

Inteligencia artificial: retos y oportunidades en los cursos de programación para ingeniería

Esteban Velilla Hernández, Juan B. Cano Quintero, Jaime A. Valencia Velásquez

**Universidad de Antioquia
Medellín, Colombia**

Resumen

La Inteligencia Artificial (IA) está transformando la enseñanza y uso de la programación en cursos de pregrado, posgrado y en la vida cotidiana, planteando tanto desafíos como oportunidades. El acceso inmediato a bases de datos y el uso de herramientas de IA, como generadores automáticos de código a partir de instrucciones lingüísticas comunes, han modificado el proceso tradicional de aprendizaje. Por un lado, estas tecnologías pueden facilitar la comprensión y uso de conceptos complejos, disminuyendo los tiempos en la solución de problemas. En este sentido, herramientas como ChatGPT, GitHub Copilot y otros sistemas avanzados pueden generar código funcional a partir de instrucciones básicas, facilitando la resolución de problemas en eventos prácticos de programación, proyectos colaborativos y “hackatones” académicos. Por otro lado, plantean interrogantes sobre la autonomía del estudiante y el desarrollo de habilidades críticas en el pensamiento algorítmico y lógico. Adicionalmente, cada vez el uso de lenguajes orientados a objetos, enmascaran significativamente la lógica de programación tradicional, dando como resultado un nuevo paradigma de programación sin la necesidad de escribir muchas líneas de código, simplemente aprovechando las bondades de este paradigma para resolver problemas enlazando o uniendo objetos según la jerarquía (como lo podemos evidenciar en programas como Scratch, GoogleSheets, o incluso librerías de Python como Pywalk, entre otras). Este ensayo analiza cómo la integración de la IA puede complementar las metodologías educativas tradicionales en la formación de futuros ingenieros, permitiendo un aprendizaje más personalizado y enfocado en la creatividad y solución de problemas. También se examinan los riesgos asociados, como la dependencia excesiva de herramientas automáticas y la necesidad de ajustar las estrategias de evaluación. La adopción efectiva de la IA en la enseñanza de la programación requiere un equilibrio entre el aprovechamiento de estas tecnologías y el fortalecimiento de habilidades fundamentales que aseguren la formación de profesionales competentes y reflexivos, permitiendo a los estudiantes enfocarse en el análisis crítico de códigos generados y su integración en proyectos que desafíen a los estudiantes a evaluar, adaptar y optimizar las soluciones sugeridas por la IA, de una manera más consciente de los principios esenciales detrás del código.

Por tanto, se describe una estrategia didáctica para la incorporación de estas herramientas que pueda ser utilizada en entornos de programación competitiva como hackatones o cursos de manera general.

Palabras clave: Inteligencia artificial (IA) en educación; Cursos de computación; estrategia didáctica

Abstract

Artificial Intelligence (AI) is transforming programming education in undergraduate and graduate courses, as well as its application in daily life, presenting both challenges and opportunities. Instant access to vast datasets and the proliferation of AI tools—such as automatic code generators responding to natural language prompts—are reshaping traditional learning processes. On one hand, these technologies can facilitate understanding and applying complex concepts, potentially reducing problem-solving time. Tools like ChatGPT, GitHub Copilot, and other advanced systems can generate functional code from basic instructions, aiding problem-solving during hands-on programming events, collaborative projects, and academic hackathons. On the other hand, they raise questions about student autonomy and the development of critical skills in algorithmic and logical thinking. Furthermore, the widespread adoption of object-oriented programming and high-level libraries can obscure traditional procedural logic. This fosters a paradigm where solutions are often constructed by integrating existing objects or library functions (e.g., using visual block-based tools like Scratch or leveraging extensive libraries within languages like Python), rather than writing large volumes of imperative code. This essay analyzes how AI integration can complement traditional methodologies in engineering education, potentially enabling more personalized learning focused on creativity and complex problem-solving. It also examines associated risks, such as excessive reliance on automated tools and the need to adjust assessment strategies. Effectively adopting AI in programming education requires balancing technology leverage with the reinforcement of fundamental skills to ensure the development of competent and reflective professionals. This approach allows students to concentrate on the critical analysis and integration of AI-generated code. By evaluating, adapting, and optimizing AI-proposed solutions within challenging projects, students can foster a deeper understanding of fundamental programming principles.

