Esfera Sabia Revista Educación Superior

ISSN: 3103-1102

Vol. 4 Núm. 1 (2026)

EDITORIAL JOGB

ESES-Vol.4. N1. 004

Pensamiento computacional y programación modular en la enseñanza del lenguaje C++

Computational Thinking and Modular Programming in C++ Language Teaching

Autores:
Tania Karina Ponce Torca
Universidad San Francisco Xavier
Sucre – Bolivia
ponce.tania@usfx.bo
https://orcid.org/0009-0009-1158-1668

Autor de correspondencia: *Tania Karina Ponce Torca*, ponce.tania@usfx.bo

Recepción: 25-febrero-2026 **Aceptación:** 18-marzo-2026 **Publicación:** 08-abril-2026

Cómo citar este artículo:
Ponce Torca, T. K. (2026). Pensamiento computacional y programación modular en la enseñanza del lenguaje C++. *Sage Sphere Higher Education*, *4*(1), 1-14. https://doi.org/10.63688/vn145s84

RESUMEN

El pensamiento computacional se posiciona como una competencia clave en la educación del siglo XXI, pero su incorporación en la enseñanza de C++ en la educación técnica superior aún presenta desafíos. En la asignatura Programación Básica (SIS100) de la Universidad San Francisco Xavier de Chuquisaca se han registrado tasas elevadas de abandono (60%-70%) y reprobación (40%-50%) entre 2016 y 2023. Este artículo propone un modelo pedagógico innovador basado en el pensamiento computacional y la programación modular, sustentado en estudios realizados en dicha asignatura. La propuesta integra metodologías activas, aprendizaje basado en juegos y desafíos inspirados en el modelo Bebras, promoviendo una enseñanza estructurada y significativa. Se presentan evidencias que respaldan la efectividad del enfoque para mejorar la comprensión de conceptos en C++. Además, se discute la viabilidad y aplicabilidad del modelo en la educación técnica, alineado con tendencias internacionales en innovación educativa y orientado al desarrollo de competencias tecnológicas avanzadas.

Palabras claves: competencia digital, educación técnica, innovación pedagógica, pensamiento computacional, programación modular.

ABSTRACT

Computational thinking is recognized as a key competence in 21st-century education, yet its integration into C++ instruction at the technical higher education level remains limited. In the course Basic Programming (SIS100) at Universidad San Francisco Xavier de Chuquisaca, high dropout rates (60%–70%) and failure rates (40%–50%) were recorded between 2016 and 2023. This article proposes an innovative pedagogical model based on computational thinking and modular programming, grounded in previous studies conducted in this course. The proposal incorporates active methodologies, game-based learning, and challenges inspired by the Bebras model, promoting a structured and meaningful approach to teaching C++. Evidence is presented showing that the integration of computational thinking and modular programming enhances students' understanding of C++ concepts. The feasibility and potential application of the proposed model in technical education are also analyzed, aligning with international trends in educational innovation and aiming to strengthen the development of advanced technological competencies.

Keywords: computational thinking, digital competence, pedagogical innovation, modular programming, technical education.

ISSN: 3103-1102

Vol. 4 Núm. 1 (2026)

1. INTRODUCCIÓN

El pensamiento computacional (PC) se ha consolidado como una habilidad clave en la educación del siglo XXI, especialmente en el ámbito de la programación y la informática. Según Wing (2017), el PC implica resolver problemas, diseñar sistemas y comprender el comportamiento humano mediante conceptos fundamentales de la informática. En la actualidad, esta competencia es considerada esencial para desenvolverse en una sociedad digital, tal como señalan Acevedo-Borrega et al. (2022) en su revisión sobre el rol del PC en entornos educativos. De manera complementaria, (Bounou et al., 2023) evidencian que el desarrollo del pensamiento computacional se asocia con un mejor rendimiento en áreas de alta complejidad, lo que refuerza su importancia en contextos formativos orientados al desarrollo de habilidades tecnológicas avanzadas. En un contexto de rápida evolución tecnológica, el PC se ha convertido en un componente esencial para aplicar nuevas tecnologías y facilitar la adaptación a entornos digitales cambiantes (Asunda et al., 2023).

