UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Revista Científica

http://revistas.udistrital.edu.co/ojs/index.php/revcie/index

CIDC

Artículo de investigación científica y tecnológica

Rúbrica basada en competencias de aprendizaje en un curso CS1 para evaluar actividades de programación CSCL

Rubric Based on Learning Outcomes for a CS1 Course to CSCL Programming Activities

Rubrica baseada em competências de aprendizagem em um curso CS1 para avaliar atividades de programação CSCL

Carlos-Giovanny Hidalgo-Suárez[1]
Víctor-Andrés Bucheli-Guerrero[2]
Hugo-Armando Ordoñez-Erazo[3]

Recibido: abril de 2022 **Aceptado:** diciembre de 2022

Para citar este artículo: Hidalgo-Suárez, C. G., Bucheli-Guerrero, V. A., y Ordoñez-Erazo, H .A. (2023). Rúbrica basada en competencias de aprendizaje en un curso CS1 para evaluar actividades de programación CSCL. *Revista Científica*, *46*(1), 134-146. https://doi.org/10.14483/23448350.20095

Resumen

Los cursos programación (CS1) tienen la tasa de mortalidad académica más alta, esto se refleja en las bajas calificaciones de los estudiantes, lo que indica que no alcancen las competencias académicas. Buscando nuevas formas de mejorar el aprendizaje de los estudiantes del curso CS1, este artículo propone una rúbrica analítica basada en competencias académicas para actividades de programación colaborativa respaldada por una herramienta de evaluación automática de código fuente que permita mejorar las calificaciones y alcanzar las competencias propuestas en el curso. Se diseñó una rúbrica con 16 criterios de evaluación que se dividieron en tres actividades que fueron presentadas por un grupo experimental (GE) de 18 estudiantes y un grupo de control (GC) de 24 estudiantes. En el GE se usó la colaboración entre estudiantes, mientras que en el GC los estudiantes trabajan de manera individual. Luego de finalizar las actividades, usando una rúbrica, se evaluaron las entregas de cada estudiante para identificar si logra los resultados de aprendizaje esperados por el curso. Los resultados demuestran que el uso de la colaboración mejora la adquisición de las competencias de aprendizaje en un 17 % más que si lo hacen de manera individual. Además, se destacan otras habilidades sociales asociadas a la colaboración, como amistad, motivación y entendimiento grupal. El desarrollo de estrategias que permita evaluar las competencias, no solo indica que el estudiante logra adquirir una habilidad, sino que también permite al estudiante identificar sus falencias en las tareas de programación.

Palabras clave: aprendizaje colaborativo; curso de programación CS1; educación tecnológica; evaluación de resultados de aprendizaje; programación de computadores.

1. Universidad del Valle (Cali-Valle del Cauca, Colombia). carlos.hidalgo@correounivalle.edu.co
2. Universidad del Valle (Cali-Valle del Cauca, Colombia). victor.bucheli@correounivalle.edu.co
3. Universidad del Cauca (Popayán-Valle, Colombia). hugoordonez@unicauca.edu.co

Abstract

Programming courses (CS1) have the highest academic mortality rate, this is reflected in the low grades of the students, which indicates that the students do not reach the academic competencies. In this sense, looking for new ways to improve the learning of the students of the CS1 course, this article proposes an analytical rubric based on academic competencies for collaborative programming activities supported by an automatic source code evaluation tool that allows to improve the qualifications and reach the competencies proposed in the course. A rubric was designed with 16 evaluation criteria that were divided into three activities which were presented by an experimental group (EG) of 18 students and a control group (CG) of 24 students. In the GE, a collaboration between students was used, while in the CG, students work individually. After finishing the activities, using an analytical rubric, the deliveries of each student were evaluated to identify if they achieved the learning results expected by the course. The results show that the use of collaboration achieves that students manage to win a learning competition in 17 % more than if they do it individually. In addition, other social skills associated with collaboration are highlighted, such as friendship, motivation and group understanding. The development of strategies that allow to evaluate the competences, not only indicate that the student manages to acquire a skill, but also allows the student to identify their shortcomings in the programming tasks.

Keywords: collaborative learning; computer programming; CS1 programming course; learning outcomes assessment; technology education.

Resumo

Os cursos de programação (CS1) têm a maior taxa de mortalidade acadêmica, isso se reflete nas notas baixas dos alunos, o que indica que eles não atingem as competências acadêmicas. o curso. Foi elaborada uma rubrica com 16 critérios de avaliação que foram divididos em três atividades que foram apresentadas por um grupo experimental (GE) de 18 alunos e um grupo controle (GC) de 24 alunos. No GE foi utilizada a colaboração entre os alunos, enquanto no GC os alunos trabalham individualmente. Após o término das atividades, por meio de uma rubrica, foram avaliadas as entregas de cada aluno para identificar se atingiram os resultados de aprendizagem esperados pelo curso. Os resultados mostram que o uso da colaboração melhora a aquisição de habilidades de aprendizagem em 17% a mais do que se o fizessem individualmente. Além disso, destacam-se outras habilidades sociais associadas à colaboração, como amizade, motivação e compreensão do grupo. O desenvolvimento de estratégias que permitam avaliar as competências, não só indica que o aluno consegue adquirir uma habilidade, como também permite ao aluno identificar as suas deficiências nas tarefas de programação.

Palavras-chaves: aprendizagem colaborativa; avaliação dos resultados da aprendizagem; curso de programação CS1; educação tecnológica; programação de computadores.

Introducción

En ingeniería es habitual encontrar asignaturas relacionadas con el aprendizaje de lenguajes de programación, ya que su uso, y la programación como tal, se considera una competencia imprescindible para el futuro profesional de un ingeniero, de cualquier especialidad. Por este motivo, el diseño de actividades que involucren herramientas para la evaluación sumativa y formativa es de ayuda para impulsar cambios en el proceso de enseñanza y aprendizaje de la programación.