Therefore, a didactic strategy is described for incorporating these tools, applicable in competitive programming environments like hackathons and general-purpose programming courses.

Keywords: Artificial Intelligence (AI) in education; programming education; didactic strategies

1. Introducción

La Inteligencia Artificial (IA) está en boca de la humanidad desde hace ya un buen rato, pero como un recurso o aplicación a disposición del público en general, es relativamente nueva y se ha convertirse en una herramienta clave en múltiples áreas del conocimiento, incluyendo la educación.



En los cursos de programación para ingeniería, su incorporación puede ofrecer nuevas posibilidades para optimizar el proceso de enseñanza-aprendizaje o por el contrario puede considerarse como un obstáculo, ya que abre todas las posibilidades de hacer “trampa” o “copiar” las respuestas de tareas que se han planteado en cursos de programación hasta el momento. Este ensayo explora los retos y oportunidades que representa la IA en este contexto, considerando tanto su potencial como las precauciones necesarias para su implementación y partiendo de las consideraciones pedagógicas y posibles panoramas futuros de los recursos computacionales que se usaran los actuales estudiantes de ingeniería (Unesco, 2021).

En el ámbito educativo, la IA actual ha permitido el desarrollo de aplicaciones que van desde asistentes virtuales hasta sistemas de tutoría inteligente. Los asistentes virtuales pueden considerarse como “programadores eficientes” y con recursos normalizados de documentación y buenas prácticas en escritura de códigos siguiendo estándares de programación con la finalidad de que éstos sean legibles, interpretables y mantenibles (uso de clases, funciones, etc), lo cual plantea una transformación educativa que cambie la óptica de los recursos relacionados con la programación de computadores y la computación numérica en general exigiendo una adaptación del meso y microcurrículo.

En el panorama de los cursos de programación en Ingeniería se enfocan en enseñar lenguajes de programación, estructuras de datos y algoritmos. Sin embargo, la enseñanza muchas veces se enfrenta a dificultades como la diversidad de niveles previos, la desmotivación y la falta de realimentación oportuna. La programación de computadores y el desarrollo de aplicaciones cada vez es más dependiente de códigos disponibles y recurso que se desarrollan frecuentemente (Imam & Alnsour, 2019), lo cual plantea la necesidad de una educación más personalizada y dinámica.

Entre las múltiples dimensiones y oportunidades de la IA en la enseñanza de programación (Bagnato et al., 2023) se puede plantear los siguientes aspectos que se consideran relevantes:

Personalización del aprendizaje: Si bien existen plataformas especializadas como Khan Academy o Coursera utilizan, entre otras, los recursos de plataformas generales de IA como ChatGPT, Gemini, o DeepSeek, entre otras, pueden adaptarse como profesores personalizados en cursos de computación de pregrado (Straková & Válek, 2024).

Evaluación automática: Herramientas como CodeGrade, Mimir o AutoGrader permiten evaluar código en tiempo real, pero también es posible adaptar las IA generales para realizar este tipo de actividades (Finnie-Ansley et al., 2023).

Asistentes de codificación: Las AI permiten de manera general asistir a estudiantes en tiempo real sugiriendo líneas de código, o detectando errores comunes, o escribiendo funciones o clases específicas. También existen programas de uso libre como Trae que puede crear aplicaciones completas (Sajji et al., 2023).

Si bien la IA tiene beneficios y oportunidades como los planteados, también tienen retos y desafíos en su implementación de los cuales podemos resaltar los siguientes (Reunanen & Nieminen, 2024):



Ética: Es un reto y una necesidad urgente para generar en el estudiante su sentido ético personal en su aprendizaje y dejar de una vez las discusiones que aún se plantean en artículos sobre la originalidad o no de los códigos en las tareas propuestas en los cursos.

Desigualdad en el acceso: No todos los estudiantes o instituciones tienen los recursos para usar tecnologías avanzadas, pero eso es impedimento para que se adopten y apropien recursos de libre uso disponibles en la red.

