



UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN DESARROLLO DE
VIDEOJUEGOS Y REALIDAD VIRTUAL

**Pensamiento Computacional y Visual Scripting
para Fortalecer Competencias de Educación
Ciudadana en Estudiantes de Enseñanza Media en
Chile**

JAVIER ESTEBAN MAHANA PALOMER

Profesor Guía: CRISTIAN VIDAL-SILVA

Memoria para optar al título de
Ingeniero en Desarrollo de Videojuegos y Realidad Virtual

Talca – Chile
Junio, 2025



UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN DESARROLLO DE
VIDEOJUEGOS Y REALIDAD VIRTUAL

**Pensamiento Computacional y Visual Scripting
para Fortalecer Competencias de Educación
Ciudadana en Estudiantes de Enseñanza Media en
Chile**

JAVIER ESTEBAN MAHANA PALOMER

Profesor Guía: CRISTIAN VIDAL-SILVA

Memoria para optar al título de
Ingeniero en Desarrollo de Videojuegos y Realidad Virtual

El presente documento fue calificado con nota: _____

Talca – Chile

Junio, 2025

TABLA DE CONTENIDOS

	página
Tabla de Contenidos	I
Índice de Figuras	III
Índice de Tablas	IV
Resumen	v
1. Introducción	1
2. Marco Teórico	4
2.1. Metodologías de enseñanza de pensamiento computacional	4
2.2. Enfoques en la Enseñanza Humanista	7
2.3. Métodos de evaluación	9
3. Metodología	11
3.1. Medición y análisis de resultados	13
3.1.1. Script para medir el éxito en los experimentos	13
3.1.2. Análisis estadístico	14
4. Intervención educativa	16
4.1. Contexto y Selección de la Muestra	16
4.2. Aplicación de la intervención	17
4.2.1. Preparación previa a las sesiones	17
4.2.2. Sesión 1: Introducción al Pensamiento Computacional	17
5. Diseño de Escenas	25
5.1. Implementación del script	25
5.1.1. Complicaciones de la implementación	26
5.2. Descripción escenas	27

6. Resultados y análisis	31
6.1. Resultados	31
6.1.1. Resultados del Script	31
6.1.2. Resultados de cuestionarios	32
6.1.3. Resultados de pruebas parciales	34
6.2. Análisis estadístico	34
6.2.1. Cuestionarios de pensamiento computacional	35
6.2.2. Pruebas de Educación Ciudadana	37
7. Conclusiones y trabajo futuro	39
Bibliografía	41
A: Cuestionarios de Medición del Pensamiento Computacional	46
A.1. Cuestionario 1	46
A.2. Cuestionario 2	49
B: Escena Preconfigurada no usada	52

ÍNDICE DE FIGURAS

	página
4.1. Clases teóricas sesión 1	18
4.2. Estudiantes creando algoritmos en papel	19
4.3. Grafo de visual scripting con traducciones	21
4.4. Estudiantes realizando la sesión 4	22
4.5. Instrucciones y reglas de la escena	23
5.1. Scriptable object y ventana de editor	26
5.2. Pantallazo de escena 2 mostrando puntos de activación de eventos y escombros	28
5.3. Pantallazo de escena 5 mostrando publicidad engañosa	30
6.1. Tabla con los resultados de los cuestionarios	32
6.2. Tabla con los delta del resultado de los cuestionarios	33
6.3. Tabla con los promedios de notas de las pruebas parciales de educación ciudadana	34
6.4. Tabla con valores p de la prueba t para los cuestionarios de pensa- miento computacional, antes y después de la intervención.	35
6.5. Tabla con valores p de la prueba t para muestras independientes, aplicada a los deltas de los cuestionarios de pensamiento computacional.	36
6.6. Tabla con valores p de la prueba t para las pruebas de educación ciudadana.	37

ÍNDICE DE TABLAS

	página
6.1. Resultados de desempeño por escena	31

RESUMEN

El pensamiento computacional es una habilidad esencial para la formación de los jóvenes en el siglo XXI. Diversos estudios internacionales indican que quienes desarrollan estas competencias desde etapas tempranas no solo se desempeñan mejor en áreas como matemáticas e informática, sino que también adquieren herramientas valiosas para enfrentar problemáticas cotidianas de forma lógica, secuencial y crítica. No obstante, el sistema educativo chileno aún no ha integrado de manera efectiva el pensamiento computacional en su currículo escolar.

Con el objetivo de incorporar estas habilidades en el contexto nacional, este trabajo propone una intervención educativa dirigida a estudiantes de enseñanza media, mediante el uso del motor de videojuegos Unity y la técnica de Visual Scripting. La propuesta combina teoría y práctica, y se fundamenta en enfoques pedagógicos activos y colaborativos, que promueven tanto el desarrollo de competencias técnicas como la reflexión sobre fenómenos sociales contemporáneos.

En particular, las actividades estuvieron contextualizadas en la asignatura de Formación Ciudadana, con foco en los riesgos que amenazan la democracia, como la desinformación, delincuencia, polarización o el debilitamiento de los derechos civiles. Se espera que los estudiantes, al resolver problemas dentro de escenarios interactivos inspirados en dichos temas, fortalezcan tanto su pensamiento computacional como su capacidad para discernir, argumentar y actuar con base en valores democráticos.

Este proyecto no solo busca mejorar el desempeño en habilidades computacionales, sino también ofrecer una metodología interdisciplinaria que vincule la tecnología con la formación cívica. A largo plazo, su implementación podría abrir nuevas vías para integrar competencias digitales en otras asignaturas, contribuyendo así a una educación más integral, crítica y acorde a los desafíos actuales de la sociedad.

El pensamiento computacional es una competencia fundamental para la formación integral de los estudiantes del siglo XXI. Diversos estudios internacionales han demostrado que su desarrollo temprano no solo mejora el rendimiento en áreas como matemáticas e informática, sino que también fortalece habilidades transversales como la resolución lógica de problemas, el pensamiento crítico y la toma de decisiones informada. Sin embargo, el sistema educativo chileno aún no ha integrado de forma efectiva este tipo de habilidades en el currículo escolar.

Con el objetivo de abordar esta brecha, esta investigación propone y evalúa una intervención educativa orientada a estudiantes de enseñanza media, basada en el uso de Visual Scripting dentro del motor de videojuegos Unity. La intervención fue diseñada de forma interdisciplinaria, integrando contenidos de la asignatura de Formación Ciudadana mediante escenarios interactivos.

La propuesta combinó actividades conectadas (con tecnología) y desconectadas (analógicas), promoviendo la colaboración, la reflexión crítica y el aprendizaje activo. El estudio se desarrolló con un enfoque cuasi-experimental, incorporando un grupo control y un grupo experimental, y evaluando el impacto mediante pruebas *t* de Student.

Los resultados muestran una mejora estadísticamente significativa en habilidades específicas del pensamiento computacional dentro del grupo experimental, particularmente en pensamiento lógico, algoritmos básicos y depuración de procesos. En contraste, no se observaron mejoras significativas en habilidades como la abstracción o la identificación de problemas, posiblemente debido a limitaciones en el tiempo de intervención y a recortes en el contenido impartido. Asimismo, se evidenció una mejora significativa en los aprendizajes de Educación Ciudadana en el grupo experimental en comparación con el grupo control, lo cual sugiere que el pensamiento computacional podría potenciar también competencias cívicas.

Finalmente, se concluye que el uso de Visual Scripting en Unity presenta barreras de accesibilidad importantes para el contexto educativo público chileno, principalmente por su disponibilidad exclusiva en inglés. Se propone, para futuras investigaciones, la exploración de herramientas similares como Blueprints en Unreal Engine, que cuentan con localización en español y podrían facilitar la implementación. Este estudio aporta evidencia sobre el potencial de una metodología interdisciplinaria que integra tecnología, ciudadanía y habilidades del siglo XXI, abriendo nuevas posibilidades para una educación más crítica, integral y contextualizada.

1. Introducción

El pensamiento computacional (CT) se define como “el proceso mental que implica formular problemas y expresar sus soluciones de manera que una computadora (ya sea una máquina o un ser humano) pueda ejecutarlas de forma eficiente” [1]. Tal como señala Yadav [2], este proceso abarca cinco habilidades clave en el ámbito educativo: la identificación y descomposición de problemas, la abstracción de la información, el pensamiento lógico, el pensamiento algorítmico y la depuración de procesos. De este modo, el pensamiento computacional se convierte en un conjunto de habilidades esenciales que toda persona debería adquirir, y cuya enseñanza debe comenzar a temprana edad [3, 4, 5]. Aquellos que desarrollan estas habilidades están mejor preparados para abordar problemas cotidianos de manera sistemática y eficiente [6].

En los últimos años, el pensamiento computacional se ha integrado progresivamente en la educación tradicional, especialmente en las áreas de STEM (Ciencia, Tecnología, Ingeniería y Matemáticas) [7, 8, 9]. El enfoque STEM promueve la enseñanza y el aprendizaje desde una perspectiva interdisciplinaria, incentivando a los estudiantes a aplicar conceptos de diferentes disciplinas para resolver problemas del mundo real [10]. Las materias STEM a menudo requieren el uso de razonamiento lógico y analítico, habilidades directamente alineadas con los principios fundamentales del pensamiento computacional, lo que ha facilitado su implementación en estos contextos [8]. Este cambio ha transformado el paradigma educativo, ya que el pensamiento computacional solía enseñarse exclusivamente a través de la programación. No obstante, la integración del pensamiento computacional en cursos STEM ha demostrado ser altamente efectiva, no solo para mejorar las habilidades de CT [11], sino también para potenciar el aprendizaje en estas materias [12, 7].

En Chile, sin embargo, el sistema educativo escolar aún no ha integrado el desarrollo del pensamiento computacional en su currículo escolar. Esta carencia es preocupante, ya que el pensamiento computacional se considera una habilidad clave para afrontar los desafíos del siglo XXI, donde el manejo de tecnologías y la capacidad para resolver problemas de manera lógica y sistemática son cada vez más necesarias [3, 4]. Aunque la relación entre el pensamiento computacional y el aprendizaje de materias STEM ha sido ampliamente investigada, existe un vacío significativo en la literatura académica respecto a su aplicación en áreas no STEM, como la formación ciudadana. Este vacío plantea una oportunidad para investigar cómo el pensamiento computacional puede enriquecer la comprensión de fenómenos sociales contemporáneos y fomentar una ciudadanía activa, crítica y reflexiva ante los riesgos que enfrenta la democracia.

Esta investigación plantea la hipótesis de que la enseñanza de competencias de pensamiento computacional mediante el uso de Visual Scripting en Unity, enmarcada dentro del contexto de la asignatura de Educación Ciudadana, mejora significativamente tanto las habilidades de pensamiento computacional como la comprensión de problemáticas cívicas, en comparación con los métodos tradicionales de enseñanza basados en instrucción expositiva y textos escolares. A partir de esta premisa, surge la pregunta de investigación que guiará este trabajo: ¿La implementación de actividades de Visual Scripting en Unity, contextualizadas en problemáticas sociales, mejora de manera significativa las competencias de pensamiento computacional y la comprensión crítica de estos fenómenos?

El objetivo general de este trabajo es desarrollar una intervención educativa basada en actividades en Unity 3D, en las que los estudiantes resuelvan desafíos vinculados a problemáticas cívicas contemporáneas mediante el uso de Visual Scripting, con el fin de fomentar competencias de pensamiento computacional y formación cívica en estudiantes de enseñanza media en Chile.

Objetivos específicos

- Diseñar una propuesta de intervención educativa que integre actividades de programación de videojuegos mediante Visual Scripting en Unity al currículo de Educación Ciudadana para estudiantes de enseñanza media.
- Implementar una intervención educativa interdisciplinaria que combine sesiones teóricas y prácticas, utilizando metodologías activas y actividades tanto

conectadas como desconectadas.

- Desarrollar un conjunto de escenas interactivas en Unity, contextualizadas en problemáticas cívicas contemporáneas, que permitan aplicar habilidades de pensamiento computacional mediante Visual Scripting.
- Desarrollar un script automatizado en Unity que permita recopilar datos en tiempo real sobre la resolución de problemas por parte de los estudiantes durante la ejecución de las escenas interactivas.
- Evaluar el impacto de la intervención en las competencias de pensamiento computacional y en la comprensión crítica de conceptos de Educación Ciudadana, mediante análisis estadístico de datos cuantitativos obtenidos antes y después de la intervención.

