancia reveló la limitada preparación de instituciones y docentes para afrontar estos desafíos, lo que contribuyó al incremento de la repitencia y la deserción escolar, además de exacerbar las desigualdades en el rendimiento académico (Melo-Becerra, y otros, 2021). Asimismo, se constató una significativa disparidad entre las competencias desarrolladas por los estudiantes de secundaria y las exigencias del mundo moderno, lo que presenta nuevas necesidades y retos al sistema educativo sobre todo en áreas como la resolución de problemas complejos y el pensamiento crítico.

En la actualidad, existe un interés creciente por parte de gobiernos nacionales, entidades locales e instituciones educativas sobre el potencial y la importancia de la formación en pensamiento computacional, robótica y programación, principalmente en estudiantes de educación secundaria. Como muestra de esto, existe un aumento de la inversión en

infraestructura educativa para las instituciones oficiales, así como la aparición de iniciativas como "Colombia Programa", impulsada por el gobierno nacional para fomentar el desarrollo de competencias en pensamiento computacional en docentes y estudiantes (Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC), 2023). Es crucial destacar que el pensamiento computacional trasciende los límites de una única asignatura, como el Área de Tecnología e Informática en el contexto colombiano, y se configura como un desarrollo transversal que requiere la integración de múltiples habilidades (Adell Segura, y otros, 2019; Barr & Stephenson, 2011). Una aproximación interdisciplinaria permite a los estudiantes comprender la aplicación de competencias y habilidades desarrolladas en diferentes asignaturas, y como puede usarlas en diversos contextos que le permitan fortalecer su capacidad para analizar y abordar problemas de su entorno.

1.2. Planteamiento del Problema

El Ministerio de Tecnologías de la Información y las Comunicaciones (Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC), 2023) ha señalado la necesidad de innovar los procesos de enseñanza-aprendizaje en las instituciones educativas colombianas, con el objetivo de fomentar la formación en programación y pensamiento computacional en los niveles de educación primaria y secundaria. En respuesta a esta prioridad, el presente trabajo propone el diseño de un proyecto de innovación educativa dirigido a estudiantes de grado octavo (edades entre 13 y 16 años) que fomente el desarrollo de habilidades fundamentales para el análisis de problemas y la formulación de soluciones a través del diseño de algoritmos y diagramas de flujo. La implementación de estas soluciones se explorará mediante el uso de herramientas tecnológicas *no code* como Micro:bit o Scratch, integrando la metodología de Aprendizaje Basado en Proyectos (ABP).

La elección del grado octavo como población objetivo se fundamenta en que los estudiantes en este rango de edad se encuentran en una etapa de desarrollo cognitivo que facilita la comprensión y aplicación de los principios del pensamiento computacional (Piaget & Inhelder, 1972); además, de que este es un momento crucial para el desarrollo de habilidades tecnológicas fundamentales y la adquisición de competencias digitales.

La selección de Micro:bit y Scratch como herramientas tecnológicas *no code* se basa en su accesibilidad y su enfoque pedagógico en los conceptos fundamentales del pensamiento computacional. Scratch, con su interfaz de programación visual basada en bloques, facilita la introducción a la lógica de la programación a través de la creación de proyectos interactivos (Maloney, y otros, 2010). Por su parte, Micro:bit permite a los estudiantes experimentar con la programación en un entorno físico, creando proyectos tangibles que conectan el mundo virtual con el real, lo que fomenta una comprensión más profunda y aplicada del pensamiento computacional (Sentance, y otros, 2017).

1.3. Objetivos Del Trabajo

Los objetivos del presente Trabajo de Fin de Master son los siguientes:

1.3.1. Objetivo General

- El objetivo general del trabajo es desarrollar una propuesta de innovación educativa basada en el Aprendizaje Basado en Proyectos (ABP) que facilite y promueva el desarrollo de habilidades de pensamiento computacional en estudiantes de grado octavo de educación secundaria, mediante la implementación de actividades prácticas y proyectos colaborativos que respondan a problemas reales y contextuales del entorno escolar.

1.3.2. Objetivos Específicos

- Realizar una revisión exhaustiva de la literatura y los estudios existentes sobre el uso de la metodología de Aprendizaje Basado en Proyectos (ABP) en el desarrollo del pensamiento computacional en estudiantes de educación secundaria, identificando enfoques efectivos y recursos pedagógicos que puedan ser aplicados en el contexto del octavo grado de educación secundaria
- Diseñar una estrategia educativa que incluya actividades y proyectos específicos basados en el ABP y el uso de herramientas *no code* (Micro:bit y Scratch), que fomenten el desarrollo de habilidades de pensamiento computacional en estudiantes de octavo grado mediante el trabajo colaborativo, buscando la alineación de estas actividades y proyectos dentro de los objetivos curriculares del Área de Tecnología e

Informática, y considerando que dichas actividades se encuentren contextualizadas en la realidad y el entorno de los estudiantes.

- Elaborar un plan de evaluación que contemple criterios e indicadores específicos que permitan medir el desarrollo del pensamiento computacional en los estudiantes, utilizando métodos cualitativos y cuantitativos para evaluar la efectividad de la estrategia propuesta y el impacto en el aprendizaje de los estudiantes.
- Evaluar la factibilidad y viabilidad de implementar la estrategia educativa propuesta en términos de recursos, infraestructura, y potenciales desafíos, proporcionando recomendaciones para asegurar su éxito y sostenibilidad en el contexto educativo de octavo grado.

2. Marco Teórico

2.1.Introducción

La acelerada revolución tecnológica de las últimas décadas, marcada por la proliferación de computadoras, internet y dispositivos móviles, ha transformado profundamente los estilos de vida y la concepción del mundo en la sociedad actual (Castells, 2000). Situaciones y actividades que parecían imposibles hace 30 años, como la comunicación sincrónica global y el comercio electrónico, se han vuelto parte de la vida cotidiana, permeando diversos ámbitos de la sociedad. Sin embargo, a pesar de la rápida adopción de estas tecnologías en la sociedad actual, en el sector educativo en Latinoamérica, y particularmente en Colombia, enfrenta una brecha significativa. Esta disparidad se atribuye principalmente a la inversión históricamente limitada en las instituciones educativas y a la necesidad de fortalecer la formación docente, especialmente en los niveles de educación primaria y secundaria (Organización para la Cooperación y el Desarrollo Económicos (OCDE), 2020).

En el contexto colombiano, una problemática educativa de gran impacto es la brecha existente entre la educación básica (primaria y secundaria) y la educación superior (Ministerio de Educación Nacional (MEN), 2016). Eso causa que haya una desconexión entre los procesos de enseñanza y aprendizaje en ambos niveles educativos, lo que genera vacíos en la fundamentación de diversas áreas del conocimiento y dificulta la transición de los estudiantes a la educación superior. Una de las áreas donde esta situación era particularmente evidente era la enseñanza del pensamiento computacional, la robótica y la programación. Inicialmente, el modelo educativo colombiano no integraba estas temáticas en la educación básica, limitando su enseñanza a programas técnicos o profesionales específicos. Esta perspectiva situó al sistema educativo colombiano en un nivel de desarrollo inferior en comparación con otros países, como España, en lo referente a la integración de competencias digitales y habilidades informáticas relevantes para la vida académica y laboral (Fundación Telefónica, 2019).

La pandemia de COVID-19 representó un punto de inflexión en esta situación. Debido a las condiciones propias de la pandemia, en Colombia se presentó el fenómeno de confinamiento obligatorio, el cual impulsó la adopción de modelos educativos a distancia y, posteriormente,

de alternancia en todas las instituciones del país. Esta transición repentina expuso la limitada preparación tanto de estudiantes como de docentes para los desafíos de la educación en línea, evidenciando la necesidad de la implementación de nuevas metodologías de educación que involucrara el uso de recursos digitales. Como consecuencia, el gobierno nacional reconoció la urgencia de replantear los paradigmas educativos para mejorar la calidad de la formación y preparar a los jóvenes para los requerimientos del siglo XXI (Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC), 2020).

2.1.1. Objetivo del Marco Teórico

El presente marco teórico tiene como objetivo fundamental proporcionar una base conceptual sólida y actualizada sobre el pensamiento computacional, la metodología de Aprendizaje Basado en Proyectos (ABP) y las herramientas tecnológicas *no code* (Micro:bit y Scratch), en el contexto de la educación secundaria. Se busca explorar teorías, modelos pedagógicos e investigaciones previas que sustenten la importancia del desarrollo del pensamiento computacional en la educación, los beneficios de la metodología ABP para el aprendizaje activo y significativo, y el potencial de las herramientas *no code* para facilitar la introducción a conceptos de programación y robótica de manera accesible y motivadora en el contexto de la educación secundaria en Colombia, principalmente en grado octavo.

2.1.2. Estructura del Marco Teórico

Para alcanzar este objetivo, el marco teórico se estructurará en los siguientes apartados principales:

- **Contexto y Antecedentes de la Educación Tecnológica en Colombia:** Se revisarán las políticas educativas y los lineamientos curriculares en el Área de Tecnología e Informática en el contexto de la educación secundaria en grado octavo; así como antecedentes sobre la necesidad de desarrollar el pensamiento computacional en estudiantes de secundaria.
- **Aprendizaje Basado en Proyectos (ABP):** Se describirá la metodología ABP, sus principios pedagógicos, las fases de implementación y sus beneficios para el aprendizaje activo, la colaboración, la resolución de problemas y la contextualización del conocimiento.

- **Pensamiento Computacional:** Se definirá el concepto de pensamiento computacional, se explorarán sus componentes fundamentales (descomposición, reconocimiento de patrones, abstracción, diseño de algoritmos) y se analizará su importancia en el desarrollo de habilidades del siglo XXI y su integración en el currículo de educación secundaria.
- **Herramientas Tecnológicas *No Code* en la Educación:** Se presentarán y analizarán las características pedagógicas de Micro:bit y Scratch, su potencial para la enseñanza de conceptos de programación y robótica sin requerir conocimientos avanzados de codificación, y su aplicabilidad en el contexto de proyectos educativos en secundaria.
- **Integración del Pensamiento Computacional y el ABP con Herramientas *No Code*:** Se explorarán las sinergias entre el pensamiento computacional, la metodología ABP y el uso de herramientas *no code*, analizando cómo su integración puede facilitar el desarrollo de habilidades en los estudiantes de secundaria.
- **Evaluación y Seguimiento del Pensamiento Computacional:** Se revisarán métodos y herramientas para evaluar el desarrollo del pensamiento computacional a partir de estudios de caso y experiencias previas en la evaluación del pensamiento computacional; además, se tendrán en cuenta consideraciones éticas y metodológicas en la evaluación educativa.

2.2. Contexto y Antecedentes de la Educación Tecnológica en Colombia

El contexto de la educación tecnológica en Colombia, particularmente en el nivel de educación secundaria y específicamente en el grado octavo, se encuentra determinado por una serie de políticas educativas y lineamientos curriculares pertenecientes al Área de Tecnología e Informática que buscan orientar y delimitar la formación de los estudiantes en lo concerniente a la educación tecnológica. La Ley General de Educación (Ley 115 de 1994) estableció la obligatoriedad de esta área como fundamental en todos los niveles de la educación básica y media (Ministerio de Educación Nacional (MEN), 1994). Sin embargo, la implementación de un currículo unificado a nivel nacional ha sido un desafío, ya que como no existen unos derechos básicos de aprendizaje (DBA) para todos los estudiantes de educación básica y media, sino que hay un conjunto de lineamientos curriculares los cuales pueden ser adaptados

con una gran diversidad de enfoques y contenidos por las diferentes educaciones educativas del país (Ministerio de Educación Nacional (MEN), 2016).

Los Lineamientos Curriculares de Tecnología e Informática (Ministerio de Educación Nacional (MEN), 2016) proporcionan un conjunto de orientaciones generales para el desarrollo de competencias tecnológicas en los estudiantes de educación básica y media. Estos lineamientos buscan promover la comprensión de los principios científicos y tecnológicos, el desarrollo de habilidades y competencias para la solución de problemas técnicos, y la apropiación crítica y responsable de las tecnologías de la información y la comunicación. Sin embargo, tal y como se mencionó anteriormente, la especificidad en cuanto a contenidos y la profundidad con la que se abordan temáticas emergentes como el pensamiento computacional queda a discreción del enfoque de las diferentes instituciones educativas.

Históricamente, la enseñanza del Área de Tecnología e Informática en la educación básica y media en Colombia se ha centrado en el manejo de herramientas ofimáticas y la alfabetización digital básica. La incorporación de conceptos más avanzados como los son la programación, el pensamiento computacional, la metodología de Aprendizaje Basado en Proyectos (ABP) y las herramientas tecnológicas *no code* (Micro:bit y Scratch), ha sido muy gradual y, en la mayoría de los casos, depende de iniciativas aisladas de docentes o de la especialidad técnica con las que cuentan algunas instituciones educativas (Sánchez, 2012). Esta situación ha generado una brecha entre las habilidades que desarrollan los estudiantes y las demandas de la sociedad actual, cada vez más digitalizada y orientada a la innovación tecnológica (ANDI, 2021).

En lo concerniente a la necesidad de desarrollar el pensamiento computacional en estudiantes de secundaria, diversos autores coinciden en la importancia que tiene como una competencia digital fundamental para el siglo XXI (Wing, 2006) (Zapata-Ros, 2015). El pensamiento computacional no se limita solamente a la programación y la codificación, sino que también involucra un conjunto de competencias adicionales como lo son la resolución de problemas, la organización de datos, la abstracción y el diseño de sistemas (ISTE (International Society for Technology in Education), 2016). El desarrollo del pensamiento computacional en edades tempranas se considera fundamental para fomentar la creatividad, el razonamiento lógico y la resolución de problemas complejos en diversas áreas del conocimiento y su aplicación en diferentes situaciones de la vida cotidiana.

El contexto de la educación tecnológica en Colombia, la necesidad de integrar el pensamiento computacional en la educación secundaria ha ganado reconocimiento en los últimos años. Iniciativas como "Colombia Programa" (Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC), 2023) impulsada por el gobierno nacional buscan fomentar el desarrollo de competencias en pensamiento computacional en docentes y estudiantes. Sin embargo, la articulación curricular del pensamiento computacional en el aula y sobre todo su implementación efectiva, especialmente en grados como octavo, aún presentan desafíos en términos de formación docente, recursos pedagógicos y estrategias de enseñanza adecuadas.

2.3.Aprendizaje Basado en Proyectos (ABP)

El Aprendizaje Basado en Proyectos (ABP) es una estrategia de pedagogía activa centrada en el estudiante, donde el proceso de enseñanza aprendizaje y el desarrollo de competencias se articulan en torno a la planificación, ejecución y evaluación de proyectos que abordan problemas o preguntas complejas y auténticas (Thomas, 2000). En esencia, el ABP promueve un aprendizaje significativo que establece conexiones entre el contenido curricular de las diferentes asignaturas y situaciones problema del mundo real; incentivando el aprendizaje autónomo, la capacidad investigativa y el aprendizaje colaborativo entre los estudiantes (Hmelo-Silver, 2004).

2.3.1. Principios Pedagógicos Fundamentales del ABP

La eficacia del ABP radica en la adhesión a principios pedagógicos esenciales:

- **El Estudiante Como Figura Central En El Aprendizaje:** El ABP redefine el rol tradicional del docente, pasando de la figura de transmisor de conocimiento a la de ser un facilitador y guía durante el proceso de aprendizaje. En este paradigma, el estudiante asume un papel activo y protagónico en la construcción de su propio conocimiento (Savery, 2015)
- **Estructuración en torno a Problemas o Preguntas Detonadoras:** Los proyectos se estructuran en torno a problemas o preguntas detonadoras planteadas por el docente, que presenten un desafío adecuado en cuanto a sus fases de implementación y que

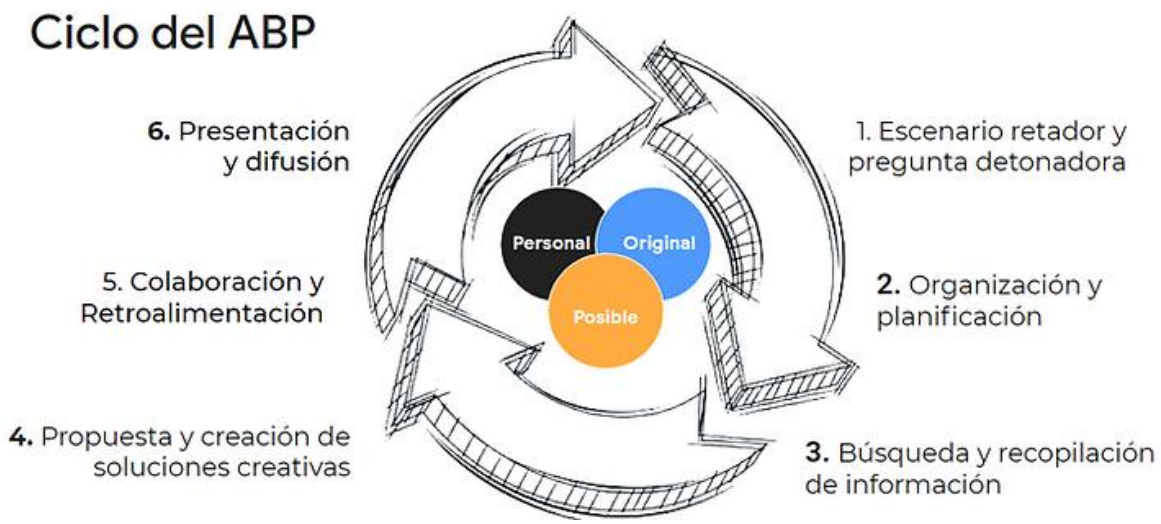
involucren situaciones o contextos que sean importantes para los estudiantes, lo cual incrementa la motivación y el nivel de compromiso de los mismos (Blumenfeld P. C., y otros, 1991)

- **Fomenta la Investigación:** El ABP impulsa a los estudiantes a involucrarse en procesos investigación y búsqueda de información pertinente de manera sistemática para lograr una comprensión profunda del problema o pregunta detonadora, y la correcta formulación de soluciones basadas en la información recopilada, desarrollando así habilidades fundamentales de pensamiento crítico y gestión de la información (Krajcik & Blumenfeld, 2006)
- **Énfasis en la Colaboración y el Trabajo en Equipo:** Uno de los componentes más importantes del ABP es la colaboración y el trabajo en equipo, ya que esto permite que los estudiantes compartan perspectivas de aprendizaje diferentes, distribuyan responsabilidades y aprendan de las contribuciones de sus compañeros, permitiendo obtener diferentes enfoques al momento de solucionar problemas, y mediante esto fortalecer sus habilidades sociales y comunicativas (Johnson & Johnson, 2009).
- **Integración De Diferentes Procesos De Evaluación:** La evaluación dentro del ABP esta compuestas por dos procesos diferentes: Por un lado, existe un proceso de retroalimentación continuo y oportuna que presente durante el desarrollo del proyecto (evaluación formativa); y por otro lado hay una valoración integral del producto final y el aprendizaje adquirido (evaluación sumativa), promoviendo la autoevaluación y la mejora continua durante todo el proceso de enseñanza aprendizaje (Wiggins, 1998).
- **Establecimiento de Conexiones con Situaciones del Mundo Real:** Al abordar problemas del mundo real que sean significativos para los estudiantes, el ABP facilita la comprensión de la importancia del conocimiento académico y la relevancia que tiene el poder transferir de una manera efectiva los aprendizajes obtenidos en el aula de clase a para solucionar problemáticas de la vida real, dándole un contexto significativo y duradero al conocimiento adquirido (Lave & Wenger, 1991)

2.3.2. Fases Típicas en la Implementación del ABP

Aunque los modelos de implementación pueden presentar variaciones, como se muestra en la Figura 1 (IESPE (Instituto de Estudios Superiores en Pedagogía y Educación Continua), 2023) el ABP generalmente se desarrolla a través de una secuencia de fases interdependientes:

Figura 1 *Ciclo del Aprendizaje Basado en Proyectos*



Fuente: (IESPE (Instituto de Estudios Superiores en Pedagogía y Educación Continua), 2023)

1. **Presentación del Problema o Pregunta Detonadora:** El docente introduce un problema, un escenario retador o una pregunta detonadora actuará como el catalizador del proyecto.
2. **Organización y Planificación del Proyecto:** Los estudiantes, con la orientación del docente, trabajaran colaborativamente planificando las tareas específicas, los recursos necesarios, la asignación de roles y la elaboración de un cronograma realista para abordar el problema.
3. **Desarrollo de la Investigación:** Los estudiantes se involucran activamente en la búsqueda, recopilación y análisis de información relevante para comprender el problema en profundidad y explorar diversas posibilidades de solución.
4. **Elaboración de la Propuesta o Solución:** Los estudiantes aplican los conocimientos y habilidades adquiridos para realizar una propuesta de solución del problema y a partir ahí se diseña y crea un producto tangible o una solución viable al problema planteado.

5. **Colaboración, Retroalimentación y Evaluación de Resultados:** Los estudiantes comunican sus proyectos a la clase o a una audiencia más amplia, y se lleva a cabo una evaluación integral del proceso y del producto, incorporando la autoevaluación, la coevaluación y la evaluación del producto final y el aprendizaje adquirido.
6. **Presentación y Difusión Sobre el Aprendizaje:** Los estudiantes realizan una reflexión metacognitiva sobre los aprendizajes clave, los desafíos superados y las estrategias para mejorar en futuros proyectos.

2.3.3. Beneficios Pedagógicos del ABP

La implementación efectiva del ABP conlleva una amplia gama de beneficios significativos para el aprendizaje de los estudiantes:

- **Promoción del Aprendizaje Activo y Significativo:** Los estudiantes se involucran directamente en la construcción activa de su conocimiento, trascendiendo el rol de receptores pasivos de información.
- **Desarrollo de Sólidas Habilidades de Colaboración:** El trabajo en equipo intensivo fomenta la comunicación efectiva, la negociación constructiva y la asunción de responsabilidad compartida.
- **Fortalecimiento de la Capacidad de Resolución de Problemas Complejos:** Los estudiantes aprenden a analizar situaciones desafiantes, a identificar múltiples soluciones potenciales y a evaluar críticamente su efectividad.
- **Contextualización Profunda del Conocimiento:** Al abordar problemas auténticos y relevantes, los estudiantes comprenden la aplicabilidad práctica y la significación del conocimiento adquirido.
- **Incremento de la Motivación y el Interés Intrínseco:** La naturaleza desafiante y la relevancia contextual de los proyectos suelen estimular la motivación y el compromiso activo de los estudiantes.
- **Cultivo del Pensamiento Crítico y la Toma de Decisiones Informadas:** La investigación rigurosa y la evaluación reflexiva requieren que los estudiantes analicen información de manera crítica, formulen argumentos sólidos y tomen decisiones fundamentadas.

- **Desarrollo de la Autonomía y la Responsabilidad Personal:** Los estudiantes aprenden a gestionar su propio proceso de aprendizaje y a asumir la responsabilidad de la consecución de sus tareas y metas.

2.4. Pensamiento Computacional

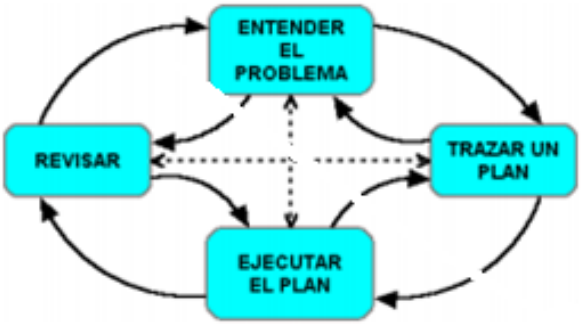
El concepto de pensamiento computacional (*Computational Thinking*) ha emergido como una habilidad fundamental en el siglo XXI, trascendiendo los límites de la informática y permeando diversas disciplinas (Wing, 2006). A pesar de su creciente reconocimiento, la definición precisa y la identificación de sus componentes principales aún presentan cierta variabilidad en la literatura académica (Adell Segura, y otros, 2019). No obstante, la esencia del pensamiento computacional radica en un conjunto de habilidades y competencias de aplicación universal que permiten abordar problemas de manera sistemática y creativa, independientemente del campo de estudio o la aplicación profesional (Wing, 2006).

Como señalan Polanco y otros (2021), el estudio de la programación informática y algorítmica ha evolucionado conceptualmente hasta consolidarse como un identificador propio: el pensamiento computacional. En este sentido, el desarrollo del pensamiento computacional se considera un componente primario y esencial al momento de introducir la programación, ya que proporciona las habilidades y competencias básicas que son comunes a diversos paradigmas y lenguajes de programación. Es importante destacar que el pensamiento computacional no depende exclusivamente del uso de sistemas informáticos, sino que se concibe como una competencia fundamental que facilita el análisis y la solución de problemas prácticos. Esto implica el desarrollo de habilidades que permiten a los individuos comprender e interactuar de una manera efectiva con diferentes tecnologías, tanto físicas como digitales, que responden a problemáticas del mundo moderno y que pueden ser programadas para encontrar soluciones (Gurises Unidos, 2017).

Un componente central del pensamiento computacional aplicado a la solución de problemas es la creación y el desarrollo de algoritmos. Para comprender esta relación, es necesario analizar las fases involucradas en la resolución de problemas y cómo estas se vinculan con la creación de algoritmos y el pensamiento computacional. Uno de los modelos más influyentes en la resolución de problemas es el modelo heurístico cíclico propuesto por Poyla (1957), que

consta de cuatro fases: comprender el problema, trazar un plan, ejecutar el plan y revisar la solución, tal y como se observa en la Figura 2 (López García, 2009, p. 8, Ilustración 1-1).