Reducción del pensamiento crítico: La automatización excesiva puede llevar a una dependencia de las herramientas y limitar el desarrollo del razonamiento lógico, puede ser una hipótesis, pero también se puede plantear que el uso de la automatización permite una mayor efectividad en el logro de objetivos y reduce tareas repetitivas (Tseng & Warschauer, 2023).

Así con los recursos de IA la perspectiva ya es una integración aún más profunda de la IA en la enseñanza y un modelo de **Programación conversacional** (Desmond et al., 2022), por darle un nombre, donde el estudiante "dialoga" con el entorno de codificación y valida sus resultados o ajusta el código a ciertos detalles.

2. Propuesta metodológica implementada

La figura 1 expone un modelo conceptual diseñado para el aprendizaje de la computación numérica, estructurado desde elementos generales y complejos, como los datos provenientes del mundo real, hasta componentes más simples y específicos, como las variables numéricas y los comandos computacionales que se muestran en más detalle con la figura 2. Este enfoque, desarrollado en el contexto de la Universidad de Antioquia, permite visualizar de manera clara cómo se transita desde la adquisición de datos hasta la aplicación de técnicas numéricas fundamentales.

En la parte superior derecha de la figura 1 se encuentran diversas fuentes de datos abiertos y experimentales, como Kaggle, el repositorio de aprendizaje automático de la Universidad de California en Irvine (UCI), el Laboratorio Nacional de Energía Renovable (NREL), la NASA y la plataforma de XM para la consulta de generación y precios de energía eléctrica. Estas plataformas proporcionan conjuntos de datos reales que sirven como punto de partida para el análisis numérico, de igual manera, algunas de estas ofrecen la opción de consultas personalizadas para acceder a las diferentes bases de datos y variables disponibles (queries). Justo debajo, se representan los diferentes tipos de estructuras de datos que permiten organizar la información recopilada. Se destacan los arrays (arreglos), vectores, matrices y escalares que conforman el paradigma del software Matlab para la manipulación de variables, así como la utilización de estructuras de tipo DataFrame de la biblioteca pandas de Python, ampliamente usadas en ciencia de datos para manejar datos tabulares.

En este sentido, se resalta que según el programa computacional a utilizar (Matlab o Python por ejemplo), las variables y datos deben llevarse a estructuras de datos que faciliten su procesamiento. Estos datos pueden provenir de diferentes fuentes como equipos de medición como osciloscopios,



estaciones meteorológicas, escáner 3D, cámaras, sensores involucrados en dispositivos móviles como celulares permitiendo hacer uso de los datos registrados por los acelerómetros, cámaras, micrófonos, sensores de orientación, GPS, entre otros y que en el caso de aplicaciones como MATLAB Mobile facilita la creación de bases de datos, lo que refuerza la conexión entre el entorno físico y el entorno computacional. Por otro lado, otra forma de generar u obtener datos es a partir de los procesos numéricos de evaluar funciones o modelos computacionales, que de igual manera, hace uso de estructuras de datos específicas para su procesamiento.

En el centro de la figura 1, se ilustra una matriz que simboliza el conjunto de variables numéricas obtenidas a partir de los datos originales. Estas variables constituyen el núcleo a partir del cual se desarrollan las diferentes técnicas de computación numérica que tradicionalmente se abordan en cursos como métodos o computación numéricos. Desde este nodo central se irradian varias ramas que representan distintos ejes temáticos fundamentales en este campo.

Así un primer trabajo que se plantea es la lectura de una base de datos y su exploración numérica y gráfica, dando una introducción a las plataformas de desarrollo como Matlab y Python consideradas en los cursos de formación de ingeniería. Por otro lado, se hace una introducción al uso de diferentes herramientas de IA para generar códigos y analizar datos, tales como Gemini integrado en Colab de Google. De esta manera, estas herramientas indican las jerarquías a través de las secciones de código (lectura, procesamiento, visualización, conclusiones, etc), fundamentales para entender la lógica computacional del ejercicio numérico y sus implicaciones computacionales.

En la parte superior izquierda, se aborda el tema de los sistemas lineales, esenciales para la resolución de múltiples problemas en ingeniería y ciencia, donde se trabaja con ecuaciones de la forma $A \cdot x = b$, sistemas lineales matriciales que son omnipresentes en el procesamiento de información y modelos matemáticos. En la parte inferior izquierda, se observa la representación de datos, que incluye visualizaciones en dos y tres dimensiones para facilitar la interpretación de los resultados.