Este documento presenta, en primer lugar, una revisión bibliográfica que contextualiza el estado del arte respecto a los métodos empleados para enseñar pensamiento computacional y contenidos humanísticos a jóvenes, así como las herramientas disponibles para su medición. A continuación, se describe la metodología utilizada en el diseño y ejecución de la intervención educativa, incluyendo las herramientas tecnológicas aplicadas, los criterios de evaluación y la estructura de las actividades. Posteriormente, se detalla la implementación del experimento en el contexto escolar, y se analizan los resultados obtenidos a partir de instrumentos cuantitativos y estadísticos. Finalmente, se presentan las conclusiones del estudio, reflexionando sobre el cumplimiento de los objetivos y el impacto de la intervención, junto con recomendaciones para futuras investigaciones y aplicaciones pedagógicas.

2. Marco Teórico

Esta sección tiene como objetivo entregar una fundamentación teórica a las metodologías y herramientas que serán utilizadas en el desarrollo del proyecto.

2.1. Metodologías de enseñanza de pensamiento computacional

El pensamiento computacional (CT, del inglés Computational Thinking) y su enseñanza han sido temas de investigación sumamente activos en los últimos años. Wang et al. (2022)[8] realizaron una revisión bibliográfica en la que examinaron 55 estudios publicados entre 2009 y 2020, con un aumento significativo de investigaciones entre 2015 y 2020. Estos estudios son de carácter empírico y abordan la integración del CT en la educación STEM. A partir de estos estudios, se destacan dos enfoques principales, ambos exitosos, para la enseñanza del pensamiento computacional: los métodos conectados (*plugged*) y los métodos desconectados (*unplugged*). Cabe destacar que estos enfoques no son excluyentes, y recientemente se han obtenido excelentes resultados combinando ambos métodos [13, 6].

Los métodos de enseñanza conectados se basan en el uso de la programación y las computadoras para enseñar los conceptos de pensamiento computacional. Entre los métodos más comunes en esta categoría se incluyen:

- Instrucción Basada en Problemas (*Problem-Based Instruction*, PBI): En este enfoque, los estudiantes resuelven problemas complejos que requieren el uso de CT. La PBI fomenta la aplicación de conceptos en situaciones del mundo real, tanto en áreas de programación como en matemáticas [14] y ciencias.

- Robótica: El uso de kits como LEGO EV3 permite a los estudiantes construir y programar robots que realizan tareas específicas. Este enfoque combina ingeniería, tecnología y programación, permitiendo a los estudiantes visualizar de manera tangible la aplicación del pensamiento computacional a través de la creación y control de robots [15].
- Modelado Computacional: Este método implica que los estudiantes creen y manipulen modelos que simulan fenómenos científicos. A través del modelado computacional, desarrollan habilidades de CT al diseñar, probar y mejorar modelos que representan sistemas complejos [16].
- Diseño de videojuegos: Los estudiantes aplican el CT creando y programando videojuegos. Este método combina la programación con el diseño creativo y la resolución de problemas. Mediante la creación de videojuegos, los estudiantes no solo aprenden principios de CT, sino también conceptos STEM, ya que los videojuegos a menudo se basan en simulaciones y modelos [17, 18, 19].

Entre los métodos revisados, este proyecto optará por el diseño de videojuegos como el enfoque principal, dado que es ampliamente reconocido como una de las herramientas conectadas más utilizadas y efectivas para desarrollar habilidades computacionales en los estudiantes [8]. Este método es altamente eficaz porque capta rápidamente la atención de los alumnos [20], proporciona retroalimentación instantánea de manera visual e intuitiva [21], y además permite a los estudiantes practicar programación y aprender conceptos STEM simultáneamente [17, 18, 19].

Es importante destacar la experiencia de Hromkovic (2016, 2017) [22, 23], quien subraya que, al enseñar pensamiento computacional de forma conectada, es fundamental utilizar una metodología de aprendizaje natural que integre los conceptos con fenómenos del mundo real. Esto facilita la asimilación orgánica de los contenidos y asegura que los estudiantes comprendan los principios subyacentes del pensamiento algorítmico y su aplicación.

El estudio de Wu (2018) [19] destacó el uso de Microsoft Kodu, una herramienta de creación de videojuegos basada en visual scripting, como particularmente efectiva para enseñar pensamiento computacional a estudiantes de secundaria. El visual scripting es una técnica de programación visual que permite a los estudiantes desarrollar algoritmos y diseñar juegos sin necesidad de escribir código, lo que lo hace más

accesible para principiantes. Esta metodología simplifica la programación, presentando las instrucciones en forma de bloques visuales, lo que permite a los estudiantes enfocarse en la lógica detrás de los algoritmos en lugar de la sintaxis. Wu observó que este enfoque no solo facilitó la enseñanza de conceptos clave como la descomposición de problemas y el reconocimiento de patrones, sino que también permitió a los estudiantes diseñar y experimentar con juegos más complejos, desarrollando habilidades de pensamiento algorítmico de manera intuitiva y entretenida.

Además, Repenning (2010)[12] proporciona una checklist para seleccionar una herramienta adecuada que permita enseñar competencias computacionales mediante el diseño de videojuegos de manera efectiva. Entre los criterios que destaca, se encuentra la necesidad de que la herramienta tenga una baja barrera de entrada, permitiendo que los estudiantes puedan crear un juego funcional rápidamente. Al mismo tiempo, la herramienta debe ofrecer un alto techo de posibilidades, incentivando la creatividad y permitiendo que los estudiantes avancen hacia proyectos más complejos a medida que adquieren más habilidades.

Considerando estos factores, se ha decidido que en este proyecto la enseñanza conectada se llevará a cabo utilizando el motor de videojuegos Unity3D, junto con sus herramientas de visual scripting, lo que facilitará tanto la comprensión de conceptos computacionales como la creación de videojuegos por parte de los estudiantes.

Los métodos de enseñanza no conectados son un enfoque que utiliza juegos, acertijos y actividades físicas para enseñar conceptos de pensamiento computacional sin necesidad de computadoras. Aunque no involucran programación directa, estas actividades fomentan habilidades clave como la descomposición de problemas y la identificación de patrones. Estos métodos han demostrado ser efectivos en la enseñanza de CT y han sido ampliamente utilizados en investigaciones, con al menos 55 estudios publicados entre 2010 y 2019 [24]. Su éxito y efectividad se atribuyen a sus características interactivas y flexibles, como se observa en estudios como el de Tim Bell (2009)[25], que resaltan su valor para la enseñanza de CT a principiantes.

Estas actividades son ideales para introducir a los principiantes al pensamiento computacional, ya que los algoritmos escritos pueden ser muy abstractos y presentar una barrera significativa para los novatos, al involucrar conceptos difíciles de comprender de inmediato [26]. Por el contrario, los métodos desconectados permiten que los estudiantes interactúen con los conceptos de manera tangible y comprensible antes de abordar problemas más complejos.

Una plataforma destacada que utiliza este enfoque es CS Unplugged ¹, un programa educativo gratuito que ofrece una variedad de actividades para enseñar CT a estudiantes de todas las edades. Estas actividades van desde juegos de papel y lápiz hasta dinámicas de grupo que simulan procesos computacionales.

Entre los ejemplos más comunes de actividades desconectadas se encuentran:

- Juego de instrucciones secuenciales: En este ejercicio, un estudiante o grupo de estudiantes deben seguir una serie de instrucciones predefinidas, simulando el comportamiento de una computadora que ejecuta un algoritmo. Un ejemplo de esta actividad podría ser guiar a un compañero para cruzar una sala, evitando obstáculos mediante comandos específicos como “izquierda”, “derecha” o “avanzar”. [25, 27]
- Descomposición de tareas complejas: Otra actividad popular consiste en que los estudiantes descompongan una tarea compleja, como cocinar una receta o armar un rompecabezas, en pasos más pequeños y secuenciales. Esta actividad enseña la importancia de la descomposición en el pensamiento algorítmico. [27]

2.2. Enfoques en la Enseñanza Humanista

En la enseñanza humanista existen dos grandes enfoques metodológicos: los métodos tradicionales y los métodos constructivistas o activos [28].

Los métodos de aprendizaje tradicionales se centran en la transmisión de conocimiento desde el profesor hacia el estudiante mediante técnicas como la lectura expositiva, las explicaciones directas y la memorización. En este enfoque, el docente es la figura principal y actúa como la fuente de conocimiento, mientras que el estudiante desempeña un rol pasivo, recibiendo y reteniendo la información para luego reproducirla en exámenes o evaluaciones. Los métodos tradicionales enfatizan la organización estructurada de contenidos y el aprendizaje lineal y acumulativo.

Estos métodos han sido predominantes en la educación formal durante siglos, ya que permiten enseñar grandes cantidades de información a muchos estudiantes de manera organizada y sistemática. Entre los principales autores que contribuyeron a los principios del enfoque tradicional están:

¹www.csunplugged.org

- *John Locke* (1693) [29], quien postulaba que los individuos nacen sin conocimientos innatos y que la mente es como una “*tabla rasa*” que se va llenando a través de la experiencia. Esta idea fundamentó el enfoque tradicional, donde el maestro “deposita” conocimiento en los estudiantes.
- *Johann Friedrich Herbart* (1806) [30], quien aportó una estructura pedagógica que establece principios para la enseñanza secuencial, el rol autoritario del docente y la importancia de la disciplina y la moral en el aprendizaje. Estos principios hicieron de la educación una actividad estructurada y planificada, guiada por el docente.
- *Edward Thorndike* (1906) [31], quien aplicó fundamentos conductistas psicológicos al enfoque educativo tradicional, promoviendo el uso de la práctica repetitiva, el refuerzo positivo y la evaluación estandarizada como técnicas clave para el aprendizaje.

El aprendizaje activo o constructivista es un enfoque pedagógico que busca involucrar a los estudiantes de manera activa en su proceso de aprendizaje; estos métodos causan una mayor participación y motivación dentro de los estudiantes, además de generar un ambiente de aprendizaje dinámico y atractivo, lo cual aumenta la motivación de los estudiantes y le da una mayor vitalidad al contenido [32].

En el ámbito de la Educación Ciudadana, también se han explorado metodologías activas que abordan riesgos concretos para la democracia, como la manipulación de la información. En este sentido, McCarthy y Anderson [28] realizaron un experimento en cursos universitarios de ciencia política, en el que los estudiantes participaron en una actividad colaborativa orientada a identificar sesgos en encuestas de opinión pública. Los estudiantes trabajaron en parejas, analizando versiones distintas de preguntas extraídas de encuestas reales. Estas preguntas habían sido formuladas con diferentes niveles de carga emocional o con términos tendenciosos, lo que permitía observar cómo pequeños cambios en el lenguaje podían influir considerablemente en los resultados.

El grupo experimental, que realizó esta actividad de análisis comparativo, fue contrastado con un grupo de control que solo recibió instrucción expositiva sobre el tema. Los resultados mostraron que los estudiantes que participaron activamente en la actividad no solo comprendieron mejor los principios del diseño de encuestas, sino

que también desarrollaron una mayor capacidad crítica frente a los mecanismos de manipulación de la opinión pública. Su desempeño fue significativamente superior en una prueba posterior que evaluaba la identificación de sesgos y la reflexión sobre el impacto de la desinformación en contextos democráticos.

2.3. Métodos de evaluación

Para este estudio, es necesario utilizar herramientas que permitan medir tanto las habilidades de pensamiento computacional de los estudiantes como sus conocimientos en el ámbito de la Educación Ciudadana. En relación con esta última, la medición en el contexto escolar chileno contempla evaluaciones regulares en esta asignatura.

La evaluación de las competencias en pensamiento computacional presenta un desafío debido a la falta de estándares consistentes y a la multidimensionalidad de estas habilidades. La complejidad del pensamiento computacional, junto con la variedad de habilidades que abarca y los distintos enfoques pedagógicos, hace que la creación de un método de evaluación universalmente aplicable sea difícil.

Tang et al. (2020) [33] revisaron 96 estudios empíricos sobre métodos de evaluación de pensamiento computacional, identificando cuatro enfoques principales:

1. **Pruebas Tradicionales:** Incluyen preguntas de opción múltiple y abiertas que evalúan conceptos como secuencias, algoritmos y lógica. Son fáciles de implementar y permiten comparaciones entre estudiantes, pero tienden a ser superficiales y no capturan el uso práctico de las habilidades. Grover y Pea (2013) [34] emplearon este método para evaluar conceptos básicos de programación, aunque señalaron que la profundidad de la evaluación era limitada.
2. **Evaluación de Portafolios:** Los estudiantes presentan proyectos en plataformas como Scratch o robótica, evaluados mediante rúbricas. Este método permite una evaluación más detallada de la aplicación práctica del pensamiento computacional, aunque requiere mucho tiempo y puede ser subjetivo. Brennan y Resnick (2012) [35] utilizaron portafolios en Scratch para analizar habilidades como la abstracción y la modularización.
3. **Encuestas:** A través de escalas de Likert, se mide la percepción de los estudiantes sobre su comprensión y actitudes hacia el pensamiento computacional.