Diversos estudios, como el de Acevedo-Borrega et al. (2022), destacan que el PC permite a los estudiantes enfrentar problemas complejos y desarrollar habilidades críticas. subrayan su papel en la integración de tecnologías avanzadas, como la inteligencia artificial, en contextos educativos. Este enfoque, que incluye habilidades como la abstracción, descomposición, reconocimiento de patrones y razonamiento algorítmico, fomenta también la creatividad y el pensamiento crítico.

En este marco, la enseñanza de lenguajes de programación como C++ representa un componente esencial en la formación universitaria en áreas tecnológicas. No obstante, persisten desafíos relacionados con metodologías centradas únicamente en contenidos sintácticos. (Barragán, 2023) advierte que, aunque el PC debe fomentarse desde edades tempranas, su incorporación efectiva en niveles universitarios aún es limitada. A su vez, la programación modular se presenta como una estrategia efectiva para dividir problemas complejos en componentes más manejables, mejorando la comprensión del desarrollo de software (Yadav, 2019).

Estudios recientes, como el análisis de casos múltiples de Leibovitch et al., 2025, muestran que las metodologías centradas en el pensamiento crítico, computacional o de resolución de problemas transforman progresivamente las creencias docentes y mejoran la forma en que

Doi: https://doi.org/10.63688/vn145s84

los estudiantes construyen conocimiento. Los autores evidencian que cuando la enseñanza se orienta hacia habilidades cognitivas explícitas, los estudiantes participan más, construyen razonamientos más profundos y obtienen mejores resultados académicos.

En el ámbito hispano y latinoamericano, la integración del pensamiento computacional ha cobrado un impulso significativo en los últimos años. Una revisión sistemática reciente de (Caisaguano Villa & Apolo Buenaño, 2025), basada en 70 estudios de acceso abierto indexados en SCOPUS, evidencia un crecimiento notable de la producción científica sobre este tema, pasando de un 11.43% en 2020 a un 37.14% en 2024. Sus resultados muestran que los países de habla hispana, especialmente España, Chile, Colombia, Perú y México, están desarrollando propuestas diversas que incluyen robótica educativa, programación visual, actividades sin computadora y enfoques STEM. No obstante, los autores advierten brechas en la formación docente, acceso a recursos y la baja representación de países latinoamericanos en la investigación, lo que subraya la necesidad de propuestas contextualizadas. Este panorama refuerza la pertinencia de desarrollar modelos pedagógicos adaptados a realidades locales como la educación técnica superior en Bolivia, donde aún persisten desafíos en la enseñanza de programación y en la apropiación del pensamiento computacional.

En la Universidad San Francisco Xavier de Chuquisaca, por ejemplo, en una asignatura de programación se registraron entre 2016 y 2024 tasas de abandono de entre el 60% y el 70%, y reprobación entre el 40% y el 50%. Situaciones de este tipo también se observan en otras carreras con características similares, lo que refuerza la necesidad de cambios metodológicos. Por tanto, la necesidad de enfoques pedagógicos innovadores en la enseñanza de la programación se vuelve urgente. Como señalan Tejera-Martínez et al. (2020) los lenguajes de programación deben utilizarse para desarrollar competencias clave como la abstracción y el razonamiento lógico. La programación modular aporta un enfoque que favorece esta construcción progresiva de conocimiento (Alvarez et al., 2023; Ricciardi, 2024).

Este artículo tiene como objetivo principal diseñar y analizar un modelo educativo innovador para la enseñanza de C++, que integra pensamiento computacional y programación modular. Se busca evaluar su impacto en el rendimiento académico, la retención estudiantil y el perfeccionamiento de habilidades cognitivas y creativas en estudiantes universitarios. El estudio se desarrolla tomando como referencia una asignatura de programación en

la educación superior donde se identificaron altos niveles de deserción y reprobación, problemática que también se presenta en otras carreras con características similares.