En la parte inferior central, se encuentra la interpolación y el ajuste de curvas, herramientas clave para estimar valores desconocidos a partir de datos conocidos. Este bloque se conecta con el tema de integración y derivación numérica, elementos fundamentales del cálculo que permiten modelar fenómenos dinámicos y acumulativos.

A continuación, en la parte inferior derecha, se abordan las ecuaciones diferenciales ordinarias (EDO) y en derivadas parciales (EDP), cuya resolución numérica es indispensable para simular sistemas físicos complejos. Finalmente, el bloque de optimización, ubicado en la esquina inferior izquierda, permite encontrar soluciones óptimas bajo restricciones dadas, siendo una técnica ampliamente utilizada en ciencia de datos, ingeniería y economía.

En conjunto, esta figura propone una ruta de aprendizaje para la computación numérica, que inicia en la realidad empírica a través de la adquisición y organización de datos, y culmina en la aplicación de técnicas numéricas avanzadas, promoviendo así una comprensión integral del campo. Los lenguajes de programación y sus elementos básicos presentados de forma esquemática en la Figura 2, los cuales muestran la forma de lógica de procesamiento realizado por el

computador a partir de las diferentes entradas para producir una salida determinada. En este sentido se resaltan los elementos fundamentales de programación que se deben reforzar e identificar en los códigos (tipos de variables, estructura de programación, etc.), así estos sean creados por asistentes de IA.

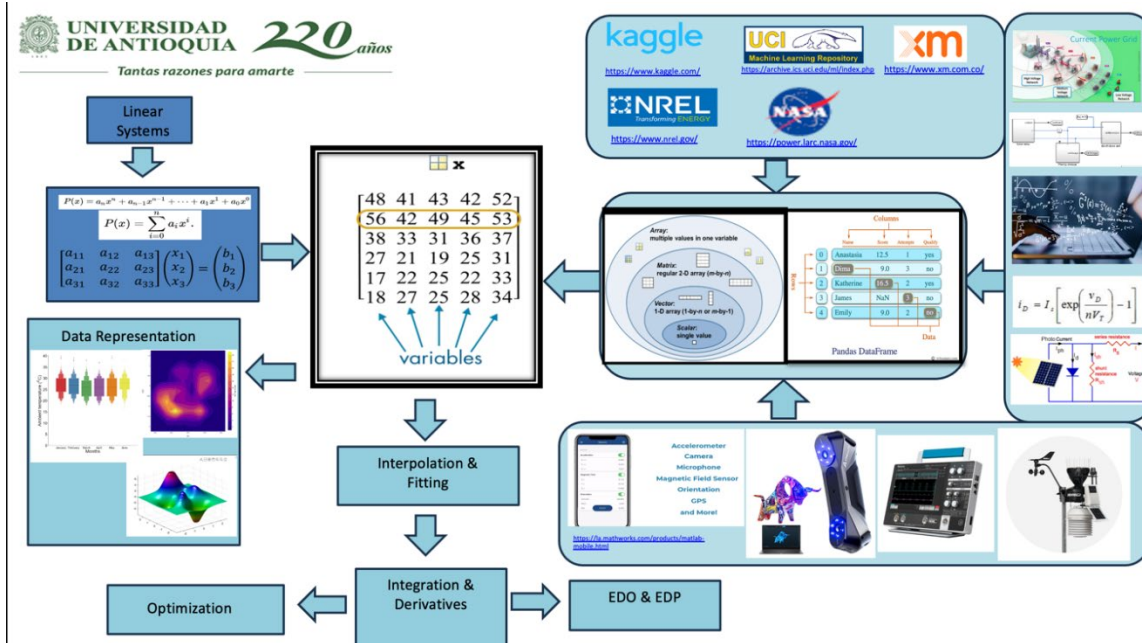


Figura 1. Resumen grafico de propuesta metodológica.