Este método permite evaluar aspectos afectivos, pero depende de la autopercepción, lo cual puede no reflejar la habilidad real. Fessakis et al. (2013) [36] emplearon este método para evaluar la motivación de los estudiantes tras un proyecto en robótica.

4. **Entrevistas y Métodos Cualitativos:** Las entrevistas (como el método “think-aloud”) permiten a los estudiantes explicar sus procesos de pensamiento y estrategias en tiempo real. Lye y Koh (2014) [37] emplearon entrevistas para estudiar estrategias de resolución de problemas algorítmicos. Aunque detalladas, son difíciles de aplicar en grupos grandes y pueden ser subjetivas.

La revisión de Tang et al. (2020) [33] sugiere que una combinación de métodos es la estrategia más efectiva para capturar las diversas dimensiones del pensamiento computacional. Por esta razón, el presente proyecto empleará una metodología de evaluación dual que combina una evaluación tradicional con un análisis práctico a través de proyectos.

3. Metodología

La metodología de este proyecto consistió en una intervención educativa de cinco sesiones en un recinto educacional, en el contexto de la asignatura de Educación Ciudadana.

El primer paso de esta metodología fue establecer contacto con el establecimiento educacional. Este debía contar con acceso a un laboratorio de computación y otorgar el permiso para realizar las intervenciones con un curso de educación media (jóvenes de entre 14 y 18 años). Luego, se coordinó la intervención con el docente encargado de la asignatura humanista en la cual se realizó la intervención, con el objetivo de comprender el contexto, las necesidades y preferencias de los estudiantes del curso, así como familiarizarse con los contenidos de la materia que se relacionaban con la propuesta.

Dado que esta metodología involucró la participación de menores de edad, fue imperativo obtener el **asentimiento informado** de los apoderados. Este documento formalizó la autorización para que sus hijos participaran en las actividades de la intervención. En el caso de participantes mayores de 18 años, se requirió su **consentimiento informado** directo, lo que les permitió aprobar formalmente su participación en las tareas de esta intervención.

Las sesiones de la intervención fueron cinco y tuvieron una duración estipulada de 90 minutos. Dada la efectividad de los enfoques no conectados para introducir el pensamiento computacional, este trabajo utilizó métodos de enseñanza desconectados en las dos primeras sesiones. De esta forma, se permitió a los estudiantes comprender la lógica detrás de los programas computacionales, antes de aplicar estos conceptos en las tres sesiones siguientes en un ambiente conectado. A continuación se presenta un resumen de cómo estuvieron estructuradas las intervenciones:

- **Sesión 1:**

Se presentó el proyecto, se aplicó una prueba diagnóstica de pensamiento computacional y se introdujo teóricamente el concepto de pensamiento computacional, relacionándolo con el contexto de la asignatura.

- **Sesión 2:**

Se realizaron actividades desconectadas de pensamiento computacional. Fue importante vincular estas actividades tanto al contexto de la asignatura como al contexto personal de los estudiantes.

- **Sesión 3:**

Se llevó a cabo la primera sesión utilizando computadores y Unity 3D. Se dio tiempo a los estudiantes para familiarizarse con el entorno computacional, y posteriormente se les planteó un desafío de baja dificultad dentro del programa.

- **Sesión 4:**

Se desarrolló una sesión centrada en la resolución de problemas y en tareas más complejas mediante el uso de Visual Scripting en Unity. En esta sesión se trabajaron principalmente las habilidades de lógica y depuración.

- **Sesión 5:**

Se realizó la última sesión conectada, en la cual se propuso una tarea compuesta por múltiples elementos del pensamiento computacional. Finalmente, se aplicó una evaluación postintervención y se llevó a cabo el cierre de la intervención.

Las sesiones que se realizaron en computadoras utilizaron **escenas preconfiguradas** en Unity 3D. Estas escenas fueron diseñadas con *scripts*, modelos 3D o *sprites* 2D previamente integrados, contando con funcionalidad básica y una estética definida. El objetivo fue que los estudiantes pudieran comenzar a trabajar de forma inmediata, sin requerir configuraciones adicionales, facilitando así el enfoque en las tareas de *Visual Scripting* y en la exploración del entorno, de manera guiada e intuitiva. Una descripción detallada de estas escenas se encuentra en el capítulo Diseño de Escenas.

3.1. Medición y análisis de resultados

Este estudio buscó medir el progreso de los estudiantes tanto en el desarrollo del pensamiento computacional como en sus competencias en Educación Ciudadana. Para obtener resultados más robustos, se utilizó un diseño comparativo con dos grupos: el **grupo experimental**, compuesto por estudiantes que participaron en la intervención educativa; y el **grupo control**, correspondiente a un curso del mismo nivel que no recibió la intervención, pero que fue evaluado en los mismos términos.

La comparación entre ambos grupos permitió establecer con mayor precisión el impacto específico de la intervención en los aprendizajes de los estudiantes.

La evaluación de competencias en Educación Ciudadana se realizó mediante las evaluaciones regulares de la asignatura, previamente planificadas por el establecimiento. Se tomó una evaluación antes del inicio de la intervención y otra al finalizarla. Mientras el grupo experimental utilizó las horas de clase de Educación Ciudadana para participar en las sesiones diseñadas en este proyecto, el grupo control continuó con sus clases regulares. Esta estrategia permitió comparar directamente la efectividad del enfoque tradicional versus la intervención basada en pensamiento computacional.

Para evaluar el pensamiento computacional, se aplicaron dos cuestionarios breves, uno antes y otro después de la intervención, a ambos grupos, diseñados para mantener un nivel de dificultad equivalente. De esta forma, se aseguró la validez de la comparación entre los resultados, ya que el periodo entre la aplicación de ambos cuestionarios fue idéntico para ambos grupos.

Cada cuestionario incluyó cinco preguntas, una por cada uno de los fundamentos clave del pensamiento computacional: pensamiento lógico, identificación de problemas, abstracción, algoritmos básicos y depuración de procesos. Las preguntas fueron diseñadas para mantenerse fuera del ámbito técnico de la programación, lo que permitió medir estas habilidades de forma más transversal, enfocándose en el razonamiento y la lógica subyacente.

Los cuestionarios pueden revisarse en el Apéndice A.

3.1.1. Script para medir el éxito en los experimentos

Con el objetivo de evaluar en tiempo real la efectividad, dificultad y tasa de éxito de las escenas preconfiguradas, se programó un *script* que registraba automáticamente si los estudiantes lograban cumplir con el objetivo de cada escena y cuánto

tiempo tardaron en completarla.

La tasa de éxito de las tareas proporcionó información clave para determinar si la dificultad de los desafíos era adecuada. Por ejemplo, una tasa de éxito muy baja podría indicar que la actividad es demasiado compleja, mientras que una tasa demasiado alta acompañada de tiempos de resolución muy cortos podría evidenciar que el desafío es demasiado sencillo.

Esta métrica resultó fundamental para ajustar y perfeccionar las escenas preconfiguradas, y convertirlas en herramientas educativas efectivas y adaptadas al nivel de los estudiantes. Además, los datos registrados por este *script* permitieron extraer conclusiones relevantes sobre el diseño de las actividades y plantear posibles mejoras para futuras implementaciones.

3.1.2. Análisis estadístico

Para esta investigación se empleó un análisis estadístico basado en la prueba t de Student, dado que se trata de una herramienta ampliamente reconocida por su eficacia en la comparación de medias entre dos grupos. Esta prueba resulta particularmente útil en diseños experimentales orientados a detectar diferencias significativas entre grupos de control y grupos experimentales [38].

El objetivo del análisis fue determinar si existieron diferencias estadísticamente significativas en los conocimientos de pensamiento computacional y en los conocimientos de educación ciudadana entre los grupos estudiados. La prueba t entrega un valor p, el cual simboliza el porcentaje de riesgo de arrojar un falso positivo. Siguiendo el criterio propuesto por Fisher, se consideró que una diferencia fue estadísticamente significativa si el valor p fue inferior a 0.05 [39].

Dado que la hipótesis de investigación planteó explícitamente que la intervención educativa generaría una mejora en los aprendizajes del grupo experimental respecto del grupo control, se utilizó una prueba t **unidireccional**. Esta elección se fundamentó en que una prueba direccional permite aumentar la potencia estadística al enfocarse exclusivamente en detectar efectos en la dirección planteada en la hipótesis, es decir, una mejora en el grupo experimental.

El procedimiento estadístico se desarrolló en dos etapas. En primer lugar, se aplicó la prueba t para muestras relacionadas (pareadas) dentro de cada grupo, comparando los resultados obtenidos antes y después de la intervención. Esto permitió evaluar si,

al interior de cada grupo, se produjeron mejoras estadísticamente significativas tras la implementación de la intervención.

Posteriormente, se calcularon las diferencias individuales (deltas) entre los puntajes pre y post intervención de cada estudiante. Con estos datos, se realizó una prueba t para muestras independientes con varianzas desiguales (Welch's t-test), ya que no se asumió homogeneidad de varianzas entre los grupos y esta variante del test ofrece mayor robustez en condiciones de varianza desigual o tamaños de muestra distintos [40]. Esta comparación permitió determinar si la magnitud del cambio en el grupo experimental fue significativamente mayor que en el grupo control, reforzando la validez de los efectos observados como producto de la intervención.

4. Intervención educativa

4.1. Contexto y Selección de la Muestra

La intervención educativa fue inicialmente planificada para ser implementada en un colegio particular pagado de enfoque científico-humanista y con mayores recursos económicos. Sin embargo, tras varias reuniones y una colaboración inicial prometedora, la participación fue inesperadamente rechazada debido a un cambio imprevisto en la disponibilidad del profesor.

Como resultado, el estudio se reorientó hacia el Liceo Javiera Carrera de Talca, un establecimiento público técnico-profesional. Este liceo cuenta con especializaciones en Mecánica Automotriz, Climatización y Administración. Este ajuste en el perfil del centro educativo y de la muestra estudiantil, con diferencias en sus características socioeconómicas y académicas del plan original, se considerará para la interpretación de los hallazgos y será debidamente contextualizado en el análisis de resultados.

En el Liceo Javiera Carrera de Talca, se consultó sobre la disponibilidad de cursos de educación media para la intervención, asignándose dos grupos de tercer año medio, con estudiantes de entre 16 y 18 años. El Tercero Medio A, especializado en Mecánica Automotriz, fue designado como el grupo experimental, principalmente debido a su disponibilidad horaria. Por su parte, el Tercero Medio B, con especialización en Climatización, fue designado como el grupo control.

Es crucial destacar un sesgo inherente a esta selección de muestra: la diferencia en la especialización técnica de ambos grupos. El grupo experimental se especializa en Mecánica Automotriz, mientras que el grupo control lo hace en Climatización. Esta distinción podría influir en las habilidades o conocimientos previos de los estudiantes de maneras que el estudio no puede controlar directamente.

Adicionalmente, hay que contextualizar que los estudiantes de este liceo provienen de un entorno socioeconómico vulnerable, lo que es un factor relevante a considerar al interpretar los resultados de la intervención.

4.2. Aplicación de la intervención

La aplicación de la intervención se detalla a continuación de forma cronológica, sesión por sesión, para facilitar la comprensión de su desarrollo.

4.2.1. Preparación previa a las sesiones

Antes de la primera sesión, fue fundamental establecer contacto y reunirse con el profesor a cargo de la asignatura de Educación Ciudadana. En estas reuniones, el profesor compartió los objetivos y el nivel de la asignatura, así como sus percepciones sobre las actitudes y preferencias de los estudiantes. Se señaló que el grupo experimental era más inquieto y participativo que el grupo control, mostrando gran interés en temas cotidianos como las redes sociales o problemáticas locales, como la delincuencia.

Durante esta etapa de preparación, también se llevó a cabo una primera interacción con los estudiantes, donde se presentó brevemente el proyecto y se entregaron los documentos de asentimiento o consentimiento informado, según la edad de cada participante.

