Asociación Colombiana de Facultades de Ingeniería

Revista Educación en Ingeniería
http://www.educacioneningenieria.org

LE1: una estrategia amistosa para un curso introductorio de programación

Jesús Pérez & Maybelin Azuaje

Laboratorio de Sistemas Discretos, Automatización e Integración. Universidad de Los Andes, Mérida, Venezuela. jesuspangulo@ula.ve, maybelin@ula.ve

Resumen— Los cursos introductorios de programación han presentado dificultades en distintas universidades de diferentes países. La asignatura "Programación 1" de la Universidad de Los Andes en Venezuela presenta dificultades relacionadas a la alta tasa de reprobados. El primer contacto con la asignatura es muy importante, por lo tanto, el objetivo de esta investigación es proporcionar una estrategia amistosa que facilite ese primer contacto. La estrategia propuesta incluye la creación de la biblioteca PR1-ULA y la planificación de diez sesiones de clases que deben ser proporcionadas en las tres primeras semanas. La estrategia fue probada en los semestres A-2017 y A-2018, obteniendo una incidencia positiva en el rendimiento de los estudiantes y permitiendo descubrir de manera temprana el potencial de los estudiantes.

Palabras Clave— introducción a la programación; curso de programación, estrategia amistosa.

Recibido: 1 de marzo de 2019. Revisado: 11 de abril de 2019. Aceptado: 3 de mayo de 2019.

LE1: A friendly strategy for an introductory programming course

Abstract— Introductory programming courses present difficulties in distinct universities of different countries. The "Programming 1" course of the University of Los Andes in Venezuela presents difficulties related to the high rate of disapproved. The first contact with the course is very important, therefore, the purpose of this research is to provide a friendly strategy facilitating that first contact. The strategy that is carried out includes the creation of the PR1-ULA library and the planning of ten class sessions which must be provided in the first three weeks. The strategy was tested in the A-2017 and A-2018 semesters, obtaining a positive impact on students' performance and allowing to discover in an early way the students' potential.

Keywords— programming introduction; programming course; friendly strategy.

1. Introducción

Los cursos introductorios de programación han presentado dificultades en el aprendizaje de estudiantes en distintas universidades de diferentes países: en Alemania, Universidad de Hamburgo [1]; en Arabia Saudita, Universidad Prince Sattam Bin Abdulaziz [2]; en Argentina, Universidad Tecnológica Nacional [3], Universidad Nacional del Nordeste [4], Universidad Nacional de Quilmes [5] y Universidad de Buenos Aires [6]; en Brasil, Universidad Federal de Paraíba [7]; en Colombia, Universidad de Medellín [8,9], Universidad de Nariño [8], Corporación Universitaria de la Costa [8] y Universidad Nacional de Colombia [10]; en Costa Rica, Universidad de Costa Rica [11]; en Chile, Universidad de Valparaiso [12]; en Malasia, Universidad de Tecnología MARA [13], Universidad Nacional de Malasia [14] y Universidad Tecnológica de Malasia [15]; en México, Instituto Tecnológico Superior Progreso [16]; en Perú, Universidad de Ciencias y Humanidades [17]; en Portugal, Universidad de Coimbra [18]; en Taiwan, Universidad Nacional de Tainan [19], [20]; en Turquía, Universidad Işık [21], Universidad Atilim [22] y Universidad Mehmet Akif Ersoy [23]; en Uruguay, Universidad ORT [24]; y en Venezuela, Universidad Central de Venezuela [25] y Universidad de Los Andes [26-28].