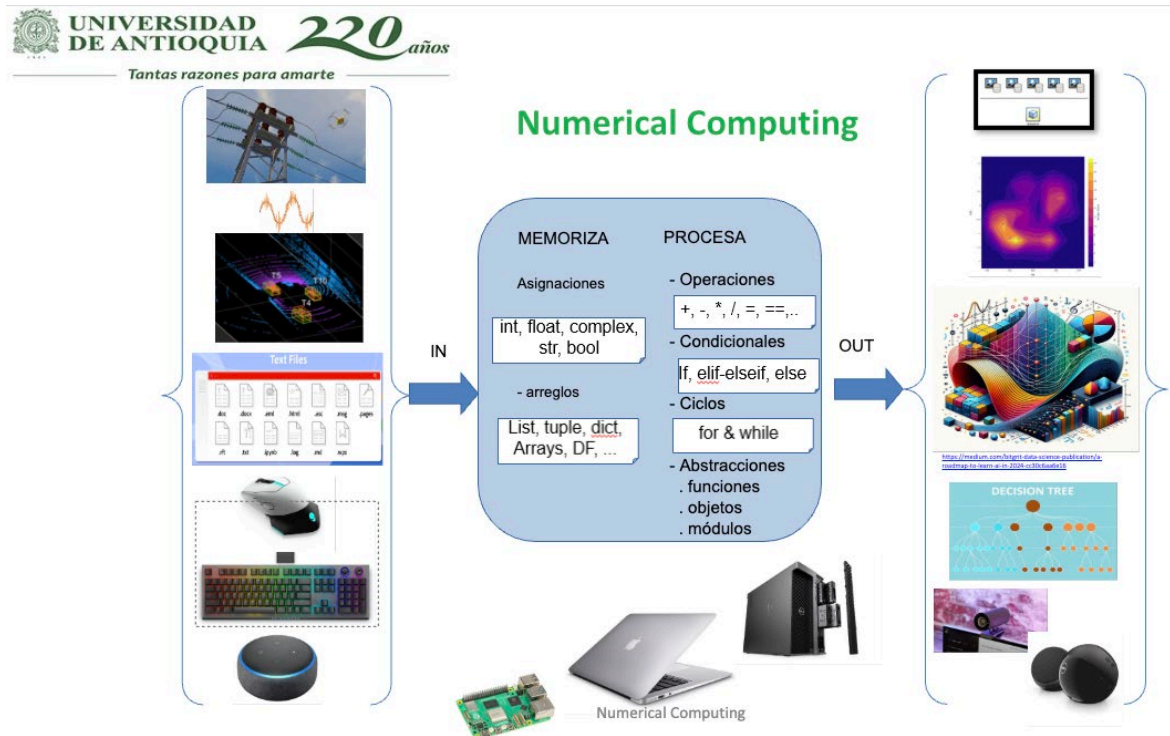


Figura 2. Conceptualización del computador y elementos básicos de programación.



3. Resultados parciales

Los resultados de esta propuesta metodológica los consideramos parciales, ya que el tiempo que consideramos para una valoración completa es de al menos 2 años, de los cuales solo se han completado al momento un año. La implementación de esta metodología de enseñanza-aprendizaje en el curso de computación numérica, basada en una progresión desde problemas reales y complejos hasta componentes computacionales simples y fundamentales, ha generado una serie de resultados positivos en el desarrollo académico, técnico y profesional de los estudiantes de ingeniería, entre los que describimos:

1. Mayor comprensión contextual de la computación numérica

Al iniciar el proceso con datos reales provenientes de fuentes reconocidas como la NASA, Kaggle o sensores móviles, los estudiantes comprenden desde el principio la relevancia práctica de la computación numérica. Este enfoque contextualiza el aprendizaje, facilitando la conexión entre la teoría matemática y su aplicación en el mundo real. De igual manera, permite la estructuración lógica de los pasos que se deben seguir para la interpretación de los datos involucrados en las bases de datos. De esta manera, se hace énfasis en la lectura de archivos o consultas de bases de datos, funciones específicas según el lenguaje computacional y la estructura de datos involucrada en cada uno de estos, como por ejemplo en el caso de pandas de Python la estructura DataFrame. Adicionalmente, una vez establecida la estructura de datos, se enfatiza en la forma de procesar los datos (filtrar, seleccionar, eliminar, etc) para luego posteriores análisis incluyendo la representación gráfica. De esta manera, se hace una introducción general a los programas computacionales, estructuras de datos, funciones y librerías requeridas para cumplir con los objetivos de la práctica de bases de datos.