4.2.2. Sesión 1: Introducción al Pensamiento Computacional

Objetivo: Presentar el proyecto, evaluar el nivel inicial de los estudiantes e introducir los conceptos de pensamiento computacional en un contexto cotidiano, vinculándolos con problemáticas que afectan a la democracia, tal como lo aborda la asignatura de Educación Ciudadana.

- **(10 min) Presentación:** Saludo y presentación personal, incluyendo experiencia y antecedentes. Introducción al proyecto, con énfasis en su objetivo, duración y las actividades a desarrollar.
- **(30 min) Evaluación diagnóstica:** Aplicación de un cuestionario (ver A.1 cuestionario 1) de selección múltiple para evaluar habilidades iniciales de pensamiento computacional.

- **(35 min) Introducción al pensamiento computacional:** Clase teórica introductoria sobre pensamiento computacional, destacando su utilidad como herramienta para analizar y resolver problemas. Se presentan los cuatro componentes clave: descomposición, abstracción, pensamiento algorítmico y depuración, con ejemplos aplicados a situaciones cotidianas y cívicas.



Figura 4.1: Clases teóricas sesión 1

- **(15 min) Cierre:** Discusión grupal sobre la relevancia del pensamiento computacional en el contexto actual de los estudiantes, y preguntas abiertas sobre situaciones que creen que podrían aplicar estas habilidades.

Comentarios: Por más que esta sesión fuera totalmente teórica, los estudiantes se mostraron muy receptivos y participativos hacia la actividad.

Sesión 2: Actividades Desconectadas

Objetivo: Desarrollar habilidades de pensamiento computacional mediante actividades prácticas sin uso de computadores.

- **(10 min) Revisión rápida de los componentes de pensamiento computacional:** Repaso de los conceptos de descomposición, abstracción y pensamiento algorítmico.

- **(35 min) Actividad de descomposición:** Los estudiantes, divididos en grupos, reciben una tarea cotidiana compleja (como “preparar una fiesta de completos” o “preparar un asado”) y deben desglosarla en pasos manejables. Se discuten y afinan los pasos con el encargado, identificando áreas de mejora.
- **(35 min) Actividad de pensamiento algorítmico:** Se crea un “laberinto” de mesas y sillas en la sala de clases y cada grupo debe crear un “algoritmo” en papel para guiar a alguien a través de la sala, usando comandos específicos como “girar a la izquierda”, “girar a la derecha” o “avanzar”. Se realiza una demostración para observar claridad y eficiencia de los algoritmos.



Figura 4.2: Estudiantes creando algoritmos en papel

- **(10 min) Reflexión y cierre:** Discusión sobre cómo las habilidades trabajadas ayudan a pensar de forma más estructurada en contextos cotidianos.

Comentarios: La primera actividad contó con una excelente participación de los estudiantes, demostrando una descomposición muy rigurosa de la tarea presentada. Sin embargo, la actividad del laberinto generó menos entusiasmo y participación. Los estudiantes mostraron cierto rechazo a crear el laberinto, aparentemente debido al esfuerzo físico que implicaba.

Para las sesiones que requerían el uso de computadores, fue necesario coordinar con el encargado de la sala de computación con al menos un día de anticipación. Durante esta instancia previa, se realizó la instalación de Unity Engine y Git, además

de clonar un repositorio previamente preparado que contenía las escenas preconfiguradas necesarias para el trabajo con los estudiantes.

Aunque el repositorio original fue modificado progresivamente a lo largo de las sesiones, para los fines de este informe se creó una nueva versión que recopila todas las escenas preconfiguradas finales. Este repositorio final se encuentra disponible en el siguiente enlace: <https://github.com/JavierMahana/JavierMahanaMemoriaTitulo>

Sesión 3: Introducción a Unity y Visual Scripting

Objetivo: Introducir a los estudiantes en la interfaz de Unity y Visual Scripting, enseñándoles a implementar secuencias básicas de acciones mediante lógica visual.

- **(45 min) Exploración de la interfaz de Unity y componentes básicos:** Presentación del inspector, la jerarquía y las diferencias entre panel de escena y vista de juego. Demostración de manipulación de cámara y objetos en escena. A los estudiantes se les entrega la escena preconfigurada 1 (ver 5.2 Introducción a Unity), y deberán crear su propio nivel, moviendo, escalando y rotando objetos para observar su impacto en la vista de juego. Aquí se podrán assets y un player controller, ya configurado. Así pueden llegar y probar sus escenas.
- **(45 min) Introducción a Visual Scripting en una Escena Interactiva:** Introducción al componente Visual Scripting (o “Script Graph”) en un objeto de la escena, explicando nodos básicos para implementar movimiento, escala y rotación. Los estudiantes aplican lo aprendido en un ejercicio donde deben implementar secuencias de movimiento, escala y rotación en la escena preconfigurada 2 (ver 5.2 Entrega de Solicitud Estudiantil).

Comentarios: Los estudiantes disfrutaron especialmente la primera escena, donde tuvieron libertad para experimentar y crear un nivel en Unity. Sin embargo, la segunda escena resultó frustrante y difícil de resolver para la mayoría del grupo. Solo un estudiante logró completar el desafío, lo que se atribuye principalmente a dos factores: en primer lugar, la barrera del idioma. La interfaz de Unity y el sistema de Visual Scripting está completamente en inglés, lo que dificultó la comprensión intuitiva y requirió apoyo constante por parte del investigador para explicar el significado y uso de cada elemento. En segundo lugar, la dificultad del desafío era relativamente alta, lo que incrementó la sensación de frustración.

Modificaciones y inconvenientes

A raíz de las dificultades observadas en la sesión anterior con la escena preconfigurada, se implementaron cambios importantes en el grafo de Visual Scripting para las siguientes sesiones. Se añadieron traducciones y explicaciones en español para diversos nodos, con el objetivo de facilitar la comprensión de su funcionalidad por parte de los estudiantes. Además, se redujo la dificultad de los desafíos, entregando previamente todos los nodos necesarios para la actividad, de modo que los estudiantes solo debían conectarlos correctamente y asignar los valores correspondientes.

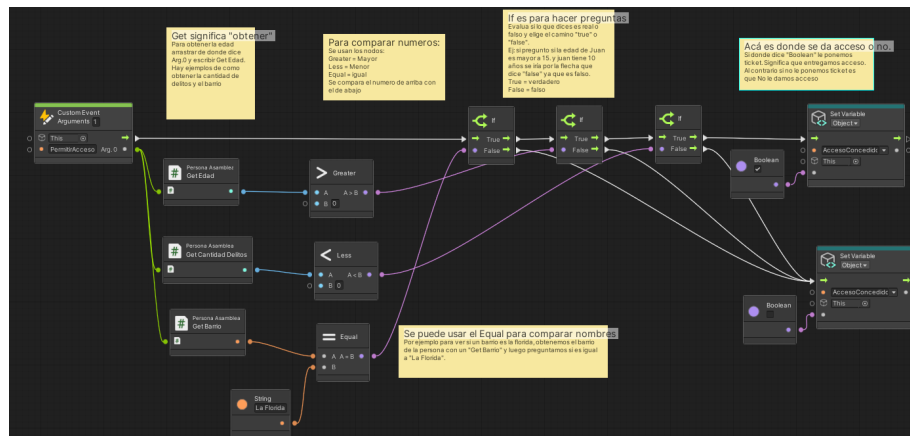


Figura 4.3: Grafo de visual scripting con traducciones

Por otro lado, se nos informó que durante la cuarta sesión solo se dispondría de 45 minutos. Ante esta limitación, se decidió eliminar la aplicación de una de las escenas. Se optó por omitir la escena preconfigurada 4, ya que sus actividades eran muy similares a las de otras escenas, por lo que su exclusión tendría un impacto mínimo en el desarrollo general de la intervención.

Sesión 4: Condicionales en Visual Scripting

Objetivo: Profundizar en el uso de Visual Scripting en Unity, abordando tareas más complejas como la lógica condicional.

- **(10 min) Repaso teórico:** Repaso de los nodos básicos vistos en la sesión anterior y explicación de la lógica computacional mediante los nodos IF.

- **(35 min) Actividad de Lógica Avanzada:**

Práctica de lógica condicional en la escena preconfigurada 3 (ver 5.2 Control de Acceso a Asamblea Vecinal Digital).



Figura 4.4: Estudiantes realizando la sesión 4

Comentarios: Las traducciones y la reducción de la complejidad resultaron ser modificaciones acertadas, ya que esta escena presentó una mayor tasa de éxito y satisfacción por parte de los estudiantes. Sin embargo, a pesar de que las instrucciones fueron explicadas al inicio de la actividad, la mayoría de los estudiantes volvió a consultarlas durante su desarrollo.

Sesión 5: Evaluación Final y Proyecto de Integración

Objetivo: Evaluar el aprendizaje y permitir a los estudiantes aplicar pensamiento computacional y Visual Scripting.

Debido a que en la sesión anterior hubo muchas dudas de parte de los estudiantes se decidió escribir en la pizarra las instrucciones y las preguntas frecuentes, de modo que los estudiantes puedan resolver sus dudas de forma autónoma al leerlas.

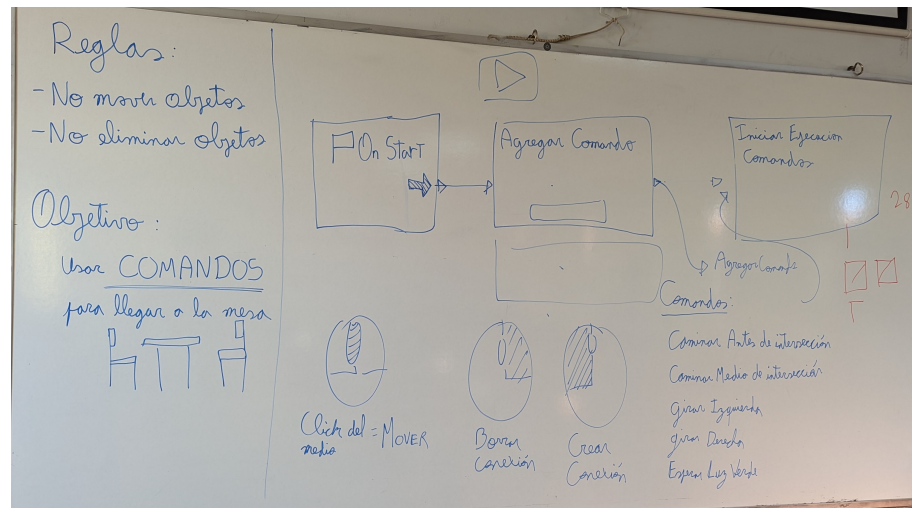


Figura 4.5: Instrucciones y reglas de la escena

- **(10 min) Explicación instrucciones y reglas de la escena**
- **(35 min) Proyecto integrador en Unity:** Los estudiantes aplican lo aprendido en un pequeño proyecto dentro de la **Escena 5: Laberinto de la Desinformación** (ver 5.3), donde deberán programar una secuencia de comandos para que un personaje logre llegar a la meta en un escenario lleno de señalizaciones con mentiras o falsas promesas. Se les desafía a utilizar la navegación dentro del motor de Unity para encontrar el lugar real de la meta, lógica condicional, secuencias ordenadas y depuración de errores, como analogía del pensamiento crítico frente a la participación informada.
- **(30 min) Evaluación final:** Aplicación de una evaluación de 30 minutos, similar a la inicial (ver A.2 cuestionario 2), para medir progreso del pensamiento computacional.
- **(15 min) Reflexión final:** Discusión grupal sobre la experiencia, recopilación de opiniones de parte de los estudiantes y preguntas sobre el rol del pensamiento computacional en la vida cívica. Los estudiantes comparten conclusiones sobre el posible impacto que esta intervención pueda tener en su vida y comentarios de que les parecieron las actividades.

Si bien la complejidad del proyecto integrador era relativamente alta, se observó una elevada tasa de éxito y satisfacción por parte de los estudiantes. En la instancia

de reflexión final, los estudiantes se mostraron muy agradecidos por la actividad, destacando que conceptos que al inicio de la intervención les eran completamente desconocidos ahora les resultaban familiares. Además, valoraron positivamente la labor del investigador, señalando que respondió a sus dudas con claridad y mostró disposición para escuchar sus inquietudes.