2. METODOLOGÍA

Durante el segundo semestre de 2025 se implementó una propuesta pedagógica basada en pensamiento computacional y programación modular en una asignatura de programación de la educación superior. La intervención se aplicó en dos grupos paralelos ya conformados por la unidad académica, lo que permitió establecer una comparación entre el enfoque tradicional de enseñanza y el modelo propuesto.

La evaluación de resultados se realizó mediante instrumentos aplicados antes y después de la intervención, construidos a partir de los pilares del pensamiento computacional. Adicionalmente, se consideraron los desempeños prácticos a través de una plataforma estructurada de codificación. La recolección de datos se efectuó a través de pruebas y registros digitales, y su análisis se basó en la comparación de promedios y la identificación de patrones de progresión entre ambos grupos.

Materiales

Se utilizaron dos tipos de materiales principales. El primero consistió en evaluaciones diagnósticas y finales, desarrolladas bajo la metodología Bebras (Bebras, 2025), que incluyen problemas relacionados con descomposición, reconocimiento de patrones, abstracción y algoritmos. El segundo material fue la plataforma de programación OmegaUp (https://omegaup.com), que permite registrar en tiempo real la resolución de problemas prácticos en lenguaje C++, y ofrece retroalimentación inmediata, trazabilidad de resultados y clasificación automática.

Como recurso principal de apoyo para el grupo experimental se empleó el libro de autoría propia titulado Programación en C++: Fortalecimiento de Habilidades a través del Pensamiento Computacional (Ponce Torca, 2024), que integra ejercicios teóricos y prácticos basados en pensamiento computacional y programación modular.

Participantes

La matrícula inicial en la asignatura alcanzó un total de 108 estudiantes universitarios, distribuidos en dos grupos de teoría conformados por sorteo institucional: 59 en el primer

grupo y 49 en el segundo. Esta diferencia moderada refleja una distribución relativamente equilibrada al inicio del semestre.

Durante el desarrollo del curso, se observó un patrón de abandono similar al registrado en gestiones anteriores, asociado principalmente a las características del enfoque tradicional de enseñanza. Al finalizar el semestre, completaron todos los requisitos para el análisis 79 estudiantes: 50 en el grupo experimental y 29 en el grupo control. La reducción fue más pronunciada en el grupo control, que pasó de una matrícula inicial considerable a un número final reducido.

Para garantizar la comparabilidad, los análisis se basaron en promedios y porcentajes de mejora, mitigando el efecto de la diferencia en el tamaño final de los grupos.

Tareas y Métodos

La intervención se estructuró en actividades teóricas y prácticas. En las sesiones teóricas, el grupo experimental trabajó con contenidos enfocados en pensamiento computacional y programación modular, mientras que el grupo control utilizó clases expositivas tradicionales. En las sesiones prácticas, ambos grupos realizaron los mismos ejercicios, diseñados e implementados en la plataforma OmegaUp. Los temas abordaron 12 tópicos esenciales de programación, incluyendo variables, estructuras selectivas y repetitivas, funciones, arreglos y cadenas. Cada tema fue desarrollado en cuatro sesiones prácticas distribuidas semanalmente.

La evaluación incluyó un pretest y postest Bebras para medir la evolución de competencias en los pilares del pensamiento computacional, así como habilidades complementarias en lenguaje y matemáticas. Además, de evaluación de laboratorio para el seguimiento del rendimiento en ejercicios de codificación en OmegaUp, con análisis por promedio, temática, y consistencia.

El análisis de los datos se realizó comparando resultados entre grupos, utilizando promedios, porcentajes de mejora y patrones de progresión. También se incorporó cuestionarios abiertos para explorar percepciones estudiantiles sobre la metodología aplicada.