Es importante conocer las competencias adquiridas por cada estudiante, cada curso de programación contiene las competencias antes nombradas, habilidades, sensibilidades, logros de aprendizaje y criterios de evaluación que se espera que el estudiante reconozca y alcance al finalizar un curso. Para lograr que el estudiante alcance las competencias, se proponen actividades, proyectos y exámenes. Sin embargo, evaluar a los estudiantes es una tarea complicada y no está bien detallada. En este sentido, aparecen las rúbricas que permiten evaluar las partes del desempeño del estudiante, desglosando sus componentes para obtener una calificación total, además de determinar el estado

del desempeño e identificar fortalezas y debilidades que permitan al estudiante mejorar su desempeño (Dümmel, Westfechtel y Ehmann, 2018).

La colaboración como estrategia de aprendizaje en el aula ha tenido buenos resultados en los cursos de programación (Chen *et al.*, 2020; Mehennaoui *et al.*, 2014), mejorando desde aspectos académicos hasta habilidades a nivel personal. Sin embargo, identificar las habilidades que adquieren los estudiantes en el trabajo individual es una tarea complicada para el profesor (Böhne, Faltin y Wagner, 2007; Kardan y Sadeghi, 2015). Por esta razón es necesario apoyar este proceso colaborativo a través de herramientas que permitan evaluar y retroalimentar, considerando los logros de aprendizaje y las competencias académicas propuestas en el curso (Pereira *et al.*, 2019).

En la literatura uno de los enfoques de aprendizaje con mayor acogida es Computer Supported Collaborative Learning (CSCL) (Mehennaoui *et al.*, 2014; Stahl, Koschmann y Suthers, 2006), surge como apoyo al aprendizaje tradicional (Demaidi, Qamhieh y Afeefi, 2019; Lee, Fong y Gordon, 2013), busca controlar el proceso para lograr que los estudiantes obtengan conocimientos de aprendizaje de manera colaborativa (Ayala, Ortiz y Osorio, 2005; Lämsä *et al.*, 2021). Este enfoque se basa en la formación de grupos y cómo estos se pueden apoyar en la tecnología para mejorar el aprendizaje y la enseñanza, que sirven de apoyo al aprendizaje basado en la colaboración, pero estos procesos se pueden intervenir para adaptarlos a las necesidades (Chen *et al.*, 2018; Coto, Mora y Collazos, 2014).

En este artículo se propone una rúbrica para evaluar el proceso de enseñanza y aprendizaje en un curso de programación que desarrolla actividades colaborativas, la cual permite que los estudiantes conozcan los objetivos de aprendizaje, junto con los criterios de evaluación necesarios para alcanzar las competencias del curso.

Este artículo está organizado de la siguiente manera. La segunda sección presenta los trabajos relacionados. La tercera sección describe la metodología, se presentan en detalle las preguntas de interés, la selección de la muestra del curso CS1 y las fases de las pruebas experimentales realizadas. La cuarta sección presenta los resultados de las pruebas experimentales desarrolladas. La quinta sección discute los resultados del trabajo y, por último, se concluye el trabajo realizado.

Los resultados del aprendizaje se expresan como declaraciones de conocimiento y habilidades individuales que los estudiantes deben poseer al final del curso en el que se inscribieron. Es un enfoque integral para organizar y operar un sistema educativo que se centra en el éxito buscado por los estudiantes al final del ciclo de aprendizaje (Goss, 2022).

Los resultados de aprendizaje de un programa están establecidos por varios niveles del equipo de gestión académica. Hay tres componentes principales: resultado de aprendizaje del programa, resultado de aprendizaje del estudiante y resultado de aprendizaje del curso. Este artículo se centra en el resultado de aprendizaje del estudiante, que es el centro del aprendizaje y a quien están ligados los resultados esperados de un curso y de un programa (Alhazbi y Hassanh, 2013; Blikstein *et al.*, 2014; Clancy *et al.*, 2003).

La evaluación de resultados de aprendizaje en cursos de programación enfatiza la importancia de que cuando un estudiante alcanza las competencias y los logros necesarios, queda con bases sólidas que permiten continuar con la formación del plan de estudios y seguramente se forma un buen profesional para la industria. Sin embargo, la forma de evaluar las competencias no suele ser una tarea sencilla, ya que involucra diferentes elementos que pueden representar una tarea de mucho tiempo. En este sentido, se realiza una revisión de la literatura en la cual se encuentran diferentes estrategias para evaluar resultados de aprendizaje.

Una de las formas tradicionales de evaluar las competencias es a través de indicadores de logros, que son actividades evaluativas que tienen porcentaje que sumados dan el total del indicador de logro (TrainCom, s.f.). Si el estudiante cumple con

más del 50 % quiere decir que ha logrado el resultado de aprendizaje. Sin embargo, esta forma requiere mucho tiempo y esfuerzo por parte del profesor; además de dejar sesgos entre lo que el estudiante entrega y lo que se evalúa objetivamente. Algunas propuestas son las de OECD ILibrary, (s.f.), Sadler (2016) y Zlatkin-Troitschanskaia, Pant y Coates (2016).

Por otro lado están los evaluadores de logros de aprendizaje a través de listas de chequeo, las cuales permiten identificar si el estudiante logra un criterio de evaluación o no. Este método ha sido uno de los más usados, pero también uno de los más criticados, ya que al ser dicotómico es difícil conocer el porcentaje exacto que el estudiante alcanza en un logro de aprendizaje. Algunos trabajos en la revisión son Carbonaro (2019), Powell y Wimmer (2015) y Tiantong y Teemuangsai (2013).

En este mismo sentido, aparece la evaluación por semáforo, donde se puede medir si el estudiante alcanza una competencia o un logro de aprendizaje basado en tres niveles: bajo, medio y alto. Este método ha sido probado en diferentes trabajos (Collins, Weber y Zambrano, 2014; Fiesler, Garrett y Beard, 2020; Ishii, Gilbride y Stensrud, 2009; Saunders, 2012) y ha sido modificado usando cuatro categorías (Means *et al.*, 2009), también usando anotaciones y calificaciones dentro de las categorías (Michel, Cater y Varela, 2009), aplicación de autoevaluación por categorías (Bhuyan y Tamir, 2020).