5. Diseño de Escenas

Las escenas preconfiguradas en Unity fueron diseñadas para ofrecer a los estudiantes un entorno de aprendizaje práctico y directo, donde pueden enfocarse en la resolución de problemas específicos. Este enfoque busca integrar de manera práctica los conceptos de programación y pensamiento algorítmico con problemáticas/situaciones cívicas, ofreciendo una experiencia de aprendizaje interdisciplinaria y accesible. Estas pueden ser accedidas en este enlace <https://github.com/JavierMahana/JavierMahanaMemoriaTitulo>

5.1. Implementacion del script

El diseño de estas escenas establece condiciones claras de "éxito", las cuales son evaluadas mediante un script automatizado. Esto permite recopilar datos sobre el progreso de los estudiantes, evaluando tanto si lograron completar la actividad como el tiempo que tardaron en hacerlo.

Para la implementación técnica de este script, se utilizó una ventana de editor, dado que su uso resultaba sencillo para los estudiantes, quienes solo debían presionar un botón dentro de dicha ventana. Esta interfaz creaba un *ScriptableObject* que contenía dos variables: un valor booleano que indicaba si el desafío había sido completado, y un número flotante que registraba el tiempo transcurrido sin lograr dicha finalización. La ventaja de utilizar un *ScriptableObject* radica en la facilidad para recopilar los datos, ya que, al finalizar cada sesión, se podía revisar cada computador de manera individual y anotar los resultados mostrados en estos assets.

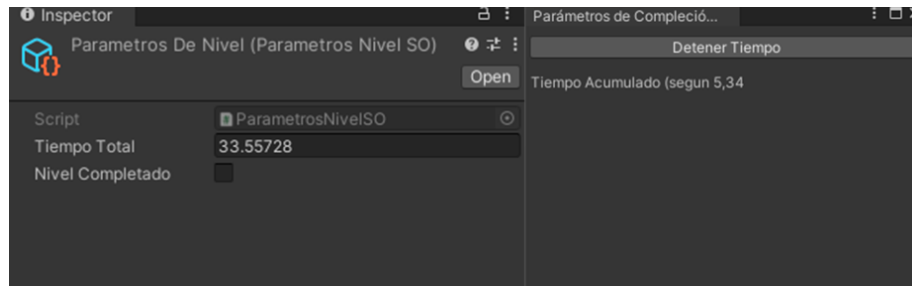


Figura 5.1: Scriptable object y ventana de editor

5.1.1. Complicaciones de la implementación

El principal problema técnico que debió resolverse durante esta implementación fue la comunicación entre los scripts del editor y los de tiempo de ejecución. Para abordar este desafío, se creó un script basado en `MonoBehaviour`, llamado `CompletarNivelBridge`, que residía dentro de la escena. Su función era monitorear en tiempo de ejecución si se cumplía la condición de éxito del nivel. Posteriormente, la ventana del editor obtenía una referencia a este objeto y consultaba si dicha condición ya se había cumplido. En caso afirmativo, los datos eran almacenados en el correspondiente *ScriptableObject*. A continuación, se presenta un fragmento con las partes más relevantes del código de la ventana del editor.

```

1 private void OnGUI(){
2     //[...] codigo que muestra el boton y maneja el tiempo
3     //parametrosData es el Scriptable Object
4     if (parametrosData != null){
5         CompletarNivelBridge completarNivelBridge =
6             FindObjectOfType<CompletarNivelBridge>();
7         if (completarNivelBridge != null && !parametrosData.
8             nivelCompletado)
9         {
10             if (completarNivelBridge.nivelCompleto){
11                 MarcarNivelComoCompletado();
12             }
13         }
14     }
15 }
16
17 public void MarcarNivelComoCompletado(){
18     if (parametrosData != null){
19         parametrosData.nivelCompletado = true;
20     }
21 }

```

```
19         EditorUtility.SetDirty(parametrosData);
20         AssetDatabase.SaveAssets();
21         AssetDatabase.Refresh();
22     }
23 }
```

Listing 5.1: Fragmento de código para la gestión del estado del nivel

5.2. Descripción escenas

Escena 1: Introducción a Unity

Esta escena se desarrolla en un escenario 3D con assets de proporciones incorrectas y posicionados sin sentido lógico.

El objetivo de esta escena es que los estudiantes se familiaricen con la interfaz de Unity y aprendan a manipular objetos y navegar por la escena, por medio de la creación de sus propios niveles utilizando los assets presentes en la escena. Debido a que esta escena se enfoca en la exploración inicial de la interfaz, esta será la única escena sin una condición de éxito.

Escena 2: Entrega de Solicitud Estudiantil

Esta escena presenta a un estudiante que recorre su liceo con el objetivo de entregar una solicitud formal al centro de estudiantes, proponiendo mejoras urgentes para el establecimiento.

En la escena, el personaje avanza automáticamente por el entorno escolar, enfrentando distintos obstáculos simbólicos que representan las dificultades cotidianas para llevar a cabo acciones de participación ciudadana. Esta actividad introduce a los estudiantes al uso de Visual Scripting y a la modificación del componente **transform** en Unity.

Durante el trayecto, se activan eventos en puntos específicos del recorrido. Cada uno de estos eventos genera un nodo en el **Script Graph**, y los estudiantes deberán programar diferentes transformaciones en el personaje para superar los obstáculos:

- En el primer evento, deberán ajustar la posición vertical del personaje para superar una acumulación de escombros.

- En el segundo evento, deberán reducir la escala del personaje para atravesar un pasillo estrecho obstruido por mobiliario.
- En el tercer y cuarto evento, deberán rotar al personaje para evitar obstáculos.

Una vez superados todos los obstáculos, el personaje llega a su destino y entrega la solicitud con éxito. Como cierre, el entorno escolar aparece transformado: limpio, iluminado y accesible, junto con el siguiente mensaje final: “Cuando participamos con convicción y propósito, incluso los espacios más olvidados pueden transformarse. No subestimes el valor de una solicitud bien hecha: es el primer paso hacia el cambio real.”

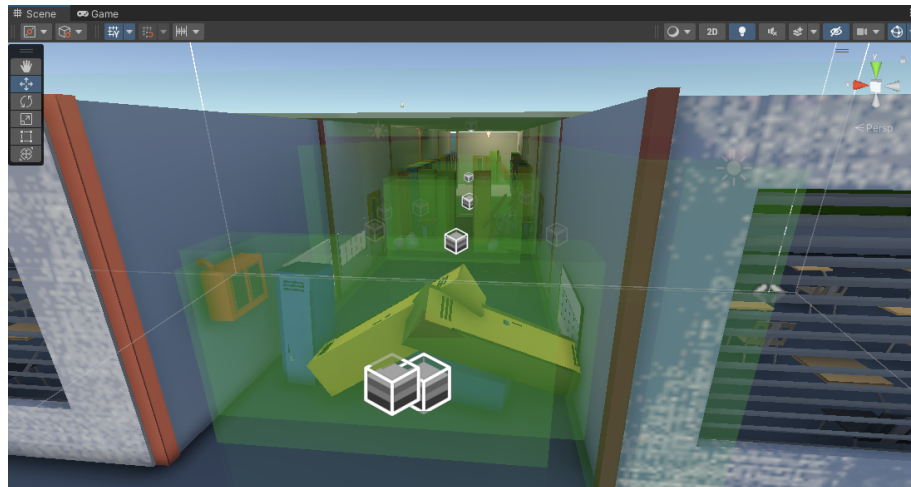


Figura 5.2: Pantallazo de escena 2 mostrando puntos de activación de eventos y escombros

Escena 3: Control de Acceso a Asamblea Vecinal Digital

Esta escena está ambientada en un centro comunitario que acoge una asamblea vecinal digital, organizada para debatir y votar propuestas ciudadanas relacionadas con la seguridad, la convivencia y el acceso a servicios públicos.

Los estudiantes deberán programar un sistema de control de acceso, aplicando lógica condicional mediante nodos IF en Visual Scripting, permitiendo el ingreso solo a quienes cumplan ciertos criterios definidos por la comunidad organizadora. Se autoriza el acceso únicamente si:

- La persona tiene más de 15 años (Operador mayor).

- Es un vecino del barrio (operador igual).
- No tiene antecedentes de comportamiento discriminatorio o violento en instancias anteriores (operador menor).

Una vez que el sistema permita ingresar únicamente a quienes cumplen con los requisitos, se cumple la condición de éxito y se muestra el siguiente mensaje: “Una comunidad democrática se construye con participación informada, reglas claras y respeto mutuo. Elegir a quién damos voz también es un acto de responsabilidad ciudadana, en especial en ambientes online.”

Escena 4: Validación Ética de Candidaturas Ciudadanas

Escena no implementada en la intervención educativa. Se puede ver su diseño en los anexos: B.

Escena 5: Programación del Recorrido hacia la Meta Ciudadana

En esta escena, los estudiantes deberán programar el comportamiento de un personaje que debe desplazarse a través de un escenario digital. Para lograrlo se cuentan con las siguientes instrucciones: como avanzar hasta antes de una intersección, caminar hasta el medio de una intersección, girar a la izquierda o a la derecha y esperar el semáforo.

Reglas del entorno:

- El escenario está lleno de señales de publicidad y anuncios engañosos que prometen soluciones fáciles y rápidas.
- El personaje solo se moverá solo con instrucciones previamente especificadas y programadas por el responsable.
- El objetivo es llegar a un lugar del escenario que será la meta.

Una vez el personaje llega a la meta se mostrará un mensaje de éxito:

“Las falsas promesas y la desinformación pueden llevarnos al vacío si no tomamos decisiones basadas en hechos y reflexión. En un mundo lleno de publicidad engañosa, el pensamiento crítico es nuestra mejor herramienta.”



Figura 5.3: Pantallazo de escena 5 mostrando publicidad engañosa

6. Resultados y análisis

6.1. Resultados

En esta sección se expondrán los resultados del experimento.

6.1.1. Resultados del Script

Escena	Total estudiantes	Estudiantes Exitosos	Minutos Promedio Éxi- to	Tasa de éxito
Escena 2	24	1	35,3	4,17 %
Escena 3	25	12	23,1	48,00 %
Escena 5	25	12	31,8	48,00 %

Tabla 6.1: Resultados de desempeño por escena

La tasa de éxito extremadamente baja en la Escena 2 sugiere dificultades iniciales en la comprensión o en el diseño de la escena. Las escenas 3 y 5 presentan tasas de éxito similares, aunque difieren en el tiempo promedio, lo que podría reflejar diferencias en la complejidad de las tareas.

6.1.2. Resultados de cuestionarios

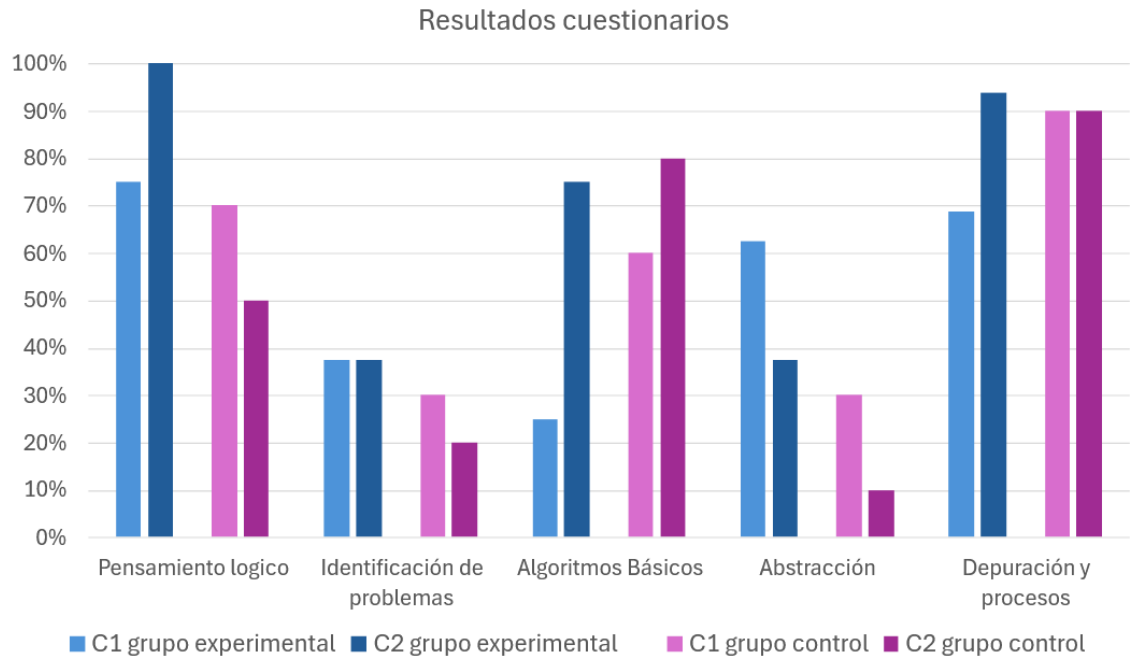


Figura 6.1: Tabla con los resultados de los cuestionarios

En el gráfico se presentan los resultados de los cuestionarios de pensamiento computacional aplicados antes (C1) y después (C2) de la intervención, tanto para el grupo experimental como para el grupo control. Se observa una mejora notable en el grupo experimental en dimensiones como **pensamiento lógico** y **algoritmos básicos**, alcanzando niveles cercanos o superiores al 90 % de respuestas correctas en la post-evaluación (C2). En contraste, el grupo control muestra niveles más bajos y estables entre C1 y C2, lo que sugiere una menor efectividad del método tradicional. Cabe destacar que en **identificación de problemas** y **abstracción**, las mejoras fueron mínimas en ambos grupos, lo cual coincide con el hecho de que estas habilidades no fueron abordadas de forma sistemática en las actividades de la intervención. La dimensión de **depuración y procesos** presenta un alto rendimiento en ambos grupos durante C2, lo que podría estar influenciado por aprendizajes compartidos u otros factores externos al diseño experimental.