3. RESULTADOS

El modelo educativo basado en pensamiento computacional y programación modular fue implementado con el propósito de evaluar su efectividad en el rendimiento de estudiantes

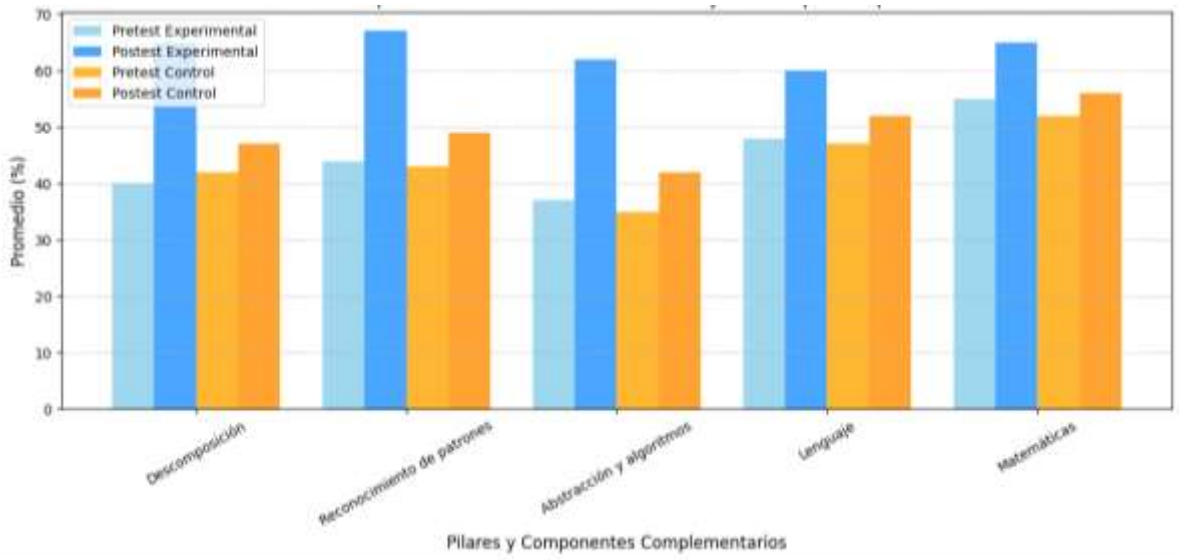
universitarios en una asignatura de programación de la educación superior. Los resultados se presentan en dos dimensiones: la evolución de competencias mediante pretest/postest y el desempeño práctico en laboratorio a lo largo de 12 temáticas.

En la primera dimensión, se evaluaron competencias en pensamiento computacional (descomposición, reconocimiento de patrones, abstracción y algoritmos), así como habilidades en lenguaje y matemáticas. Ambos grupos iniciaron el semestre con niveles similares y una matrícula relativamente equilibrada; sin embargo, a lo largo del curso se registró una reducción más pronunciada en la participación del grupo control. Al cierre del semestre, el grupo experimental mostró mejoras más significativas:

- En abstracción y algoritmos, pasó del 45% al 60% (+15%), mientras que el grupo control avanzó del 42% al 49% (+7%).
- En lenguaje y matemáticas, el grupo experimental mejoró en 12% y 10%, frente a 5% y 4% del grupo control, respectivamente.

Figura 1

Comparación de Promedios Pretest y Postest por grupo



Nota. Resultados descriptivos en formato de barras agrupadas. Elaboración propia.

En cuanto al desempeño practico, durante el semestre, los estudiantes abordaron 12 temas esenciales en programación estructurada. Se evaluaron semanalmente en la plataforma OmegaUp, registrando su promedio por tema. Los resultados muestran un mejor rendimiento general del grupo experimental, especialmente en temáticas avanzadas.