También se encuentran las rúbricas, en las cuales se proponen diferentes niveles que pertenecen a un criterio de evaluación. La evaluación de la competencia o el logro de aprendizaje se alcanza en porcentaje con la suma de todos los criterios de evaluación. Este método de evaluación requiere un diseño que puede llevar mucho tiempo, pero es el que presenta mejores resultados para conocer el estado actual de los estudiantes. Algunos trabajos presentados son: Cateté, Snider y Barnes (2016), Lakas y Belkacem (2021), Saito *et al.* (2021), Wei, Saab y Admiraal (2021), en otros trabajos han usado rúbricas evaluativas (Mustapha *et al.*, 2016), rúbricas basadas en realimentación (Allen y Tanner, 2006), rúbricas basadas en razonamientos lógicos de competencias académicas (Palmer, Bach y Streifer, 2014).

Así mismo aparecen nuevas estrategias que combinan rúbricas con trabajo colaborativo. En Kilgour *et al.* (2020) se propone una rúbrica que permite evaluar las competencias académicas en un grupo, donde cada estudiante logre un criterio de evaluación y si es necesario puede recibir ayuda de sus compañeros de grupo. Por otro lado, Fleming (2008) muestra que el trabajo colaborativo aporta significativamente a la evaluación por competencias. Finalmente, en Allen y Knight (2009) se propone un método que integra el trabajo colaborativo, la formación de grupos y la evaluación de resultados de aprendizaje para mejorar un curso de programación. En todos los casos que se usó una rúbrica y la colaboración, se indicó que el estudiante logra identificar cuáles fueron sus falencias y mejorar su calificación.

Metodología

En esta sección se identifican las preguntas de interés que motivaron a integrar una rúbrica para evaluar resultados de aprendizaje de estudiantes de programación CS1. La rúbrica se implementó en dos grupos, uno para evaluar las competencias donde se usa la colaboración y otro para el trabajo individual. Se hizo la selección de los grupos de control y experimental que permitieron realizar experimentos para obtener los resultados.

Preguntas de interés

Este trabajo se realiza sobre el curso CS1, donde algunas actividades evaluativas son colaborativas, lo cual tiene cierto grado de dificultad para el profesor a la hora de evaluar los logros de aprendizaje que adquiere cada estudiante, en este sentido es importante que se integren nuevas estrategias que permitan evaluar actividades de programación individual y colaborativas, con el fin de apoyar a los estudiantes en su desempeño formativo. En

este artículo surge la pregunta de interés: ¿cómo evaluar competencias de aprendizaje en tareas de programación colaborativas de un curso CS1?

Población y muestra

El CS1 es el primer curso de programación del plan de estudios del programa de Ingeniería de Sistemas de la Universidad del Valle (Cali, Colombia). Este curso consta de 4 horas de clase por semana (2 teóricas y 2 prácticas) y 8 horas de trabajo autónomo. En total, los estudiantes tienen que estudiar 128 horas cada semestre (16 semanas). El objetivo general de este curso es aprender e implementar estructuras de datos y estructuras de control, así como desarrollar el pensamiento algorítmico para resolver problemas computacionales. El lenguaje de programación del curso es C++. La estructura curricular del curso se compone de 3 competencias académicas (C), 4 indicadores de logro (IL) y 19 criterios de evaluación (CE).

Cada semestre, aproximadamente 50 estudiantes toman el curso, en el semestre I del año 2019, 40 estudiantes se matricularon al curso CS1, los cuales fueron seleccionados para el experimento presentado en este artículo. Los 40 estudiantes fueron divididos en dos grupos de forma aleatoria: 24 estudiantes de control (grupo de control = GC) y 16 estudiantes de experimento (grupo experimental = GE).

Rúbrica para una actividad del curso CS1

Estructura de la rúbrica

En la Tabla 1 se propone la estructura de la rúbrica, en la cual se presentan jerárquicamente de mayor a menor: competencia, logro de aprendizaje y criterios de evaluación del curso CS1. Este diseño curricular se toma del curso de la Escuela de Ingeniería de Sistemas y Computación de la Universidad del Valle (García Retana, 2011). Cada competencia está ligada a uno o varios logros de aprendizaje, y estos están sujetos a los criterios de evaluación que los define el profesor del curso.

Tabla 1. Estructura de la rúbrica para el curso CS1

C: competencia del curso					
LA: logro de aprendizaje (%)					
CE: criterio de evaluación	Programador 100 %	Teikirisi 85 %	Puro Stack Overflow 65 %	Memento Quod Genus 35 %	Novato 0 %

Los niveles de evaluación serán acordes al criterio de evaluación, donde en cada nivel se presente claramente el concepto que se está midiendo. Teniendo en cuenta que esta rúbrica es diseñada para estudiantes de programación, hemos puesto niveles (ascendentes) que sean del agrado de los estudiantes, con el fin de hacer divertida la evaluación, definimos 5 niveles:

- Nivel 1. Novato: el estudiante que no tiene los conocimientos necesarios para lograr el mínimo del criterio de evaluación (su solución aparentemente no está relacionada con el problema en cuestión).
- Nivel 2. Memento Quod Genus: el estudiante necesita revisar las guías y presentaciones de clase para mejorar (muestra cierta comprensión de una parte del problema).
- Nivel 3. Puro Stack Overflow: es una comunidad, en la cual los desarrolladores pueden encontrar soluciones a problemas comunes de programación en diferentes lenguajes (la solución presenta algunos errores / parcialmente correcta).
- Nivel 4. Teikirisi: (del inglés *take it easy*, muchas veces en programación por el afán de entregar nos queda faltando o sobrando instrucciones sintácticas propias de los lenguajes de programación “”,.;’’. (algunos errores “tontos” en el código para pasar los casos de prueba).
- Nivel 5. Programador: llega a este nivel cuando la solución es completamente correcta y eficiente.

Cada CE, dentro de la rúbrica, debe tener un porcentaje. En este caso, asignaremos los valores

a cada nivel (nivel 1: 0 %, nivel 2: 35 %, nivel 3: 65 %, nivel 4: 85 %, nivel 5: 100 %). El promedio de todos los CE nos da el porcentaje del LA. Si el LA es mayor de 80 %, el estudiante alcanza la competencia.