Figura 2 *Interpretación dinámica y cíclica de las etapas planteadas por Polya para resolver problemas*



Fuente: (López García, 2009, pág. 8)

Según este modelo, la resolución de un problema requiere un proceso de análisis que implica la comprensión detallada del problema, la descomposición del mismo en sus componentes esenciales para facilitar la ideación y el diseño de un plan de solución factible, la ejecución rigurosa de dicho plan y, finalmente, la revisión exhaustiva para verificar si la solución obtenida responde adecuadamente a la situación planteada inicialmente. Una característica distintiva de este modelo es su naturaleza cíclica, que permite retornar a cualquiera de las fases previas en caso de identificar obstáculos, errores o la necesidad de replantear el proceso con base en nueva información o restricciones del problema.

Es fundamental reconocer que, a diferencia de una percepción inicial, este modelo no está intrínsecamente ligado a una disciplina específica. Su aplicación a problemas prácticos del mundo real a menudo requiere un enfoque multidisciplinar, integrando conocimientos matemáticos y científicos, así como habilidades lingüísticas y competencias técnicas que contribuyen de manera conjunta a la solución deseada.

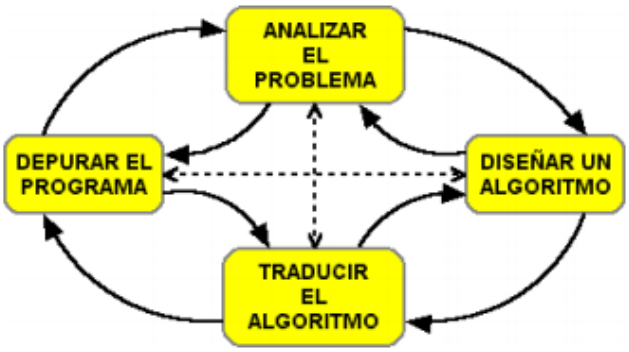
Una vez comprendido el modelo de Polya, podemos definir un algoritmo y analizar su relación con este. Un algoritmo se define como un conjunto finito de pasos secuenciales que siguen un conjunto de reglas establecidas, cuya ejecución permite la solución de un problema o la realización de una tarea específica. Si bien el concepto de algoritmo tiene raíces históricas en el álgebra, su aplicación trasciende las matemáticas, siendo una herramienta cognitiva utilizada de manera constante, tanto consciente como inconscientemente, en la solución de problemas cotidianos. Ejemplos comunes incluyen una receta culinaria, las instrucciones para operar maquinaria compleja o la planificación de una ruta de viaje óptima.

Para que un algoritmo sea programable y aplicable en el pensamiento computacional, debe cumplir con una serie de características principales (Baena, 2014): estar bien definido, ser finito, ser sencillo, ser efectivo, ser eficiente y poseer un número de entradas y salidas claramente delimitadas.

Como se muestra en la Figura 3 (López García, 2009, p. 11, Ilustracion 1-3) existe un paralelismo significativo entre las etapas del desarrollo de un programa a partir de un algoritmo programable y el modelo heurístico de Polya, donde se encuentran las siguientes equivalencias:

- Analizar el problema < - > Entender el problema
- Diseñar un algoritmo < - > Trazar un plan
- Traducir el algoritmo < - > Ejecutar el plan
- Depurar el programa < - > Revisar

Figura 3 *Fases para elaborar un programa a partir de un algoritmo programable.*



Fuente: (López García, 2009, pág. 11)

El proceso de desarrollar un algoritmo programable implica tomar el enunciado del problema y, a partir de él, formularlo claramente, identificar los datos disponibles, las restricciones y condiciones relevantes, determinar los resultados esperados y, con base en esta información, definir los procesos y acciones necesarias para lograr la solución. De manera similar al modelo de Polya, se ha establecido un modelo heurístico para el análisis de problemas a partir de su enunciado (Schunk, 1997), que enfatiza la importancia de la formulación del problema, la identificación de los resultados esperados, la planificación a partir de los datos disponibles y sus restricciones, y los procesos necesarios para alcanzar una posible solución al problema, tal y como se muestra en la Figura 4 (López García, 2009, p. 11, Ilustracion 1-3).

Figura 4 *Etapas a desarrollar en la fase de análisis de un problema (Entenderlo)*

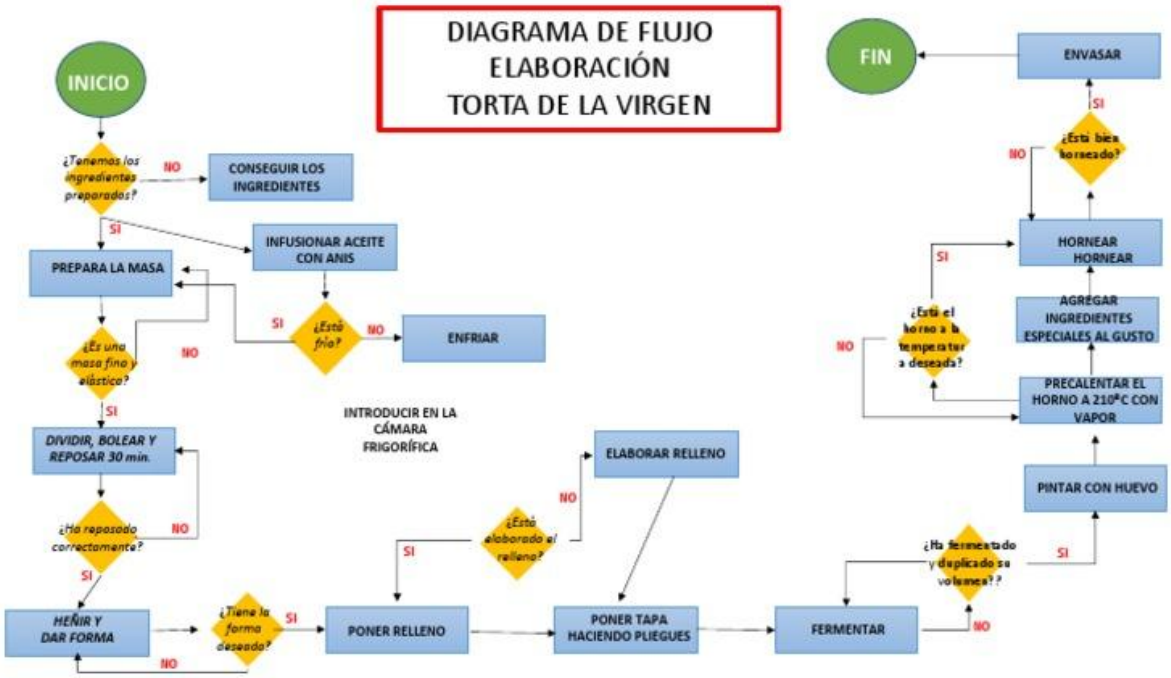
Fuente: (López García, 2009, pág. 11)

Es crucial destacar que el análisis de problemas, en este contexto, no requiere necesariamente un dominio avanzado de las matemáticas. Más bien, exige un enfoque multidisciplinar que abarca competencias como el manejo numérico, la teoría de conjuntos, habilidades lingüísticas, conocimientos científicos e incluso competencias técnicas para interpretar la información del enunciado y construir el algoritmo programable de manera efectiva.

2.5. Herramientas Tecnológicas *No Code* En La Educación

Una característica fundamental de los algoritmos es la flexibilidad con la que se pueden representar, permitiendo su expresión a través de diversas metodologías, siendo los diagramas de flujo y el pseudocódigo dos de las más comunes. Si bien el pseudocódigo presenta una mayor cercanía a las estructuras de los lenguajes de programación formales, en el ámbito educativo (sobre todo en la educación secundaria) los diagramas de flujo ofrecen una mayor versatilidad ya que por su naturaleza visual se facilita la introducción didáctica a la creación de algoritmos programables, permitiendo a los estudiantes comprender de manera más intuitiva la lógica secuencial y las estructuras de control mediante la identificación de bloques funcionales básicos, sin la necesidad de conocer ningún tipo de codificación (ProgramacionColVia, s.f.). Esto se puede aplicar a cualquier tipo de problema practico, tal y como se muestra a continuación en la Figura 5 (Gutiérrez León, 2024), en la cual se representa el proceso de elaboración de una torta de la Virgen por medio de un algoritmo en diagrama de flujo.

Figura 5 *Representación en diagrama de flujo del algoritmo para la receta de una torta de la virgen*



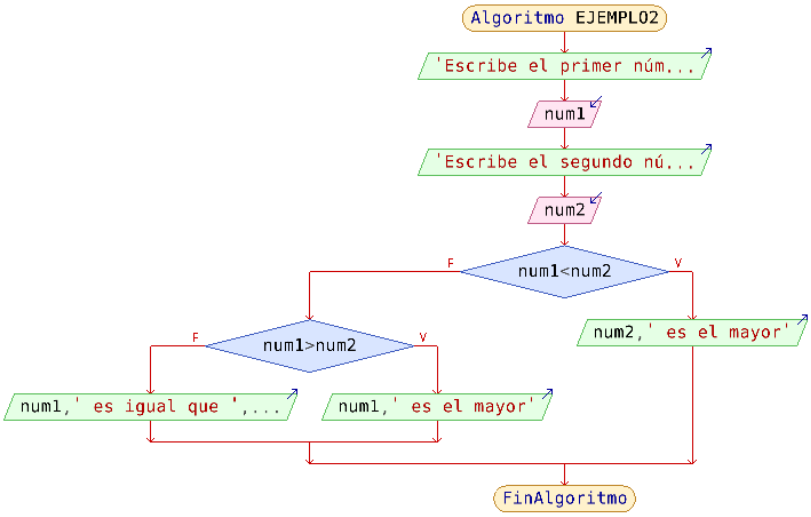
Fuente: (Gutiérrez León, 2024)

La elaboración de algoritmos programables mediante diagramas de flujo se apoya significativamente en el desarrollo de competencias lingüísticas y relacionales, así como en la comprensión de la teoría de conjuntos, más allá de la tradicional concepción de la programación como un dominio eminentemente matemático y científico. La clave radica en la capacidad de definir instrucciones precisas y formular expresiones condicionales como relaciones lógicas entre las variables definidas en el algoritmo.

Una herramienta pedagógica ampliamente utilizada para la enseñanza de la implementación de algoritmos programables a través de diagramas de flujo es PSeInt. Este software fue diseñado específicamente para la introducción al pensamiento computacional y los primeros pasos en programación, ofreciendo una interfaz intuitiva que permite la implementación de algoritmos en pseudocódigo, complementada con un editor de diagramas de flujo. La utilización de bloques funcionales visualmente similares a los empleados en la elaboración de diagramas de flujo básicos facilita el aprendizaje y la posterior transferencia de estos conceptos a otros entornos.

Un ejemplo del uso de este software se puede evidenciar en la Figura 6 (Librería CATEDU, 2022), que muestra un algoritmo en diagrama de flujo desarrollado en PSeInt que pregunta dos números diferentes al usuario, los compara, y luego muestra el número mayor en pantalla.

Figura 6 *Diagrama de flujo de un programa en PSeInt que a partir de dos números compara cuál es mayor y lo muestra en pantalla.*



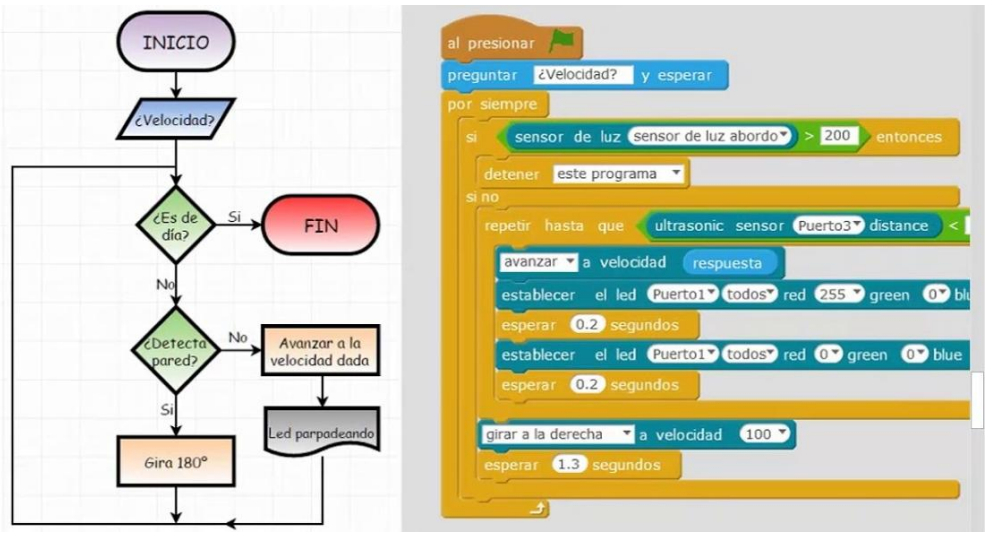
Fuente: (Librería CATEDU, 2022)

Una vez que los estudiantes comprenden los fundamentos de la creación de algoritmos programables y su representación visual, la transición a otras herramientas y plataformas, incluyendo lenguajes de programación y tecnologías *no code*, se simplifica debido a la similitud en la lógica de los bloques funcionales.

2.5.1. Diagramas de Flujo y Tecnologías *No Code*

En el campo de la programación, las tecnologías *no code* han ganado un impulso significativo. Estas tecnologías emergen como una alternativa dentro del concepto de Desarrollo Rápido de Aplicaciones (RAD), respondiendo a las limitaciones del desarrollo de software tradicional, caracterizado por su lentitud y la necesidad de un amplio conocimiento en codificación, lenguajes y paradigmas de programación (Moreno, s.f.). La idea detrás de esta metodología busca facilitar la creación de programas, permitiendo a profesionales de diversas áreas construir aplicaciones sin la necesidad de un profundo conocimiento de programación. Esto se logra mediante interfaces gráficas intuitivas donde los usuarios manipulan elementos visuales (bloques) mediante acciones de arrastrar y soltar. Cada bloque representa una acción o instrucción programática, identificada por formas, colores y etiquetas que facilitan la creación de programas funcionales para sitios web, aplicaciones móviles y software en general (Omatech, 2022)

Figura 7 *Diagrama de flujo y programa no code equivalente para programación de un Arduino usando Scratch*



Fuente: (Juegos Robótica, s.f)

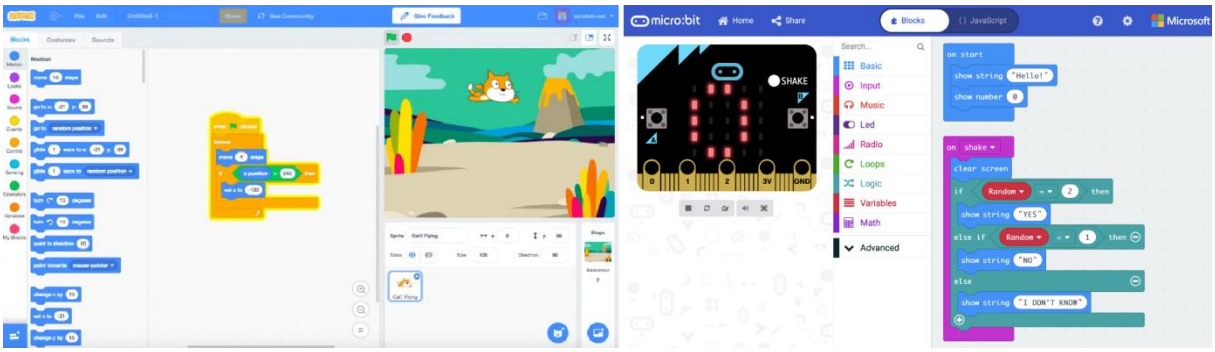
Una de las ventajas destacadas de las tecnologías *no code* es su estrecha relación con los diagramas de flujo. En ambos enfoques, cada elemento o bloque representa una acción o instrucción específica dentro del programa. Esta similitud permite una transición fluida y sencilla en la construcción de programas utilizando tecnologías *no code* directamente a partir de diagramas de flujo de algoritmos programables. La equivalencia entre las instrucciones específicas en ambos formatos facilita un método de partes casi intercambiables. Un ejemplo de esto se muestra en la Figura 6 (Juegos Robótica, s.f), donde se puede observar la construcción de un algoritmo en diagrama de flujo y su respectiva implementación en un programa no code para Arduino usando Scratch.

La tecnología *no code* ha encontrado aplicaciones en diversos campos, incluyendo la educación. Su propósito inicial de permitir a personas sin una formación extensa en ciencias de la computación crear programas se alinea perfectamente con los objetivos educativos de introducir conceptos de programación a estudiantes de diferentes niveles sin la barrera de la codificación tradicional. El requisito fundamental para esto es el desarrollo de una lógica programable basada en la creación de diagramas de flujo a partir de algoritmos programables.

2.6. Integración del Pensamiento Computacional y el ABP con Herramientas *No Code*

Scratch y Micro:bit son dos de las herramientas *no code* más comúnmente empleadas en educación. Ambas plataformas ofrecen entornos amigables con interfaces intuitivas y herramientas sencillas y visualmente similares tal y como se ve en la Figura 8 (l3tcrafteducacion.com, 2022; microbit.microlog.es, 2019), lo que facilita a los estudiantes la identificación y comprensión de las funciones de los diferentes bloques utilizados al crear programas. Este modelo de programación por bloques permite la introducción de conceptos básicos de programación y la creación de algoritmos programables a estudiantes de diversos rangos de edad.

Figura 8 *Programación usando diagramas de bloques en Scratch (Izquierda) y MicroBIT (Derecha)*

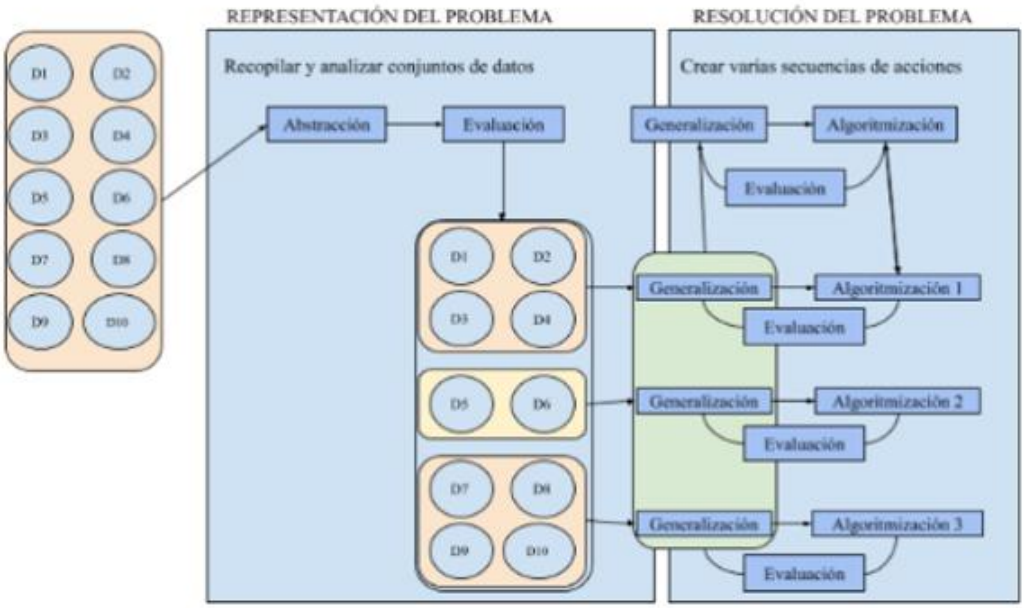


Nota: (a) Ejemplo de programación usando Scratch Fuente: (l3tcrafteducacion.com, 2022) – (b) Ejemplo de programación usando Micro:bit Fuente: (microbit.microlog.es, 2019)

La implementación de proyectos educativos basados en Scratch y Micro:bit puede articularse de manera efectiva con los modelos de resolución de problemas propuestos por Polya (1957) y Schunk (1997). Al enfrentarse a un desafío o tarea usando estas herramientas tecnológicas no code, los estudiantes deben comprender el problema (identificar los objetivos del proyecto), trazar un plan (diseñar la lógica del algoritmo utilizando bloques), ejecutar el plan (programar en Scratch o Micro:bit) y revisar (probar y depurar su código). Este proceso cíclico fomenta la reflexión metacognitiva y la mejora iterativa, tal como se destaca en la obra de Wiggins (1998) sobre la evaluación formativa.

Además, la perspectiva de Ortega-Ruipérez (2020) subraya cómo el pensamiento computacional permite descomponer los procesos de pensamiento durante la resolución de problemas, tal y como se aprecia en la Figura 9 (Ortega-Ruipérez, 2020, p. 135, Figura 2). Al utilizar Scratch y Micro:bit en el contexto del Aprendizaje Basado en Proyectos (ABP), los estudiantes pueden aplicar este enfoque para abordar problemas reales y contextuales, tal como se delineó en el objetivo general de este TFM. La naturaleza lúdica y creativa de estas herramientas motiva la participación activa y el aprendizaje significativo, permitiendo a los estudiantes experimentar con conceptos abstractos de programación y robótica de una manera tangible y relevante para su entorno.

Figura 9 *Representación simplificada de los procesos de pensamiento en la resolución de problemas aplicando el pensamiento computacional.*



Fuente: (Ortega-Ruipérez, 2020, pág. 135)

En este sentido, la integración de Scratch y Micro:bit como herramientas *no code* dentro de una estrategia de enseñanza basada en el ABP se presenta como un enfoque prometedor para el desarrollo del pensamiento computacional en estudiantes de educación secundaria. Al basarse en modelos pedagógicos sólidos como los de Polya, Schunk y Ortega, y al aprovechar la conexión entre algoritmos, diagramas de flujo y la programación por bloques, se puede facilitar la creación de algoritmos programables que respondan a problemas auténticos y promuevan el desarrollo de habilidades esenciales para el siglo XXI.

2.7. Evaluación y Seguimiento del Pensamiento Computacional

La evaluación del desarrollo del pensamiento computacional en estudiantes de educación secundaria representa un desafío complejo pero crucial para comprender la efectividad de las intervenciones pedagógicas y para guiar la mejora continua de los procesos de enseñanza y aprendizaje (Brennan & Resnick, 2012). Dada la naturaleza multifacética del pensamiento computacional, que abarca habilidades cognitivas, metacognitivas y creativas, se requiere la implementación de métodos y herramientas de evaluación diversos y complementarios (Román-González, y otros, 2017).

2.7.1. Métodos y Herramientas para la Evaluación del Pensamiento Computacional:

La literatura académica y las experiencias previas en la evaluación del pensamiento computacional sugieren una variedad de enfoques:

- **Evaluación Basada en Tareas y Proyectos:** Una de las formas más directas de evaluar el pensamiento computacional es a través del análisis del desempeño de los estudiantes en tareas y proyectos de programación o resolución de problemas computacionales (Weintrop, y otros, 2016). Esto puede incluir la evaluación de la eficiencia, la claridad, la corrección y la creatividad de las soluciones propuestas, así como el proceso seguido para llegar a ellas. El análisis de los artefactos digitales creados por los estudiantes en plataformas como Scratch o Micro:bit puede proporcionar información valiosa sobre su capacidad para aplicar los componentes del pensamiento computacional (Brennan & Resnick, 2012).
- **Cuestionarios y Pruebas de Opción Múltiple:** Si bien el pensamiento computacional es inherentemente práctico, los cuestionarios y las pruebas de opción múltiple pueden utilizarse para evaluar la comprensión conceptual de sus componentes y principios (Román-González, y otros, 2017). Estas herramientas pueden diseñarse para medir la capacidad de los estudiantes para identificar ejemplos de descomposición, reconocimiento de patrones, abstracción y diseño de algoritmos en contextos específicos.
- **Rúbricas de Evaluación:** El uso de rúbricas detalladas permite establecer criterios claros y transparentes para evaluar diferentes aspectos del pensamiento computacional en las tareas y proyectos de los estudiantes (ISTE (International Society

for Technology in Education), 2016). Las rúbricas pueden centrarse en dimensiones como la planificación, la implementación, la depuración, la comunicación y la reflexión sobre el proceso de resolución de problemas.

- **Portafolios de Evidencias:** La recopilación sistemática de los trabajos de los estudiantes a lo largo del tiempo en un portafolio puede ofrecer una visión longitudinal de su progreso en el desarrollo del pensamiento computacional (Barlex & Carr, 2007). El portafolio puede incluir proyectos, diagramas de flujo, reflexiones y autoevaluaciones.
- **Observación Directa:** La observación estructurada del comportamiento de los estudiantes mientras trabajan en actividades de pensamiento computacional puede proporcionar información cualitativa valiosa sobre sus estrategias de resolución de problemas, su colaboración y su compromiso (NRC (National Research Council), 2011).
- **Entrevistas y Grupos Focales:** Las entrevistas individuales o los grupos focales con estudiantes pueden explorar sus percepciones sobre el pensamiento computacional, sus experiencias al aplicarlo y sus desafíos y logros (Cohen, y otros, 2018).