Doi: https://doi.org/10.63688/vn145s84

Esfera Sabia Revista Educación Superior

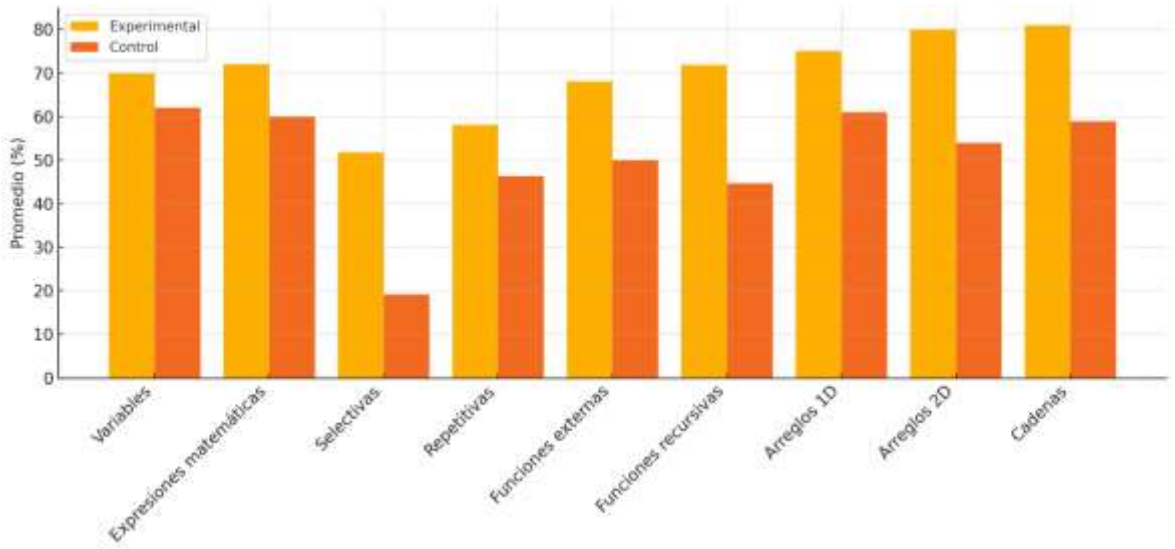
ISSN: 3103-1102

Vol. 4 Núm. 1 (2026)

- En cadenas y arreglos bidimensionales, el grupo experimental obtuvo 80.91% y 79.92%, frente a 58.89% y 53.91% del grupo control.
- En funciones recursivas, el grupo experimental logró 71.88%, mientras que el grupo control 44.61%.

Figura 2

Promedios por temática en laboratorio

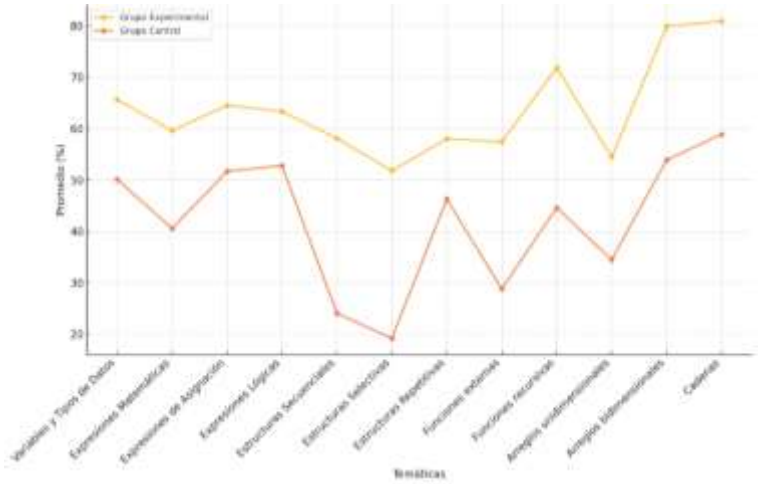


Nota. Resultados descriptivos en formato de barras agrupadas. Elaboración propia.

La progresión temática y patrones de aprendizaje se mantuvieron estables en el grupo experimental a lo largo del semestre, evidenciando una progresión uniforme. En cambio, el grupo control presentó mayor dispersión, particularmente en temáticas avanzadas.

Figura 3

Progresión de Promedios por Temáticas entre Grupos



Doi: https://doi.org/10.63688/vn145s84

Nota. Resultados descriptivos en formato de barras agrupadas. Elaboración propia.

Para establecer la relación entre asistencia y rendimiento, se analizó cómo varían los promedios según tres rangos de asistencia (alta, media y baja) en ambos grupos. En el grupo experimental, los estudiantes con asistencia alta (75% o más de las sesiones) alcanzaron promedios cercanos al 85%, mientras que en el grupo control, con niveles similares de asistencia, el promedio se mantuvo alrededor del 60%. Es decir, aun comparando estudiantes que asistieron con regularidad, quienes trabajaron con el modelo basado en pensamiento computacional y programación modular obtuvieron alrededor de 20 puntos porcentuales adicionales, especialmente en contenidos avanzados como funciones recursivas y manejo de cadenas.