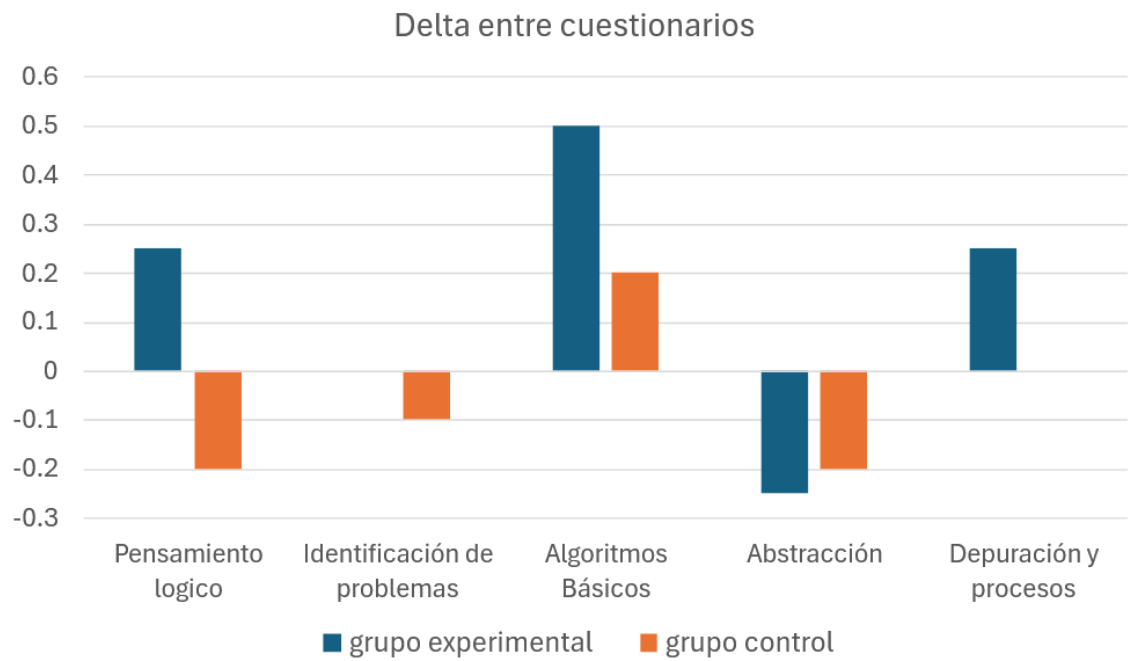


Figura 6.2: Tabla con los delta del resultado de los cuestionarios

El gráfico muestra la diferencia de puntajes (delta) entre los resultados pre y post intervención de los cuestionarios de pensamiento computacional, comparando al grupo experimental y al grupo control. Se observa que el grupo experimental presenta mejoras notorias en las habilidades de pensamiento lógico, algoritmos básicos y depuración y procesos, mientras que el grupo control muestra estancamiento o incluso retrocesos en la mayoría de las categorías.

6.1.3. Resultados de pruebas parciales

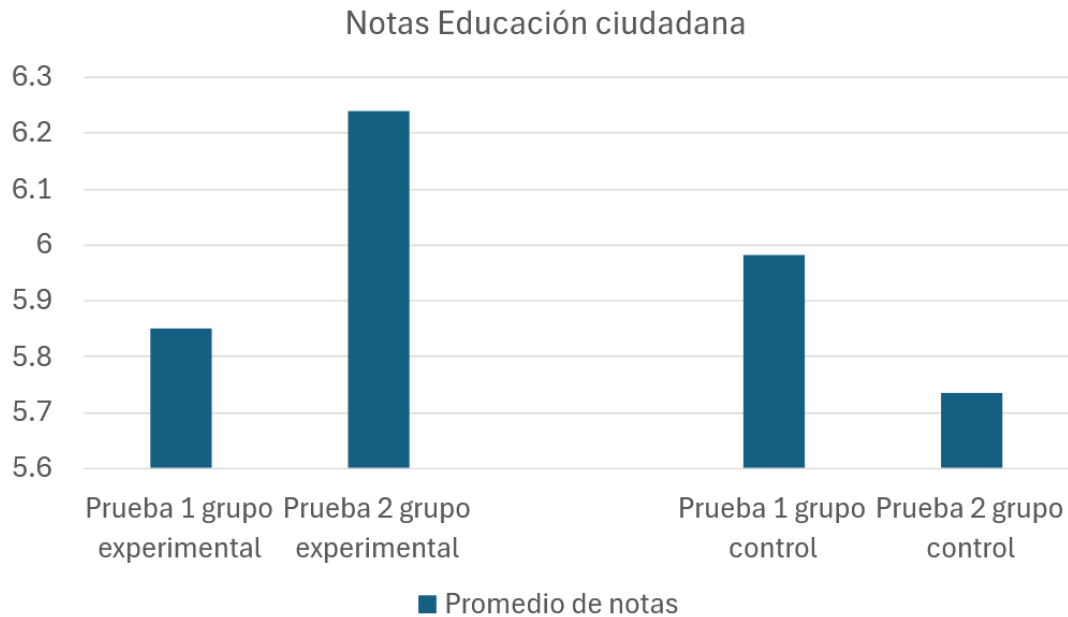


Figura 6.3: Tabla con los promedios de notas de las pruebas parciales de educación ciudadana

En el gráfico se presentan los promedios de notas obtenidas en Educación ciudadana por ambos grupos, antes y después de la intervención. Se evidencia una mejora significativa en el grupo experimental, que pasó de un promedio de 5,85 en la primera evaluación a 6,24 en la segunda. En contraste, el grupo control muestra una disminución en su desempeño, bajando de un 5,98 a un 5,73.

6.2. Análisis estadístico

Tal como se detalló en la sección de metodología, el análisis estadístico se desarrolló en dos fases. La primera consistió en evaluar si la intervención generó un impacto significativo dentro de cada grupo (experimental y control), comparando los resultados obtenidos antes y después de la intervención. La segunda fase se centró en comparar la magnitud de la mejora relativa entre ambos grupos, con el fin de determinar si los cambios observados en el grupo experimental fueron significativamente superiores a los del grupo control.

Para ambos análisis se utilizó la prueba t de Student, implementada a través de la fórmula PRUEBA.T de Excel, utilizando tanto su variante para muestras relacionadas como la de muestras independientes, según el caso.

6.2.1. Cuestionarios de pensamiento computacional

En primer lugar, se aplicó una prueba t para muestras relacionadas (pareadas) dentro de cada grupo, contrastando los puntajes obtenidos en los cuestionarios de pensamiento computacional antes y después de la intervención. Este análisis permite evaluar si existieron cambios estadísticamente significativos en las respuestas de los mismos participantes tras la implementación del programa educativo.

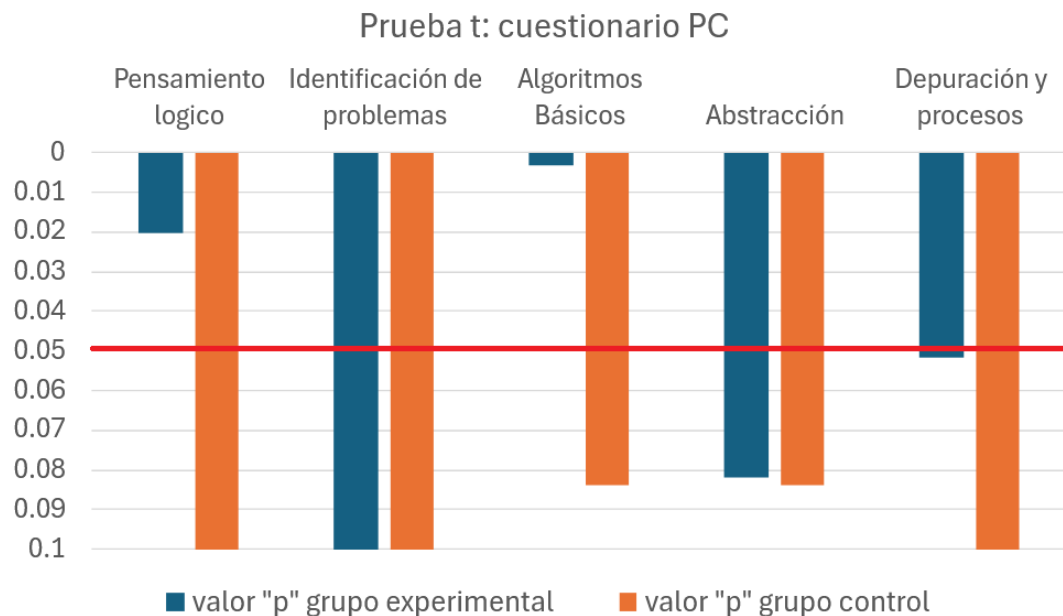


Figura 6.4: Tabla con valores p de la prueba t para los cuestionarios de pensamiento computacional, antes y después de la intervención.

Los resultados muestran que en el grupo control no se observó ninguna diferencia estadísticamente significativa en las respuestas, ya que todos los valores p fueron superiores a 0.05. Por el contrario, en el grupo experimental se registraron mejoras significativas en las preguntas asociadas a pensamiento lógico y algoritmos básicos. Además, en la dimensión de habilidades de depuración y procesos, el valor p fue de 0.0518, lo que si bien no cumple estrictamente con el umbral convencional de 0.05,

se encuentra lo suficientemente cerca como para sugerir una posible mejora, aunque con menor nivel de certeza estadística.

En la segunda fase del análisis se buscó comparar directamente los cambios obtenidos por ambos grupos, utilizando los *deltas* (diferencias entre los puntajes post y pre intervención) como variable de interés. Para ello, se aplicó una prueba t para muestras independientes con varianzas desiguales (Welch's t-test), permitiendo evaluar si el grupo experimental presentó una mejora significativamente mayor que el grupo control.

Este tipo de análisis es relevante porque no solo permite verificar mejoras dentro de cada grupo, sino que también controla la influencia de factores externos que podrían haber afectado a ambos grupos por igual, otorgando mayor solidez a los resultados obtenidos.

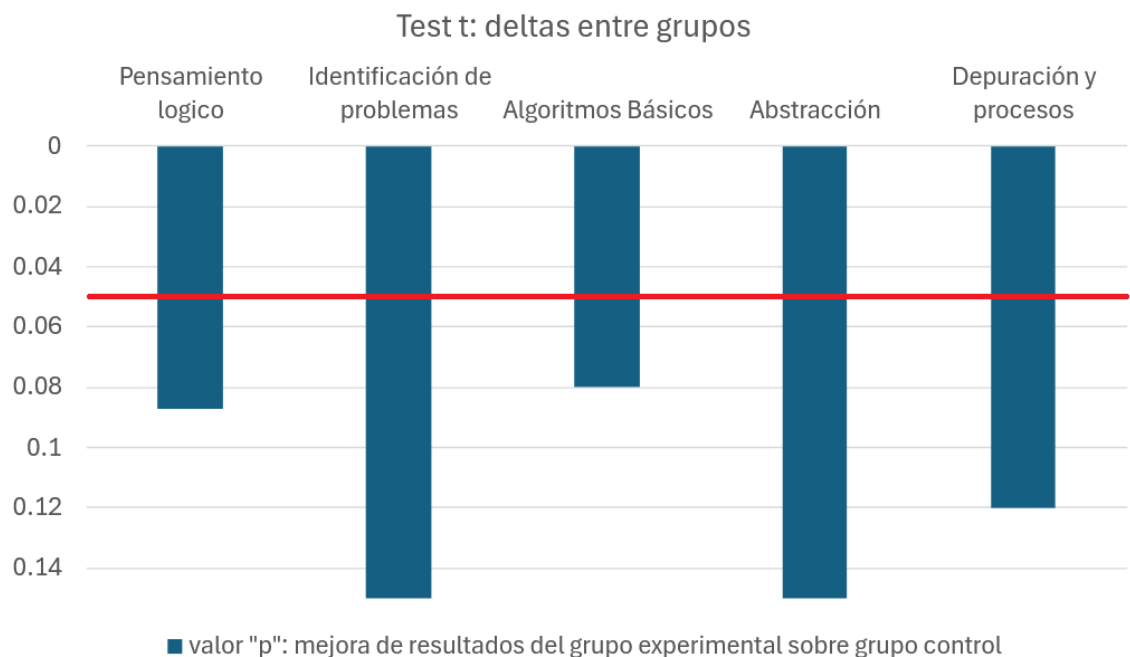


Figura 6.5: Tabla con valores p de la prueba t para muestras independientes, aplicada a los deltas de los cuestionarios de pensamiento computacional.