2.7.2. Consideraciones Éticas y Metodológicas en la Evaluación Educativa:

La evaluación del pensamiento computacional, al igual que cualquier forma de evaluación educativa, debe llevarse a cabo considerando principios éticos y metodológicos fundamentales (Popham, 2018):

- **Validez:** Los instrumentos y métodos de evaluación deben medir de manera precisa y significativa los aspectos del pensamiento computacional que se pretenden evaluar.
- **Confiabilidad:** Los resultados de la evaluación deben ser consistentes y estables a lo largo del tiempo y entre diferentes evaluadores.
- **Equidad:** La evaluación debe ser justa y equitativa para todos los estudiantes, teniendo en cuenta sus diversas experiencias y antecedentes. Se deben evitar sesgos culturales, lingüísticos o de género en el diseño y la aplicación de las evaluaciones.
- **Transparencia:** Los criterios de evaluación y los propósitos de la evaluación deben ser claros y comunicados de manera efectiva a los estudiantes y a otros actores relevantes.

- **Retroalimentación:** La evaluación debe proporcionar retroalimentación formativa y oportuna a los estudiantes para apoyar su aprendizaje y desarrollo.
- **Privacidad y Confidencialidad:** Los datos de la evaluación deben manejarse con respeto a la privacidad y la confidencialidad de los estudiantes.
- **Uso Responsable de los Resultados:** Los resultados de la evaluación deben utilizarse de manera ética y responsable para informar la toma de decisiones pedagógicas y mejorar la calidad de la enseñanza.

En el contexto del Aprendizaje Basado en Proyectos (ABP) y el uso de herramientas *no code* como Scratch y Micro:bit, la evaluación del pensamiento computacional puede integrarse de manera natural a través del análisis de los proyectos desarrollados por los estudiantes, la observación de su proceso de trabajo colaborativo y la reflexión sobre sus aprendizajes. El diseño de rúbricas específicas para evaluar las dimensiones del pensamiento computacional en estos contextos puede ser una estrategia efectiva para proporcionar una retroalimentación significativa y para realizar un seguimiento del progreso de los estudiantes a lo largo del tiempo.

3. Contextualización Y Diseño Del Proyecto De Innovación

La creciente relevancia del pensamiento computacional como competencia digital esencial para el siglo XXI (Wing, 2006), junto con la necesidad de implementar metodologías pedagógicas activas que promuevan el aprendizaje significativo y la autonomía estudiantil, fundamentan la presente propuesta de innovación. En este sentido, se puede considerar que el Aprendizaje Basado en Proyectos (ABP) es una estrategia pedagógica idónea para articular la adquisición de conocimientos y el desarrollo de competencias orientado a la resolución de problemas (Thomas, 2000); mientras que las herramientas tecnológicas no code ofrecen un acceso intuitivo a la creación de algoritmos programables, facilitando la aproximación a la programación y la experimentación en un entorno tecnológico (Omatech, 2022). En este contexto, la presente propuesta de innovación se alinea directamente con el objetivo general de este Trabajo de Fin de Estudios (TFM): Desarrollar el pensamiento computacional en estudiantes de educación secundaria de grado octavo mediante una estrategia pedagógica de enseñanza basada en proyectos (ABP) y la utilización de herramientas tecnológicas no code como Scratch y Micro:bit.

Esta propuesta de innovación educativa, está diseñada con el fin de ser implementada en la Institución Educativa “Instituto Montenegro” en respuesta a la necesidad detectada de fortalecer las habilidades de pensamiento lógico y la comprensión de los fundamentos de la programación, se basa en la premisa de que la integración del ABP con herramientas no code puede proporcionar un entorno de aprendizaje motivador que permita cultivar en los estudiantes habilidades del pensamiento computacional, incluyendo la descomposición, el reconocimiento de patrones, la abstracción y el diseño de algoritmos (Adell Segura, y otros, 2019). Al involucrar a los estudiantes en la resolución de problemas prácticos a través de proyectos y permitirles materializar sus ideas mediante la programación visual, se aspira a fomentar una comprensión más profunda y un aprendizaje más significativo de estos conceptos, preparándolos para desenvolverse de manera competente en un mundo crecientemente digitalizado.

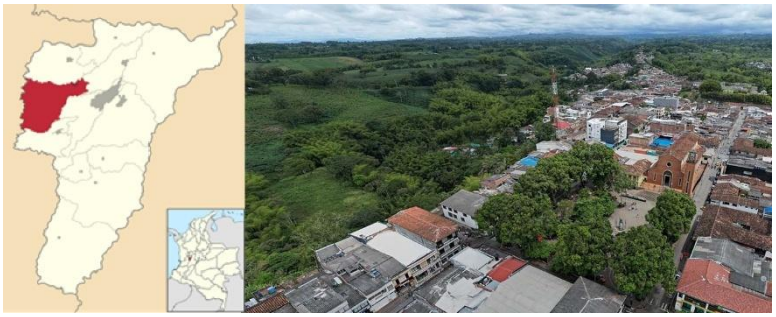
3.1.Contextualización

El proyecto de innovación se plantea para su implementación en la Institución Educativa "Instituto Montenegro", un establecimiento público de carácter urbano ubicado en la ciudad de Montenegro, en el departamento del Quindío, Colombia

3.1.1. Descripción del Entorno Socio-Geográfico

La Institución Educativa "Instituto Montenegro" se localiza en el corazón de la zona urbana de Montenegro, un municipio del departamento del Quindío, el cual forma parte del reconocido "eje cafetero" colombiano, tal y como se muestra en la Figura 10(a) (Wikipedia, s.f). Esta región se caracteriza por una economía predominantemente agrícola, centrada en el cultivo del café, especialmente en municipios con una extensa área rural como Montenegro. Adicionalmente, como consecuencia del paisaje cultural cafetero de la región, el ecoturismo se posiciona como una actividad económica secundaria importante, aprovechando las haciendas cafeteras y parques temáticos cercanos, como se puede observar en la Figura 10(b) (Invest in Armenia, 2024).

Figura 10 *Contexto Geográfico y Paisajístico del Municipio de Montenegro, Quindío.*



Nota: (a) Mapa de Ubicación Geográfica de Montenegro, Quindío. Fuente: (Wikipedia, s.f) – (b) Fotografía del municipio de Montenegro, Quindío mostrando el Paisaje Cultural Cafetero. Fuente: (Invest in Armenia, 2024)

3.1.2. Características de la Población Estudiantil y Contexto Social

La Institución Educativa "Instituto Montenegro" opera con recursos de la Secretaría de Educación Departamental del Quindío y cuenta con dos sedes: La sede principal que es para la educación secundaria y media, la que se observa en la Figura 11 (I.E "Instituto Montenegro", 2020), y la sede "Rafael Uribe Uribe" para primaria. La institución se encuentra ubicada en la Carrera 5 N° 11-07, cerca de puntos de referencia como la estación de bomberos y la plaza municipal, la institución atiende principalmente a estudiantes provenientes de los estratos socioeconómicos 1 y 2 del municipio, principalmente de barrios aledaños como La Isabela y La Soledad.

Figura 11 *Institución Educativa "Instituto Montenegro"*

Fuente: (I.E "Instituto Montenegro", 2020)

La institución ofrece formación educativa a aproximadamente 420 estudiantes en preescolar y primaria, y a 600 estudiantes en básica secundaria y media. En el área de Tecnología e Informática, se asignan dos (2) horas semanales para cada grupo. Los estudiantes de secundaria, con edades comprendidas entre los 6 y 18 años, proviene mayoritariamente de familias con recursos económicos limitados, a menudo con ausencia de uno o ambos progenitores y bajos niveles de escolaridad. Para muchas de estas familias, la educación pública representa la única vía para la formación de sus hijos. Si bien el acceso a recursos tecnológicos en los hogares ha aumentado gradualmente tras la pandemia de COVID-19, especialmente el uso de *smartphones*, la disponibilidad de computadores y conexiones a internet de calidad sigue siendo limitada.

En cuanto al rendimiento específico en el área de Tecnología e Informática, no se dispone de resultados de pruebas internas o externas estandarizadas. Sin embargo, la observación del desempeño de los estudiantes al ingresar al grado sexto y a la media técnica revela una carencia de conocimientos teóricos y prácticos en el uso del computador, lo que dificulta el desarrollo de los planes de estudio previstos. Esta situación se atribuye, en parte, a la falta de acceso a recursos tecnológicos en el hogar para la práctica y la consulta.

3.1.3. Recursos e Instalaciones Disponibles

Como institución pública, la Institución Educativa “Instituto Montenegro” depende de los recursos e instalaciones proporcionados por la Secretaría de Educación Departamental del Quindío. Actualmente, para la básica secundaria y media, la institución cuenta con dos salas de informática equipadas con computadores portátiles individuales para cada estudiante.

Adicionalmente, desde el año 2020, la Secretaría de Educación Departamental del Quindío ha implementado un programa de mejora de la infraestructura y los recursos de las instituciones

educativas, incluyendo la dotación de Aulas STEAM (Ciencia, Tecnología, Ingeniería, Arte y Matemáticas) con el objetivo de fomentar la implementación de proyectos pedagógicos orientados al desarrollo de habilidades y competencias digitales, en consonancia con los lineamientos del Ministerio de Educación Nacional (SEDQuindio, 2022). A la fecha, se han instalado tres Aulas STEAM en la Institución Educativa "Instituto Montenegro", destinadas a las áreas de Lenguaje, Matemáticas y Tecnología e Informática. Si bien el proceso de capacitación docente para el uso de estos espacios se ha iniciado y las aulas se encuentran completamente equipadas, aún no se han implementado proyectos pedagógicos de manera formal en estas nuevas instalaciones.

3.2. Personas Destinatarias del Proyecto de Innovación

Los beneficiarios directos de este proyecto de innovación serán estudiantes de grado octavo de la Institución Educativa "Instituto Montenegro". Este nivel educativo se ha seleccionado considerando que los estudiantes se encuentran un momento crucial para el desarrollo de habilidades de pensamiento lógico y la introducción a conceptos de programación.

El grupo de estudiantes de grado octavo se compone de aproximadamente 70 estudiantes, distribuidos en dos grupos. La propuesta de innovación educativa se diseñará como un curso completo para la enseñanza del pensamiento lógico programable, con el objetivo de que los estudiantes desarrollen proyectos grupales de aprendizaje colaborativo. La implementación de estos proyectos se realizará utilizando herramientas tecnológicas *no code* como Scratch o Micro:bit, previamente analizadas en el marco teórico como entornos amigables y efectivos para la iniciación en la programación y el desarrollo del pensamiento computacional (Ortega-Ruipérez, 2020)

Se tendrá especial consideración a la inclusión de estudiantes con Necesidades Específicas de Apoyo Educativo (NEAE) dentro de los grupos. La metodología ABP y las herramientas *no code* ofrecen flexibilidad para adaptar las actividades y los niveles de desafío a las diversas necesidades de aprendizaje. Se implementarán estrategias de agrupamiento flexible, apoyo individualizado y la utilización de recursos visuales y multisensoriales para garantizar la participación y el progreso de todos los estudiantes, promoviendo un entorno de aprendizaje inclusivo y equitativo (Florian & Black-Hawkins, 2011).

3.3. Objetivos, Contenidos y Competencias Básicas/Clave del Proyecto de Innovación

El presente proyecto de innovación pedagógica se articula en torno a un objetivo general y una serie de objetivos específicos que buscan fomentar el desarrollo del pensamiento computacional en estudiantes de educación secundaria mediante la implementación de una estrategia pedagógica denominada Aprendizaje Basado en Proyectos (ABP) y la utilización de herramientas tecnológicas *no code* como Scratch y Micro:bit. Estos objetivos se centran en los aprendizajes que se espera adquieran los estudiantes como resultado de su participación en la propuesta.

3.3.1. Objetivo General

Desarrollar el pensamiento computacional en estudiantes de grado octavo de la Institución Educativa "Instituto Montenegro" mediante la implementación de una estrategia pedagógica denominada Aprendizaje Basado en Proyectos (ABP) que integre la creación de algoritmos programables y su representación a través de diagramas de flujo, culminando en su implementación práctica utilizando herramientas tecnológicas *no code* como Scratch y/o Micro:bit.

3.3.2. Objetivos Específicos

Para alcanzar el objetivo general propuesto, se establecen los siguientes objetivos específicos:

- Introducir a los estudiantes en los conceptos clave del pensamiento computacional, incluyendo la descomposición de problemas, el reconocimiento de patrones, la abstracción y el diseño de algoritmos (Wing, 2006).
- Capacitar a los estudiantes en el diseño y formulación de algoritmos secuenciales y condicionales para la resolución de problemas específicos (Baena, 2014).
- Enseñar a los estudiantes el uso de los diagramas de flujo como una herramienta visual para representar de una manera comprensible la lógica y la secuencia de los algoritmos programables (ProgramacionColVia, s.f.).
- Familiarizar a los estudiantes con el uso de plataformas *no code* como Scratch y/o Micro:bit para la implementación práctica de los algoritmos diseñados en el desarrollo de actividades (Brennan & Resnick, 2012).

- Guiar a los estudiantes en el desarrollo y la ejecución de proyectos basados en problemas o desafíos relevantes para su contexto, aplicando los principios del pensamiento computacional y las herramientas *no code* (Thomas, 2000).
- Promover el trabajo en equipo y la comunicación efectiva entre los estudiantes durante el desarrollo de los proyectos (Johnson & Johnson, 2009).
- Estimular en los estudiantes la reflexión sobre su propio proceso de aprendizaje, identificando fortalezas, debilidades y estrategias de mejora que fomenten el desarrollo del pensamiento computacional (Flavell, 1976).

3.3.3. Contenidos del Proyecto

Los contenidos que se abordarán en este proyecto se organizan en torno a los siguientes ejes temáticos:

1. **Introducción al Pensamiento Computacional:**
    - Definición y relevancia del pensamiento computacional.
    - Componentes fundamentales: descomposición, reconocimiento de patrones, abstracción, diseño de algoritmos.
    - Aplicaciones del pensamiento computacional en la vida cotidiana y en diversas disciplinas.
2. **Fundamentos de Algoritmia:**
    - Concepto de algoritmo y sus características (definido, finito, sencillo, efectivo, eficiente, entradas y salidas).
    - Tipos de algoritmos: secuenciales, condicionales.
    - Variables y tipos de datos básicos.
    - Operadores lógicos y relacionales.
3. **Representación de Algoritmos con Diagramas de Flujo:**
    - Simbología básica de los diagramas de flujo (inicio/fin, proceso, decisión, entrada/salida, conector).

- Construcción de diagramas de flujo para algoritmos secuenciales y condicionales.
- Traducción de algoritmos a diagramas de flujo y viceversa.

4. **Introducción a Plataformas *No Code* (Scratch y/o Micro:bit):**
    - Interfaz y entorno de programación por bloques.
    - Bloques de control, movimiento, apariencia, sonido, sensores (según la plataforma).
    - Implementación de estructuras secuenciales y condicionales utilizando bloques.
5. **Aprendizaje Basado en Proyectos (ABP):**
    - Definición y principios del ABP.
    - Fases del desarrollo de un proyecto (identificación del problema, planificación, ejecución, presentación, evaluación).
    - Estrategias de trabajo colaborativo.
6. **Aplicación del Pensamiento Computacional en Proyectos:**
    - Identificación y definición de problemas o desafíos relevantes.
    - Diseño de soluciones algorítmicas para los problemas planteados.
    - Implementación de las soluciones utilizando Scratch y/o Micro:bit.
    - Prueba y depuración de los proyectos.
    - Presentación y evaluación de los resultados.

3.3.4. Competencias Básicas/Clave (según el contexto educativo colombiano):

Este proyecto contribuye al desarrollo de diversas competencias básicas y clave, enmarcadas en los lineamientos del Ministerio de Educación Nacional de Colombia (Ministerio de Educación Nacional (MEN), 2022), entre las que se destacan:

- **Competencias en Ciencia, Tecnología, Ingeniería y Matemáticas (STEM):** A través de la resolución de problemas, el diseño de algoritmos y la implementación de proyectos tecnológicos.

- **Competencia Digital:** Mediante el uso de herramientas tecnológicas *no code* y la comprensión de los principios fundamentales de la programación.
- **Aprender a aprender:** Fomentando la autonomía en el proceso de aprendizaje, la reflexión metacognitiva y la capacidad de buscar y gestionar información.
- **Competencias sociales y cívicas:** Promoviendo el trabajo colaborativo, el respeto por las ideas de los demás y la participación activa en la resolución de problemas grupales.
- **Comunicación lingüística:** A través de la necesidad de expresar claramente las ideas, describir los algoritmos y presentar los resultados de los proyectos.
- **Pensamiento crítico y resolución de problemas:** Desarrollando la capacidad de analizar situaciones, identificar problemas, proponer soluciones creativas y evaluar su efectividad.
- **Creatividad e innovación:** Estimulando la generación de ideas originales y la búsqueda de soluciones novedosas a los desafíos planteados en los proyectos.

3.4. Metodología del Proyecto de Innovación

La implementación de la presente propuesta de innovación se fundamenta en el principio metodológico del Aprendizaje Basado en Proyectos (ABP), una estrategia pedagógica centrada en el estudiante que promueve la autonomía, el aprendizaje colaborativo y el desarrollo de competencias comunicativas (Thomas, 2000). El ABP se articula mediante la presentación de situaciones problema contextualizadas en el mundo real y preferiblemente en el entorno de los estudiantes (López Belarrinaga & Martínez Pérez, 2023), las cuales se convierten en el eje central del proceso de aprendizaje; esta metodología se caracteriza por su flexibilidad, permitiendo la estructuración del proceso en diferentes fases (Presentación, Planificación, Investigación, Elaboración, Retroalimentación y Presentación) impulsando el desarrollo de la autonomía y el trabajo colaborativo, estas fases culminan en la creación de un producto tangible que busca ofrecer una solución práctica a la problemática planteada en la situación inicial (IESPE (Instituto de Estudios Superiores en Pedagogía y Educación Continua), 2023).

3.4.1. Formas de Agrupamiento: Aprendizaje Colaborativo

Para el desarrollo de los proyectos, se fomentará el aprendizaje colaborativo mediante la conformación de grupos de trabajo de tres integrantes. Se indicará a los estudiantes la importancia de la asignación de roles específicos dentro de cada grupo, incentivándolos a considerar sus fortalezas y habilidades individuales para optimizar el desempeño colectivo. Los roles propuestos son:

- Coordinador: Responsable de la comunicación y portavoz del grupo.
- Secretario: Encargado de la organización y documentación de las tareas.
- Evaluador: Responsable de analizar de forma crítica y realista los avances y posibles obstáculos.
- Encargado: Responsable de la gestión y cuidado de los materiales utilizados (rol común a todos los integrantes).
- Investigador: Responsable de la búsqueda de información relevante para asegurar la viabilidad del proyecto (rol común a todos los integrantes).

En caso de que los estudiantes no logren conformar los grupos de manera autónoma, la asignación se realizará de forma aleatoria mediante herramientas digitales para garantizar la equidad y la participación de todos.

3.4.2. Recursos para el Desarrollo del Proyecto

El desarrollo de la presente propuesta de innovación se llevará a cabo utilizando los recursos disponibles en la Institución Educativa “Instituto Montenegro”. Se emplearán tanto la sala de informática del Área de Tecnología e Informática como las Aulas STEAM, las cuales cuentan con los equipos informáticos necesarios, incluyendo computadores portátiles individuales para cada estudiante. Adicionalmente, se utilizarán las placas programables Micro:bit con que cuenta la institución educativa, que permitirán a los estudiantes los procesos de experimentación e implementación práctica de los algoritmos programables que diseñen durante el desarrollo de sus proyectos.

3.4.3. Incorporación de las Actividades en la Programación Curricular y Áreas Implicadas

Las actividades propuestas en este proyecto se integrarán dentro de la programación del Área de Tecnología e Informática para el grado octavo. Dada la naturaleza interdisciplinar del pensamiento computacional y el ABP, se buscarán coordinaciones con otras áreas del currículo, como Matemáticas (para la lógica y resolución de problemas), Lengua Castellana (para la comunicación y documentación), y Ciencias Naturales (para la contextualización de problemas y la exploración de fenómenos).

La implementación de este proyecto implica una adaptación metodológica dentro del Área de Tecnología e Informática, pasando de un enfoque más tradicional a uno centrado en la investigación, la colaboración y la creación de soluciones prácticas por parte de los estudiantes. Se dedicarán las horas de clase del área a las diferentes fases del ABP: identificación del problema, planificación, diseño de algoritmos (incluyendo su representación en diagramas de flujo), implementación en Scratch o Micro:bit, pruebas, presentación y evaluación de los proyectos.

La utilización de las Aulas STEAM como espacios de aprendizaje activo permitirá un uso diferenciado de los recursos tecnológicos y la promoción de un ambiente de experimentación y creatividad. La capacitación previa de los docentes en el uso de estos espacios y en la metodología ABP facilitará la implementación efectiva del proyecto.

3.5.Cronograma Del Proyecto De Innovación

La temporalización general de la presente propuesta de innovación se ha diseñado para su implementación a lo largo de doce (12) semanas lectivas, correspondientes a un semestre académico dentro del Área de Tecnología e Informática. Esta distribución temporal busca asegurar que los estudiantes dispongan del tiempo necesario para la comprensión teórica, la práctica guiada y el desarrollo de los proyectos propuestos.

Cada sesión semanal se estructurará en dos momentos: una hora (1) dedicada a la fundamentación teórica y la explicación de los conceptos por parte del docente, incluyendo ejercicios de práctica y modelación, y una hora (1) de aplicación práctica por parte de los estudiantes, donde desarrollarán los productos digitales específicos para cada actividad.

La siguiente tabla presenta el cronograma de las actividades principales a desarrollar a lo largo del semestre:

Tabla 1 *Cronograma de Realización de Actividades*

Actividad a Desarrollar	Numero de Sesiones Utilizadas											
	Sesión 1	Sesión 2	Sesión 3	Sesión 4	Sesión 5	Sesión 6	Sesión 7	Sesión 8	Sesión 9	Sesión 10	Sesión 11	Sesión 12
Actividad 1: Estrategias Desconectadas - Fundamentos de la Lógica Programable	X											
Actividad 2: Estructurando el Pensamiento - De la Lógica al Pseudocódigo Usando PSeInt		X	X									
Actividad 3: Codificación Visual - Programación por Bloques con Scratch				X	X							
Actividad 4: Sensores y Actuadores - Programando Soluciones para el Mundo Real mediante Micro:bit						X	X	X				
Actividad 5: Desafío de Innovación - Soluciones Tecnológicas para el Entorno Escolar									X	X	X	X

La temporización de los productos digitales propuesta responde a una progresión en el aprendizaje cuidadosamente estructurada, que permite al estudiante transitar de manera gradual desde la resolución de algoritmos lógicos básicos hasta la construcción de sistemas físicos complejos. Esta planificación garantiza la sincronización de la Bitácora Digital como un eje transversal y procesual, donde cada entrega parcial actúa como un hito de evidencia que documenta la evolución del pensamiento computacional del alumnado. Asimismo, esta distribución asegura una alineación con el modelo TPACK, al otorgar los tiempos necesarios para que los estudiantes asimilen el conocimiento tecnológico (TK) de cada herramienta (PSeInt, Scratch y Micro:bit) de forma integrada con el contenido lógico (CK) y la estrategia pedagógica de proyecto (PK), favoreciendo una apropiación tecnológica significativa

Tabla 2 *Temporización de Elaboración de Productos Digitales*

Actividad	Producto Digital / Entregable	Herramienta Principal	Cronograma (Semanas)
Actividad 1: Estrategias Desconectadas - Fundamentos de la Lógica Programable	Registro de algoritmos físicos y guía de rutas lógicas (PDF).	Cámara / Editor de texto	Semana 1
Actividad 2: Estructurando el Pensamiento - De la Lógica al Pseudocódigo Usando PSeInt	Set de archivos de pseudocódigo (.psc) y diagramas de flujo exportados.	PSeInt	Semanas 2 y 3
Actividad 3: Codificación Visual - Programación por Bloques con Scratch	Proyecto de narrativa interactiva o minijuego publicado (URL).	Scratch Online	Semanas 4 y 5
Actividad 4: Sensores y Actuadores - Programando Soluciones para el Mundo Real mediante Micro:bit	Código fuente (.hex) y video demostrativo del prototipo físico.	MakeCode / Micro:bit	Semanas 6, 7 y 8
Actividad 5: Desafío de Innovación - Soluciones Tecnológicas para el Entorno Escolar	Producto Final: Bitácora Digital integrada (Google Sites), código del proyecto y presentación multimedia.	Ecosistema Scratch / Micro:bit	Semanas 9, 10, 11 y 12

Para asegurar el seguimiento del progreso de los estudiantes, se han integrado momentos clave para la evaluación formativa y sumativa a lo largo del cronograma. La siguiente tabla detalla estos momentos, especificando la actividad evaluada, el tipo de evaluación y la sesión correspondiente, así como ejemplos de los instrumentos que se podrían utilizar

Tabla 3 *Cronograma de Evaluación*

Actividad / Fase	Semana	Tipo de Evaluación	Instrumento de Evaluación	Producto / Evidencia Evaluada
Fase Inicial	1	**Diagnóstica**	Cuestionario de entrada / Observación directa	Saberes previos sobre lógica y algoritmos.
Actividad 1: Estrategias Desconectadas	1	**Formativa**	Registro de observación y rúbrica de participación	Desempeño en retos físicos y guía de rutas lógicas.
Actividad 2: PSeInt (Pseudocódigo)	2 - 3	**Formativa**	Lista de Chequeo: Heteroevaluación (Docente)	Archivos .psc y coherencia en diagramas de flujo.
Actividad 3: Scratch (Eventos)	4 - 5	**Sumativa / Formativa**	Lista de Chequeo: Heteroevaluación y Coevaluación	Proyecto de narrativa interactiva publicado (URL).
Actividad 4: Micro:bit (Hardware)	6 - 8	**Formativa / Sumativa**	Lista de Chequeo: Heteroevaluación y Coevaluación	Prototipo físico funcionando y video de evidencia.
Actividad 5: Proyecto Final (ABP)	9 - 11	**Formativa (Procesual)**	Bitácora Digital y Tutorías de seguimiento	Avances semanales, depuración de código y diseño.
Cierre: Feria de Innovación	12	**Sumativa (Final)**	Lista de Chequeo: Heteroevaluación (Sustentación)	Producto Final integrado y exposición técnica.
Evaluación Transversal	1 - 12	**Metacognitiva**	Diana de Evaluación / Autoevaluación	Trabajo colaborativo, actitud y resolución de problemas.