Cuando la asistencia se sitúa en un rango medio (entre 50% y 74% de las sesiones), el promedio del grupo experimental desciende a aproximadamente 75%, mientras que el grupo control se ubica en torno al 55%. Finalmente, en los estudiantes con asistencia baja (menos del 50%), los promedios se reducen aún más, llegando a cerca del 65% en el grupo experimental y 50% en el grupo control. Esta tendencia muestra, por un lado, que la asistencia sigue siendo un factor determinante para el rendimiento en ambos grupos y, por otro, que el modelo pedagógico implementado en el grupo experimental ofrece una ventaja sostenida: incluso con niveles de asistencia media o baja, sus promedios se mantienen sistemáticamente por encima de los del grupo control.

Las estructuras selectivas y repetitivas fueron las áreas de menor rendimiento en ambos grupos. Aun así, el grupo experimental mostró mejores resultados: 51.81% y 58.00%, frente a 19.17% y 46.32% del grupo control. El grupo experimental alcanzó un 85% de cumplimiento de los objetivos curriculares del curso, frente al 60% del grupo control.

En la retroalimentación cualitativa de estudiantes se categorizaron los comentarios de los estudiantes en función de cuatro dimensiones: claridad metodológica, utilidad de ejercicios, dificultades y conexión teoría-práctica. La mayoría de los comentarios positivos provinieron del grupo experimental.

4. DISCUSIÓN

Los resultados evidencian que la integración del pensamiento computacional y la programación modular tuvo un efecto positivo y sostenido en el rendimiento de los

estudiantes universitarios. El grupo experimental mostró avances consistentes en competencias asociadas al diseño algorítmico, la abstracción y la resolución de problemas complejos. Estos avances se corresponden con lo planteado por Wing (2006) y con los aportes de Zapata-Ros (2019), quienes destacan que el pensamiento computacional favorece la descomposición, la abstracción y la organización lógica del razonamiento. Asimismo, Villalustre & Cueli (2023) demostraron que el fortalecimiento del pensamiento computacional mejora el rendimiento en algoritmos con condicionales y bucles, contenidos que suelen presentar mayor complejidad en los primeros cursos de programación.

A esta evidencia se suma un estudio reciente publicado en Education Sciences, el cual reporta una correlación significativa entre el desarrollo del pensamiento computacional y un mejor desempeño en áreas STEM (ciencia, tecnología, ingeniería y matemáticas), especialmente en temáticas que requieren altos niveles de análisis (Bounou et al., 2023). Este hallazgo refuerza el comportamiento observado en el grupo experimental, particularmente en los contenidos avanzados de C++ que demandan abstracción y razonamiento estructurado.

La progresión estable del grupo experimental respalda la idea de que la programación modular favorece una secuenciación de contenidos más clara y un aprendizaje acumulativo, en línea con lo sugerido por Ricciardi (2024) y Alvarez et al. (2023). En contraste, la variabilidad del grupo control, junto con la disminución de su participación hacia el final del semestre, refleja las limitaciones de modelos centrados exclusivamente en la transmisión de contenidos. En esa misma dirección, Zitha et al. (2023) advierten que los enfoques tradicionales, basados en clases magistrales y currículos de “talla única”, tienden a generar desinterés y menor compromiso estudiantil, mientras que las estrategias innovadoras y colaborativas incrementan la participación y favorecen el dominio de conceptos clave.

El análisis entre asistencia y desempeño confirma la relevancia del trabajo guiado y la continuidad en la práctica. La relación entre una asistencia alta y mejores resultados en temáticas avanzadas sugiere que el modelo no solo mejora el aprendizaje técnico, sino que también incrementa la permanencia y el compromiso académico. Este patrón coincide con lo expuesto por (Vásquez et al., 2022), quienes señalan que el progreso sostenido en actividades cognitivamente demandantes, como la resolución estructurada de problemas, depende de la participación constante y del acompañamiento metodológico.