Cada competencia equivale a un 100 %, este se divide en LA, donde cada L da un porcentaje dependiendo del aporte a la competencia; la suma de todos debe ser el 100 %. Asimismo, cada LA tiene un peso en el curso. En la Tabla 2 se muestra el LA y el porcentaje de evaluación.

Tabla 2. Pesos de evaluación por logros de aprendizaje.

Logro de aprendizaje	**Porcentaje de evaluación**
Implementación de clases y objetos	25 %
Identificación de relaciones entre clases y objetos	25 %
Implementación de estructuras de datos	50 %

Cantidad de competencias, logros de aprendizaje y criterios a evaluar por actividad

Por cada actividad a evaluar se deberá elegir un número de criterios a evaluar por logros de aprendizaje. Según Carvajal-Ortiz, Florian-Gaviria y Díaz (s.f.), elegir estos criterios debe ser una tarea en la cual el profesor piense la forma más simple de que el estudiante entienda el criterio, y que no se agote leyendo y evaluando, por ello es bueno contar entre 4 y 8 criterios de evaluación. Asimismo, los logros de aprendizaje deberán ser elegidos teniendo en cuenta la actividad propuesta, donde sea afín el conocimiento técnico con el conocimiento teórico, en Powell y Wimmer (2015), por cada competencia al menos debe haber entre 1 y 3 logros de aprendizaje. Por otro lado, para elegir el número de competencias, no se define un límite, sin embargo, en Wei *et al*. (2021) se menciona que las competencias por lo general están definidas en orden de conocimientos y tiempo de curso, así que dependiendo de eso, se puede elegir el número de competencias.

En este artículo se diseñaron 3 actividades, las cuales se relacionan con 3 competencias, 4 logros de aprendizaje y 20 criterios de evaluación que se especifican a continuación:

Actividad 1: *identificar la reproducción de una bacteria*.

Actividad 2: *en un mundo virtual existen cuatro personajes, cada uno con diferente funcionalidad de ataque y defensa. Cada personaje deberá tener una cantidad de peleas para poder ganar y conquistar el mundo.*

Actividad 3: *identificar el crecimiento y el decrecimiento entre predador y presa o presa y predador usando las ecuaciones de Lotka-Volterra.*

Tabla 3. Estructura de la rúbrica para la actividad 1.

C: Crea nuevos tipos de datos (clases) cuando los que ofrece el lenguaje no son suficientes para modelar el problema.					
LA: Implementación de clases y objetos (%)					
CE: Criterio de evaluación	Programador	Teikirisi	Puro Stack Overflow	Memento Quod Genus	Novato
Implementa una clase					
Usa encapsulación					
Implementa atributos e interfaces de una clase					
Implementa relaciones entre clases y objetos					

Tabla 4. Estructura de la rúbrica para la actividad 2.

C: Usa los tipos de datos básicos y los agregados que proporciona el lenguaje para modelar correctamente los atributos de los objetos.					
LA: Implementación de propiedades condicionales, repetitivas y recursivas (%)					
CE: Criterio de evaluación	Programador	Teikirisi	Puro Stack Overflow	Memento Quod Genus	Novato
Implementa programas condicionales					
Compara entre aplicaciones repetitivas y recursivas					
Aplica referencia para acceder a los objetos					
Compara el costo y el beneficio de estructuras repetitivas versus recursivas					
Conoce la estructura adecuada para resolver un programa					

Tabla 5. Estructura de la rúbrica para la actividad 3.

C: Mapea un problema real en un conjunto de objetos con sus relaciones, usando los tres tipos de polimorfismo y herencia para evitar estructuras condicionales y para desacoplar objetos.					
LA: Implementación de herencia y polimorfismo (%)					
CE: Criterio de evaluación	Programador	Teikirisi	Puro Stack Overflow	Memento Quod Genus	Novato
Utiliza subclases para diseñar jerarquías de clases					
Usa agrupación para diseñar clases complejas					
Razona sobre el flujo de control de un programa usando despacho dinámico					
Conoce la diferencia entre herencia y subtipo					
Define y usa tipos genéricos en la resolución de un problema					

Evaluación

Se seleccionó el evaluador automático de código fuente INGInious M-iDEA (Hidalgo y Bucheli, 2019) porque esta plataforma tiene una interfaz apropiada para administrar cursos y tareas de programación de computadoras. Tiene un juez virtual que permite a los estudiantes presentar soluciones de código fuente, evaluar y recibir comentarios automáticamente en tiempo real. La evaluación automática del código se basa en un método de comparación entre casos de prueba (entradas) y casos de prueba esperados (salidas). Además, la plataforma permite guardar registros sobre las calificaciones logradas, el número de intentos de respuesta, los tiempos promedio, el puntaje de calificación y el indicador de estado de cada actividad presentada por los estudiantes.

Luego de que el estudiante envía su código fuente al evaluador automático, el profesor descarga todas las entregas de los estudiantes. Posteriormente, usando la rúbrica, el profesor evalúa de forma manual cada entrega de cada estudiante.

Colaboración

Con base en Llanos, Hidalgo y Bucheli (2022) se implementó una estrategia de colaboración para un curso de programación CS1, donde el profesor propone una actividad de programación con un límite de tiempo. Los estudiantes que transcurrido un tiempo no hayan avanzado o terminado, tienen la posibilidad de solicitar ayuda de un compañero ya que haya logrado resolver la actividad de programación. El estudiante que ayuda, solo lo puede hacer a través del lenguaje natural, no puede mostrar o pasar el código de solución.

Experimentos

Con la definición previa de las actividades (ver sección Cantidad de competencias, logros de aprendizaje y criterios a evaluar por actividad), se plantean tres experimentos. Para cada actividad, ambos grupos tienen un tiempo límite de 2 horas de forma individual en el caso del GC, y de forma colaborativa en el caso del GE. Cada estudiante debe enviar su solución a través del evaluador automático INGInious M-iDEA. Posteriormente el profesor evalúa por cada estudiante los CE alcanzados. En la Tabla 6 se detallan los experimentos realizados.