Los resultados de esta prueba muestran que ninguna de las dimensiones evaluadas de pensamiento computacional en el grupo experimental presentó una mejora estadísticamente significativa respecto del grupo control. Esto indica que, si bien se

observaron mejoras internas en el grupo experimental, estas no fueron lo suficientemente distintas como para atribuir con certeza su origen exclusivo a la intervención educativa.

6.2.2. Pruebas de Educación Ciudadana

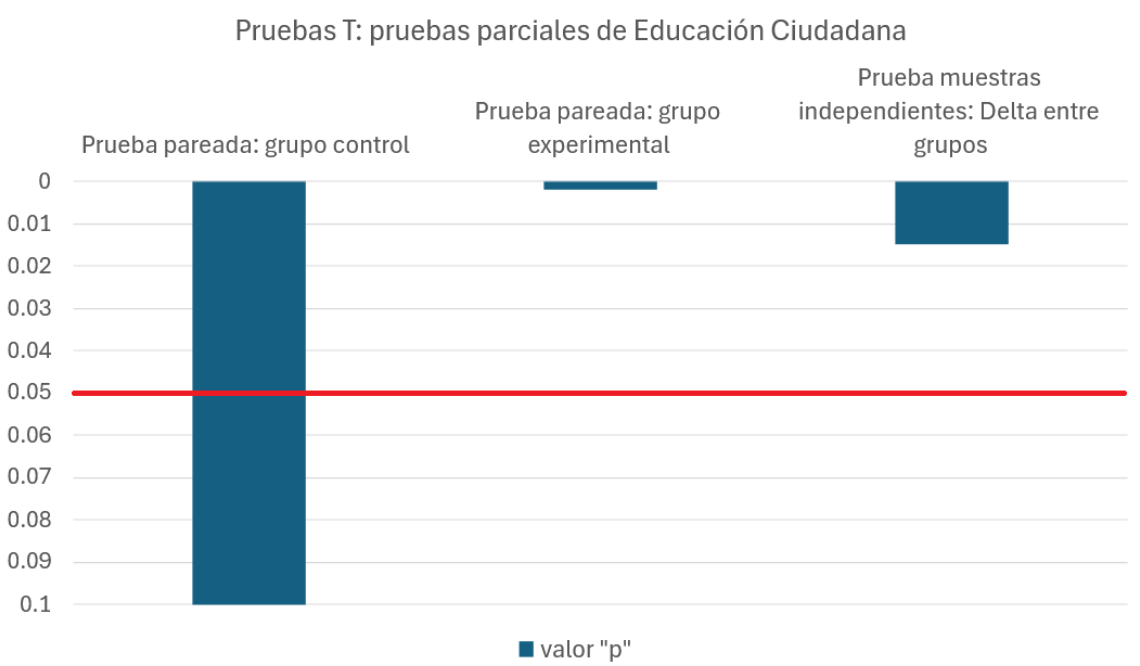


Figura 6.6: Tabla con valores p de la prueba t para las pruebas de educación ciudadana.

Los resultados obtenidos a partir de las pruebas t aplicadas a las evaluaciones parciales de la asignatura de Educación Ciudadana revelan una diferencia clara entre los grupos. En el grupo control no se observó ninguna mejora estadísticamente significativa al comparar los puntajes obtenidos antes y después de la intervención, ya que el valor p fue superior al umbral de 0.05.

En contraste, el grupo experimental mostró una mejora altamente significativa, con un valor p de 0.002, lo que indica que la intervención tuvo un impacto positivo relevante en los aprendizajes de esta área.

Además, al comparar los deltas de ambos grupos mediante una prueba t para muestras independientes, se encontró que la diferencia de mejoras fue también

estadísticamente significativa, con un valor p de 0.015. Esto sugiere que el grupo experimental no solo mejoró individualmente, sino que lo hizo en una magnitud superior a la del grupo control, fortaleciendo la evidencia a favor del efecto positivo de la intervención educativa.

7. Conclusiones y trabajo futuro

Los resultados obtenidos permiten concluir que la hipótesis planteada se cumplió de manera parcial. En cuanto al desarrollo de habilidades de pensamiento computacional, se observaron mejoras estadísticamente significativas en algunas dimensiones, mientras que otras no presentaron cambios relevantes. Específicamente, las habilidades de pensamiento lógico, algoritmos básicos y depuración de procesos mostraron avances notables. Estas competencias fueron abordadas tanto en las actividades desconectadas como en las conectadas de la intervención.

Por otro lado, las habilidades de identificación de problemas y abstracción no evidenciaron mejoras significativas. Esto puede explicarse en parte porque no fueron trabajadas directamente en las actividades desconectadas, y además, la escena preconfigurada 4 (diseñada para abordar la identificación de problemas) tuvo que ser eliminada por razones logísticas vinculadas al establecimiento educativo, reduciendo así la cobertura de este contenido.

A partir de estos resultados, se concluye que la intervención educativa podría beneficiarse de una mayor duración. Se estima que al incorporar al menos una sesión adicional enfocada en actividades desconectadas, y otra dedicada a actividades conectadas, se lograría un desarrollo más completo de todas las habilidades de pensamiento computacional propuestas.

Respecto a la herramienta utilizada **Visual Scripting en Unity**, se concluye que no resulta adecuada para el contexto de la educación pública chilena. Una de las principales limitaciones es que esta interfaz solo se encuentra disponible en inglés, idioma que no forma parte de las competencias dominantes del estudiantado en el currículo nacional. Esta barrera lingüística elimina gran parte del carácter intuitivo y accesible que se asocia comúnmente al **Visual Scripting de Unity**, y que origi-

nalmente motivó su elección para esta investigación. Como alternativa para futuras investigaciones, se sugiere considerar el uso de **Blueprints en Unreal Engine 5**, un sistema de scripting visual similar, pero que sí cuenta con interfaz en español, lo cual podría facilitar su integración en entornos educativos hispanohablantes.

Asimismo, se identificó que para lograr una implementación exitosa de las sesiones conectadas fue necesario un proceso constante de adaptación basado en la observación de las necesidades, ritmos y preferencias del grupo experimental. Esto evidencia que, además de una planificación inicial clara, es fundamental mantener una flexibilidad metodológica que permita realizar ajustes sesión por sesión. En este sentido, el uso de herramientas de evaluación en tiempo real (como el script automatizado utilizado en esta investigación) se vuelve clave para monitorear el progreso del estudiantado y tomar decisiones pedagógicas oportunas.

Finalmente, en lo que respecta al componente de **Educación Ciudadana**, los resultados muestran una mejora significativa en el grupo experimental en comparación con el grupo control. Este hallazgo sugiere que la enseñanza de pensamiento computacional mediante herramientas tecnológicas puede tener un impacto positivo también en áreas ajenas al ámbito STEM. Sin embargo, es importante señalar que las evaluaciones aplicadas en esta dimensión estuvieron sujetas a los criterios definidos por el docente de la asignatura, sobre los cuales esta investigación no tuvo control. Por esta razón, se reconoce que los resultados pueden contener cierto sesgo. Para futuras investigaciones se recomienda diseñar e implementar instrumentos de evaluación estandarizados, similares a los cuestionarios aplicados para medir pensamiento computacional, que permitan obtener resultados más objetivos y comparables.

Bibliografía

- [1] Jeannette M Wing. Computational thinking benefits society. 40th anniversary blog of social issues in computing. *Verfügbar unter <http://socialissues.cs.toronto.edu/index.html>* % 3Fp, 279, 2014.
- [2] Aman Yadav, Chris Mayfield, Ninger Zhou, Susanne Hambruch, and John T Korb. Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, 14(1):1–16, 2014.
- [3] Jeannette M. Wing. Computational thinking. *Communications of the ACM*, 49(3):33–35, 2006.
- [4] Jeannette M Wing. Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881):3717–3725, 2008.
- [5] Shuchi Grover. The 5th ‘c’ of 21st century skills? try computational thinking (not coding). *EdSurge*, 2018. Accessed: 2024-10-09.
- [6] Mario Dumančić, Nataša Rogulja, and Krešo Tomljenović. The importance of algorithmic thinking development in primary education. In *2024 47th MIPRO ICT and Electronics Convention (MIPRO)*, pages 653–658, 2024.
- [7] Pratim Sengupta, John S Kinnebrew, Satabdi Basu, Gautam Biswas, and Douglas Clark. Integrating computational thinking with k-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18:351–380, 2013.

- [8] Chunyan Wang, Ji Shen, and Joseph Chao. Integrating computational thinking in stem education: A literature review. *International Journal of Science and Mathematics Education*, 20(8):1949–1972, 2022.
- [9] Aikaterini Bounou, Konstantinos Lavidas, Vassilis Komis, Stamatis Papadakis, and Polyxeni Manoli. Correlation between high school students’ computational thinking and their performance in stem and language courses. *Education Sciences*, 13(11):1101, 2023.
- [10] Gitta Siekmann. What is stem? the need for unpacking its definitions and applications. *National Centre for Vocational Education Research (NCVER)*, 2016.
- [11] Kai Orton, David Weintrop, Elham Beheshti, Michael Horn, Kemi Jona, and Uri Wilensky. Bringing computational thinking into high school mathematics and science classrooms. In *Proceedings of the International Conference of the Learning Sciences (ICLS)*, pages 705–712. International Society of the Learning Sciences, Singapore, 2016.
- [12] Alexander Repenning, David Webb, and Andri Ioannidou. Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM technical symposium on Computer science education*, pages 265–269, 2010.
- [13] Elisa Nadire Caeli and Aman Yadav. Unplugged approaches to computational thinking: A historical perspective. *TechTrends*, 64(1):29–36, 2020.
- [14] Christina Pei, David Weintrop, and Uri Wilensky. Cultivating computational thinking practices and mathematical habits of mind in lattice land. *Mathematical thinking and learning*, 20(1):75–89, 2018.
- [15] Marina Umaschi Bers, Louise Flannery, Elizabeth R Kazakoff, and Amanda Sullivan. Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & education*, 72:145–157, 2014.
- [16] Bahadir Namdar and Ji Shen. Modeling-oriented assessment in k-12 science education: A synthesis of research from 1980 to 2013 and new directions. *International Journal of Science Education*, 37(7):993–1023, 2015.

- [17] Amy K Hoover, Jackie Barnes, Borna Fatehi, Jesús Moreno-León, Gillian Puttick, Eli Tucker-Raymond, and Casper Harteveld. Assessing computational thinking in students' game designs. In *Proceedings of the 2016 annual symposium on computer-human interaction in play companion extended abstracts*, pages 173–179, 2016.
- [18] Jacqueline Leonard, Alan Buss, Ruben Gamboa, Monica Mitchell, Olatokunbo S Fashola, Tarcia Hubert, and Sultan Almughyirah. Using robotics and game design to enhance children's self-efficacy, stem attitudes, and computational thinking skills. *Journal of Science Education and Technology*, 25:860–876, 2016.
- [19] Min Lun Wu. Educational game design as gateway for operationalizing computational thinking skills among middle school students. *International Education Studies*, 11(4):15–28, 2018.
- [20] Ann Bremner. Singing and gaming to math literacy. *Teaching Children Mathematics*, 19(9):582–584, 2013.
- [21] Stephen Cooper, Wanda Dann, and Randy Pausch. Developing algorithmic thinking with alice. In *The proceedings of ISECON*, volume 17, pages 506–539. Citeseer, 2000.
- [22] Juraj Hromkovič, Tobias Kohn, Dennis Komm, and Giovanni Serafini. Combining the power of python with the simplicity of logo for a sustainable computer science education. In *Informatics in Schools: Improvement of Informatics Knowledge and Perception: 9th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2016, Münster, Germany, October 13-15, 2016, Proceedings 9*, pages 155–166. Springer, 2016.
- [23] Juraj Hromkovic, Tobias Kohn, Dennis Komm, Giovanni Serafini, et al. Algorithmic thinking from the start. *Bulletin of EATCS*, 1(121), 2017.
- [24] Ali Battal, Gülgün Afacan Adanır, and Yasemin Gülbahar. Computer science unplugged: A systematic literature review. *Journal of Educational Technology Systems*, 50(1):24–47, 2021.
- [25] Tim Bell, Jason Alexander, Isaac Freeman, and Mick Grimley. Computer science unplugged: school students doing real computing without computers. *New Zealand Journal of Applied Computing and Information Technology*, 13(1):20–29, 2009.