Pese a los avances, en ambos grupos se identificaron áreas de dificultad recurrente, como estructuras selectivas y repetitivas. Estas temáticas combinan un alto grado de abstracción con una aplicación inmediata, lo que coincide con los hallazgos de Díaz et al. (2023), quienes evidencian que los estudiantes presentan mayores obstáculos en tareas que requieren razonamiento lógico y procesos cognitivos de alto nivel. Esto plantea la necesidad de incorporar recursos complementarios como visualizadores de código o simuladores interactivos para reforzar la comprensión y reducir la carga cognitiva.

Asimismo, investigaciones recientes con docentes, como la de Tongal et al. (2024), muestran que el fortalecimiento del pensamiento computacional se asocia con prácticas pedagógicas creativas, colaborativas y orientadas a la resolución de problemas, elementos que dialogan con la configuración del modelo implementado y con el rol activo que asumió el profesorado en su aplicación. En conjunto, los resultados confirman que un enfoque pedagógico basado en pensamiento computacional y programación modular puede mejorar el rendimiento, reducir la deserción y fortalecer la motivación estudiantil en contextos de educación superior. La estructura modular favoreció una comprensión secuencial y aplicada de los contenidos, incluso en temáticas de alta demanda cognitiva, y permitió que los estudiantes desarrollaran competencias clave como la abstracción, el razonamiento lógico y la resolución de problemas complejos. Aunque persistieron dificultades en áreas como las estructuras selectivas y repetitivas, estos desafíos abren la posibilidad de incorporar herramientas visuales y recursos interactivos que reduzcan la carga cognitiva. En síntesis, los hallazgos posicionan este estudio como un avance inicial en la validación de un modelo pedagógico flexible y transferible a otras asignaturas y contextos académicos.

5. CONCLUSIÓN

La integración de metodologías activas, aprendizaje basado en juegos y desafíos Bebras mejora la comprensión de conceptos C++ al promover la abstracción, análisis y resolución estructurada de problemas, alineándose con competencias tecnológicas del siglo XXI.

Este enfoque innovador demuestra viabilidad en educación técnica superior boliviana, con potencial transferible a otras asignaturas STEM, al fomentar el compromiso estudiantil y la permanencia académica frente a modelos tradicionales.

Doi: https://doi.org/10.63688/vn145s84

La propuesta posiciona al pensamiento computacional como solución estratégica para contextos con deserción elevada, respaldada por evidencias locales (2016-2023) y tendencias internacionales en innovación educativa.

REFERENCIAS BIBLIOGRÁFICAS

Acevedo-Borrega, J., Valverde-Berrocoso, J., & Garrido-Arroyo, M. C. (2022). Computational thinking and educational technology: A scoping review of the literature. *Education Sciences, 12*(1), 39. https://doi.org/10.3390/educsci12010039

Alvarez, C., Samary, M. M., & Wise, A. F. (2023). Modularization for mastery learning in CS1: A 4-year action research study. *Journal of Computing in Higher Education, 36*(2), 1–??. https://doi.org/10.1007/s12528-023-09366-1

Asunda, P., Faezipour, M., Tolemy, J., & Do Engel, M. T. (2023). Embracing computational thinking as an impetus for artificial intelligence in integrated STEM disciplines through engineering and technology education. *Journal of Technology Education, 34*(2), 43–63. https://doi.org/10.21061/jte.v34i2.a.3

Barragán, E. A. (2023). Pensamiento computacional y programación en la formación de estudiantes desde edades tempranas. *Revista Educación, 47*(2), 775–793. https://doi.org/10.15517/revedu.v47i2.53645

Bebras, C. (2025). *Bebras*. https://www.bebras.org/

Bounou, A., Lavidas, K., Komis, V., Papadakis, S., & Manoli, P. (2023). Correlation between high school students' computational thinking and their performance in STEM and language courses. *Education Sciences, 13*(11), 1101. https://doi.org/10.3390/educsci13111101