Resultados

En este artículo se diseñó y probó una rúbrica que permitió evaluar las competencias de aprendizaje en actividades de programación. La rúbrica permite evaluar en dos diferentes contextos, tanto para la forma colaborativa como para la forma tradicional, donde los estudiantes trabajan de forma individual. En la figura 1 y la figura 2, forma colaborativa e individual respectivamente, se muestran los estudiantes que participaron en cada grupo y por cada uno se muestran los criterios de evaluación que alcanzaron en cada competencia, además, en las figuras se observan cinco colores con una letra, el verde es el nivel 5 (programador) y el rojo el nivel 1 (novato), los otros colores representan los niveles 2, 3 y 4. Por último en las figuras aparece la columna de porcentaje (%), la cual indica la medida lograda por cada estudiante, si es mayor al 80 %, quiere decir que el estudiante logró adquirir la mayoría de los CE esperados y con esto se evidencia que obtuvo la competencia de aprendizaje.

Tabla 6. Experimentos para el grupo de control (GC) y el grupo experimental (GE).

Experimento	ID actividad	Tiempo	GC	CE
1	Actividad 1	2 horas	Herramienta de evaluación + rúbrica	Herramienta de evaluación + rúbrica + actividad colaborativa
2	Actividad 2			
3	Actividad 3			

	Competencia 1						Competencia 2							Competencia 3								
	LA 1						LA 2							LA 3								
Estudiantes	CE 1	CE 2	CE 3	CE 4	CE 5	%	CE 1	CE 2	CE 3	CE 4	CE 5	CE 6	%	CE 1	CE 2	CE 3	CE 4	CE 5	CE 6	CE 7	CE 8	%
Estudiante 1	p	p	p	p	p	100	p	p	p	p	p	p	100	p	p	p	p	p	T	p	p	72,5
Estudiante 2	T	T	p	p	p	92	T	p	p	p	p	S	106	T	T	p	p	p	T	p	p	72,5
Estudiante 3	S	T	p	p	p	86	S	p	S	p	p	m	86	S	T	p	p	p	T	p	p	72,5
Estudiante 4	S	T	p	N	p	68	S	p	S	S	p	m	76	S	T	p	m	p	T	p	p	63,75
Estudiante 5	p	p	p	p	p	100	p	p	p	S	p	m	96	p	p	S	m	p	p	p	p	60
Estudiante 6	p	p	p	m	p	86	p	p	p	m	p	m	92	p	p	S	m	T	p	m	T	46,25
Estudiante 7	T	T	p	m	p	78	p	p	S	m	p	m	82	T	T	S	p	T	p	m	p	57,5
Estudiante 8	T	T	p	m	N	60	p	p	S	m	S	m	72	T	T	S	p	T	N	N	S	37,5
Estudiante 9	T	T	p	p	p	92	p	p	p	p	S	p	110	T	T	p	p	m	m	p	p	57,5
Estudiante 10	T	N	p	N	p	60	p	N	p	N	p	m	70	T	N	p	N	m	m	p	p	46,25
Estudiante 11	T	T	p	p	p	92	p	T	p	p	p	m	102	T	T	p	p	m	m	p	p	57,5
Estudiante 12	p	p	p	p	p	100	p	p	p	S	p	m	96	p	p	m	p	T	p	p	p	63,75
Estudiante 13	S	N	p	N	p	54	S	N	p	S	p	p	82	S	N	m	N	m	m	S	S	25
Estudiante 14	S	T	p	p	p	86	S	T	p	p	p	S	96	S	m	m	p	m	m	m	S	33,75
Estudiante 15	S	N	p	m	N	40	S	N	p	p	S	S	72	S	m	p	p	m	m	N	m	37,5
Estudiante 16	N	S	p	p	N	54	N	S	p	p	S	N	64	N	m	p	p	N	N	N	N	30
MEDIA						86							89									57,5

Figura 1. Logros de aprendizaje del grupo experimental.

	Competencia 1					
	LA 1					
Estudiantes	**CE 1**	**CE 2**	**CE 3**	**CE 4**	**CE 5**	**%**
Estudiantes	p	p	S	S	p	80
Estuidante 1	m	m	m	N	N	22
Estuidante 2	m	m	m	S	S	38
Estuidante 3	S	S	N	S	S	42
Estuidante 4	S	S	N	S	S	42
Estuidante 5	S	m	m	m	m	34
Estuidante 6	p	p	p	p	p	100
Estuidante 7	p	p	T	T	p	92
Estuidante 8	p	p	N	p	p	82
Estuidante 9	p	p	p	p	S	90
Estuidante 10	p	S	T	T	p	82
Estuidante 11	p	p	p	p	S	90
Estuidante 12	p	S	S	S	p	70
Estuidante 13	m	m	N	T	m	36
Estuidante 14	T	S	S	S	T	62
Estuidante 15	N	T	N	p	p	60
Estuidante 16	N	T	S	S	S	48
Estuidante 17	N	S	p	p	p	72
Estuidante 18	T	p	N	T	S	64
Estuidante 19	S	p	N	T	T	64
Estuidante 20	T	p	N	S	T	64
Estuidante 21	S	p	N	S	N	44
Estuidante 22	S	T	N	S	N	40
Estuidante 23	S	T	p	p	S	76
Estuidante 24	S	N	T	T	p	64
MEDIA						64

	Competencia 2						
	LA 2						
CE 1	**CE 2**	**CE 3**	**CE 4**	**CE 5**	**CE 6**	**%**
S	S	S	S	S	m	46,7
m	m	m	S	m	m	33,3
m	m	S	m	m	m	33,3
N	N	N	m	m	N	16,7
m	m	S	S	T	T	53,3
m	m	S	m	m	m	33,3
p	p	p	p	p	p	100,0
p	p	T	p	p	T	93,3
p	p	S	p	p	p	91,7
T	T	S	p	p	T	81,7
T	T	N	T	T	T	68,3
p	p	p	p	p	p	100,0
p	p	S	N	S	S	60,0
T	m	m	S	S	S	48,3
T	T	S	S	S	S	60,0
N	N	N	N	N	N	10,0
S	S	T	T	T	T	70,0
N	T	T	N	T	T	56,7
T	T	T	S	S	S	65,0
S	S	N	S	S	S	43,3
m	m	T	T	T	N	51,7
N	N	N	N	N	N	10,0
m	T	T	T	m	m	55,0
N	S	N	p	N	p	46,7
T	N	T	p	T	T	71,7
						58,3