3.6. Actividades Del Proyecto De Innovación

La secuencia didáctica de este proyecto de innovación se fundamenta en el marco **TPACK** (Mishra & Koehler, 2006), el cual postula que la integración de las TIC debe ser una intersección equilibrada entre el conocimiento disciplinar, pedagógico y tecnológico. Bajo este enfoque, las actividades articulan sinérgicamente el **Pensamiento Computacional** (Contenido), el **Aprendizaje Basado en Proyectos** (Pedagogía) y herramientas como **PSeInt, Scratch y Micro:bit** (Tecnología). De esta manera, los recursos digitales dejan de ser un agregado superficial para convertirse en mediadores pedagógicos esenciales, permitiendo que los estudiantes de grado octavo del Instituto Montenegro transformen sus ideas en soluciones digitales funcionales y significativas para su contexto.

3.6.1. Actividad 1: Estrategias Desconectadas - Fundamentos de la Lógica Programable

- **Duración de la actividad:** 1 sesión de 2 horas
- **Objetivos específicos de la actividad:**
    - Identificar y estructurar los pasos finitos de un algoritmo para resolver problemas de navegación básica.
    - Ejecutar secuencias de instrucciones de forma física para comprender la relación entre el código (instrucción) y la acción (resultado).
    - Desarrollar habilidades de pensamiento algorítmico y resolución de problemas mediante el uso de lógica espacial y secuenciación, sin el uso de dispositivos digitales.
    - Fomentar el análisis crítico y la evaluación de soluciones secuenciales propuestas por pares.
- **Contenidos de la actividad:**
    - Abstracción y Descomposición: Identificación de las partes esenciales de un problema.
    - Algoritmos y Secuenciación: Creación de órdenes lógicas y cronológicas con el fin de elaborar algoritmos secuenciales orientados a la solución de problemas prácticos.
    - Depuración (Debugging): Localización y resolución de fallos en la secuencia lógica de un algoritmo secuencial.
- **Descripción de la actividad:**

Esta actividad se diseña como un puente cognitivo entre el pensamiento humano y el lenguaje de las máquinas, permitiendo un primer acercamiento al pensamiento computacional. La intención es que el estudiante de grado octavo experimente en que consiste la lógica programable antes de enfrentarse a una herramienta de programación como tal, ya sea un entorno de codificación formal y/o una tecnología no code; y que mediante esto comprenda que la programación no es solo "escribir código", sino estructurar el pensamiento para solucionar problemas prácticos. Para ello, se emplea el modelo TPACK, garantizando que la tecnología potencie la pedagogía activa y el contenido disciplinar:

Tabla 4 *Integración del Modelo TPACK en la Actividad 1*

Componente	Descripción de la Integración
Conocimiento del Contenido (CK)	Se centra en la **lógica algorítmica**. El estudiante debe aprender a traducir una intención ("quiero llegar allá") en una serie de comandos específicos y limitados. Se abordan conceptos de geometría (ejes cartesianos), lógica secuencial y optimización de rutas.
Conocimiento Pedagógico (PK)	Se aplica el **Aprendizaje Basado en Proyectos (ABP)** en su fase inicial de investigación-acción y la metodología de **Andamiaje**, permitiendo que el alumno se familiarice con la lógica algorítmica antes de enfrentarse a la complejidad de una interfaz de software.
Conocimiento Tecnológico (TK)	Aunque es una actividad desconectada, el diseño del recurso actúa como una **Tecnología Educativa Analógica** que modela el funcionamiento de un sistema computacional.
Integración (TPACK)	El estudiante no solo aprende lógica secuencial, sino que la experimenta a través de una metodología activa que transforma un tablero físico en un sistema de procesamiento de información. La integración TPACK se manifiesta cuando la limitación técnica del recurso (Tarjetas de Instrucciones) obliga al estudiante a desarrollar **capacidades cognitivas de abstracción y descomposición de manera estructurada**.

- **Desarrollo detallado de la secuencia didáctica:**
    - Momento de Apertura - Análisis y Abstracción (20 min):

En este momento inicial, el docente le presentara a los estudiantes la actividad desconectada los estudiantes el juego *Reto Serpientes y Tesoros* (ver Anexo 2, sección a), el cual es un recurso diseñado para introducir los conceptos fundamentales del Pensamiento Computacional (PC) de manera analógica.

La actividad utiliza un tablero físico de 6x6 celdas organizado bajo un sistema de coordenadas cartesianas. Los estudiantes deben guiar un avatar que representa a uno de sus compañeros (Robot) desde un punto de inicio hasta diversos objetivos (tesoros), evitando obstáculos (serpientes) y siguiendo una sintaxis de comandos limitada.

El docente presenta el *Tablero De Juego* y el *Recuadro de Instrucciones*. Se explica que el lenguaje de programación del reto consta de comandos básicos de movimiento: *Avanzar, Girar Derecha, Girar Izquierda y Tomar*. Se enfatiza que, al ser una actividad desconectada, la "depuración" debe realizarse de manera mental antes de plasmar el algoritmo en papel.

- Momento de Desarrollo - Programación Humana y Ejecución (70 min):

Fase de Codificación: Cada grupo de trabajo recibe un set de "Tarjetas de Instrucciones" (*Inicio, Avanzar, Girar Derecha, Girar Izquierda, Tomar y Fin*) y debe dividirse en tres roles (Programador, Robot, Analista de Errores). Utilizando el tablero visto en la fase anterior, los grupos de trabajo deben analizar cuales son las rutas optimas para moverse en el tablero. Deben codificar su solución utilizando la simbología definida y, posteriormente, formalizar el proceso mediante un diagrama de flujo que represente la lógica de toma de decisiones para esquivar los obstáculos.

Fase de Ejecución (El Robot Humano): Los grupos se trasladan a una cuadrícula física marcada en el suelo del aula. Un estudiante asume el rol de "Robot" y solo puede moverse si su equipo le entrega la tarjeta correcta. El "Programador" lee la secuencia y el "Robot" la ejecuta.

Fase de Depuración: Si el "Robot" choca con un obstáculo físico (previamente colocado según el diseño del *Tablero de Juego*), el grupo debe detenerse, el "Analista de Errores" debe identificar en qué paso falló la lógica y corregir la secuencia en su bitácora proponiendo una nueva solución.

- Momento de Cierre - Validación Digital y Reflexión (30 min):

Cada equipo realiza la validación de su código simbólico con el fin de verificar si el algoritmo desarrollado es funcional. La actividad concluye con una plenaria donde se reflexiona sobre una pregunta orientadora: *¿Por qué una computadora no puede "adivinar" lo que queremos si le damos una instrucción ambigua?* Se concluye con una reflexión sobre cómo la planificación paso a paso en el tablero es la base para la programación que realizarán en entornos digitales como PSeInt y Scratch.

- **Recursos:**
  - Tecnológicos: Computador docente, Videobeam, Tablero De Juego y Recuadro de Instrucciones *Reto Serpientes y Tesoros,* Tarjetas de Instrucciones.
  - Físicos: Cinta de enmascarar para la cuadrícula en el piso, tarjetas impresas de comandos, obstáculos de plástico (conos) y la Bitácora de Proyecto de cada equipo.
  - Personales: Docente del Área de Tecnología e Informática.

- **Evaluación de la actividad:**

Se realizará una evaluación de carácter formativo y cualitativo:

Heteroevaluación: El docente utilizará una lista de chequeo (Tabla 11) para verificar si la secuencia lógica final es correcta, si el equipo logró identificar errores de forma autónoma y el uso correcto del lenguaje técnico (algoritmo, instrucción, depuración) durante el debate.

Autoevaluación: A través de la Diana de Evaluación – Trabajo Colaborativo (Figura 12), los estudiantes valorarán su participación en la resolución de problemas y actitud frente al error.

Además, se realizará una retroalimentación mediante cuestionarios anónimos buscando la opinión de los estudiantes sobre la metodología, los recursos, los tiempos y la relevancia de la propuesta.

3.6.2. Actividad 2: Estructurando el Pensamiento - De la Lógica al Pseudocódigo Usando PSeInt

- **Duración de la actividad:** 2 sesiones de 2 horas cada una.
- **Objetivos específicos de la actividad:**
  - Formalizar algoritmos diseñados en lenguaje natural mediante el uso de pseudocódigo y diagramas de flujo.
  - Manipular estructuras secuenciales y asignación de variables en el entorno de desarrollo PSeInt.
  - Validar la lógica de programación mediante la ejecución paso a paso y la detección de errores de sintaxis.
  - Desarrollar la capacidad de abstracción al representar procesos mediante simbología técnica estandarizada.
- **Contenidos de la actividad:**
  - Representación Formal: Simbología de diagramas de flujo y reglas de escritura en pseudocódigo.
  - Entorno de Desarrollo (IDE): Interfaz de PSeInt, consola de salida y editor de diagramas.
  - Lógica de Programación: Variables, constantes y estructuras secuenciales (Entrada - Proceso - Salida).

- **Descripción de la actividad:**

La presente actividad se constituye como un eslabón muy importante en el desarrollo del pensamiento computacional en los estudiantes, ya que consolida los conceptos lógicos aprendidos de forma manual en la actividad anterior y los traduce a una representación técnica y estructurada necesaria en el diseño de algoritmos formales. Su importancia radica en dotar al estudiante de las herramientas cognitivas necesarias para traducir procesos mentales abstractos en estructuras algorítmicas rigurosas y estandarizadas. Al emplear el pseudocódigo y la diagramación, los estudiantes no solo organizan sus ideas, sino que comienzan a interiorizar las reglas de sintaxis, el orden jerárquico y la lógica indispensables para la resolución de problemas mediante sistemas computacionales. Bajo el modelo TPACK, esta fase asegura que la tecnología sea el mediador que permite visualizar y depurar la propia estructura de pensamiento, como se detalla en la siguiente matriz de integración:

Tabla 5 *Integración del Modelo TPACK en la Actividad 2*

Componente	Descripción de la Integración
Conocimiento del Contenido (CK)	Se centra en la **sintaxis lógica formal**. El estudiante debe comprender que el orden de las instrucciones y las variables usadas determinan el éxito del algoritmo desarrollado.
Conocimiento Pedagógico (PK)	Se implementa la estrategia de **Andamiaje Cognitivo** y la progresión **"Usa-Modifica-Crea"**. Se facilita la transición del pensamiento concreto al abstracto mediante la visualización dual (texto y gráfico).
Conocimiento Tecnológico (TK)	El recurso central es el software **PSeInt**. Este mediador tecnológico permite la simulación inmediata y la traducción automática entre pseudocódigo y diagramas de flujo, facilitando la autocrítica del código.
Integración (TPACK)	La tecnología (PSeInt) actúa como el **espejo de la lógica**. Permite que la pedagogía ("Usa-Modifica-Crea") haga visible el contenido (algoritmos), ofreciendo un entorno donde el estudiante experimenta el rigor técnico sin la complejidad de un lenguaje de programación comercial.

Para el desarrollo de esta actividad se propone el uso del software PSeInt, ya que esta herramienta cumple como un punto intermediado entre la dinámica de las actividades desconectadas (Actividad 1) y la programación visual por bloques en Scratch (Actividad 2). Este entorno permite que el estudiante traslade la lógica desarrollada mediante el uso de

instrucciones simples en la actividad anterior hacia una estructura de algoritmo en diagrama de flujo que a su vez se encuentra vinculado con un pseudocódigo formal. De esta manera, el uso de la herramienta PSeInt actúa como el puente necesario para la creación de algoritmos formales, asegurando que los estudiantes comprendan la rigurosidad sintáctica y el uso de las secuencias de control antes de enfrentarse a la abstracción propia de los entornos de programación en bloque.

- **Desarrollo detallado de la secuencia didáctica:**

Sesión 1: Exploración y Modelado (120 min)

- Momento de Apertura (30 min):

El docente realiza un breve repaso de la Actividad 1. A través del videobeam, presenta la interfaz de PSeInt y demuestra cómo un algoritmo de "preparar un café" (visto anteriormente) se traduce a pseudocódigo. Se hace énfasis en que la computadora requiere precisión absoluta: "Si no definimos la variable, el programa no sabe qué procesar".

- Momento de Desarrollo - Fases "Usa" y "Modifica" (70 min):

Fase "Usa": Los estudiantes ejecutan un programa predeterminado (ej. un sumador simple). Observan la ejecución paso a paso, viendo cómo el cursor se mueve por las diferentes instrucciones y/o por los bloques del diagrama de flujo.

Fase "Modifica": Se les plantea el reto de alterar el código para que, en lugar de sumar, realice el promedio de tres notas escolares. Deben identificar que necesitan crear nuevas variables y modificar la fórmula matemática. La tecnología les brinda retroalimentación inmediata si olvidan un punto y coma o una comilla.

Después que los grupos de trabajo creen sus propios algoritmos como posible solución al reto, el docente presentará como modelo el Algoritmo *Calculo_Promedio_Instituto_Montenegro* (ver Anexo 2, sección b), permitiendo a los estudiantes observar la ejecución del código y la generación automática del diagrama de flujo, facilitando así la comprensión de las estructuras condicionales.

- Momento de Cierre (20 min):

Reflexión sobre la importancia de la representación gráfica. Los estudiantes activan el botón de "Dibujar diagrama de flujo" en PSeInt y comparan la estructura visual con su código escrito.

Sesión 2: Creación y Validación (120 min)

- Momento de Apertura (20 min):

El docente plantea un problema de contexto real para el Instituto Montenegro (ej. un sistema para calcular el descuento en la cafetería según la edad del estudiante). Se discuten las entradas necesarias (precio, edad) y la salida esperada.

- Momento de Desarrollo - Fase "Crea" (80 min):

Los estudiantes, en grupos de trabajo, diseñan el algoritmo desde cero en PSeInt. Deben: 1) Escribir el pseudocódigo, 2) Generar el diagrama de flujo y 3) Realizar la "prueba de escritorio" usando la función de ejecución paso a paso del software para asegurar que no haya errores.

- Momento de Cierre (20 min):

Los equipos exportan su diagrama de flujo como imagen y lo cargan en su Bitácora Digital (Google Sites/Docs), añadiendo una breve descripción de los problemas que enfrentaron y cómo los resolvieron (Metacognición).

- **Recursos:**
  - Tecnológicos: Sala de informática, software PSeInt (instalado en local), videobeam, Bitácora Digital de Proyecto.
  - Físicos: Guía rápida de sintaxis PSeInt (ficha de comandos/bloques), cuaderno de bocetos para diagramas manuales.
  - Personales: Docente del Área de Tecnología e Informática.
- **Evaluación de la actividad:**

Se aplicará una evaluación de desempeño y producto:

Heteroevaluación: El docente utilizará una lista de chequeo (Tabla 12) para verificar la correcta declaración de variables y la coherencia lógica del diagrama de flujo final.

Autoevaluación: A través de la Diana de Evaluación – Trabajo Colaborativo (Figura 12), los estudiantes valorarán su participación en la resolución de problemas y actitud frente al error.

Coevaluación: los grupos de trabajo intercambian sus programas y deben intentar "romper" el código de sus compañeros introduciendo datos inesperados, fomentando la cultura de la depuración. Para esto se usará una lista de chequeo (Tabla 13) con la que los estudiantes

puedan evaluar el trabajo de sus compañeros de manera estructurada y enfocada en los criterios clave.

Portafolio: Registro de la evidencia gráfica en la Bitácora Digital, evaluada según la rúbrica de productos digitales.

Además, se realizará una retroalimentación mediante cuestionarios anónimos buscando la opinión de los estudiantes sobre la metodología, los recursos, los tiempos y la relevancia de la propuesta.

3.6.3. Actividad 3: Codificación Visual - Programación por Bloques con Scratch

- **Duración de la actividad:** 2 sesiones de 2 horas cada una.
- **Objetivos específicos de la actividad:**
    - Aplicar conceptos de pensamiento computacional (eventos, bucles y condicionales) en un entorno de programación visual.
    - Diseñar narrativas interactivas mediante la coordinación de múltiples objetos y scripts paralelos.
    - Desarrollar la creatividad digital al integrar elementos multimedia (sonido, imagen y movimiento) con lógica de programación.
    - Fortalecer la capacidad de depuración de errores en entornos de ejecución inmediata.
- **Contenidos de la actividad:**
    - Programación por Bloques: Interfaz de Scratch, categorías de bloques y área de scripts.
    - Lógica de Eventos: Inicio por bandera verde, interacción con teclado y mensajes entre objetos.
    - Control de Flujo: Bucles infinitos (por siempre), bucles condicionados y estructuras de decisión.
    - Multimedia Interactiva: Gestión de disfraces, escenarios y efectos de sonido vinculados al código.

- **Descripción de la actividad:**

Esta actividad representa un hito dentro del proyecto, ya que supone una transición desde la lógica puramente estructural (estudiada en PSeInt en la actividad anterior) hacia una lógica aplicada y creativa. Su relevancia reside en demostrar al estudiante que la programación no es solo una sucesión de comandos lineales, sino una poderosa herramienta de expresión capaz de materializar ideas complejas de forma visual y altamente interactiva. En este entorno, el alumno se enfrenta al reto de gestionar la concurrencia, donde diversas secuencias de código se ejecutan en paralelo simultáneamente, exigiendo un nivel superior de abstracción y coordinación lógica.

Bajo el modelo TPACK, se propone a Scratch no solo como un software técnico, sino como una herramienta pedagógica donde convergen el diseño estético y el rigor algorítmico. Esta fase se apoya en el construccionismo, permitiendo que el alumno aprenda haciendo y convierte el error en una oportunidad de depuración. Al finalizar, la posibilidad de compartir estas creaciones en una comunidad global de aprendizaje refuerza la dimensión social del conocimiento y el empoderamiento digital, tal como se desglosa detalladamente a continuación:

Tabla 6 *Integración del Modelo TPACK en la Actividad 3*

Componente	Descripción de la Integración
Conocimiento del Contenido (CK)	Se centra en la **lógica de eventos y paralelismo**. El estudiante debe comprender cómo diferentes hilos de ejecución (scripts) pueden ocurrir simultáneamente y responder a acciones del usuario.
Conocimiento Pedagógico (PK)	Se aplica el **Aprendizaje Basado en Proyectos** bajo la estrategia **"Usa-Modifica-Crea"**. Se fomenta el aprendizaje por descubrimiento y la colaboración mediante la técnica del "Remix" o mejora de proyectos ajenos.
Conocimiento Tecnológico (TK)	El recurso central es el entorno **Scratch**. Esta plataforma permite una retroalimentación visual inmediata (el "escenario") y facilita la comprensión de la sintaxis mediante la metáfora de piezas de rompecabezas.
Integración (TPACK)	La tecnología (Scratch) funciona como el **catalizador de la creatividad lógica**. Integra la pedagogía (construccionismo) con el contenido (bucles y eventos), permitiendo que el error sea visible y corregible de forma lúdica y significativa.

- **Desarrollo detallado de la secuencia didáctica:**

Sesión 1: Exploración del Entorno y Lógica de Eventos (120 min)

- Momento de Apertura (30 min):

El docente presenta Scratch como un entorno donde los diagramas de flujo de la clase anterior se vuelven "piezas físicas" (bloques). Se realiza una demostración de cómo mover un objeto y hacerlo hablar mediante el bloque de "al hacer clic en bandera verde". Se enfatiza la relación: Evento → Acción.

- Momento de Desarrollo - Fases "Usa" y "Modifica" (70 min):

Fase "Usa": Los estudiantes acceden a un proyecto base sencillo (ej. un personaje que camina). Deben analizar cómo están conectados los bloques.

Fase "Modifica": Se les reta a cambiar el fondo, añadir un segundo personaje y lograr que ambos dialoguen usando bloques de "esperar" y "enviar mensaje". Aquí experimentan con el paralelismo: dos códigos ejecutándose al mismo tiempo.

- Momento de Cierre (20 min):

Puesta en ejecución de los proyectos modificados. El docente introduce el concepto de "depuración visual": si el personaje no se detiene, ¿qué bloque de control nos falta?

Sesión 2: Desarrollo de la Narrativa Interactiva (120 min)

- Momento de Apertura (20 min):

Para ejemplificar la lógica de eventos y la sincronización de diálogos, el docente presentará como modelo el recurso *Reto Scratch* (ver Anexo 2, sección c), el cual servirá como guía para que los estudiantes comprendan la lógica de eventos y el uso de mensajes de difusión en la narrativa.

Se plantea el reto principal: crear una historia breve o un minijuego que incluya al menos un bucle, un evento de teclado y un cambio de escenario. Se discute la importancia de planificar la historia antes de programar.

- Momento de Desarrollo - Fase "Crea" (80 min):

Organizados en los grupos de trabajo asignados, los estudiantes desarrollan su proyecto. Deben aplicar el concepto de abstracción para decidir qué disfraces y sonidos son esenciales.

El docente actúa como facilitador, orientando a los equipos que encuentran errores lógicos en sus bucles de control.

- Momento de Cierre (20 min):

Los estudiantes publican su proyecto en el "Estudio de la Clase" de Scratch y añaden el enlace a su Bitácora Digital, acompañándolo de una reflexión sobre cómo lograron que los objetos interactuaran entre sí.

- **Recursos:**
  - Tecnológicos: Computadores con acceso a Scratch (online u offline), conexión a internet, videobeam, Bitácora Digital de Proyecto.
  - Físicos: Guía visual de bloques (categorías de colores), guion gráfico (storyboard) en papel para la fase de diseño.
  - Personales: Docente del Área de Tecnología e Informática.
- **Evaluación de la actividad:**

Se aplicará una evaluación de desempeño y producto:

Heteroevaluación: Uso de una lista de chequeo (Tabla 14) centrada en la eficiencia del código (evitar redundancias) y el uso correcto de bloques de control y eventos.

Autoevaluación: A través de la Diana de Evaluación – Trabajo Colaborativo (Figura 12), los estudiantes valorarán su participación en la resolución de problemas y actitud frente al error.

Coevaluación: Cada grupo de trabajo realizara la "Prueba de Usuario" para el proyecto de otro equipo y deja un comentario positivo y una sugerencia de mejora en Scratch. Para esto se usará una lista de chequeo (Tabla 15) con la que los estudiantes puedan evaluar el trabajo de sus compañeros de manera estructurada y enfocada en los criterios clave.

Portafolio: Calidad de la narrativa y correcto funcionamiento de los scripts registrados en la respectiva Bitácora Digital, evaluada según la rúbrica de productos digitales.

Además, se realizará una retroalimentación mediante cuestionarios anónimos buscando la opinión de los estudiantes sobre la metodología, los recursos, los tiempos y la relevancia de la propuesta.

3.6.4. Actividad 4: Sensores y Actuadores - Programando Soluciones para el Mundo Real mediante Micro:bit

- **Duración de la actividad:** 3 sesiones de 2 horas cada una.
- **Objetivos específicos de la actividad:**
    - Comprender la interacción entre software y hardware mediante el uso de una placa de desarrollo (Micro:bit).
    - Analizar y procesar datos provenientes de sensores (luz, temperatura, movimiento) para la toma de decisiones algorítmicas.
    - Implementar estructuras condicionales basadas en rangos y umbrales de datos físicos externos.
    - Prototipar una solución tecnológica tangible que responda a una necesidad detectada en el entorno escolar.
- **Contenidos de la actividad:**
    - Computación Física: Concepto de entrada procesamiento y salida.
    - Hardware Educativo: Arquitectura básica de la placa de desarrollo Micro:bit
    - Entornos de Programación para Hardware: Uso de Microsoft MakeCode como simulador y cargador de código.
    - Lógica de Datos: Lectura de sensores de temperatura, acelerómetro y brújula.
- **Descripción de la actividad:**

Esta actividad se constituye como la cúspide técnica del proyecto de innovación, pues representa el salto cualitativo de la programación lógica hacia la interacción con el mundo físico. Su importancia radica en que permite al estudiante visualizar el pensamiento computacional no solo como una herramienta de software, sino como el motor que dota de "inteligencia" a los objetos cotidianos. Al programar hardware real, los estudiantes trascienden el código en pantalla para enfrentarse a las variables e imprevistos del entorno físico, lo que exige un rigor superior en los procesos de depuración (*debugging*) y una capacidad de abstracción capaz de modelar fenómenos naturales en datos digitales. Bajo el modelo TPACK, la placa de desarrollo Micro:bit actúa como el puente mediador que une la

pedagogía basada en el diseño de prototipos con la ciencia de los datos físicos, facilitando una comprensión integral de la tecnología como una capa de inteligencia mediadora de la realidad. Esta transición no solo consolida las competencias técnicas, sino que fomenta un pensamiento sistémico donde el estudiante comprende la relación causa-efecto entre el algoritmo lógico y la respuesta física del dispositivo, tal como se desglosa detalladamente a continuación:

Tabla 7 *Integración del Modelo TPACK en la Actividad 4*

Componente	Descripción de la Integración
Conocimiento del Contenido (CK)	Se enfoca en la **computación física y la gestión de entradas/salidas**. El estudiante aprende cómo los sensores traducen fenómenos físicos (luz, calor) en datos procesables por un algoritmo mediante condicionales.
Conocimiento Pedagógico (PK)	Se aplica el **Aprendizaje Basado en Proyectos** con un enfoque en **prototipado rápido**. Se promueve la autonomía y la iteración constante mediante la estrategia "Usa-Modifica-Crea".
Conocimiento Tecnológico (TK)	Los recursos centrales son la placa de desarrollo **Micro:bit** y el editor **MakeCode**. El simulador permite previsualizar el comportamiento físico antes de la descarga del código al hardware real.
Integración (TPACK)	La tecnología (Micro:bit) permite que la pedagogía (ABP) dé sentido al contenido (lógica de sensores). Se logra una integración total donde el estudiante evidencia que la competencia digital tiene un impacto directo en la solución de problemas físicos.