Los problemas más comunes de los cursos de programación en esas universidades son: desmotivación de los estudiantes [3,9,13,14,19,22,23,27,29], dificultad de aprendizaje de la programación [4,11,13-16,20,21,28], alta tasa de reprobados [1,2,12,17,25] y debilidades en el pensamiento computacional [7,26,30,31]. En Venezuela, en la Universidad de Los Andes, en la carrera de Ingeniería de Sistemas, también se presentan dificultades en el curso introductorio "Programación 1", cuyo objetivo principal es que los estudiantes desarrollen habilidades en el análisis, diseño y construcción de programas codificados en el lenguaje de programación C para resolver problemas con programación estructurada. En los últimos semestres la preocupación está relacionada con la alta tasa de reprobados, lo cual se evidencia en las secciones con registro completo de las estadísticas. Por una parte, en el semestre B-2015, en las secciones 2 y 4 reprobaron el 68.6% y 83.3% de los estudiantes matriculados, respectivamente; y por otra parte, en el semestre A-2016, en la sección 1 reprobó el 75% y en la sección 3 reprobó el 86.2% de los matriculados.

El primer contacto con la programación es muy importante para el desarrollo de toda la asignatura [32], por lo tanto, el objetivo de esta investigación es proporcionar una estrategia amistosa que facilite ese primer contacto. La metodología que se lleva a cabo en este estudio, incluye una revisión de investigaciones recientes, para conocer las tendencias actuales sobre el aprendizaje de la programación en el ámbito universitario. A partir de allí, se propone la estrategia LE1, basada en una biblioteca que permite programar animaciones

Como citar este artículo: Pérez, J. and Azuaje, M., LE1: una estrategia amistosa para un curso introductorio de programación. Educación en Ingeniería, 14(28), pp. 45-53, Marzo - Julio de 2019.

con audios y colores, utilizando sólo nueve instrucciones. La estrategia LE1 incluye la planificación de 10 sesiones cortas de clases, cuyo contenido también se facilita en 10 videos cortos y complementarios.

La organización del artículo está divida en cinco secciones: estado del arte, donde se presentan las investigaciones recientes relacionadas con cursos introductorios de programación; estrategia LE1, que describe la biblioteca propuesta y las sesiones de clases dirigidas a los estudiantes; experimento, donde se muestran los pasos que se aplican para probar el efecto de la biblioteca sobre el rendimiento académico de los estudiantes; resultados, para presentar los valores estadísticos y correlacionales; discusión, donde se analizan y comparan los resultados, y finalmente, las conclusiones.

2. Estado del arte

Los problemas más comunes encontrados en las investigaciones relacionadas a cursos introductorios de programación en el ámbito universitario se pueden clasificar en cuatro dimensiones: desmotivación de los estudiantes, dificultad en el aprendizaje de la programación, alta tasa de reprobados en las asignaturas y debilidades en el pensamiento computacional (considerado como una habilidad fundamental del siglo *XXI* [18]).

Entre las propuestas para atacar la *desmotivación* se encuentran: involucrar a los estudiantes en el desarrollo de nuevas herramientas para complementar la gestión del curso [3]; proporcionar un sistema basado en juegos para el salón de clases mediante una herramienta (Kahoot) que permite crear y compartir cuestionarios [13]; suministrar estrategias motivacionales para facilitar la interacción entre el profesor y los estudiantes, a partir de los conceptos de constructivismo y gamificación [9]; utilizar la gamificación mediante una aplicación web para enseñar programación y visualizar videos tanto en el salón clases como a distancia [19]; incorporar un robot social en el salón de clases para desempeñar algunas actividades docentes [27]; y desarrollar juegos basados en problemas de la vida real con el entorno de programación Scratch [22].

La *dificultad de aprendizaje* que presentan los estudiantes ha promovido las siguientes iniciativas: desarrollo de actividades lúdicas y utilización de juegos serios [4]; aplicación de reglas de juego para el proceso de aprendizaje, tales como puntuaciones, barras de progresos, niveles, entre otros [14]; metodología de estudio orientada a la planificación estratégica y la práctica autónoma [28]; y desarrollo de un esquema de actividad de aprendizaje basada en web, donde los estudiantes pueden aprender los conceptos de programación de manera interactiva, practicar las habilidades de codificación haciendo uso de Scratch y abordar las actividades mediante mensajes en línea [20].