	Competencia 3								
	LA 3								
CE 1	**CE 2**	**CE 3**	**CE 4**	**CE 5**	**CE 6**	**CE 7**	**CE 8**	**%**
p	p	S	S	S	S	S	m	35
m	m	T	T	m	S	m	m	37,5
m	m	m	m	S	m	m	m	25
N	N	N	N	N	m	m	N	12,5
m	m	m	m	S	S	T	T	40
m	m	m	m	S	m	m	m	25
p	p	p	p	p	p	p	p	75
p	p	p	p	p	p	p	p	75
p	p	p	p	p	p	p	p	75
N	N	N	S	N	p	N	p	35
T	T	T	T	T	T	T	T	60
p	p	p	p	p	N	T	m	52,5
T	T	T	T	S	S	S	T	48,75
m	m	m	m	N	T	m	m	26,25
T	S	T	m	m	N	T	m	32,5
N	S	S	T	S	S	S	T	45
S	S	T	T	T	T	T	m	53,75
S	S	T	T	p	T	T	p	65
p	p	T	T	m	S	T	m	43,75
N	S	S	T	T	S	p	S	51,25
T	T	T	S	S	T	T	T	52,5
T	T	T	p	T	p	T	p	67,5
N	N	S	S	T	S	S	T	45
T	T	N	T	N	T	T	T	42,5
T	T	S	S	S	S	N	T	36,25
								46,9

Figura 2. Logros de aprendizaje del grupo de control.

Los resultados de usar la colaboración (GE) en comparación con la forma individual (GC) muestran que el trabajo colaborativo apoya en adquisición de conocimientos y competencias de aprendizaje, se observa que más estudiantes del GE adquieren el nivel más alto de "programador (P)" y "teikirisi (T)" (ver figura 1), mientras que en el GC se aprecia que los estudiantes en su mayoría adquieren el nivel medio "memento quod genus (M)" y "puro stack overflow" (S).

En investigaciones previas (Llanos, Hidalgo y Bucheli, 2022; Hidalgo *et al.*, 2021) se ha evidenciado que la colaboración mejora tres aspectos importantes: 1) las calificaciones del estudiante, 2) el conocimiento homogéneo en el grupo y 3) las habilidades blandas del estudiante. En este artículo se encuentra que la colaboración también logra que los estudiantes alcancen las competencias académicas, que los estudiantes mejoran significativamente sus conocimientos y que todo el curso mejora, esto se observa en la media de cada competencia (Tabla 7)

Tabla 7. Comparación de media entre el GE y el GC.

Competencia académica	Media GC	Media GE
1	64	86
2	58,3	89
3	46,9	57,5

Finalmente, el uso de la rúbrica en los cursos de programación puede ser un recurso para que el estudiante tome decisiones y tomar medidas a tiempo para alcanzar las competencias. Por otra parte, el profesor puede identificar a tiempo los estudiantes que no estén adquiriendo una competencia de aprendizaje, y así puede crear estrategias y mejorar los métodos de enseñanza propuestos en el curso, con el fin de mejorar la dinámica y prevenir que los estudiantes no tengan las capacidades en programación. Desde el curso, se puede evaluar si los logros de aprendizaje y criterios de evaluación están ajustados al contenido del mismo, además de identificar el aporte de las competencias de un curso al futuro profesional.

Discusión

Este trabajo presenta una rúbrica para estimular las habilidades en la programación y alcanzar competencias académicas de estudiantes del curso CS1. Al usar esta rúbrica, fue posible probar y evaluar la estrategia propuesta, permitiendo realizar seguimiento sobre el proceso de aprendizaje, identificando si se logran los conceptos propuestos de programación en el código fuente, que permitirán a futuro obtener información para desarrollar estadísticas descriptivas sobre la calificación del grupo y el tiempo de entregas por cada taller de programación.

Por otro lado, el uso de la estrategia colaborativa apoyada con la herramienta de evaluación automática, integra la evaluación formativa y la sumativa. Por un lado, la evaluación automática permite identificar el nivel alcanzado en el logro de aprendizaje de una actividad y, por otro lado, los resultados de la rúbrica permiten identificar claramente los resultados esperados, además, sirve de apoyo al profesor para tomar decisiones oportunas que beneficien el aprendizaje de los estudiantes.

El trabajo colaborativo entre estudiantes surge como apoyo para conseguir un resultado que resulta más difícil de conseguir individualmente. Además, el trabajo colaborativo permite que haya mayor homogeneidad de los estudiantes en los logros de aprendizaje del curso. La estrategia planteada centra el aprendizaje del estudiante en la experiencia de colaboración y reflexión individual, mejorando habilidades sociales de comunicación, liderazgo, confianza en los demás, capacidad de intercambio de roles, entre otras. Esto transforma el espacio académico tradicional en un espacio lúdico y divertido.

En el aprendizaje de la programación, especialmente en el curso CS1, es necesario que el estudiante sea constante en la práctica para lograr alcanzar las competencias necesarias del curso. Para esto la academia cuenta con profesores idóneos que enseñan las bases necesarias; sin embargo, el tiempo limitado de interacción entre el profesor y el estudiante no permite que el estudiante reciba retroalimentación permanente sobre sus procesos de aprendizaje. En este sentido, contar con herramientas que permitan evaluar un código fuente y dar realimentación inmediata es un insumo muy importante para la enseñanza de la programación. Esto logra acelerar el proceso de aprendizaje, facilitar la adquisición de competencias académicas e incentivar actitudes de aprendizaje autónomo en los estudiantes.

Conclusiones

El diseño de rúbricas que integren la colaboración y permitan evaluar las competencias de aprendizaje mejora significativamente el entendimiento y las calificaciones del estudiante, además de las habilidades interpersonales que incentivan a mejorar los cursos de programación.

En el desarrollo y diseño de la rúbrica es importante que haya iteraciones que permitan observar la relación entre los contenidos del curso y el estudiante, esto se puede evidenciar a través de herramientas de evaluación de código fuente, donde se guardan los registros de las entregas, que al analizarse permiten tomar decisiones oportunas que beneficien al estudiante.