2. Desarrollo de pensamiento analítico y resolución de problemas

La progresión hacia variables, estructuras de datos y técnicas como interpolación, integración, optimización y resolución de ecuaciones diferenciales, fomenta en los estudiantes el pensamiento lógico y la capacidad de resolución de problemas complejos, competencias fundamentales en la formación de ingenieros.

3. Fortalecimiento del aprendizaje activo y autónomo

El uso de herramientas accesibles como MATLAB Mobile y entornos de datos abiertos promueve la autonomía del estudiante, permitiéndole experimentar con sus propios datos y avanzar en su proceso de aprendizaje a su propio ritmo. Este enfoque favorece también el aprendizaje activo y el trabajo por proyectos.

4. Mejora de la alfabetización en ciencia de datos y programación

Al integrar conceptos como matrices, arrays, estructuras DataFrame y visualización de datos, los estudiantes desarrollan habilidades clave en análisis numérico, manejo de datos y programación científica, áreas cada vez más demandadas en la ingeniería moderna.

5. Interdisciplinariedad y motivación

La diversidad de fuentes de datos y áreas de aplicación (energía, medio ambiente, movilidad, salud, etc.) facilita la conexión de la computación numérica con otras disciplinas, aumentando la motivación del estudiante al ver que sus conocimientos tienen impacto en múltiples contextos.

6. Preparación para la industria y la investigación

Esta metodología prepara a los futuros ingenieros no solo para enfrentar problemas técnicos con herramientas numéricas avanzadas, sino también para trabajar con datos reales en entornos industriales o de investigación, una habilidad crucial en la ingeniería del siglo XXI.

Tabla 1. Valoración parcial de resultados

Afirmación	Totalmente de acuerdo	De acuerdo	Indiferente	En desacuerdo	Totalmente en desacuerdo
1. Mayor comprensión contextual de la computación numérica	85.0%	13.0%	2.0%	0.0%	0.0%
2. Desarrollo de pensamiento analítico y resolución de problemas	74.0%	11.0%	10.0%	5.0%	0.0%
3. Fortalecimiento del aprendizaje activo y autónomo	88.0%	12.0%	0.0%	0.0%	0.0%
4. Mejora de la alfabetización en ciencia de datos y programación	95.0%	5.0%	0.0%	0.0%	0.0%
5. Interdisciplinariedad y motivación	70.0%	5.0%	17.0%	3.0%	5.0%
6. Preparación para la industria y la investigación	55.0%	35.0%	5.0%	5.0%	0.0%

La Tabla 1 resume los resultados de las encuestas que consideramos parciales, ya que, como se dijo previamente, la implementación de esta metodología solo lleva al momento 1 año y la valoración real en estas habilidades las hemos podido obtener con egresados en otras investigaciones sobre metodologías anteriormente. (Valencia et al 2019)

4. Conclusiones

En conjunto, la implementación de esta metodología puede enriquecer significativamente el proceso formativo de los estudiantes de ingeniería, al promover un aprendizaje más profundo, aplicado y conectado con la realidad, preparándolos para enfrentar con solvencia los desafíos técnicos y sociales de su futuro profesional y los resultados reales solo podrán valorarse cuando este grupo de jóvenes este afrontando de manera profesional los desafíos de la ingeniería actual