La *alta tasa de reprobados* se ha intentado subsanar con estrategias que combinan distintos métodos de aprendizaje, tales como: aplicación de métodos de investigación y acción participativa para la resolución grupal de exámenes de semestres pasados [17]; rediseño del curso para combinar conferencias con un entorno de programación en línea y videos instructivos cortos [1]; modificación curricular para definir estrategias de enseñanza y aprendizaje, resolución de problemas y apoyo para los estudiantes [25]; e implementación de un taller de diseño de juegos utilizando el entorno de programación de Scratch [12].

Las *debilidades en el pensamiento computacional* han sido estimuladas de la siguiente manera: desarrollando un sistema asistente de lenguajes de programación visual, que puede capturar la pantalla de trabajo del alumno y proporcionar asistencia en el momento apropiado para conocer y estudiar los conceptos de pensamiento computacional que son aprendidos por los estudiantes [30]; estudiando dos enfoques para enseñar pensamiento computacional (guía y exploración), y desarrollando un juego basado en Scratch llamado "Dragon Architect", el cual es un hibrido entre los enfoques estudiados [31].

En general, las soluciones a los problemas encontrados en cursos introductorios de programación están orientadas hacia la creación e incorporación de nuevas estrategias, con el propósito de proporcionar mejores posibilidades o alternativas a los estudiantes, cuyos resultados frecuentemente son positivos. Por ejemplo, para el problema de razonamiento lógico presentado en la asignatura "Programación 1", con la estrategia de un juego serio, se obtuvieron mejores resultados [26]. En el mismo orden de ideas, para el problema de alta tasa de reprobados en la asignatura "Programación 1", en esta investigación se crea e incorpora una estrategia que ha sido denominada LE1.

3. Estrategia LE1

La estrategia LE1 (acrónimo de "**L**aboratorio de Sistemas Discretos Automatización e Integración **E**strategia **1**") tiene como propósito facilitar el primer contacto de los estudiantes con la asignatura "Programación 1" en las tres primeras semanas de clases. LE1 se cataloga como una estrategia amistosa por dos razones: primero, proporciona una biblioteca denominada PR1-ULA que permite programar animaciones con audios y colores, utilizando sólo nueve instrucciones; y segundo, presenta la biblioteca en sólo 10 sesiones cortas de clases, cuyo contenido también se facilita en 10 videos cortos y complementarios.

3.1. Diseño e implementación de PR1-ULA

En la asignatura "Programación 1", cuyo objetivo es resolver problemas utilizando el lenguaje de programación C, se necesita una estrategia amistosa en aras de facilitar el primer acercamiento de los estudiantes con la programación en concordancia con los conceptos que los estudiantes deben aprender durante el desarrollo del curso. Por lo tanto, los requerimientos principales de esa estrategia son: permitir la familiarización con la terminal de Linux y utilizar el lenguaje C. En ese sentido, se realiza el diseño de la biblioteca PR1-ULA (acrónimo de Programación 1 - Universidad de Los Andes) para que permita reflejar en la terminal de Linux tres aspectos: animaciones, colores y audios.

El diseño de la biblioteca se realiza con sólo nueve instrucciones: limpiar, saltar, espaciar, imprimir, imprimirColor, imprimirEntero, esperar, esperarMilisegundos y reproducirAudio. Sus prototipos fueron establecidos para reflejar en la terminal de Linux (con la menor cantidad de instrucciones) las animaciones, los colores y los audios. En la Tabla 1 se presentan los nueve prototipos con su respectiva descripción.

Tabla 1.
Prototipos de la biblioteca PR1-ULA.

Prototipo	Descripción
void limpiar();	Borra todo lo que se muestra en la terminal de Linux.
void saltar(int);	Imprime los saltos de línea indicados.
void espaciar(int);	Imprime los espacios indicados.
void imprimir(char *);	Muestra un mensaje por la terminal.
void imprimirColor(char *,char *,char *);	Muestra un mensaje con los colores indicados de fondo y letra.
void imprimirEntero(int);	Muestra el número indicado por la terminal de Linux.
void esperar(int);	Detiene la ejecución del programa durante la cantidad de segundos indicados.
void esperarMilisegundos(int);	Detiene la ejecución del programa durante la cantidad de milisegundos indicados.
void reproducirAudio(char *);	Reproduce un archivo de audio con extensión mp3.