Cuando el estudiante se encuentra con actividades de programación retadoras es necesario que cuente con apoyos, y es preciso que un compañero de curso sea quien brinde la ayuda, permitiendo así integrar a los estudiantes para encontrar una solución colaborativa, además de fomentar el aprendizaje y el apoyo mutuo.

Contribucín de Autoria

Carlos-Giovanny Hidalgo-Suárez: Investigación, Conceptualización, Análisis formal, Metodología, Software, Validación, Escritua – Borrador original.
Víctor-Andrés Bucheli-Guerrero: Conceptualización, Metodología, Supervisión, Validación, Escritura – revisión y edición.
Hugo-Armando Ordoñez-Erazo: Conceptualización, Metodología, Supervisión, Escritura – revisión y edición.

Referencias

Alhazbi, S., Hassanh, M. (2013). Fostering self-regulated learning in introductory computer programming course. En *WCCCE 2013: Western Canadian Conference on Computing Education*.

Allen, S., Knight, J. (2009). A method for collaboratively developing and validating a rubric. *International Journal for the Scholarship of Teaching and Learning*, *3*(2), e210. https://doi.org/10.20429/ijsotl.2009.030210

Allen, D., Tanner, K. (2006). Rubrics: Tools for making learning goals and evaluation criteria explicit for both teachers and learners. *CBE—Life Sciences Education*, *5*(3), 197-203. https://doi.org/10.1187/cbe.06-06-0168

Ayala, G., Ortiz, M., Osorio, M. (2005). Agent modelling for CSCL environments using answer sets programming. En *Sixth Mexican International Conference on Computer Science (ENC'05)*, 214-221. https://doi.org/10.1109/ENC.2005.9

Bhuyan, M. H., Tamir, A. (2020). Evaluating COs of computer programming course for OBE-based BSc in EEE program. *International Journal of Learning and Teaching*, *12*(2), 86-99. https://doi.org/10.18844/ijlt.v12i2.4576

Blikstein, P., Worsley, M., Piech, C., Sahami, M., Cooper, S., Koller, D. (2014). Programming pluralism: Using learning analytics to detect patterns in the learning of computer programming. *Journal of the Learning Sciences*, *23*(4), 561-599. https://doi.org/10.1080/10508406.2014.954750

Böhne, A., Faltin, N., Wagner, B. (2007). Distributed group work in a remote programming laboratory—A comparative study. *International Journal of Engineering Education*, *23*(1), 162-170

Carbonaro, A. (2019). Good practices to influence engagement and learning outcomes on a traditional introductory programming course. *Interactive Learning Environments*, *27*(7), 919-926. https://doi.org/10.1080/10494820.2018.1504307

Carvajal-Ortiz, L., Florian-Gaviria, B., Díaz, J. F. (s.f.). Modelos, métodos y prototipo de software para el apoyo del diseño, evaluación y análisis de aprendizajes en gestión curricular de la educación superior basada en competencias.

Cateté, V., Snider, E., Barnes, T. (2016). Developing a rubric for a creative CS principles lab. En *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, 290-295. https://doi.org/10.1145/2899415.2899449

Chen, J., Wang, M., Kirschner, P. A., Tsai, C.-C. (2018). The role of collaboration, computer use, learning environments, and supporting strategies in CSCL: A meta-analysis. *Review of Educational Research*, *88*(6), 799-843. https://doi.org/10.3102/0034654318791584

Chen, Y.-H., Lee, W.-C., Tseng, C.-H., Deng, L. Y., Chang, C.-Y., Lee, L.-H. (2020). Cognitive learning performance assessment and analysis with CSCL applied on the NetGuru platform and CSPL applied on the TAoD platform for the network experiment class. *Journal of Supercomputing*, *76*(1), 16-46. https://doi.org/10.1007/s11227-019-02836-3

Clancy, M., Titterton, N., Ryan, C., Slotta, J., Linn, M. (2003). New roles for students, instructors, and computers in a lab-based introductory programming course. *ACM SIGCSE Bulletin*, *35*(1), 132-136. https://doi.org/10.1145/792548.611951

Collins, D., Weber, J., Zambrano, R. (2014). Teaching business ethics online: Perspectives on course design, delivery, student engagement, and assessment. *Journal of Business Ethics*, *125*, 513-529. https://doi.org/10.1007/s10551-013-1932-7

Coto, M., Mora, S., Collazos, C. (2014). Evaluation of the collaboration process from an individual and collaborative perspective. En *Proceedings of the XV International Conference on Human Computer Interaction*, 1-9. https://doi.org/10.1145/2662253.2662342

Demaidi, M. N., Qamhieh, M., Afeefi, A. (2019). Applying blended learning in programming courses. *IEEE Access*, *7*, 156824-156833. https://doi.org/10.1109/ACCESS.2019.2949927

Dümmel, N., Westfechtel, B., Ehmann, M. (2018). Effects of a preliminary programming course on students' performance. *ECSEE'18:*

Proceedings of the 3rd European Conference of Software Engineering Education, 77-86. https://doi.org/10.1145/3209087.3209088

Fiesler, C., Garrett, N., Beard, N. (2020). What do we teach when we teach tech ethics? A syllabi analysis. En *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 289-295. https://doi.org/10.1145/3328778.3366825

Fleming, D. L. (2008). Using best practices in online discussion and assessment to enhance collaborative learning. *College Teaching Methods & Styles Journal*, *4*(10), e5573. https://doi.org/10.19030/ctms.v4i10.5573

García Retana, J. Á. (2011). Modelo educativo basado en competencias: importancia y necesidad. *Actualidades Investigativas en Educación*, *11*(3), e10225. https://doi.org/10.15517/aie.v11i3.10225

Goss, H. (2022). Student learning outcomes assessment in higher education and in academic libraries: A review of the literature. *The Journal of Academic Librarianship*, *48*(2). https://doi.org/10.1016/j.acalib.2021.102485

Hidalgo, C., Bucheli, V. (2019). Modelo soportado en inteligencia artificial para el desarrollo de actividades de aprendizaje activo basadas en colaboración asistida por computador (M-IDEA). En *17th LACCEI International Multi-Conference for Engineering, Education, and Technology*.