5. Referencias

Artículos de revistas

- Bagnato, A., Cicchetti, A., Berardinelli, L., Bruneliere, H., & Eramo, R. (2023). AI-augmented Model-Based Capabilities in the AIDOaRt Project. *ACM SIGAda Ada Letters*, 42(2), 99–103. <https://doi.org/10.1145/3591335.3591349>
- Desmond, M., Duesterwald, E., Isahagian, V., & Muthusamy, V. (2022). A No-Code Low-Code Paradigm for Authoring Business Automations Using Natural Language. <http://arxiv.org/abs/2207.10648>
- Finnie-Ansley, J., Denny, P., Luxton-Reilly, A., Santos, E. A., Prather, J., & Becker, B. A. (2023). My AI Wants to Know if This Will Be on the Exam: Testing OpenAI's Codex on CS2 Programming Exercises. *ACM International Conference Proceeding Series*, 97–104. <https://doi.org/10.1145/3576123.3576134>
- Imam, A. T., & Alnsour, A. J. (2019). The Use of Natural Language Processing Approach for Converting Pseudo Code to C# Code. *Journal of Intelligent Systems*, 29(1), 1388–1407. <https://doi.org/10.1515/jisys-2018-0291>
- Reunanen, T., & Nieminen, N. (2024). Artificial Intelligence as a Catalyst: A Case Study on Adaptive Learning in Programming Education. *Human Factors, Business Management and Society*. <https://doi.org/10.54941/ahfe1004957>
- Sajji, A., Rhazali, Y., & Hadi, Y. (2023). A methodology of automatic class diagrams generation from source code using Model-Driven Architecture and Machine Learning to achieve Energy Efficiency. *E3S Web of Conferences*, 412, 01002. <https://doi.org/10.1051/e3sconf/202341201002>
- Straková, N., & Válek, J. (2024). Chatbots as a Learning Tool. *R&E-SOURCE*, 245–265. <https://doi.org/10.53349/resource.2024.is1.a1259>
- Tseng, W., & Warschauer, M. (2023). AI-writing tools in education: if you cant beat them, join them. *Journal of China Computer-Assisted Language Learning*, 3(2), 258–262. <https://doi.org/10.1515/jccall-2023-0008>

Libros

- UNESCO (2021). AI and education: Guidance for policymakers. [unesdoc.unesco.org](https://unesdoc.unesco.org/ark:/48223/pf0000376709). <https://unesdoc.unesco.org/ark:/48223/pf0000376709>

Memorias de congresos

- Encuentro Internacional de Educación en Ingeniería ACOFI 2024 (EIEI ACOFI 2024) <https://acofi.edu.co/eiei2024/memorias/>
- 2989 EXPERIENCIA DE MICRO CURRÍCULO EN MÉTODOS NUMÉRICOS PARA EL PROGRAMA DE INGENIERÍA ELÉCTRICA BASADO EN PROYECTOS DE AULA Y EN APRENDIZAJE BASADO EN PROBLEMAS. Jaime Alejandro Valencia Velásquez, Noé Alejandro Mesa Quintero. Encuentro Internacional de Educación en Ingeniería ACOFI 2019 https://acofi.cloudbiteca.com/pmb/opac_css/index.php?lvl=notice_display&id=1047

Fuentes electrónicas

- <https://www.codegrade.com/> Consultada 21 de abril 2025
- <https://www.mimirhq.com/classroom/test-cases> Consultada 18 de abril de 2025.



- <https://github.com/eecs-autograder/autograder.io/blob/master/README.md> Consultado 18 de abril del 2025.
- <https://autograder.ucsd.edu> Consultado 18 de abril del 2025.
- <https://www.trae.ai/> Consultado 30 de marzo de 2025.
- University of Hong Kong. (1997, June). Final report: Ad Hoc Group for Learning Technologies. Consultado el 21 de mayo de 2002 en http://www.hku.hk/caut/Homepage/itt/5_Reports/5_1AdHoc.htm

Sobre los autores

- **Esteban Velilla Hernández:** Ingeniero electricista, Máster en ingeniería, PhD en materiales de la universidad de Antioquia. Profesor titular. esteban.velilla@udea.edu.co
- **Juan Bernardo Cano Quintero:** Ingeniero Electrónico, Doctor en Ingeniería electrónica. Profesor Titular de la UdeA. bernardo.cano@udea.edu.co
- **Jaime A. Valencia Velásquez.** Ingeniero electricista, Ms Matemáticas, PhD en ingeniería eléctrica de la universidad Politécnica de Cataluña. Profesor titular. bernardo.cano@udea.edu.co

Los puntos de vista expresados en este artículo no reflejan necesariamente la opinión de la Asociación Colombiana de Facultades de Ingeniería.

Copyright © 2025 Asociación Colombiana de Facultades de Ingeniería (ACOFI)