- **Desarrollo detallado de la secuencia didáctica:**

Sesión 1: Arquitectura de la placa de desarrollo Micro:bit (120 min)

- Momento de Apertura (30 min):

Presentación física de la Micro:bit. El docente explica la relación entre los diagramas de flujo (Actividad 2), los bloques (Actividad 3) y cómo estos ahora controlarán electricidad y luz en la la placa de desarrollo Micro:bit y se representarán en matriz LED de la misma.

- Momento de Desarrollo - Fase "Usa" (70 min):

Los estudiantes exploran la interfaz del editor MakeCode. Programaran una secuencia sencilla de iconos y animaciones en la matriz LED que se active al presionar los botones A y B. Realizan una primera descarga del archivo .hex a la tarjeta física.

- Momento de Cierre (20 min):

Se realizará una reflexión sobre la diferencia que existe entre ver el resultado en el simulador del editor MakeCode y verlo implementado en el hardware real de la placa de desarrollo Micro:bit.

Sesión 2: Lógica de Sensores (120 min)

- Momento de Apertura (20 min):

Introducción a los sensores. El docente muestra cómo la placa de desarrollo Micro:bit "siente" el entorno mediante el sensor de luz y el acelerómetro.

- Momento de Desarrollo - Fase "Modifica" (80 min):

Los estudiantes trabajaran sobre un código base de "Alarma de movimiento". Deben modificarlo para que funcione como un "Medidor de luz": si la luz es baja, la placa de desarrollo Micro:bit debe encender un mensaje de "Estudiar con luz". Aquí deben ajustar valores numéricos (umbrales) para que el sensor reaccione correctamente según la iluminación del salón.

- Momento de Cierre (20 min):

Debate sobre cómo los sensores en la vida real (puertas automáticas, luces de calle) usan esta misma lógica de "Si (valor) < (umbral)".

Sesión 3: Prototipado y Solución de Contexto (120 min)

- Momento de Apertura (20 min):

El docente utilizará el simulador de la interfaz del editor MakeCode para presentar como modelo el recurso *Termómetro Digital* (ver Anexo 2, sección d). Este recurso permitirá a los alumnos observar cómo el código reacciona a cambios de temperatura antes de proceder a la fase de prototipado físico.

Planteamiento del reto de creación: "Un asistente tecnológico para el Instituto Montenegro". Los grupos de trabajo eligen un proyecto tecnológico simple que puedan implementar en la la placa de desarrollo Micro:bit como por ejemplo un contador de pasos para educación física, un juego simple como dados o piedra - papel -tijera, o unas luces direccionales que se puedan usar en la bicicleta.

- Momento de Desarrollo - Fase "Crea" (80 min):

Organizados en los grupos de trabajo asignados, los estudiantes diseñan el algoritmo desde cero, lo validan en el simulador del editor MakeCode y lo ejecutan en la placa de desarrollo Micro:bit. Deben integrar al menos dos tipos de entrada (sensor + botón). El docente supervisa la lógica de los condicionales.

- Momento de Cierre (20 min):

Los estudiantes documentan su prototipo mediante una fotografía o clip de video corto que se anexa a su Bitácora Digital, describiendo la lógica aplicada y el problema resuelto.

- **Recursos:**
  - Tecnológicos Computadores con el editor MakeCode, placa de desarrollo Micro:bit (una por grupo de trabajo), cables USB, Bitácora Digital de Proyecto.
  - Físicos: Guía técnica de sensores, materiales para el soporte del prototipo (opcional).
  - Personales: Docente del Área de Tecnología e Informática.
- **Evaluación de la actividad:**

Se aplicará una evaluación de desempeño y producto:

Heteroevaluación: Se aplicará la lista de chequeo (Tabla 16) diseñada para verificar el uso de sensores y la correcta descarga del código al hardware.

Autoevaluación: A través de la Diana de Evaluación – Trabajo Colaborativo (Figura 12), los estudiantes valorarán su participación en la resolución de problemas y actitud frente al error.

Coevaluación: Los equipos intercambiaran sus placas de desarrollo Micro:bits para verificar si el dispositivo de sus compañeros responde correctamente a los estímulos físicos (luz o movimiento). Para esto se usará una lista de chequeo (Tabla 17) con la que los estudiantes puedan evaluar el trabajo de sus compañeros de manera estructurada y enfocada en los criterios clave.

Portafolio: Calidad de la narrativa y correcto funcionamiento de los scripts registrados en la respectiva Bitácora Digital, evaluada según la rúbrica de productos digitales.

Además, se realizará una retroalimentación mediante cuestionarios anónimos buscando la opinión de los estudiantes sobre la metodología, los recursos, los tiempos y la relevancia de la propuesta.

3.6.5. Actividad 5: Desafío de Innovación - Soluciones Tecnológicas para el Entorno Escolar

- **Duración de la actividad:** 4 sesiones de 2 horas cada una.
- **Objetivos específicos de la actividad:**
  - Integrar los conceptos de lógica algorítmica, programación por bloques y computación física en la creación de un proyecto original.
  - Resolver problemas del contexto escolar mediante el diseño y construcción de prototipos tecnológicos.
  - Documentar de manera sistemática el proceso de ingeniería, desde la ideación hasta la ejecución final, en la Bitácora Digital.
  - Sustentar públicamente la funcionalidad y relevancia de la solución tecnológica desarrollada.
- **Contenidos de la actividad:**
  - Integración de Saberes: Uso simultáneo de lógica, Scratch y/o Micro:bit.
  - Metodología de Proyecto: Fases de ideación, diseño, desarrollo, pruebas y socialización.
  - Comunicación Técnica: Elaboración de presentaciones y demostraciones de prototipos.
- **Descripción de la actividad:**

La presente actividad constituye la síntesis integradora de toda la secuencia didáctica, representando el escenario donde los estudiantes de grado octavo convergen los saberes técnicos y cognitivos construidos en las fases precedentes. En este cierre, se movilizan los fundamentos de lógica algorítmica de las estrategias desconectadas, la capacidad de formalización técnica desarrollada en PSeInt, la creatividad narrativa y lógica de eventos de Scratch, y la interacción con el mundo material a través de la computación física con Micro:bit. Esta convergencia permite que el estudiante trascienda la mera ejecución de comandos para convertirse en un arquitecto de soluciones digitales, capaz de articular el pensamiento sistémico con la resolución de problemáticas de su entorno bajo un enfoque de impacto social.

Bajo la metodología de Aprendizaje Basado en Proyectos (ABP), los estudiantes asumen un rol protagónico como agentes de cambio, enfrentando el desafío de identificar y resolver problemas reales dentro del contexto del Instituto Montenegro. La importancia de esta fase final radica en el fomento de la autonomía y el pensamiento crítico, permitiendo que los estudiantes realicen la transición definitiva del "aprender a programar" al "programar para resolver". Se espera que, al finalizar esta actividad, los alumnos alcancen una competencia integral que les permita diseñar, prototipar y sustentar una solución tecnológica funcional, demostrando un dominio sólido del pensamiento computacional aplicado a la realidad. Este proceso de integración total se articula bajo el modelo TPACK de la siguiente manera:

Tabla 8 *Integración del Modelo TPACK en la Actividad 5*

Componente	Descripción de la Integración
Conocimiento del Contenido (CK)	Se enfoca en la **síntesis de la lógica computacional**. El estudiante aplica de forma transversal variables, bucles, condicionales y gestión de datos físicos para resolver una situación problema.
Conocimiento Pedagógico (PK)	Se basa en el **Aprendizaje Cooperativo y el Design Thinking**. La pedagogía se centra en la autonomía del equipo, donde el docente actúa únicamente como mentor y facilitador del proceso creativo.
Conocimiento Tecnológico (TK)	El alumnado decide el ecosistema tecnológico a emplear (Scratch, Micro:bit o una integración de ambos). La tecnología es ahora una **extensión de su capacidad creativa** para materializar una solución.
Integración (TPACK)	Representa la **convergencia total**. El proyecto final es la evidencia de cómo la tecnología potencia una pedagogía constructivista para dominar un contenido técnico complejo, logrando una innovación educativa real en el aula.

- **Desarrollo detallado de la secuencia didáctica:**

Sesión 1: Ideación y Planificación (120 min)

  - Momento de Apertura (30 min):

El docente presenta el desafío final, el cual involucra una problemática real dentro del contexto del Instituto Montenegro. Se definen las áreas de impacto (medio ambiente, seguridad escolar, salud o convivencia).

Como referencia de un producto final de alta calidad y que sea pertinente dentro del contexto escolar, el docente presentará como modelo el recurso *Reto Reciclaje Instituto Montenegro* (ver Anexo 2, sección e). Este recurso demuestra cómo integrar variables y listas junto con la manipulación de objetos en Scratch para resolver una necesidad como lo es aprender la correcta clasificación de residuos para su reciclaje.

- Momento de Desarrollo - Fase "Lluvia de ideas" (70 min):

Los grupos de trabajo seleccionan un problema y diseñan el Diagrama de Flujo de su solución. Deben decidir qué herramientas usarán (Scratch para software, Micro:bit para hardware).

- Momento de Cierre (20 min):

Los grupos de trabajo realizaran un discurso inicial de presentación del proyecto ("Elevator Pitch") ante sus compañeros de la clase para recibir sugerencias rápidas que les ayuden a abordar la solución del desafío final.

Sesión 2: Desarrollo y Codificación (120 min)

- Momento de Apertura (20 min):

Organización de recursos. Los grupos de trabajo preparan sus estaciones de trabajo y revisan su Bitácora Digital sobre todos los elementos necesarios para implementar la solución, buscando integrar las sugerencias que encuentren más acertadas entre las enunciadas en el momento de cierre de la sesión anterior.

- Momento de Desarrollo - Fase "Crea" (80 min):

Programación intensiva. Los estudiantes traducen sus diagramas de flujo a código (MakeCode o Scratch). Comienza la construcción de la estructura física del prototipo (si aplica).

- Momento de Cierre (20 min):

Registro de avances en la Bitácora Digital, resaltando los obstáculos técnicos encontrados, así como también plantean posibles métodos para superar dichos obstáculos.

Sesión 3: Refinamiento y Pruebas (120 min)

- Momento de Apertura (20 min) - Fase "Control de Calidad":

Se explican los criterios de eficiencia de código y usabilidad para que los estudiantes analicen cómo se pueden aplicar estos conceptos a sus algoritmos. El objetivo es mejorar la

implementación de la solución desarrollada hasta el momento, optimizando la lógica de programación y definir estrategias que permitan superar los obstáculos técnicos encontrados en las fases previas.

- Momento de Desarrollo (80 min):

Pruebas de funcionamiento y depuración (*debugging*). Los equipos deben someter su prototipo o solución digital a condiciones reales. Se finaliza la documentación y se prepara el material visual para la feria.

- Momento de Cierre (20 min):

Validación final por parte del docente para asegurar que el proyecto cumple con los mínimos técnicos requeridos.

Sesión 4: Feria de Innovación y Socialización (120 min)

- Momento de Apertura (20 min):

Montaje del escenario de exposición en el aula o laboratorio.

- Momento de Desarrollo (80 min) – Fase "Showcase":

Cada equipo presenta su proyecto, explica la lógica aplicada y realiza una demostración en vivo. Se aplican los instrumentos de coevaluación entre pares.

- Momento de Cierre (20 min):

Reflexión final dirigida por el docente sobre el camino recorrido desde la primera actividad desconectada hasta el prototipo final.

- **Recursos:**
  - Tecnológicos: Estaciones de trabajo con acceso a internet, plataformas de programación según elección del equipo (Scratch o Microsoft MakeCode), tarjetas Micro:bit con sus respectivos cables de datos, y acceso a la Bitácora Digital de Proyecto (Google Sites o entorno colaborativo similar) para el registro de evidencias.
  - Físicos: Sensores y actuadores externos (si el proyecto lo requiere), materiales de papelería, herramientas básicas de ensamble y elementos reciclables para el diseño del soporte físico de los prototipos (enfoque Maker), guías de diseño

de proyectos, plantillas para el storyboard o diagrama de flujo final, y rúbricas de evaluación compartidas previamente con el alumnado.

- Personales: Docente del Área de Tecnología e Informática.

- **Evaluación de la actividad:**

La evaluación de esta fase final es de carácter integral, sumativa y auténtica, centrada en la capacidad de transferencia de los aprendizajes a situaciones reales. Se organiza a través de tres dimensiones complementarias:

Heteroevaluación (Sustentación y Producto): El docente aplicará una lista de chequeo (Tabla 18) para valorar la funcionalidad técnica del prototipo, el rigor de la lógica programable empleada y la calidad de la sustentación oral durante la Feria de Innovación. Se pondrá especial énfasis en la capacidad del equipo para explicar la resolución de los conflictos técnicos surgidos durante el desarrollo.

Coevaluación (Validación entre Pares): Mediante una lista de chequeo (Tabla 19), cada equipo evaluará el proyecto de sus compañeros desde la perspectiva del usuario final. Este ejercicio permite medir el impacto social y la usabilidad de las soluciones propuestas, fomentando una cultura de retroalimentación constructiva.

Evaluación de Producto (Bitácora Digital): Se realizará una revisión del portafolio digital de cada equipo, el cual debe contener la trazabilidad completa del proyecto: desde el planteamiento del problema y los diagramas lógicos, hasta el código final y el registro audiovisual del funcionamiento del prototipo; para esto se usará la rúbrica de evaluación de productos digitales (Tabla 20).

Además, se realizará una retroalimentación mediante cuestionarios anónimos buscando la opinión de los estudiantes sobre la metodología, los recursos, los tiempos y la relevancia de la propuesta.

3.7. Evaluación de los Procesos de Enseñanza-Aprendizaje

La evaluación en este proyecto de innovación se concibe como un proceso continuo, formativo y auténtico, diseñado para medir no solo la adquisición de conceptos técnicos (Scriven, 1991), sino también el desarrollo de competencias de pensamiento computacional y habilidades del siglo XXI. Se aleja de la evaluación punitiva tradicional para convertirse en una herramienta de andamiaje que guía al estudiante a través de las cinco actividades de la secuencia didáctica.

La estrategia de evaluación adoptada se concibe de manera integral, abarcando tanto el desempeño de los estudiantes como la valoración del propio proyecto de innovación.

3.7.1. Enfoque Evaluativo y Tipología

De acuerdo con el modelo de Aprendizaje Basado en Proyectos (ABP), la evaluación se estructura en tres fases clave:

- **Evaluación Diagnóstica (Semana 1):** Realizada al inicio de la Actividad 1 para identificar los saberes previos de los estudiantes de grado octavo en cuanto a lógica y resolución de problemas.
- **Evaluación Formativa (Semanas 2 a 11):** Es el eje central del proyecto. Se apoya en la retroalimentación constante durante el uso de PSeInt, Scratch y Micro:bit. Aquí, el error es visto como una oportunidad de depuración (debugging) y mejora lógica.
- **Evaluación Sumativa (Semana 12):** Representada en la "Feria de Innovación", donde se valora el prototipo final y la capacidad de los estudiantes para sustentar sus soluciones tecnológicas frente a problemas de su contexto real.

3.7.2. Instrumentos de Evaluación y Evidencias

Para garantizar la objetividad y el rigor académico, se han diseñado instrumentos específicos que capturan el progreso del estudiante en cada componente del marco TPACK:

- **Listas de Chequeo de Heteroevaluación:** Utilizadas por el docente en cada una de las actividades para verificar criterios técnicos como la sintaxis en pseudocódigo, la lógica de eventos en Scratch y la gestión de sensores en Micro:bit.
- **Listas de Chequeo de Coevaluación:** Instrumentos aplicados entre pares en las actividades 3, 4 y 5. Fomentan el pensamiento crítico y la capacidad de análisis de código ajeno, promoviendo la retroalimentación constructiva.

- **Bitácora Digital (Portafolio de Evidencias):** Es el producto transversal del proyecto (Google Sites). En ella, los estudiantes deben cargar progresivamente sus archivos (.psc, .hex, URLs de Scratch) y reflexiones metacognitivas. La Bitácora permite evaluar la trazabilidad del aprendizaje desde la fase desconectada hasta el prototipo final.
- **Diana de Evaluación:** Herramienta visual para la autoevaluación del trabajo colaborativo, permitiendo a los equipos reflexionar sobre su desempeño en roles, manejo del tiempo y resolución de conflictos.

3.7.3. Matriz de Evaluación de Productos Digitales

La evaluación se materializa a través de la entrega de productos digitales específicos que demuestran el dominio de las competencias propuestas. La siguiente matriz (Tabla 9) resume la relación entre la actividad, el producto y el instrumento predominante.

Este sistema de evaluación integral asegura que el cumplimiento de los objetivos de aprendizaje esté alineado con la temporización de 12 semanas, garantizando que cada estudiante reciba la retroalimentación necesaria para transformar su pensamiento intuitivo en competencias digitales avanzadas y significativas.

Tabla 9 *Matriz de Evaluación de Productos Digitales*

Fase / Actividad	Producto Digital	Tipo de Evidencia	Instrumento Clave
I. Lógica	Guía de algoritmos	Documento técnico	Observación y Rúbrica
II. Formalización	Código PSeInt (.psc)	Lógica formal	Lista de Chequeo (Hetero)
III. Creatividad	Proyecto Scratch (URL)	Lógica de eventos	Lista de Chequeo (Co-eval)
IV. Interacción	Código Micro:bit (.hex)	Computación física	Lista de Chequeo (Hetero)
V. Innovación	Bitácora y Prototipo	Solución integrada	Rúbrica de Producto Final

3.8. Medidas de Atención a la Diversidad

La atención a la diversidad es un principio fundamental que guía el diseño de la presente propuesta de innovación, reconociendo las diferentes capacidades, ritmos y estilos de aprendizaje del alumnado (Tomlinson, 2001). Se han integrado estrategias pedagógicas y el

uso de tecnologías educativas para ofrecer un entorno de aprendizaje inclusivo y adaptable a las necesidades individuales.

Adaptaciones Generales con Soporte Tecnológico:

- Flexibilidad en la presentación de la información: Se utilizarán diversos formatos (textual, visual, auditivo) en los materiales y las instrucciones de las actividades (videos explicativos, presentaciones multimedia, infografías), permitiendo a los estudiantes acceder al contenido de la manera que les resulte más accesible (Rose & Meyer, 2002).
- Múltiples formas de participación: Las actividades se diseñarán para fomentar la participación a través de diferentes modalidades (individual, grupal, oral, escrita, digital), permitiendo a los estudiantes demostrar su comprensión y habilidades de diversas maneras (Gardner, 1993). Herramientas como Flipgrid para la expresión oral y escrita, Jamboard para la colaboración visual y Padlet para la recopilación multimedia ofrecen alternativas para la participación.
- Opciones para la acción y la expresión: Se ofrecerán diferentes opciones para que los estudiantes desarrollen y presenten sus productos digitales (documentos de texto, presentaciones, videos, diagramas de flujo en PSeInt, implementaciones en Scratch o Micro:bit), permitiéndoles elegir el medio que mejor se adapte a sus fortalezas e intereses.
- Apoyo visual y estructuración: Se proporcionarán organizadores gráficos, listas de verificación y pautas claras para la realización de las tareas, lo que beneficiará especialmente a estudiantes con dificultades de aprendizaje o que requieren una mayor estructuración.
- Ritmo de aprendizaje flexible: El diseño de las actividades, aunque con una temporalización general en el cronograma, permitirá cierto grado de flexibilidad en el ritmo de trabajo individual, especialmente en las actividades en casa y en la elaboración de los productos digitales. El seguimiento continuo a través de la bitácora permitirá identificar necesidades individuales y ofrecer apoyo específico.

Adaptaciones Específicas Potenciales:

Si bien no se conocen las necesidades específicas del alumnado de antemano, se contemplan las siguientes adaptaciones que podrían implementarse según sea necesario:

- Para estudiantes con dificultades de aprendizaje:
    - Descomposición de tareas: Dividir las actividades complejas en tareas más pequeñas y manejables.
    - Apoyos visuales adicionales: Proporcionar esquemas, mapas conceptuales y otros recursos visuales para facilitar la comprensión.
    - Tiempos extendidos: Ofrecer más tiempo para la realización de las tareas y la entrega de los productos digitales, si es necesario.
    - Tutoría individualizada: Brindar apoyo personalizado y retroalimentación específica para abordar las dificultades individuales.
    - Uso de software de apoyo: Considerar el uso de software de lectura de pantalla, reconocimiento de voz o herramientas de organización, según las necesidades específicas.
- Para estudiantes con altas capacidades:
    - Actividades de ampliación: Proponer tareas que permitan una mayor profundización en los temas y la exploración de conceptos avanzados.
    - Proyectos de investigación autónomos: Ofrecer la posibilidad de desarrollar proyectos individuales con mayor autonomía y complejidad.
    - Roles de liderazgo: Fomentar su participación como líderes en los grupos de trabajo, promoviendo la enseñanza entre pares.

Evaluación Inclusiva:

Las estrategias de evaluación se diseñarán para ser inclusivas, considerando las diferentes formas en que los estudiantes demuestran su aprendizaje. El uso de rúbricas claras y la variedad de instrumentos de evaluación (observación, productos digitales, autoevaluación, coevaluación) buscan ofrecer una visión completa del progreso de cada estudiante, más allá de una única medida (Gipps, 1994). Se realizarán ajustes en los instrumentos y los criterios de evaluación si fuera necesario para asegurar una valoración justa y equitativa.

4. Evaluación Del Proyecto De Innovación

La evaluación del proyecto de innovación es crucial para determinar su viabilidad, anticipar desafíos y proyectar sus beneficios educativos. Esta propuesta, centrada en el desarrollo del pensamiento computacional mediante ABP y herramientas tecnológicas, se considera viable dada su alineación curricular y su potencial para un aprendizaje activo.

Viabilidad y Posibles Dificultades:

La viabilidad se apoya en un diseño pedagógico sólido basado en el uso del Aprendizaje Basado En Proyectos (ABP) y el uso de los recursos tecnológicos con los que cuenta la institución. Sin embargo, podrían surgir desafíos como la disponibilidad y funcionamiento de los recursos tecnológicos, que se mitigarán con familiarización inicial y planes de contingencia. La gestión del tiempo y el ritmo de aprendizaje diverso requerirán flexibilidad y apoyo individualizado, facilitado por la evaluación formativa y la bitácora. Mantener la motivación en los estudiantes ante la complejidad de algunas tareas se incentivará mediante la planeación de actividades progresivas y el reconocimiento de logros.

Prospectiva y Beneficios Futuros (Corto y Mediano Plazo):

A pesar de los desafíos, el proyecto ofrece beneficios significativos:

- **A Corto Plazo**: Desarrollo de habilidades de pensamiento computacional y competencias digitales, mayor compromiso estudiantil y fomento del trabajo colaborativo.
- **A Mediano Plazo**: Potencial mejora del rendimiento académico, mayor autonomía y capacidad de aprendizaje, preparación para futuros estudios STEM y el mundo laboral, y una cultura de innovación tecnológica en el aula.

En conclusión, la evaluación del proyecto subraya su potencial para cultivar habilidades esenciales del siglo XXI. El proyecto requerirá exigirá una planificación cuidadosa y reflexión docente continúa representada en una planificación adaptativa y un seguimiento constante para superar los desafíos y maximizar los beneficios educativos a corto y mediano plazo.