Hidalgo, C. G., Bucheli, V. A., Restrepo, F., González, F. A. (2021). Estrategia de enseñanza basada en la colaboración y la evaluación automática de código fuente en un curso de programación CS1. *Investigación e Innovación en Ingenierías*, *9*(1), e4185. https://doi.org/10.17081/invinno.9.1.4185

Kardan, A., Sadeghi, H. (2015). Modeling the learner group formation problem in computer-supported collaborative learning using mathematical programming. En *8th National and 5th International Conference on E-Learning and e-Teaching*. https://doi.org/10.1109/ICELET.2014.7040616

Kilgour, P., Northcote, M., Williams, A., Kilgour, A. (2020). A plan for the co-construction and collaborative use of rubrics for student learning. *Assessment & Evaluation in Higher Education*, *45*(1), 140-153. https://doi.org/10.1080/02602938.2019.1614523

Lakas, A., Belkacem, A. N. (2021). A framework for course-embedded assessment for evaluating learning outcomes of a network programming course. En *IEEE Global Engineering Education Conference*, 989-995. https://doi.org/10.1109/EDUCON46332.2021.9454129

Lämsä, J., Uribe, P., Jiménez, A., Caballero, D., Hämäläinen, R., Araya, R. (2021). Deep networks for collaboration analytics: Promoting automatic analysis of face-to-face interaction in the context of inquiry-based learning. *Journal of Learning Analytics*, *8*(1), e7118. https://doi.org/10.18608/jla.2021.7118

Lee, G., Fong, W. W., Gordon, J. (2013). Blended learning: The view is different from student, teacher, or institution perspective. En *Lecture Notes in Computer Science* (LNCS, 8038, pp. 356-363). https://doi.org/10.1007/978-3-642-39750-9_33

Llanos, J. M., Hidalgo, C. G., Bucheli, V. A. (2022). Strategy based on computer-supported collaborative learning to form workgroups automatically in an introductory programming course (CS1). *Revista Facultad de Ingeniería*, *31*(61), e14368. https://doi.org/10.19053/01211129.v31.n61.2022.14368

Means, B., Toyama, Y., Murphy, R., Bakia, M., Jones, K. (2009). *Evaluation of Evidence-Based Practices in Online Learning: A Meta-Analysis and Review of Online Learning Studies* [Grade Thesis]. Centre for Learning Technology, Reino Unido. https://repository.alt.ac.uk/629/

Mehennaoui, Z., Lafifi, Y., Seridi, H., Boudria, A. (2014). A new approach for grouping learners in CSCL systems. En *International Conference on Multimedia Computing and Systems*, 628-632. https://doi.org/10.1109/ICMCS.2014.6911143

Michel, N., Cater, J. J., Varela, O. (2009). Active versus passive teaching styles: An empirical study of student learning outcomes. *Human Resource Development Quarterly*, 20(4), 397-418. https://doi.org/10.1002/hrdq.20025

Mustapha, A., Samsudin, N. A., Arbaiy, N., Mohammed, R., Hamid, I. R. (2016). Generic assessment

rubrics for computer programming courses. *Turkish Online Journal of Educational Technology - TOJET*, *15*(1), 53-68

OECD iLibrary. (s.f.). Assessment of learning outcomes in higher education: A comparative review of selected practices. *OECD Education Working Papers*. https://www.oecd-ilibrary.org/content/paper/244257272573

Palmer, M. S., Bach, D. J., Streifer, A. C. (2014). Measuring the promise: A learning-focused syllabus rubric. *To Improve the Academy*, *33*(1), 14-36. https://doi.org/10.1002/tia2.20004

Pereira, F. D., Oliveira, E., Cristea, A., Fernandes, D., Silva, L., Aguiar, G., Alamri, A., Alshehri, M. (2019). Early dropout prediction for programming courses supported by online judges. In S. Isotani, E. Millán, A. Ogan, P. Hastings, B. McLaren, & R. Luckin (Eds.), *Artificial Intelligence in Education* (vol. 11626, pp. 67-72). Springer. https://doi.org/10.1007/978-3-030-23207-8_13

Powell, L. M., Wimmer, H. (2015). Evaluating the effectiveness of self-created student screencasts as a tool to increase student learning outcomes in a hands-on computer programming course. *Information Systems Education Journal*, *13*(5), 106-111

Sadler, D. R. (2016). Three in-course assessment reforms to improve higher education learning outcomes. *Assessment & Evaluation in Higher Education*, *41*(7), 1081-1099. https://doi.org/10.1080/02602938.2015.1064858

Saito, D., Yajima, R., Washizaki, H., Fukazawa, Y. (2021). Validation of rubric evaluation for programming education. *Education Sciences*, *11*(10), e656. https://doi.org/10.3390/educsci11100656

Saunders, L. (2012). Faculty perspectives on information literacy as a student learning outcome. *The Journal of Academic Librarianship*, *38*(4), 226-236. https://doi.org/10.1016/j.acalib.2012.06.001

Stahl, G., Koschmann, T., Suthers, D. (2006). *Computer-Supported Collaborative Learning: An Historical Perspective*

Tiantong, M., Teemuangsai, S. (2013). Student team achievement divisions (STAD) technique through the moodle to enhance learning achievement. *International Education Studies*, *6*(4), 85-92. https://doi.org/10.5539/ies.v6n4p85

TrainCom. (s.f.). Indicators of achievement. http://train-com.de/traincom/english/competency/achievement.rsys

Wei, X., Saab, N., Admiraal, W. (2021). Assessment of cognitive, behavioral, and affective learning outcomes in massive open online courses: A systematic literature review. *Computers & Education*, *163*, e104097. https://doi.org/10.1016/j.compedu.2020.104097

Zlatkin-Troitschanskaia, O., Pant, H. A., Coates, H. (2016). Assessing student learning outcomes in higher education: Challenges and international perspectives. *Assessment & Evaluation in Higher Education*, *41*(5), 655-661. https://doi.org/10.1080/02602938.2016.1169501