- [26] Tony Jenkins. On the difficulty of learning to program. In *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences*, volume 4, pages 53–58. Citeseer, 2002.
- [27] Christian P Brackmann, Marcos Román-González, Gregorio Robles, Jesús Moreno-León, Ana Casali, and Dante Barone. Development of computational thinking skills through unplugged activities in primary school. In *Proceedings of the 12th workshop on primary and secondary computing education*, pages 65–72, 2017.
- [28] J Patrick McCarthy and Liam Anderson. Active learning techniques versus traditional teaching styles: Two experiments from history and political science. *Innovative higher education*, 24:279–294, 2000.
- [29] John Locke. *Some Thoughts Concerning Education*. A. and J. Churchill, London, 1693.
- [30] Johann Friedrich Herbart. *The science of education: Its general principles deduced from its aim and the aesthetic revelation of the world*, volume 21. DC Heath & Company, 1806.
- [31] Edward Lee Thorndike. *Principles of teaching based on psychology*. New York: AG Seiler, 1906.
- [32] Charles C Bonwell and James A Eison. *Active learning: Creating excitement in the classroom. 1991 ASHE-ERIC higher education reports*. ERIC, 1991.
- [33] Xiaodan Tang, Yue Yin, Qiao Lin, Roxana Hadad, and Xiaoming Zhai. Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, 148:103798, 2020.
- [34] Shuchi Grover and Roy Pea. Computational thinking in k–12: A review of the state of the field. *Educational researcher*, 42(1):38–43, 2013.
- [35] Karen Brennan and Mitchel Resnick. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, volume 1, page 25, 2012.

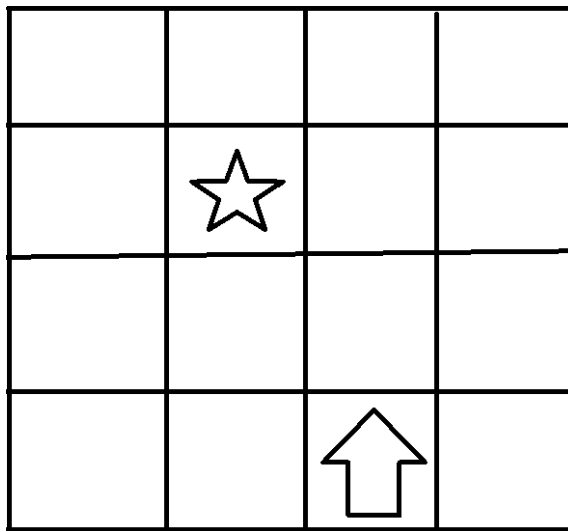
- [36] Georgios Fessakis, Evangelia Gouli, and Elisavet Mavroudi. Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education*, 63:87–97, 2013.
- [37] Sze Yee Lye and Joyce Hwee Ling Koh. Review on teaching and learning of computational thinking through programming: What is next for k-12? *Computers in human behavior*, 41:51–61, 2014.
- [38] Andy Field. *Discovering statistics using IBM SPSS statistics*. Sage publications limited, 2024.
- [39] Ronald Aylmer Fisher and Ronald A Fisher. *The design of experiments*. Springer, 1971.
- [40] Robert M West. Best practice in statistics: Use the welch t-test when testing the difference between two groups. *Annals of clinical biochemistry*, 58(4):267–269, 2021.

A. Cuestionarios de Medición del Pensamiento Computacional

A.1. Cuestionario 1

1. Pensamiento Lógico (Secuencia Lógica):

Un jugador quiere programar un robot para moverse por una cuadrícula. El robot puede avanzar y retroceder una casilla, y girar a la derecha e izquierda. En la siguiente imagen se representa la cuadrícula con el robot, donde el robot es la flecha y debe alcanzar la estrella. ¿Cuál de las siguientes secuencias lleva al robot a la estrella?



A) Avanzar, girar a la izquierda, retroceder, avanzar

- B) Avanzar, avanzar, girar a la derecha, retroceder
- C) Girar a la derecha, avanzar, retroceder, avanzar
- D) Avanzar, avanzar, girar a la derecha, avanzar

2. Identificación de Problemas:

Tienes una lista desordenada de nombres de estudiantes que debes ordenar alfabéticamente para organizar las inscripciones en un evento escolar. ¿Cuál sería el primer paso lógico para resolver este problema?

- A) Dividir la lista en partes más pequeñas.
- B) Contar cuántos nombres hay en la lista.
- C) Comparar dos nombres de la lista y decidir su orden relativo.
- D) Pedir a cada estudiante que se ordene a sí mismo en el listado.

3. Algoritmos Básicos:

Quieres encontrar un nombre específico en una lista de asistencia ordenada alfabéticamente. ¿Qué harías primero para encontrar el nombre de manera más rápida?

- A) Revisar cada nombre de la lista uno por uno desde el principio hasta encontrar el correcto.
- B) Dividir la lista a la mitad y verificar en qué mitad está el nombre, repitiendo el proceso hasta encontrarlo.
- C) Revisar la lista al azar hasta encontrar el nombre que buscas.
- D) Empezar desde el final de la lista y leer hacia atrás.

4. Abstracción:

Estás organizando un evento escolar en el que participan varios equipos de diferentes tamaños. Solo necesitas saber cuántos equipos hay y el nombre de cada equipo. ¿Qué información sería suficiente para registrar?

- A) El nombre del equipo y el número de participantes en cada uno.
- B) El nombre de cada equipo, el número de participantes, y la dirección de cada participante.

C) La lista de actividades de cada equipo y los nombres de todos los integrantes.

D) La fecha de cumpleaños de cada participante y sus calificaciones.

5. Depuración y Procesos:

Un estudiante está revisando su tarea de matemáticas y nota que el resultado de una operación no coincide con el que debería tener. ¿Cuál sería el mejor enfoque para encontrar el error?

A) Revisar cada paso de la operación para ver si cometió algún error de cálculo.

B) Dejar el resultado como está y continuar con la siguiente pregunta.

C) Cambiar el resultado sin revisar los pasos para intentar obtener el número correcto.

D) Pedir a otro estudiante que haga toda la tarea desde el principio.

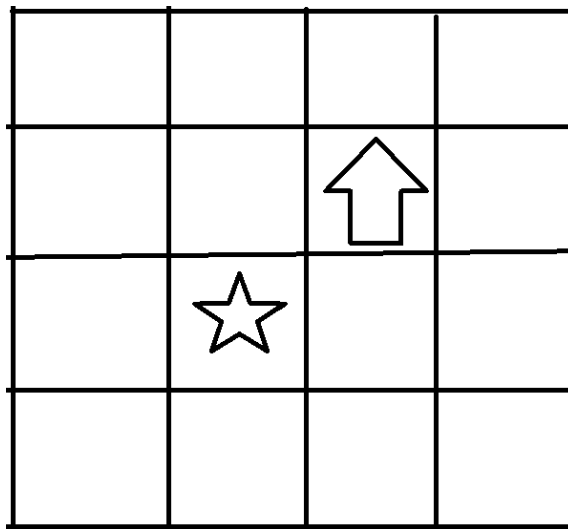
Respuestas:

1. B 2. C 3. B 4. A 5. A

A.2. Cuestionario 2

1. Pensamiento Lógico (Secuencia Lógica):

Un jugador quiere programar un robot para moverse por una cuadrícula. El robot puede avanzar y retroceder una casilla, y girar a la derecha e izquierda. En la siguiente imagen se representa la cuadrícula con el robot, donde el robot es la flecha y debe alcanzar la estrella. ¿Cuál de las siguientes secuencias lleva al robot a la estrella?



- A) Girar a la izquierda, avanzar, girar a la izquierda, retroceder
- B) Retroceder, girar a la izquierda, retroceder, avanzar
- C) Girar a la izquierda, avanzar, girar a la izquierda, avanzar
- D) Girar a la derecha, retroceder, girar a la izquierda, avanzar

2. Identificación de Problemas:

Tienes una calculadora que realiza operaciones matemáticas básicas, pero necesitas calcular raíces cuadradas. ¿Cuál de las siguientes opciones describe mejor cómo identificar el problema?

- A) Probar operaciones básicas en la calculadora hasta obtener el resultado.
- B) Determinar que la función de raíz cuadrada no está incluida y que debes buscar otra herramienta.

- C) Pedir ayuda a alguien para hacer el cálculo.
- D) Usar una calculadora diferente de inmediato, sin evaluar el problema.

3. Algoritmos Básicos:

Necesitas asignar asientos a los estudiantes en una sala de examen. Cada asiento tiene un número, y quieres asegurarte de que todos ocupen el asiento correcto según la lista. ¿Qué harías?

- A) Dejar que los estudiantes elijan cualquier asiento disponible.
- B) Llamar a cada estudiante en orden y asignarle su asiento correspondiente en función del número en la lista.
- C) Hacer que todos entren y se sienten lo más rápido posible, sin importar el orden.
- D) Colocar los números de los asientos al azar y pedir a los estudiantes que se acomoden.

4. Abstracción:

Estás diseñando una aplicación que muestre el estado del clima en distintas ciudades. Para que funcione correctamente, ¿qué datos deberías mostrar en la pantalla principal?

- A) La temperatura, el estado del cielo (despejado, nublado, etc.) y la ciudad.
- B) La temperatura, la velocidad del viento, la humedad y una lista de eventos de la ciudad.
- C) La temperatura, el historial climático del último mes y los puntos turísticos de la ciudad.
- D) Todos los datos meteorológicos disponibles para esa ciudad en las últimas 24 horas.

5. Depuración y Procesos:

Un estudiante está escribiendo un informe y nota que algunas palabras están subrayadas en rojo, indicando un posible error ortográfico. ¿Qué debería hacer para corregirlo?

- A) Ignorar las palabras subrayadas y continuar escribiendo el informe.

- B) Revisar cada palabra subrayada y corregir la ortografía en caso de ser necesario.
- C) Eliminar todo el texto y escribirlo nuevamente.
- D) Cambiar el formato del texto para que el subrayado desaparezca.

Respuestas:

1. C 2. B 3. B 4. A 5. B

B. Escena Preconfigurada no usada

Escena 4: Validación Ética de Candidaturas Ciudadanas

Esta escena simula una plataforma digital en la que se revisan candidaturas para un cargo de representación en un consejo comunal. El sistema fue diseñado para asegurar que solo personas íntegras y comprometidas puedan postularse. Sin embargo, el código de verificación contiene errores que permiten validar candidaturas que no cumplen con los principios básicos de participación ética, lo que podría generar corrupción, clientelismo o pérdida de confianza ciudadana.

Los estudiantes deberán depurar el sistema, que actualmente evalúa de forma incorrecta las condiciones para autorizar una candidatura, y corregir los errores para que solo los postulantes adecuados sean aprobados.

Condiciones correctas para aceptar una candidatura:

- La persona no ha sido sancionada por malas prácticas en cargos anteriores.
- La candidatura cuenta con al menos 20 firmas de respaldo ciudadano reales y verificadas.
- El postulante ha presentado una declaración pública.

Errores presentes en el código original:

- Se utilizan operadores **OR** en lugar de **AND**, permitiendo la aprobación con solo una condición cumplida.

- La condición que verifica el respaldo ciudadano permite aprobar candidaturas con menos de 20 firmas.

Una vez que los errores sean corregidos, el sistema solo aprobará candidaturas transparentes y responsables. Al cumplirse la condición de éxito, se mostrará un mensaje de éxito.

“Cuando las reglas están mal hechas, la corrupción encuentra el camino libre. Depurar el sistema es también una forma de proteger la democracia.”