Sin embargo, para ofrecer una visión más completa y crítica, se presenta a continuación una matriz DAFO que sintetiza los principales factores internos y externos que podrían influir en el éxito del proyecto:

Tabla 10 *Matriz DOFA de Evaluación del Proyecto de Innovación*

Factores Internos	**Fortalezas**	**Debilidades**
Entorno y Contexto	- Relevancia del pensamiento computacional en el siglo XXI. - Interés creciente en metodologías activas y el uso de tecnología educativa. - Posibilidad de colaboración con otras áreas y la comunidad educativa. - Disponibilidad de recursos educativos abiertos y plataformas en línea.	- Posibles limitaciones en el acceso equitativo a la tecnología fuera del aula (brecha digital). - Actitudes previas negativas hacia la programación o la tecnología en algunos estudiantes o familias.
Factores Externos	**Oportunidades**	**Amenazas**
Capacidades y Recursos	- Diseño pedagógico ABP y aprendizaje activo. - Integración tecnológica motivadora (PSeInt, Scratch, Micro:bit). - Cronograma detallado y secuenciado. - Evaluación integral con diversos instrumentos. - Fomento del trabajo colaborativo. - Atención a la diversidad y flexibilidad.	- Posible necesidad de formación docente específica. - Dependencia de la infraestructura tecnológica. - Posible resistencia inicial al cambio metodológico. - Gestión de la heterogeneidad del aula puede requerir ajustes significativos.
Entorno y Contexto	- Relevancia del pensamiento computacional en el siglo XXI. - Interés creciente en metodologías activas y el uso de tecnología educativa. - Posibilidad de colaboración con otras áreas y la comunidad educativa. - Disponibilidad de recursos educativos abiertos y plataformas en línea.	- Currículo sobrecargado que dificulta la dedicación del tiempo necesario. - Rapidez del cambio tecnológico que puede requerir actualizaciones constantes y adaptación de materiales.

5. Conclusiones

5.1.Conclusiones

El presente proyecto de innovación surge de la imperante necesidad de fortalecer los procesos de enseñanza-aprendizaje en Colombia, específicamente en lo referente a la incorporación de la programación y el desarrollo del pensamiento computacional en la educación básica secundaria. En este contexto, y aprovechando iniciativas gubernamentales como "Colombia Programa" y las inversiones en infraestructura educativa como las aulas STEAM del Instituto Montenegro, se propuso diseñar una estrategia de enseñanza basada en el Aprendizaje Basado en Proyectos (ABP) para estudiantes de grado octavo. Se concluye que el diseño de esta secuencia didáctica progresiva constituye una respuesta efectiva a dicha necesidad, destacando como valor diferenciador la creación de recursos tecno-pedagógicos de autoría propia (Anexo 2). Estos objetos de aprendizaje, diseñados específicamente para el contexto institucional, demuestran una alta competencia digital docente al transformar conceptos abstractos en herramientas tangibles y contextualizadas.

Este trabajo se fundamenta en la premisa de que la coyuntura actual exige propuestas innovadoras que utilicen proactivamente los recursos tecnológicos disponibles. El objetivo general se centró en impulsar el desarrollo del pensamiento computacional alineándose con los objetivos nacionales y las directrices del Proyecto Educativo Institucional. Bajo esta perspectiva, la propuesta alcanza una sinergia total bajo el modelo TPACK, demostrando que la tecnología no es un fin, sino un mediador. Al articular el conocimiento pedagógico (ABP) con el tecnológico (programación no-code) y el disciplinar (lógica y algoritmia), se logra un equilibrio que potencia el aprendizaje, permitiendo que el estudiante pase de ser un consumidor pasivo a un prosumidor tecnológico capaz de resolver problemas reales.

En relación con los objetivos específicos planteados, se diseñó una propuesta metodológica que integra el ABP como eje central. A través de la solución de situaciones problema, los estudiantes transitan por fases de análisis, diseño de algoritmos, representación mediante diagramas de flujo e implementación mediante la progresión "usa – modifica – crea". Este enfoque no solo fomenta la autonomía y el aprendizaje colaborativo, sino que impulsa el desarrollo de habilidades del siglo XXI. El fortalecimiento del pensamiento computacional

mediante este diseño didáctico promueve el pensamiento crítico, la comunicación asertiva y el pensamiento sistémico, preparando a los alumnos no solo en alfabetización digital, sino en estructuras mentales útiles para cualquier disciplina y para los desafíos de la vida cotidiana.

Una de las principales aportaciones del proyecto de innovacion radica en la articulación de tecnologías no code (Scratch y Micro:bit) con la representación algorítmica formal, proceso en el cual la herramienta PSeInt actúa como el mediador técnico-pedagógico fundamental entre las actividades desconectadas y la programación visual. Esta transición permite que el estudiante formalice la lógica simbólica antes de enfrentarse a la abstracción de los bloques utilizados en Scratch y Micro:bit. La relevancia de este trabajo reside en su potencial para transformar la enseñanza en instituciones públicas con recursos limitados, demostrando que es posible implementar procesos de vanguardia aprovechando la formación docente y las aulas STEAM. Además, se destaca la viabilidad y el rigor técnico de la propuesta; a pesar de no haber sido implementada en la práctica, la elaboración de prototipos funcionales, códigos depurados y guías detalladas (Anexos 1 y 2) asegura que el proyecto cuenta con una ruta de aplicación clara, robusta y lista para ser ejecutada con éxito en la institución.

En definitiva, este proyecto de innovación se presenta como una estrategia pedagógica viable y pertinente para el contexto educativo del Instituto Montenegro. Su enfoque en el ABP y las tecnologías no code ofrece un camino accesible y motivador que contribuye significativamente al proyecto de vida de los estudiantes. Al integrar de manera coherente el rigor técnico de la ingeniería de software con la flexibilidad del diseño instruccional, se logra una propuesta de empoderamiento digital que prepara a los ciudadanos del mañana para ser competentes, creativos y resilientes frente a los desafíos constantes del siglo XXI.

5.2. Limitaciones Y Prospectiva

El presente Trabajo de Fin de Máster (TFM) se centra en el diseño de una propuesta de innovación para el desarrollo del pensamiento computacional en estudiantes de grado octavo, utilizando el Aprendizaje Basado en Proyectos y herramientas de programación no code como Scratch y Micro:bit. Si bien se considera que la propuesta ofrece un marco sólido y pertinente, es importante reconocer sus alcances y limitaciones.

Entre los alcances de esta propuesta se destaca la articulación de una metodología activa como el ABP con herramientas tecnológicas accesibles y motivadoras. Se logra diseñar una secuencia de actividades que progresa desde el análisis de problemas hasta la implementación de soluciones mediante diagramas de flujo y programación por bloques. Además, se contempla una estrategia de evaluación integral y medidas de atención a la diversidad.

Sin embargo, existen limitaciones importantes. El diseño de la propuesta, con la restricción de un tiempo de implementación limitado (establecido en el cronograma general del proyecto), no permite una exploración exhaustiva de todas las potencialidades de las herramientas seleccionadas. En particular, la propuesta no profundiza en la creación de bloques funcionales propios en Micro:bit (análogos a las funciones en programación funcional) ni explora en detalle las capacidades avanzadas de Scratch, como el manejo de escenarios, la animación y la interacción compleja con objetos. Una implementación que abarcara la totalidad del año lectivo permitiría una exploración más profunda y el desarrollo de proyectos con un mayor nivel de complejidad y reto para los estudiantes.

En cuanto a la prospectiva, este TFM abre diversas líneas de trabajo futuro e investigación. Una de ellas sería la transición progresiva desde la programación no code hacia el aprendizaje de lenguajes de programación formales. Este paso requeriría una estrategia de enseñanza diferenciada, posiblemente manteniendo el ABP pero incorporando prácticas pedagógicas distintas a las estrategias desconectadas y la progresión usa – modifica – crea, que fueron centrales en la presente propuesta. El aprendizaje de la codificación requeriría que los estudiantes aprendieran a escribir instrucciones por sí mismos, sin la analogía directa con los bloques funcionales.

En este sentido, la exploración de lenguajes de programación como Ruby o Python podría ser una dirección interesante para futuras propuestas de innovación, dada su sintaxis legible y su aplicabilidad en diversos contextos. Sin embargo, la implementación de una propuesta de esta naturaleza requeriría una planificación aún más extensa, probablemente abarcando también la totalidad del año lectivo, para asegurar una apropiación significativa de los conceptos y la adquisición de habilidades de codificación sólidas.

En resumen, aunque la presente propuesta de innovación ofrece un camino viable para el desarrollo del pensamiento computacional a través del ABP y las herramientas no code, sus limitaciones en cuanto a la profundidad de exploración de las herramientas y la transición a

lenguajes formales abren interesantes líneas de investigación y desarrollo futuro. Estas futuras líneas podrían enfocarse en estrategias pedagógicas específicas para la enseñanza de la codificación y la exploración de lenguajes de programación textuales en el contexto de proyectos significativos para los estudiantes.

Referencias Bibliográficas

Adell Segura, J., Llopis Nebot, M. Á., Esteve Mon, F. M., & Valdeolivas Novella, M. G. (02 de 01 de 2019). El debate sobre el pensamiento computacional en educación. *RIED. Revista Iberoamericana de Educación a Distancia, 22*(1), 171–186. Obtenido de https://www.redalyc.org/articulo.oa?id=331459398009

ANDI. (2021). *Habilidades Digitales en Colombia*. Obtenido de https://www.andi.com.co/uploads/gan_habilidadesdigitales_col_v8.pdf

Baena, G. M. (2014). *Antología de algoritmos computacionales.* Editorial Académica Española.

Barlex, D., & Carr, M. (2007). *Learning through portfolios.* Routledge.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48-54. Obtenido de https://www.researchgate.net/publication/247924673_Bringing_computational_thinking_to_K-12_what_is_Involved_and_what_is_the_role_of_the_computer_science_education_community

Bers, M. U. (2018). *Coding as a playground: Programming and computational thinking in early childhood.* Routledge.

Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., & & Palincsar, A. (1991). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist, 26*(3-4), 369-398. Obtenido de https://www.tandfonline.com/doi/abs/10.1080/00461520.1991.9653139

Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., & Palincsar, A. (21 de 11 de 2011). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist, 26*(3-4), 369-398. doi:10.1080/00461520.1991.9653139

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association*, (págs. 1-25).

Brookfield, S. D. (1995). *Becoming a critically reflective teacher.* Jossey-Bass.

Castells, M. (2000). *La era de la información: economía, sociedad y cultura. Vol. 1: La sociedad red.* (Segunda ed.). Madrid, España: Alianza Editorial, S. A. Obtenido de https://amsafe.org.ar/wp-content/uploads/Castells-LA_SOCIEDAD_RED.pdf

Castro, P., & Garzón, M. (29 de 09 de 2017). Política Educativa Colombiana, una medición de calidad lejos del contexto. Obtenido de https://www.economiacolombiana.co/desarrollo-futuro/brechas-de-calidad-en-la-educacion-un-analisis-a-nivel-nacional-y-territorial-medicion-a-traves-de-las-pruebas-saber-11-3328

Cohen, L., Manion, L., & Morrison, K. (2018). *Research methods in education* (Octava ed.). Routledge.

Díaz-Barriga Arceo, F., & Hernández Rojas, G. (2010). *Estrategias docentes para un aprendizaje significativo: Interpretación constructivista* (Tercera ed.). McGraw-Hill.

Flavell, J. H. (1976). Metacognition and cognitive monitoring: A new area of cognitive–developmental inquiry. *American Psychologist, 34*(10), 906–911.

Florian, L., & Black-Hawkins, K. (2011). Exploring inclusive educational practice. *British Educational Research Journal, 37*(5), 813-828.

Fundación Telefónica. (2019). *Sociedad Digital en España 2018.* Penguin Random House Grupo Editorial. Obtenido de https://publiadmin.fundaciontelefonica.com/index.php/publicaciones/add_descargas?tipo_fichero=pdf&idioma_fichero=es_es&pais=Espa%C3%B1a&title=Sociedad+Digital+en+Espa%C3%B1a+2018&code=655&lang=es&file=Sociedad_Digital_Espana_2018.pdf&_gl=1*ttwbk6*_ga*MTEz

Gardner, H. (1993). *Multiple intelligences: The theory in practice.* Basic Books.

Gipps, C. (1994). *Beyond testing: Towards a theory of educational assessment.* Falmer Press.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38-43. Obtenido de http://multimedia.uoc.edu/carlos/chipro/wp-content/uploads/2013/10/38.full_.pdf

Gurises Unidos. (06 de 2017). *Gurises Unidos.* Obtenido de https://www.gurisesunidos.org.uy/wp-content/uploads/2017/11/PensamientoComputacional.pdf

Gutiérrez León, C. Z. (10 de 05 de 2024). *Diagrama de Flujo 2 [Imagen]*. Obtenido de Scribd: https://es.scribd.com/document/564418906/DIAGRAMA-DE-FLUJO-2

Hmelo-Silver, C. E. (2004). Problem-based learning: What and how do students learn? *Educational Psychology Review, 16*(3), 235-266. doi:https://doi.org/10.1023/B:EDPR.0000034022.16470.f3

I.E "Instituto Montenegro". (24 de 11 de 2020). *Imagen de Perfil "Instituto Montenegro" [Fotografia].* Obtenido de Facebook: https://www.facebook.com/ieins.montenegro3/photos

IESPE (Instituto de Estudios Superiores en Pedagogía y Educación Continua). (2023). *El Aprendizaje Basado en Proyectos como medio para lograr los aprendizajes [Imagen]*. Obtenido de https://www.iespe.mx/post/el-aprendizaje-basado-en-proyectos-como-medio-para-lograr-los-aprendizajes

Invest in Armenia. (12 de 2024). *Datos generales y cifras del Quindío [Imagen].* Obtenido de Invest in Armenia - Montenegro: https://investinarmenia.org/wp-content/uploads/2024/12/img-photos-montenegro-2-quindio-invest-in-armenia.jpg

ISTE (International Society for Technology in Education). (2016). *ISTE Standards for Students*. Obtenido de ISTE Standards for Students: https://www.iste.org/standards/for-students

Johnson, D. W., & Johnson, R. T. (2009). An overview of cooperative learning. In J. Cohen, A. M. Sprague, & J. Bellanca (Eds.). *he book of learning and teaching: Leading edge applications of cognitive science to enhance student learning*, 19-31.

Juegos Robótica. (s.f). *Diagrama de flujo y programa no code equivalente para programación de un Arduino usando Scratch [Imagen].* Obtenido de JuegosRobotica.es: https://juegosrobotica.es/diagrama-de-flujo/

Krajcik, J. S., & Blumenfeld, P. C. (2006). *Project-based learning. In R. K. Sawyer (Ed.).* Cambridge University Press.

l3tcrafteducacion.com. (10 de 01 de 2022). *Scratch, el lenguaje de programación perfecto para empezar [imagen]*. Obtenido de L3T Craft Educación: https://l3tcrafteducacion.com/wp-content/uploads/2021/12/0_356zWDYXZyGbmQxa_-1024x583.png

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation.* Cambridge University Press.

Librería CATEDU. (28 de 10 de 2022). *Ejemplo 2: Diagrama de flujo de un programa que a partir de dos números compara cuál es mayor y lo muestra en pantalla [Imagen].* Obtenido de Fundamentos de programación estructurada con PSeInt y Scratch: https://libros.catedu.es/books/fundamentos-de-programacion-estructurada-con-pseint-y-scratch/page/diagramas-de-flujo

López Belarrinaga, S., & Martínez Pérez, B. (09 de 10 de 2023). *Orientaciones Metodológicas Para El Diseño De Experiencias De Aprendizaje - Aprendizaje Basado en Proyectos (ABP)*. Obtenido de CEDEC - Proyecto EDIA: https://descargas.intef.es/cedec/proyectoedia/guias/contenidos/orientaciones_metodologia/aprendizaje_basado_en_proyectos_abp.html

López García, J. C. (01 de 10 de 2009). Ilustración 1-1: Interpretación dinámica y cíclica de las etapas planteadas por Poyla para resolver problemas. *Algoritmos y Programación (Guía para Docentes).* Obtenido de https://www.calameo.com/read/000170621716489806b76

López García, J. C. (01 de 10 de 2009). Ilustración 1-3: Fases para elaborar un programa a partir de un algoritmo programable. *Algoritmos y Programación (Guía para Docentes).* Obtenido de https://www.calameo.com/read/000170621716489806b76

López García, J. C. (01 de 10 de 2009). Ilustración 1-5: Etapas a desarrollar en la fase de análisis de un problema (Entenderlo). *Algoritmos y Programación (Guía para docentes).* Obtenido de https://www.calameo.com/read/000170621716489806b76

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education (TOCE), 10*(1), 1-15. Obtenido de

https://www.researchgate.net/publication/234820368_The_Scratch_Programming_Language_and_Environment

Melo-Becerra, L. A., Ramos-Forero, J. E., Rodríguez-Arenas, J. L., & Zárate-Solano, H. M. (02 de 11 de 2021). Efecto de la pandemia sobre el sistema educativo: El caso de Colombia. *Borradores de Economía, 1179*. Obtenido de Investigaciones Economicas - Banco de la Republica - Colombia: https://repositorio.banrep.gov.co/handle/20.500.12134/10225

microbit.microlog.es. (03 de 2019). *Micro:bit Programación [Imagen]*. Obtenido de Micro:bit Microlog.es: https://microbit.microlog.es/wp-content/uploads/2019/03/makecode-microbit.png

Ministerio de Educación Nacional (MEN). (08 de 02 de 1994). *Ley General de Educación*. Obtenido de https://www.mineducacion.gov.co/1621/articles-85906_archivo_pdf.pdf

Ministerio de Educacion Nacional (MEN). (05 de 2008). *Guía No. 30 Ser competente en tecnología: ¡una necesidad para el desarrollo!* Obtenido de https://www.mineducacion.gov.co/1780/articles-160915_archivo_pdf.pdf

Ministerio de Educación Nacional (MEN). (2013). *Competencias TIC para el desarrollo profesional Docente.* Obtenido de https://www.mineducacion.gov.co/1621/articles-339097_archivo_pdf_competencias_tic.pdf

Ministerio de Educación Nacional (MEN). (2016). *Lineamientos Curriculares de Tecnologías e Informática.* Bogotá, Colombia.

Ministerio de Educación Nacional (MEN). (19 de 09 de 2022). *Orientaciones curriculares para el área de tecnología e informática en educación básica y media*. Obtenido de https://www.colombiaaprende.edu.co/sites/default/files/files_public/2022-11/Orientaciones_Curricures_Tecnologia.pdf

Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC). (2020). *Implementación de tecnologías digitales para aprender en las sedes Educativas Públicas.* Obtenido de https://www.mintic.gov.co/portal/715/articles-326715_documento__00.pdf

Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC). (14 de 12 de 2023). *Presidente Gustavo Petro lanza 'Colombia Programa', iniciativa que formará a 896.000 estudiantes y 11.200 docentes en pensamiento computacional*. Obtenido de MinTIC - Sala de Prensa: https://www.mintic.gov.co/portal/inicio/Sala-de-prensa/Noticias/333210:Presidente-Gustavo-Petro-lanza-Colombia-Programa-iniciativa-que-formara-a-896-000-estudiantes-y-11-200-docentes-en-pensamiento-computacional

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record, 108*(6), 1017-1054. doi:https://doi.org/10.1111/j.1467-9620.2006.00684.x

Moreno, J. J. (s.f.). *Flokzu*. Recuperado el 15 de 05 de 2024, de Low code vs No Code - La Guia Definitiva: https://flokzu.com/es/bpm-es/low-code-y-no-code-plataformas-de-workflow-y-bpm-dan-aire-a-it/

NRC (National Research Council). (2011). *Learning science through computer games and simulations.* National Academies Press.

Omatech. (16 de 05 de 2022). *Omatech Blog*. Obtenido de ¿Qué es No-Code y Low-Code? Definición y diferencias.: https://www.google.com/search?q=https://www.omatech.com/blog/que-es-no-code-y-low-code-definicion-y-diferencias/

Organización para la Cooperación y el Desarrollo Económicos (OCDE). (2020). *Education at a Glance 2020: OECD Indicators.* OECD Publishing.

Ortega-Ruipérez, B. (28 de 07 de 2020). Figura 2: Representación simplificada de los procesos superiores de pensamiento en la resolución de problemas aplicando el pensamiento computacional. *Pedagogía del Pensamiento Computacional desde la Psicología: un Pensamiento para Resolver Problemas.* Obtenido de https://institucional.us.es/revistas/cuestiones/29_2/misce_2.pdf

Ortega-Ruipérez, B. (28 de 07 de 2020). Pedagogía del Pensamiento Computacional desde la Psicología: un Pensamiento para Resolver Problemas. *Cuestiones Pedagógicas, 2*(29), 130-144. Obtenido de https://institucional.us.es/revistas/cuestiones/29_2/misce_2.pdf

Piaget, J., & Inhelder, B. (1972). *Psicología del niño.* Paris, Francia: Presses Universitaires.

Polanco, N., Ferrer, S., & Fernández, M. (2021). Aproximación a una definición de pensamiento computacional. *Revista Iberoamericana de Educación a Distancia*, 55-76. Recuperado el 05 de 06 de 2024, de https://revistas.uned.es/index.php/ried/article/view/27419/22029

Polya, G. (1957). *How to solve it: A new aspect of mathematical method* (Segunda ed.). Princeton University Press.

Popham, W. J. (2018). *Classroom assessment: What teachers need to know* (Novena ed.). Pearson.

ProgramacionColVia. (s.f.). *Diagramas De Flujo, Pseudocódigo Y Pruebas De Escritorio*. Recuperado el 05 de 06 de 2024, de https://programacioncolvia.weebly.com/diagramas-de-flujo-pseudocodigo-y-pruevas-de-escritorio.html

PTesParaTalentos. (s.f). *PT es Para Talentos - PRINCIPIOS BÁSICOS PARA EL ABP (APRENDIZAJE BASADO EN PROYECTOS)*. Obtenido de https://ptesparatalentos.wordpress.com/2015/10/30/orientaciones-para-el-abp-aprendizaje-basado-en-proyectos/

Rojas-Ospina, T., Ochoa-Angrino, S., & Tovar-Aguirre, A. (2023). Formación en promoción de autonomía y apoyo pedagógico durante confinamiento por covid-19. *Revista Latinoamericana de Ciencias Sociales, Niñez y Juventud,, 21*(2), 226-259. Obtenido de https://www.redalyc.org/journal/773/77376320010/html/

Román-González, M., Moreno-León, J., & Robles, S. (2017). Computational thinking through robotics in primary education. *Education and Information Technologies, 22*(4), 2053-2077.

Rose, D. H., & Meyer, A. (2002). *Teaching every student in the digital age: Universal design for learning.* Association for Supervision and Curriculum Development.

Sánchez, J. (2012). La enseñanza de la informática en la educación básica y media en Colombia: un análisis de las políticas y las prácticas pedagógicas. *Revista Educación y Pedagogía, 24*(63), 111-126.

Savery, J. R. (2015). Overview of problem-based learning: Definitions and distinctions. *Interdisciplinary Journal of Problem-Based Learning, 9*(1), 3-9.

Schunk, D. H. (1997). *Learning theories: An educational perspective* (Segunda edición ed.). Mexico: Prentice Hall.

Scriven, M. (1991). *Evaluation Thesaurus.* Sage Publications.

SEDQuindio. (01 de 2022). *Noticias 2022 - A través de la secretaría TIC, la Gobernación hará entrega de 630 portátiles para 63 aulas Steam.* Obtenido de https://www.quindio.gov.co/noticias-2022/noticias-enero-2022/a-traves-de-la-secretaria-tic-la-gobernacion-hara-entrega-de-630-portatiles-para-63-aulas-steam

Sentance, S., Sinclair, J., & Bödeker, H. (2017). Raspberry Pi and micro:bit: A quick start guide for teachers. *Computing at School (CAS)*.

Tekman Education. (03 de 10 de 2021). *Modelos pedagógicos: Qué son y cuáles son los fundamentales en educación*. Obtenido de https://www.tekmaneducation.com/: https://www.tekmaneducation.com/modelos-pedagogicos-en-educacion/

Thomas, J. W. (3 de 2000). *A review of research on project-based learning.* Obtenido de The Autodesk Foundation: http://www.autodesk.com/foundation

Tomlinson, C. A. (2001). *How to differentiate instruction in mixed-ability classrooms.* (Segunda ed.). Association for Supervision and Curriculum Development.

Vargas, N. (07 de 06 de 2022). *Wikindio - Instituto Montenegro [Fotografía].* Obtenido de https://wikindio.com/instituto-montenegro/

Vieira, C., Gómez, M., Canu, M., & Duque, M. (06 de 2019). *Pensamiento Computacional.* Obtenido de STEM-Academia (Pequeños Científicos): https://www.pequenoscientificos.org/uploads/7/6/6/4/76644211/pensamiento_computacional.pdf

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science teachers. *Journal of Science Education and Technology, 25*(1), 127-147.

Wiggins, G. P. (1998). *Educative assessment: Designing assessments to inform and improve student performance.* Jossey-Bass Publishers.

Wikipedia. (s.f). *[Mapa - Localización de Montenegro en el Quindío].* Obtenido de Wikipedia - Montenegro (Quindío): https://es.wikipedia.org/wiki/Montenegro_(Quind%C3%ADo)

Wing, J. M. (01 de 03 de 2006). Computational Thinking. *Communications of the ACM, 49*(3), 33-35. doi:10.1145/1118178.1118215

Zapata-Ros, M. (15 de 09 de 2015). Pensamiento computacional: Una nueva alfabetización digital. *RED-Revista de Educación a Distancia, 46*(4). Obtenido de http://www.um.es/ead/red/46/zapata.pdf

ANEXOS

Anexo 1. Ejemplos de Instrumentos de Evaluación

a. Lista de Chequeo Heteroevaluación - Actividad 1

Este instrumento permitirá al docente valorar de manera objetiva el desempeño de los equipos durante la fase de estrategias desconectadas, centrando la atención en los hitos del pensamiento computacional y el trabajo colaborativo.