Caisaguano Villa, G. G., & Apolo Buenaño, D. E. (2025). Integración del pensamiento computacional en contextos educativos: Revisión sistemática de literatura en artículos de acceso abierto de los últimos 10 años en SCOPUS. *Reincisol, 4*(8), 339–369. https://doi.org/10.59282/reincisol.v4(8)339-369

Díaz, O. R. F., Lechuga, G. D., González, M. E., & Rodríguez, A. A. C. (2023). Pensamiento computacional versus pensamiento matemático: Correlación en aprendizaje de estudiantes de educación media en Colombia. *Revista de Ciencias Sociales, 29*(3), 98–111. https://doi.org/10.31876/rcs.v29i3.40700

Leibovitch, Y. M., Beencke, A., Ellerton, P. J., McBrien, C., Robinson-Taylor, C. L., & Brown, D. J. (2025). Teachers' (evolving) beliefs about critical thinking education during professional learning: A multi-case study. *Thinking Skills and Creativity, 56*. https://doi.org/10.1016/j.tsc.2024.101725

Ricciardi, A. (2024). Programming fundamentals: The power of modular development. *DEV Community*. https://dev.to/alex_ricciardi/programming-fundamentals-the-power-of-modular-development-4c6d

Tejera-Martínez, F., Aguilera, D., & Vílchez-González, J. M. (2020). Lenguajes de programación y desarrollo de competencias clave: Revisión sistemática. *Revista Electrónica de Investigación Educativa, 22*, 1–12. https://doi.org/10.24320/redie.2020.22.e27.2869

Tongal, A., Yıldırım, F. S., Özkara, Y., Say, S., & Erdoğan, Ş. (2024). Examining teachers' computational thinking skills, collaborative learning, and creativity within the framework of sustainable education. *Sustainability, 16*(22), 9839. https://doi.org/10.3390/su16229839

Vásquez, Q., Jacinto, A., Tarrillo, H., & Enrique, H. (2022). Resolución de problemas con el método matemático de Polya: La aventura de aprender. *Revista de Ciencias Sociales, 28*(Especial 5), 75–86. https://doi.org/10.31876/rcs.v28i.38146

Villalustre, L., & Cueli, M. (2023). Assessing the computational thinking of pre-service teachers: A gender and robotics programming experience analysis. *Education Sciences, 13*(10), 1032. https://doi.org/10.3390/educsci13101032

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2017). Computational thinking's influence on research and education for all. *Italian Journal of Educational Technology, 25*(2), 7–14. https://doi.org/10.17471/2499-4324/922

Yadav, A. (2019). Computer science pedagogical content knowledge: Characterizing teacher performance. *ACM Transactions on Computing Education, 19*. https://doi.org/10.1145/3303770

Zapata-Ros, M. (2019). Computational thinking unplugged. *Education in the Knowledge Society, 20*(1). https://doi.org/10.14201/eks2019_20_a18

Zitha, I., Mokganya, G., & Sinthumule, O. (2023). Innovative strategies for fostering student engagement and collaborative learning among extended curriculum programme students. *Education Sciences, 13*(12), 1196. https://doi.org/10.3390/educsci13121196

Conflicto de Intereses: Los autores afirman que no existen conflictos de intereses en este estudio y que se han seguido éticamente los procesos establecidos por esta revista. Además, aseguran que este trabajo no ha sido publicado parcial ni totalmente en ninguna otra revista.

FINANCIAMIENTO

Los autores no recibieron financiamiento para el desarrollo de esta investigación.

CONTRIBUCIÓN DE AUTORÍA:

Nombres de autores e iniciales: Tania Karina Ponce Torca (TKPT).

1. Conceptualización: (TKPT)
2. Curación de datos: (TKPT)
3. Análisis formal: (TKPT)
4. Adquisición de fondos: (TKPT)
5. Investigación: (TKPT)
6. Metodología: (TKPT)
7. Administración del proyecto: (TKPT)
8. Recursos: (TKPT)
9. Software: (TKPT)
10. Supervisión: (TKPT)
11. Validación: (TKPT)
12. Visualización: (TKPT)
13. Redacción – Borrador original: (TKPT)
14. Redacción – Revisión y edición: (TKPT)