Tabla 11 *Lista de Chequeo Heteroevaluación - Actividad 1*

Proyecto: Desarrollo de Pensamiento Computacional **Actividad: Estrategias Desconectadas - Fundamentos de la Lógica Programable**					
Equipo: _____		**Fecha: _____**			
#	**Criterio a Evaluar**	**Sí**	**No**	**Parcial**	**Observaciones / Evidencias**
1	**Lógica y Secuenciación:** ¿El algoritmo propuesto resuelve el reto planteado en el recurso de Genially siguiendo una secuencia lógica?				
2	**Abstracción:** ¿El equipo identifica correctamente los obstáculos y puntos de recolección antes de iniciar la programación física?				
3	**Depuración (Debugging):** ¿Identifican y corrigen errores en la secuencia de instrucciones de forma autónoma durante la ejecución física?				
4	**Uso de Lenguaje Técnico:** ¿Utilizan términos como "algoritmo", "instrucción" o "secuencia" adecuadamente durante el desarrollo de la actividad?				
5	**Cumplimiento de Roles:** ¿Se evidencia respeto por las funciones asignadas (Programador, Robot, Analista de Errores)?				
6	**Validación Tecnológica:** ¿Logran trasladar con éxito la solución física al mediador digital para su verificación final?				

b. Lista de Chequeo Heteroevaluación - Actividad 2

Este instrumento permite al docente verificar no solo el resultado final del algoritmo, sino el proceso técnico de traducción y simulación, asegurando que el estudiante ha comprendido las reglas de formalización que exige el entorno PSeInt.

Tabla 12 *Lista de Chequeo Heteroevaluación - Actividad 2*

Proyecto: Desarrollo de Pensamiento Computacional **Actividad: Estructurando el Pensamiento - De la Lógica al Pseudocódigo Usando PSeInt**					
Equipo: _____		**Fecha: _____**			
#	**Criterio a Evaluar**	**Sí**	**No**	**Parcial**	**Observaciones / Evidencias**
1	**Sintaxis y Estructura:** ¿El pseudocódigo presenta una estructura formal correcta (palabras clave *Inicio*, *Definir*, *Escribir*, *Leer*, *Fin*)?				
2	**Manejo de Variables:** ¿Se definen y asignan correctamente los tipos de datos (numéricos, caracteres) necesarios para la solución del problema?				
3	**Coherencia Gráfica:** ¿Existe una relación lógica y exacta entre el pseudocódigo escrito y el diagrama de flujo generado en PSeInt?				
4	**Simulación y Depuración:** ¿Utiliza la función de "ejecución paso a paso" para verificar el flujo de la información e identificar posibles cuellos de botella?				
5	**Resolución del Reto:** ¿El algoritmo final resuelve de manera efectiva el problema planteado (ej. conversor de unidades o cálculo de promedios)?				
6	**Documentación Digital:** ¿Se evidencia la carga de capturas de pantalla y reflexiones del proceso en la Bitácora Digital (Google Sites/Docs)?				

c. Lista de Chequeo Coevaluación - Actividad 2

Esta lista de chequeo está diseñada para que los estudiantes puedan evaluar el trabajo de sus compañeros de manera estructurada y enfocada en los criterios clave.

Instrucciones para los estudiantes:

1. Lean cuidadosamente el algoritmo y el diagrama de flujo del grupo que van a evaluar.
2. Para cada criterio, marquen "Sí" si consideran que se cumple y "No" si no se cumple o si hay aspectos a mejorar.
3. En la columna de "Comentarios / Sugerencias", proporcionen una breve explicación de su elección y ofrezcan comentarios o sugerencias constructivas que permitan mejorar el trabajo del otro grupo.
4. Sean respetuosos y enfocados en ofrecer una retroalimentación que ayude al aprendizaje.

Tabla 13 *Lista de Chequeo Coevaluación - Actividad 2*

Proyecto: Desarrollo de Pensamiento Computacional **Actividad: Estructurando el Pensamiento - De la Lógica al Pseudocódigo Usando PSeInt**					
Equipo: _____				**Fecha: _____**	
#	**Criterio a Evaluar**	**Valoración**		**Comentarios / Sugerencias**	
		SI	**NO**		
	ALGORITMO				
1	¿El algoritmo presenta una secuencia lógica de pasos?				Explica brevemente si la secuencia es clara y si sigue un orden lógico para resolver el problema.
2	¿Se identifican claramente las entradas del problema?				¿Están todas las entradas necesarias especificadas?
3	¿Los procesos o acciones están definidos con claridad?				¿Se entiende claramente qué se debe hacer en cada paso?
4	¿Se identifica claramente la salida o resultado?				¿El algoritmo produce el resultado esperado para el problema planteado?
5	¿El algoritmo resuelve el problema propuesto?				Justifica si la solución algorítmica es adecuada para el problema.
	DIAGRAMA DE FLUJO				
6	¿Se utiliza el símbolo de inicio y fin correctamente?				¿Están marcados claramente el inicio y el final del diagrama?
7	¿Los procesos se representan con el símbolo adecuado?				¿Se utiliza el rectángulo para representar las acciones o los cálculos?
8	¿Las entradas/salidas se representan correctamente?				¿Se utiliza el paralelogramo para indicar la entrada o salida de datos?
9	¿Las decisiones se representan con el rombo?				¿Se utilizan los rombos para representar las condiciones o las preguntas?
10	¿El flujo del diagrama es claro y fácil de seguir?				¿Las flechas indican la dirección del flujo de manera lógica y sin ambigüedades?
11	¿El diagrama de flujo representa fielmente el algoritmo propuesto?				¿Existe una correspondencia clara entre los pasos del algoritmo y los elementos del diagrama?

d. Lista de Chequeo Heteroevaluación - Actividad 3

Esta lista de chequeo permite al docente medir la capacidad de los estudiantes para transitar de una lógica lineal (PSeInt) a una lógica basada en eventos y paralelismo (Scratch). Este instrumento garantiza que la valoración no se limite al resultado estético, sino que profundice en la eficiencia y estructura del código desarrollado.

Tabla 14 *Lista de Chequeo Heteroevaluación - Actividad 3*

Proyecto: Desarrollo de Pensamiento Computacional **Actividad: Codificación Visual - Programación por Bloques con Scratch**					
Equipo: _____		**Fecha: _____**			
#	**Criterio a Evaluar**	**Sí**	**No**	**Parcial**	**Observaciones / Evidencias**
1	**Lógica de Eventos:** ¿Utiliza correctamente los bloques de inicio (bandera verde, teclas, mensajes) para desencadenar acciones?				
2	**Control de Flujo:** ¿Implementa bucles (*por siempre, repetir*) y condicionales (*si... entonces*) para dotar de inteligencia al comportamiento de los personajes?				
3	**Paralelismo y Sincronización:** ¿Se evidencia la ejecución simultánea de varios hilos de código y el uso de bloques de espera o mensajes para coordinar diálogos?				
4	**Eficiencia Algorítmica (Depuración):** ¿El código es limpio y evita la redundancia de bloques? ¿Se han corregido errores lógicos de movimiento o interacción?				
5	**Integración Multimedia:** ¿Aprovecha las herramientas de Scratch para personalizar disfraces, escenarios y sonidos en función de la narrativa?				
6	**Competencia Digital (Publicación):** ¿El proyecto ha sido compartido correctamente en el estudio de la clase y el enlace está registrado en la Bitácora Digital?				

e. Lista de Chequeo Coevaluación - Actividad 3

Esta lista de chequeo está diseñada buscando que los estudiantes desarrollen competencias de pensamiento crítico al analizar la lógica y creatividad en los proyectos de sus pares, promoviendo la retroalimentación constructiva y la cultura del 'Remix' propia de la comunidad Scratch."

Instrucciones para los estudiantes:

1. Lean cuidadosamente el algoritmo y el diagrama de flujo del grupo que van a evaluar.
2. Para cada criterio, marquen "Sí" si consideran que se cumple y "No" si no se cumple o "Parcial" si cumple, pero hay aspectos a mejorar.
3. En la columna de "Comentarios para mejorar / Fortalezas", proporcionen una breve explicación de su elección y ofrezcan comentarios constructivos que permitan mejorar el trabajo del otro grupo o señalen las fortalezas a destacar del proyecto de sus pares.
4. Sean respetuosos y enfocados en ofrecer una retroalimentación que ayude al aprendizaje.

Tabla 15 *Lista de Chequeo Coevaluación - Actividad 3*

Proyecto: Desarrollo de Pensamiento Computacional **Actividad: Codificación Visual - Programación por Bloques con Scratch**					
Equipo: _____		**Fecha: _____**			
#	**Criterio de Evaluación (Nivel de Logro)**	**Sí**	**No**	**Parcial**	**Comentarios para mejorar / Fortalezas**
1	**Interactividad:** ¿El proyecto inicia correctamente al presionar la bandera verde y responde a las teclas o clics del usuario?				
2	**Lógica de Bloques:** ¿Se nota el uso de bucles (repetir) y condicionales (si... entonces) para que los personajes actúen de forma inteligente?				
3	**Originalidad:** ¿La historia o el juego presenta elementos creativos en sus disfraces, escenarios o sonidos que lo hacen único?				
4	**Claridad Narrativa:** ¿Se entiende claramente de qué trata la historia o cuáles son las reglas del juego al momento de ejecutarlo?				
5	**Fluidez del Código:** ¿Los personajes se mueven y hablan de forma coordinada (sin que se atropellen los diálogos o las acciones)?				
6	**Colaboración:** ¿El grupo de trabajo evaluado demuestra haber trabajado de manera colaborativa para resolver los retos técnicos del proyecto?				

f. Lista de Chequeo Heteroevaluación - Actividad 4

Esta lista de chequeo permite al docente evaluar la capacidad de los estudiantes para gestionar la incertidumbre del mundo real, verificando si son capaces de traducir variables físicas (como luz o temperatura) en estructuras de control precisas. Este proceso asegura que el aprendizaje trascienda la simulación digital y se consolide en una ejecución técnica rigurosa sobre el hardware real.

Tabla 16 *Lista de Chequeo Heteroevaluación - Actividad 4*

Proyecto: Desarrollo de Pensamiento Computacional Actividad: Sensores y Actuadores - Programando Soluciones para el Mundo Real mediante Micro:bit					
Equipo: _____		**Fecha: _____**			
#	**Criterio a Evaluar**	**Sí**	**No**	**Parcial**	**Observaciones / Evidencias**
1	**Gestión de Entradas:** ¿Utiliza correctamente los sensores (luz, temperatura, acelerómetro, etc) para activar funciones del programa?				
2	**Lógica de Umbrales:** ¿Configura de manera adecuada los valores numéricos (comparadores mayor/menor que) para que el sensor responda al entorno?				
3	**Uso de Actuadores:** ¿Logra representar la información de salida de forma clara a través de la matriz LED o sonidos?				
4	**Tránsito Simulador-Hardware:** ¿Demuestra capacidad para descargar y ejecutar el código desde el entorno MakeCode hacia la tarjeta física?				
5	**Documentación Audiovisual:** ¿Se evidencia en la Bitácora el registro del prototipo funcionando en un entorno real (foto/video)?				

g. Lista de Chequeo Coevaluación - Actividad 4

Esta lista de chequeo está diseñada buscando que los estudiantes desarrollen habilidades comunicativas y de pensamiento crítico, obligando al evaluador a analizar la usabilidad y la eficiencia de soluciones ajenas. Al interactuar con el hardware de otros equipos mediante una "Prueba de Usuario", se promueve una reflexión colectiva sobre la funcionalidad tecnológica y la empatía en el diseño, elementos clave en la formación de ciudadanos digitalmente competentes.

Instrucciones para los estudiantes:

1. Lean cuidadosamente el algoritmo y el diagrama de flujo del grupo que van a evaluar.
2. Para cada criterio, marquen "Sí" si consideran que se cumple y "No" si no se cumple o "Parcial" si cumple, pero hay aspectos a mejorar.
3. En la columna de "Comentarios para mejorar / Fortalezas", proporcionen una breve explicación de su elección y ofrezcan comentarios constructivos que permitan mejorar el trabajo del otro grupo o señalen las fortalezas a destacar del proyecto de sus pares.
4. Sean respetuosos y enfocados en ofrecer una retroalimentación que ayude al aprendizaje.

Tabla 17 *Lista de Chequeo Coevaluación - Actividad 4*

Proyecto: Desarrollo de Pensamiento Computacional Actividad: Sensores y Actuadores - Programando Soluciones para el Mundo Real mediante Micro:bit					
Equipo: _____		**Fecha: _____**			
#	**Criterio de Evaluación**	**Sí**	**No**	**Parcial**	**Comentarios para mejorar / Fortalezas**
1	**Funcionalidad:** ¿El dispositivo del equipo compañero cumple con la tarea propuesta (ej. avisar si hay mucha luz o si se mueve)?				
2	**Interactividad:** ¿La Micro:bit responde de forma inmediata cuando interactuamos con sus sensores o botones?				
3	**Claridad del Mensaje:** ¿Se entiende lo que muestra la pantalla LED o el sonido sin necesidad de que el equipo lo explique?				
4	**Ingenio y Creatividad:** ¿Usaron los bloques de programación de forma creativa para que el prototipo fuera más interesante o eficiente?				

h. Lista de Chequeo Heteroevaluación - Actividad 5

La heteroevaluación de este proyecto final se sustenta en la valoración integral de las competencias adquiridas durante toda la secuencia didáctica. El instrumento busca medir no solo la funcionalidad técnica del producto, sino el rigor en la documentación y la capacidad de síntesis del estudiante. Se evalúa la 'solución' como un todo coherente, donde el código, el hardware y la sustentación oral demuestran que los estudiantes han interiorizado el pensamiento computacional como una herramienta para la transformación de su entorno, alineándose con los estándares de calidad que exige un proyecto de innovación educativa.

Tabla 18 *Lista de Chequeo Heteroevaluación - Actividad 5*

Proyecto: Desarrollo de Pensamiento Computacional **Actividad: Desafío de Innovación - Soluciones Tecnológicas para el Entorno Escolar**					
Equipo: _____		**Fecha: _____**			
#	**Criterio de Evaluación**	**Sí**	**No**	**Parcial**	**Observaciones / Evidencias**
1	**Complejidad Lógica:** ¿El proyecto integra de forma efectiva bucles, condicionales y eventos para resolver el problema?				
2	**Uso de Herramientas:** ¿Se evidencia un dominio adecuado de las plataformas elegidas (Scratch/Micro:bit) según la solución planteada?				
3	**Funcionalidad:** ¿El prototipo realiza las tareas para las que fue diseñado sin errores críticos en la ejecución?				
4	**Documentación:** ¿La Bitácora Digital contiene el proceso completo (diseño, código y reflexión) de manera organizada?				
5	**Sustentación:** ¿El equipo explica con claridad y lenguaje técnico el funcionamiento de su proyecto durante la feria?				

i. Lista de Chequeo Coevaluación - Actividad 5

La coevaluación se fundamenta en el aprendizaje dialógico y la crítica constructiva entre pares. Al evaluar proyectos ajenos, los estudiantes deben poner en práctica su capacidad de análisis técnico para identificar aciertos y áreas de mejora en lógicas de programación distintas a las propias. Este ejercicio fortalece la ética del trabajo colaborativo y la comunicación asertiva, permitiendo que el estudiante reconozca el valor de la diversidad de soluciones para un mismo problema, un principio fundamental en la cultura maker y el desarrollo de software.

Instrucciones para los estudiantes:

1. Lean cuidadosamente el algoritmo y el diagrama de flujo del grupo que van a evaluar.
2. Para cada criterio, marquen "Sí" si consideran que se cumple y "No" si no se cumple o "Parcial" si cumple, pero hay aspectos a mejorar.
3. En la columna de "Comentarios para mejorar / Fortalezas", proporcionen una breve explicación de su elección y ofrezcan comentarios constructivos que permitan mejorar el trabajo del otro grupo o señalen las fortalezas a destacar del proyecto de sus pares.
4. Sean respetuosos y enfocados en ofrecer una retroalimentación que ayude al aprendizaje.

Tabla 19 *Lista de Chequeo Coevaluación - Actividad 5*

Proyecto: Desarrollo de Pensamiento Computacional **Actividad: Desafío de Innovación - Soluciones Tecnológicas para el Entorno Escolar**					
Equipo: _____		**Fecha: _____**			
#	**Criterio de Evaluación**	**Sí**	**No**	**Parcial**	**Comentarios para mejorar / Fortalezas**
1	**Impacto:** ¿La solución propuesta es útil para el problema del colegio que intentaron resolver?				
2	**Originalidad:** ¿El equipo usó la tecnología de una forma creativa o diferente a lo visto en clase?				
3	**Calidad de Presentación:** ¿Los materiales y la explicación del equipo fueron claros y fáciles de entender?				
4	**Trabajo en Equipo:** ¿Se nota que todos los integrantes del grupo conocen el proyecto y participaron en su creación?				

j. Modelo de Cuestionario de Opinión de los Estudiantes (para finalizar cada actividad)

Este es un modelo genérico que se puede adaptar para cada actividad, enfocándose en los aspectos específicos que se quiera evaluar en cada una de ellas.

Estimado/a estudiante:

Agradecemos tu participación en esta actividad. Tu opinión es muy importante para mejorar el desarrollo de este proyecto. Por favor, responde con sinceridad a las siguientes preguntas. (Puedes utilizar una escala de 1 a 5, donde 1 es "Muy en desacuerdo" y 5 es "Muy de acuerdo", o proporcionar opciones cerradas y abiertas según la pregunta).

Sobre la Actividad en General:

1. ¿Qué tanto te gustó esta actividad? (Escala del 1 al 5)
2. ¿Consideras que los objetivos de la actividad fueron claros? (Sí/No/Parcialmente)
3. ¿El tiempo dedicado a la actividad fue suficiente? (Sí/No/Demasiado poco/Demasiado)
4. ¿El nivel de dificultad de la actividad fue el adecuado? (Demasiado fácil/Adecuado/Demasiado difícil)
5. ¿Qué fue lo que más te gustó de esta actividad? (Pregunta abierta)

6. ¿Qué fue lo que menos te gustó o te resultó más difícil de esta actividad? (Pregunta abierta)

Sobre la Metodología y los Recursos:

7. ¿Consideras que la forma en que se desarrolló la actividad (clase teórica, trabajo en grupo, etc.) fue efectiva para tu aprendizaje? (Escala del 1 al 5)
8. ¿Los recursos utilizados (videos, materiales digitales, herramientas, etc.) fueron útiles para comprender los contenidos y realizar las tareas? (Escala del 1 al 5)
9. ¿La explicación del docente fue clara y te ayudó a realizar la actividad? (Escala del 1 al 5)
10. ¿Tuviste algún problema técnico con las herramientas digitales? (Sí/No. Si sí, explica brevemente cuál fue el problema)

Sobre los Productos Digitales Desarrollados (Adaptar esta sección a los productos digitales específicos de cada actividad):

11. ¿Consideras que la elaboración del [Nombre del Producto Digital] te ayudó a comprender mejor los conceptos? (Escala del 1 al 5)
12. ¿La herramienta utilizada para crear el [Nombre del Producto Digital] fue fácil de usar? (Escala del 1 al 5)
13. ¿Consideras que el [Nombre del Producto Digital] fue una forma útil de aplicar lo aprendido? (Escala del 1 al 5)
14. (Incluir preguntas similares para cada producto digital desarrollado en la actividad)

Sobre el Desarrollo del Pensamiento Computacional y el Proyecto:

15. ¿Consideras que esta actividad te ayudó a desarrollar tu pensamiento computacional? (Escala del 1 al 5)
16. ¿Encuentras útil lo aprendido en esta actividad para la solución del problema del proyecto? (Escala del 1 al 5)
17. ¿Estás motivado/a para seguir trabajando en el proyecto? (Escala del 1 al 5)

Comentarios Adicionales:

18. ¿Tienes algún otro comentario o sugerencia sobre esta actividad o sobre el proyecto en general? (Pregunta abierta)

¡Muchas gracias por tu colaboración!

Consideraciones:

- **Anonimato:** Se debe asegurar a los estudiantes que sus respuestas serán anónimas para fomentar la sinceridad.
- **Adaptación:** Se deben modificar las preguntas y las opciones de respuesta para que se ajusten específicamente a los objetivos, las tareas y los productos de cada actividad.
- **Análisis:** Se debe planificar cómo se van a recoger y analizar los datos de estos cuestionarios para identificar tendencias y áreas de mejora.

k. Rubrica de Evaluación - Productos Digitales

Esta rúbrica de evaluación se puede adaptar dependiendo de los productos digitales que se vaya a recoger en cada una de las actividades. La valoración final del desempeño de los grupos de trabajo durante cada una de las actividades se obtendrá mediante el promedio de lo obtenido en los diferentes productos digitales, así como su desempeño en las diferentes actividades de clase.

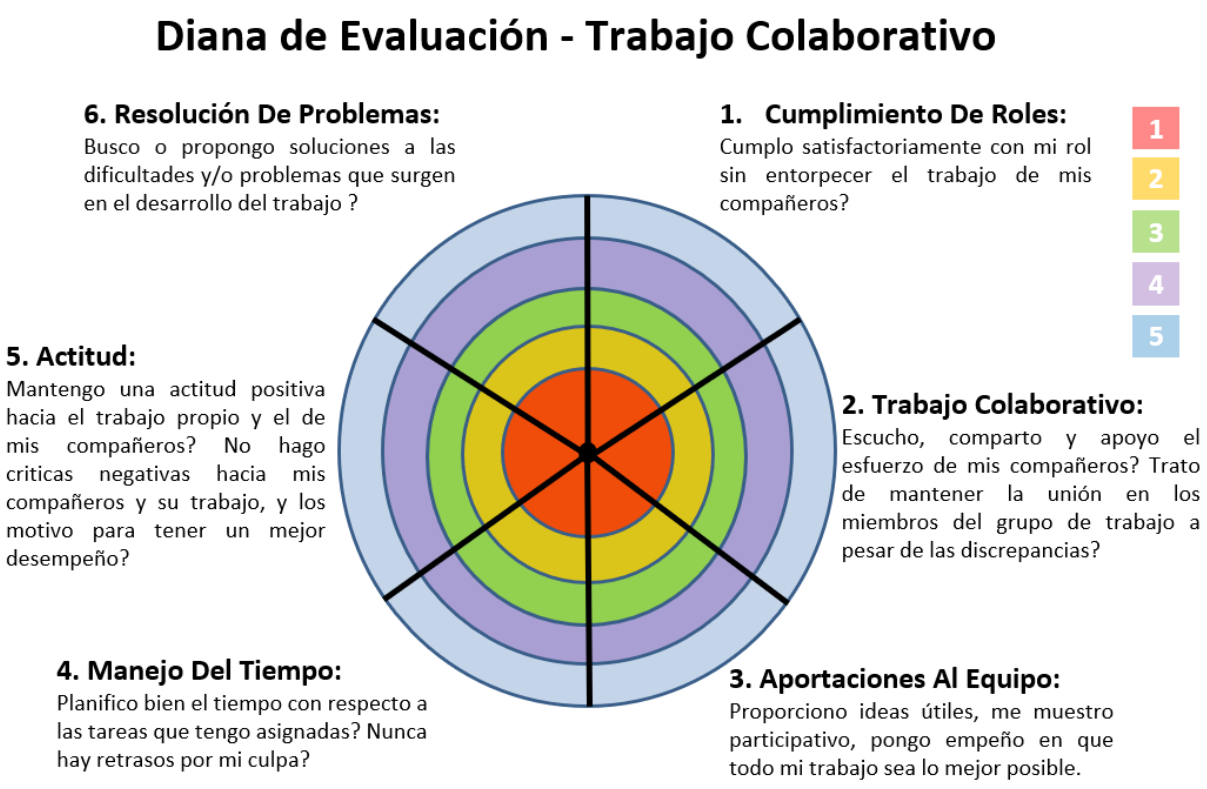
Tabla 20 *Ejemplo Rubrica de Evaluación - Productos Digitales*

Criterios de Evaluación	Nivel 1 (Insuficiente)	Nivel 2 (Regular)	Nivel 3 (Aceptable)	Nivel 4 (Bueno)	Nivel 5 (Excelente)	PUNTUACIÓN
Inclusión de información necesaria	Faltan datos clave; no se incluye información relevante o se omite gran parte de la información requerida.	Faltan algunos datos importantes o hay confusión en la presentación de la información.	Se incluye la mayoría de la información requerida de manera clara y organizada.	Incluye toda la información necesaria de forma completa y precisa.	Presenta información adicional, detallada y enriquecedora.	5
Organización adecuada	No hay estructura clara; falta cohesión entre las ideas y pasos.	La organización es adecuada en general, pero algunos elementos pueden presentar desorden o falta de coherencia.	Los pasos están bien organizados, aunque podría haber una mejora en la presentación o secuencia de la información.	La estructura y la secuencia de los pasos son claras y lógicas.	La presentación muestra una estructura excelente y una secuencia lógica perfectamente organizada.	5
Facilitación de aprendizaje colaborativo	Las actividades no fomentan la colaboración entre los estudiantes; la interacción es mínima.	Las actividades pueden permitir cierta colaboración, pero no promueven la interacción significativa entre los estudiantes.	Las actividades facilitan la colaboración entre los estudiantes, aunque puede haber mejoras en la interacción.	Las actividades fomentan el aprendizaje colaborativo de manera efectiva.	Las actividades están diseñadas de manera excepcional para fomentar la colaboración y el trabajo en equipo.	5
Herramientas TIC adecuadas	Las herramientas digitales seleccionadas no son relevantes o no se aplican correctamente a las fases del proyecto.	Algunas herramientas digitales se aplican, pero no son completamente adecuadas para cada fase del proyecto.	La mayoría de las herramientas digitales elegidas son adecuadas para las diferentes etapas del proyecto.	Las herramientas digitales seleccionadas son altamente adecuadas y apoyan eficazmente cada fase del proyecto.	Las herramientas digitales elegidas son excepcionales, mejorando significativamente cada etapa del proyecto.	5
Redacción, formato y citación	Presenta una redacción desordenada o errores graves de formato y citación.	La redacción y el formato son aceptables, pero se pueden mejorar. Algunos errores de citación.	Cumple con las normas de redacción y formato, pero con algunas áreas de mejora.	La redacción es clara y precisa, el formato es consistente y hay una correcta citación de fuentes.	La redacción es excelente, el formato es impecable y la citación de fuentes es ejemplar.	5
ORTOGRAFÍA: Penalización de -0,25 puntos por cada cinco faltas de acentuación. Penalización de - 0,10 puntos por cada falta ortográfica						PROMEDIO (Nota Final) 5

l. Diana de Evaluación - Trabajo Colaborativo

La diana de evaluación para trabajo colaborativo es un instrumento gráfico que permite a los estudiantes autoevaluar y coevaluar su desempeño y el de sus compañeros en diferentes aspectos clave del trabajo en equipo. A través de una escala numérica del 1 al 5, donde 1 representa una valoración baja y 5 una valoración alta, los estudiantes pueden reflexionar sobre su cumplimiento de roles, la efectividad del trabajo colaborativo general, la calidad y cantidad de sus aportaciones al equipo, la eficiencia en el manejo del tiempo, su actitud durante el proceso y su capacidad para la resolución de problemas dentro del grupo. Esta herramienta visual facilita la identificación de fortalezas y áreas de mejora tanto a nivel individual como colectivo, promoviendo la reflexión metacognitiva y la mejora continua de las dinámicas de trabajo en equipo.

Figura 12. *Diana de Evaluación - Trabajo Colaborativo*



Anexo 2. Recursos Tecno-Pedagógicos Utilizados en las Actividades

a. Reto Serpientes y Tesoros - Actividad 1

- Actividad Vinculada: Estrategias Desconectadas - Fundamentos de la Lógica Programable
- Nombre del Proyecto: Kit de Comandos Primitivos: "El Robot-Humano".
- Herramienta: Recurso de diseño gráfico para impresión o visualización en tabletas/tablero.

Tabla 21 *Ejemplo Tablero de Juego Reto Serpientes y Tesoros - Actividad 1*

	A	B	C	D	E	F
6	💎		🐍		💎	
5		🐍				🐍
4			💎	🐍		
3	🐍				🐍	💎
2	💎	🐍				
1			💎	🐍		INICIO

(Leyenda: 💎 = Tesoro / 🐍 = Serpiente)

- Descripción Técnica: Se trata de un conjunto de tarjetas de comandos estandarizados (Fichas de Sintaxis) que representan acciones básicas: Avanzar, Girar_Izquierda, Girar_Derecha, Tomar_Objeto, Soltar_Objeto. El recurso incluye un Tablero de Juego *Reto Serpientes y Tesoros* (un mapa de cuadrícula) donde los estudiantes deben organizar las fichas en una secuencia lógica para que un compañero (el robot) cumpla una misión sin errores.

Tabla 22 *Recuadro de Instrucciones Reto Serpientes y Tesoros - Actividad 1*

Comando	Acción	Representación Visual
INICIO	Marca el punto de partida en la cuadrícula.	🟢
AVANZAR	Desplaza el cursor una (1) casilla hacia adelante.	↑
GIRAR_DER	Gira el cursor 90° a la derecha sin cambiar de casilla.	↻
GIRAR_IZQ	Gira el cursor 90° a la izquierda sin cambiar de casilla.	↺
TOMAR	Recoge el tesoro en la posición actual.	💎
FIN	Finaliza la ejecución del algoritmo.	🛑

- Valor Pedagógico (PK): Introduce los conceptos de Algoritmos (secuencia de pasos) y Sintaxis (reglas de lenguaje). Ayuda a los estudiantes a entender que las máquinas no "suponen" intenciones, sino que ejecutan instrucciones precisas.
- Justificación TPACK: Representa el Conocimiento Pedagógico del Contenido (PCK), donde el docente utiliza un diseño visual (Tecnología de diseño) para mediar un concepto lógico complejo (Contenido) sin necesidad de hardware, preparando el terreno cognitivo para la programación formal.
- Ejemplo Solución Codificada: Ruta al Tesoro 1 (Casilla C1)

  Partiendo desde la casilla de INICIO (F1) y evitando la serpiente ubicada en D1, el algoritmo resultante es:

  - 🟢 (INICIO en F1)
  - ↺ (GIRAR_IZQ - Orientación Oeste)
  - ↑ (AVANZAR a E1)
  - ↻ (GIRAR_DER - Orientación Norte para esquivar serpiente en D1)
  - ↑ (AVANZAR a E2)
  - ↺ (GIRAR_IZQ - Orientación Oeste)
  - ↑ (AVANZAR a D2)
  - ↑ (AVANZAR a C2)
  - ↺ (GIRAR_IZQ - Orientación Sur para esquivar serpiente en B2)
  - ↑ (AVANZAR a C1)
  - 💎 (TOMAR Tesoro)
  - 🛑 (FIN)

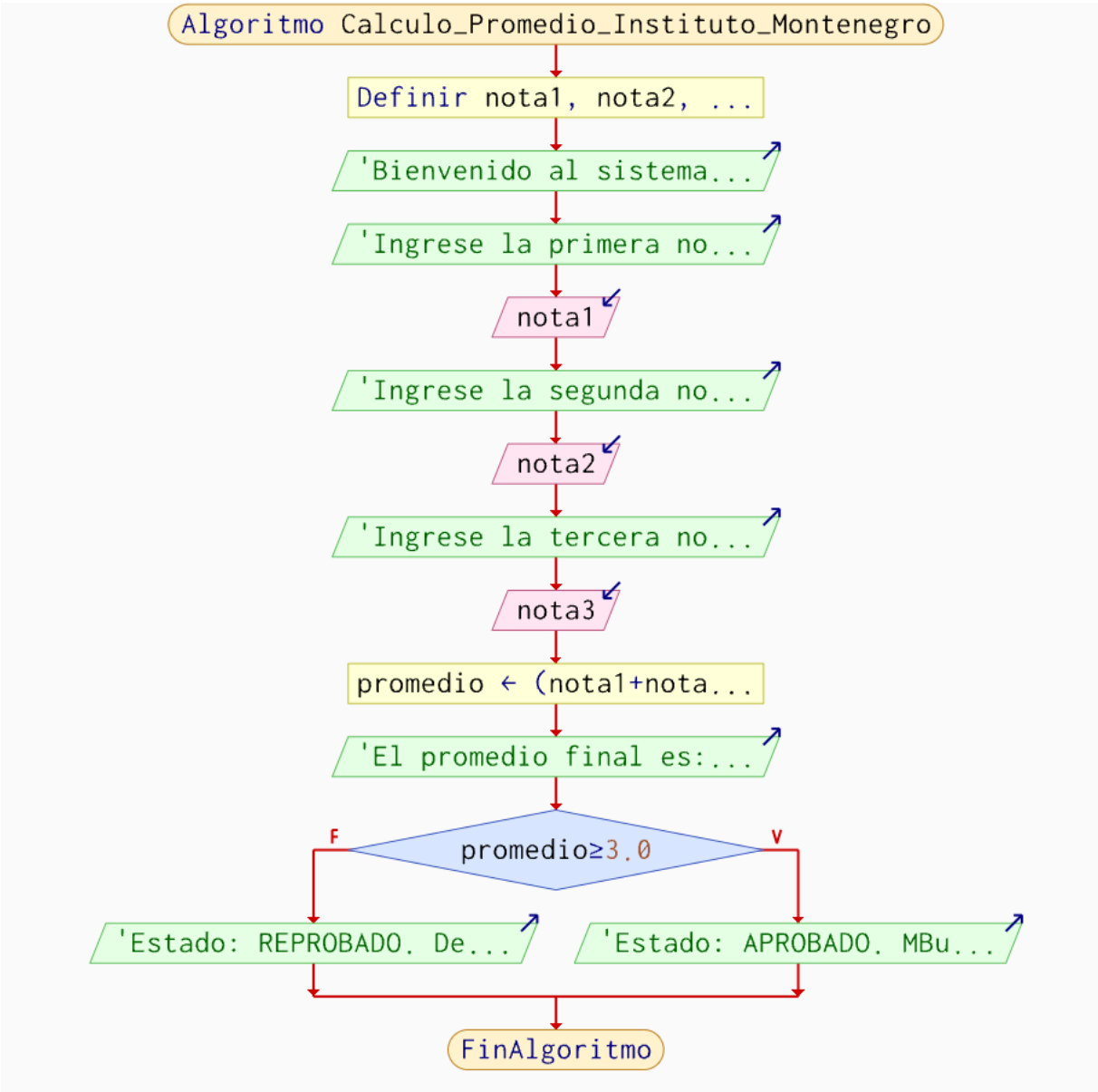
b. Algoritmo Calculo_Promedio_Instituto_Montenegro - Actividad 2

- Actividad Vinculada: Estructurando el Pensamiento - De la Lógica al Pseudocódigo Usando PSeInt
- Nombre del Proyecto: Algoritmo de Calificación y Seguimiento Académico.
- Herramienta: PSeInt (Configuración de perfil: Flexible o Estricto).
- Descripción Técnica: El algoritmo solicita al usuario el ingreso de tres notas, calcula el promedio aritmético y, mediante una estructura condicional simple (Si-Entonces), determina si el estudiante ha "Aprobado" o debe "Reforzar".

Tabla 23 *Pseudocodigo Calculo_Promedio_Instituto_Montenegro - Actividad 2*

Algoritmo Calculo_Promedio_Instituto_Montenegro
Definir nota1, nota2, nota3, promedio Como Real Escribir "Bienvenido al sistema de notas del Instituto Montenegro" Escribir "Ingrese la primera nota (0.0 a 5.0):" Leer nota1 Escribir "Ingrese la segunda nota:" Leer nota2 Escribir "Ingrese la tercera nota:" Leer nota3 promedio <- (nota1 + nota2 + nota3) / 3 Escribir "El promedio final es: ", promedio Si promedio >= 3.0 Entonces Escribir "Estado: APROBADO. ¡Buen trabajo!" Sino Escribir "Estado: REPROBADO. Debes asistir a tutorías." FinSi FinAlgoritmo

Figura 13 *Diagrama de Flujo Calculo_Promedio_Instituto_Montenegro - Actividad 2*



- Valor Pedagógico (PK): Este recurso trabaja la algoritmia pura. El estudiante debe entender el orden secuencial de las instrucciones y cómo la computadora procesa datos de entrada para generar una salida lógica.
- Justificación TPACK: El uso de PSeInt como tecnología (TK) permite que el contenido lógico (CK) se visualice de forma textual antes de pasar a los bloques, facilitando una transición pedagógica (PK) suave hacia Scratch.

c. Reto Scratch - Actividad 3

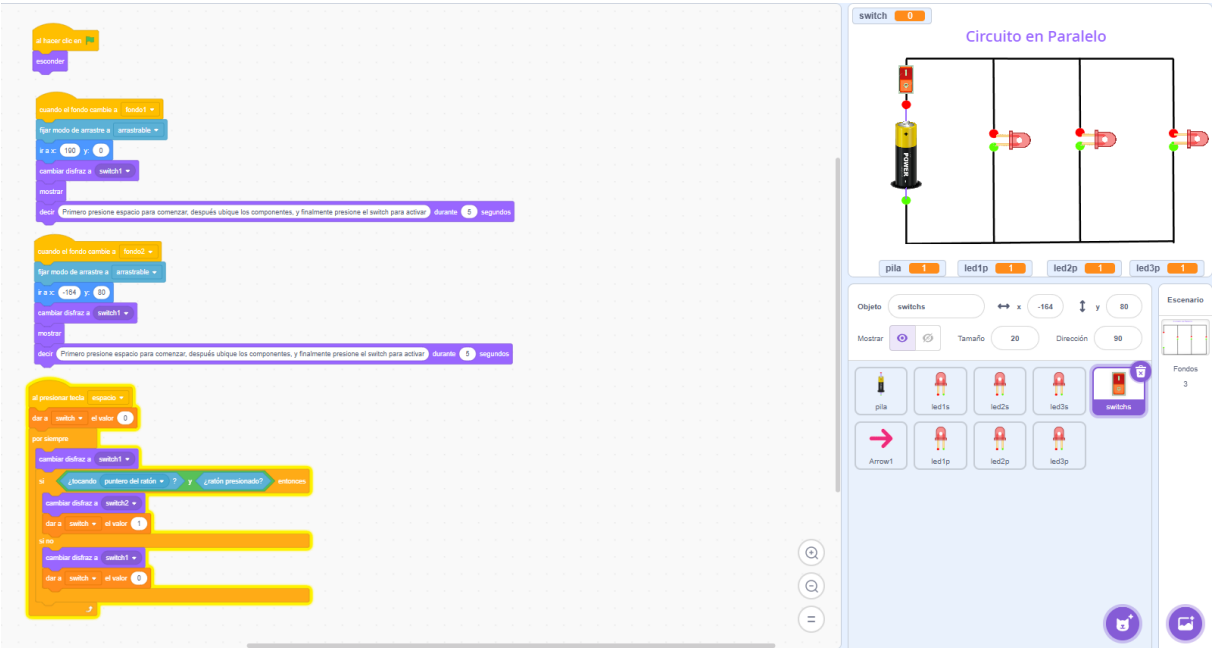
- Actividad Vinculada: Codificación Visual - Programación por Bloques con Scratch
- Nombre del Proyecto: Lógica de Eventos - Introducción a la Programación Visual.
- Herramienta: Scratch 3.0.
- Enlace de Acceso: https://scratch.mit.edu/projects/1045859576
- Descripción Técnica: El proyecto utiliza el paradigma de programación orientada a eventos para gestionar la interacción. Implementa mensajes de difusión (*broadcast*) para la sincronización de hilos paralelos y emplea coordenadas cartesianas para el control de movimiento.

Figura 14 *Pantalla Instrucciones Reto Scratch - Actividad 3*

La actividad consiste en un ejercicio simple que permite la comprensión de los circuitos en serie y en paralelo. Contiene 3 escenarios, uno de presentación (Taller de Electrónica), uno para el circuito en serie y el otro para circuito en paralelo. Cuando comienza cada escenario aparece una caja de dialogo con las instrucciones (Primero presione espacio para comenzar, después ubique los componentes, y finalmente presione el switch para activar). En la parte inferior aparece un indicador que permite verificar si la pila o los diferentes Leds se encuentran bien ubicados (0 si no están bien conectados, 1 cuando están conectados correctamente). Al momento de realizar la prueba al activar el switch si los leds están bien conectados y pasa corriente por el

circuito estos leds se iluminan y aparece un indicador visual para esto. Cuando se supera el primer escenario correspondiente al circuito en serie aparece una flecha que permite pasar al siguiente escenario correspondiente al circuito en paralelo, en este caso por la característica del circuito no es necesario que todos los leds estén bien ubicados para que se ilumine al momento de activar el switch.

Figura 15 *Bloques de Programación Reto Scratch - Actividad 3*



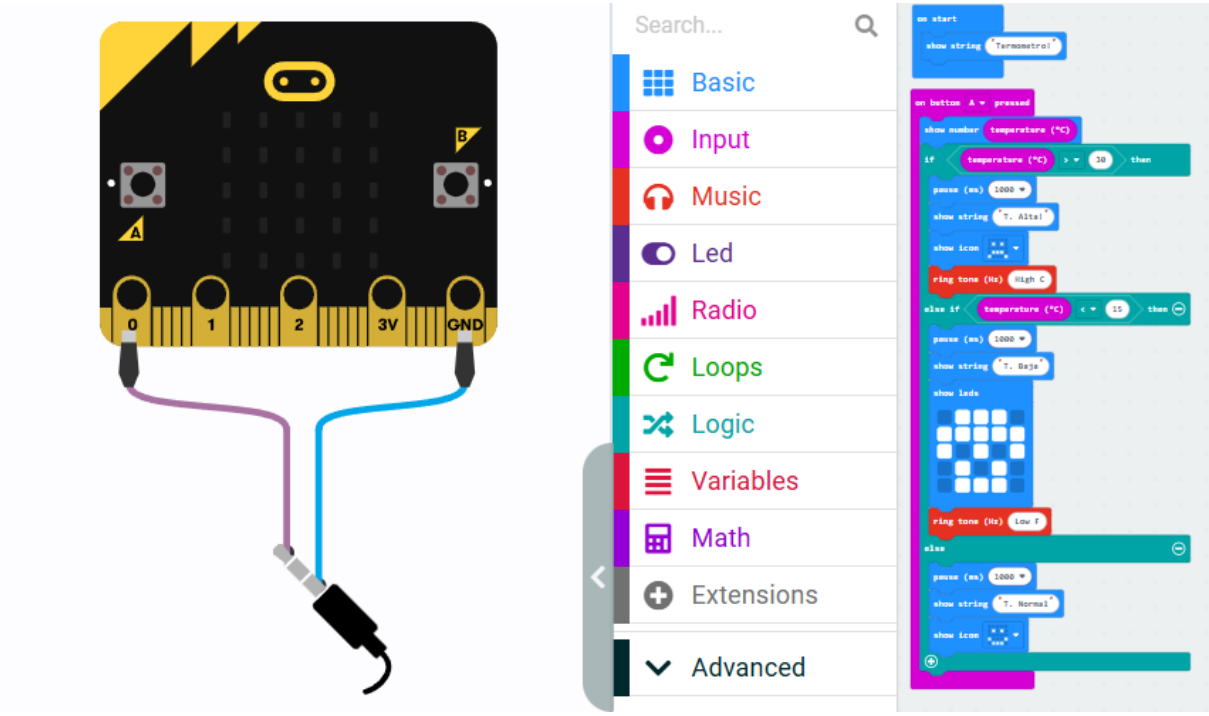
- Valor Pedagógico (PK): Fomenta la abstracción y la descomposición, permitiendo que el estudiante visualice cómo bloques lógicos se traducen en acciones narrativas mediante el uso de lógica de eventos
- Justificación TPACK: Representa el Conocimiento Tecnológico (TK) usando Scratch puesto al servicio del Conocimiento del Contenido (CK) sobre algoritmos de control de flujo, permitiendo una pedagogía de construcción creativa (PK).

d. Termómetro Digital - Actividad 4

- Actividad Vinculada: Sensores y Actuadores - Programando Soluciones para el Mundo Real mediante Micro:bit
- Nombre del Proyecto: Monitorización de Entornos Reales Mediante el Uso de Sensores
- Herramienta: Microsoft MakeCode para Micro:bit.
- Enlace de Acceso: https://makecode.microbit.org/S02721-35118-09999-50447

- Descripción Técnica: Implementa una estructura condicional basada en umbrales físicos. El código lee la temperatura ambiente (Input) y genera una respuesta visual en la matriz LED (Output), demostrando el funcionamiento de sistemas automáticos.

Figura 16 *Bloques de Programación En Micro:bit Termómetro Digital - Actividad 4*



Al iniciar la placa de desarrollo Micro:bit muestra la palabra “Termómetro” en la matriz LED, para iniciar a usar la placa se debe presionar el botón A, luego muestra la temperatura y dependiendo del valor de esta muestra un mensaje basado en diferentes umbrales: Si la temperatura está por encima de 30 °C muestra la frase “T. Alta” y luego muestra un emoji; si la temperatura está por debajo de 15 °C muestra la frase “T. Baja” y luego muestra un emoji; y si la temperatura se encuentra entre 15 °C y 30 °C muestra la frase “T. Normal” y luego muestra un emoji

- Valor Pedagógico (PK): Representa la integración entre software y hardware (TK-CK), permitiendo a los estudiantes experimentar la programación como una herramienta de interacción con el mundo material.
- Justificación TPACK: Evidencia la transición del software al hardware. Aquí, el TPACK se manifiesta en la capacidad del docente para mostrar cómo un concepto abstracto (condicionales) tiene un impacto físico medible, facilitando el aprendizaje basado en la experimentación directa.

e. Reto Reciclaje Instituto Montenegro - Actividad 5

- Actividad Vinculada: Desafío de Innovación - Soluciones Tecnológicas para el Entorno Escolar
- Nombre del Proyecto: Integración ABP y Solución de Contexto
- Herramienta: Scratch 3.0 (Ecosistema Integrado).
- Enlace de Acceso: https://scratch.mit.edu/projects/1045834202
- Descripción Técnica: Proyecto complejo de tipo "Serious Game" que integra gestión de variables (puntaje), listas de datos y detección de colisiones. El código está organizado modularmente para resolver un problema de sostenibilidad socio-ambiental real del Instituto Montenegro mediante el pensamiento sistémico.

Figura 17 *Pantalla Instrucciones Reto Reciclaje Instituto Montenegro - Actividad 5*

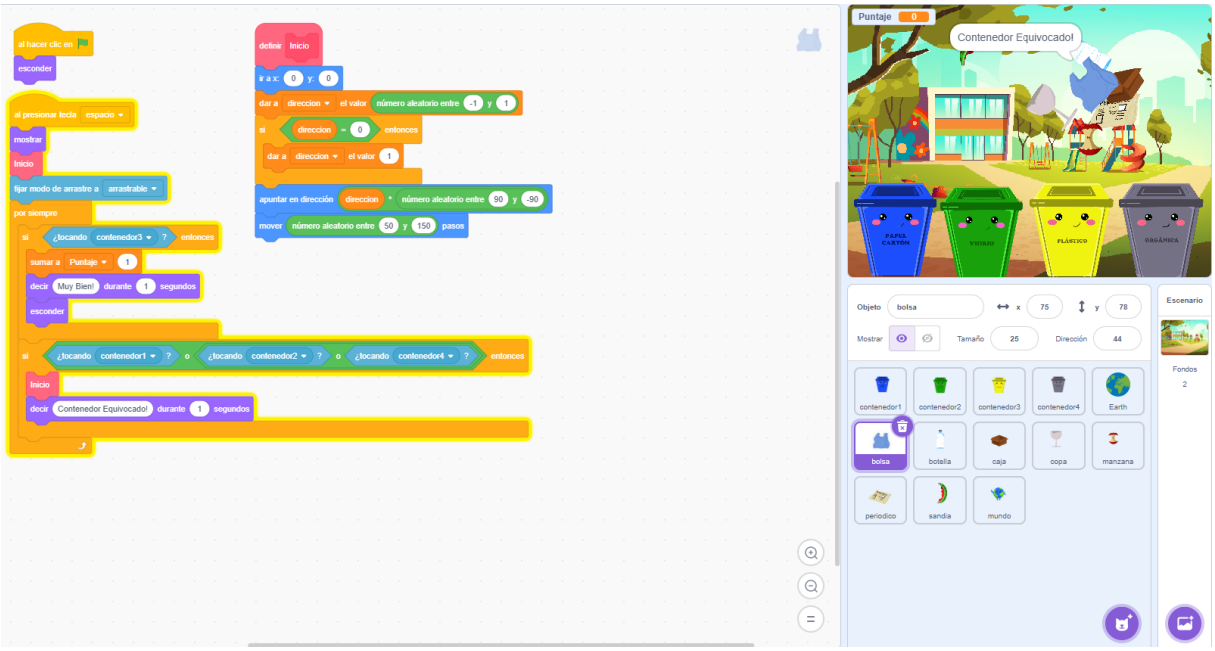
La actividad consiste en un juego que permite aprender a clasificar los residuos para su reciclaje. Contiene 2 escenarios, uno de presentación (Aprendamos a Reciclar!) y el otro para el juego de reciclaje. Se pasa de la presentación haciendo click en cualquier parte del escenario. Cuando comienza el escenario del juego hay que hacer click en el símbolo del planeta el cual da las instrucciones, con pausas de 5 segundos entre cada caja de dialogo. Luego hay que dar espacio para continuar. Hay cuatro contenedores diferentes en la parte inferior, y en el centro del escenario aparecen varios tipos de residuos en ubicaciones al azar que hay que arrastrar al contenedor correspondiente.

Figura 18 *Pantalla de Ejecución Reto Reciclaje Instituto Montenegro - Actividad 5*

Si se coloca el residuo en el contenedor equivocado sale el mensaje “Contenedor Equivocado!” y el residuo es ubicado nuevamente en una posición al azar, si se coloca en el contenedor correcto sale el mensaje “Muy Bien!” y se agrega un punto; al alcanzar un puntaje de 7 puntos sale una imagen de un planeta sonriendo con el mensaje “Muy Bien! Has Aprendido A Clasificar Residuos Para Reciclaje!!!”

Figura 19 *Pantalla Final Reto Reciclaje Instituto Montenegro - Actividad 5*

Figura 20 *Bloques de Programación Reto Reciclaje Instituto Montenegro - Actividad 5*



- Valor Pedagógico (PK): Es la máxima expresión del ABP. Los estudiantes analizan este recurso para comprender cómo el pensamiento computacional puede aplicarse a una problemática social de su propia institución.
- Valor en el Marco TPACK (Convergencia Total): Este proyecto es la culminación de la secuencia. Integra el contenido (ecología y lógica), la tecnología (programación avanzada) y la pedagogía (ABP), demostrando cómo el estudiante puede convertirse en un prosumidor de tecnología con impacto social.