Fuente: Los autores.

Las animaciones se reflejan utilizando la técnica "stop motion" (consiste en simular movimientos mediante una serie de imágenes fijas que pasan rápidamente de manera sucesiva), mediante la combinación de las instrucciones *limpiar* (para borrar todo lo que se muestra en la terminal), *saltar y espaciar* (para ubicar el cursor en la terminal), e *imprimir* (para mostrar los mensajes en la terminal).

Los ocho colores disponibles (negro, rojo, verde, magenta, azul, morado, cian y blanco) tienen efecto tanto en el fondo como en la letra, y la unidad mínima es un carácter, es decir, un carácter perteneciente a la tabla ASCII sólo puede tener un color de letra y un color de fondo, de manera que, para la construcción de animaciones, cada carácter es análogo a un pixel. Por otra parte, los audios deben tener extensión mp3 y tienen que estar almacenados en la misma carpeta donde se encuentra el archivo ejecutable del programa para que puedan ser reproducidos.

La implementación de la biblioteca se realiza utilizando los archivos cabecera "stdio.h" y "stdlib.h", generando como resultado una biblioteca con dos archivos (pr1-ula.h y pr1 ula.o), de manera que los programas se realizan en un archivo con extensión ".c", donde únicamente debe incluirse al archivo cabecera de la biblioteca "pr1-ula.h". La compilación del programa se realiza utilizando el siguiente comando: "gcc pr1 ula.o programa.c –o programa". La reproducción de los audios se hace con mpg123, por lo tanto, debe estar instalado en el computador donde se ejecuta el programa. El "Hola mundo" de la biblioteca PR1-ULA (programa más básico), está constituido por cinco líneas que se describen en la Tabla 2.

Tabla 2.
Programa "Hola mundo" utilizando la biblioteca PR1-ULA.

Línea	Instrucción	Descripción
1	#include "pr1-ula.h"	Incluye la biblioteca.
2	int main(){	Declara la función principal.
3	imprimir("Hola mundo");	Muestra el mensaje "Hola mundo en la terminal de Linux.
4	return 0;	Indica que se ejecutó correctamente la función.
5	}	Finaliza la función principal.

Fuente: Los autores.

3.2. Diseño e implementación de las sesiones de clases

En la programación estructurada se ven reflejados tres conceptos [33]: secuencias, decisiones y repeticiones. Por lo tanto, la utilización de la biblioteca PR1-ULA en la asignatura "Programación 1" debe involucrar estos conceptos, de manera que la noción de secuencias estará implícita durante la realización de todos los programas, porque cada instrucción es ejecutada según el orden de aparición en el programa; las decisiones aparecerán únicamente mediante la decisión simple; y las repeticiones serán introducidas a través del repita para. Adicionalmente, la utilización de la biblioteca PR1-ULA abarca algunos otros conceptos que deben aprenderse durante el curso de manera introductoria, tales como: variables, procedimientos y parámetros. La relación de los conceptos que debe abarcar la asignatura y los conceptos presentados con la biblioteca PR1-ULA se muestran en la Tabla 3.

El proceso de enseñanza de la biblioteca PR1-ULA a los estudiantes se estructura en diez sesiones, donde cada sesión consiste en desarrollar uno de los conceptos mencionados anteriormente y realizar las prácticas respectivas, para alcanzar las competencias correspondientes que se muestran a continuación:

- **Sesión 1:** Escribe, compila y ejecuta programas que muestran mensajes por la terminal con la biblioteca PR1-ULA, utilizando el editor de texto "gedit", el compilador "gcc" y la terminal de Linux, respectivamente.
- **Sesión 2:** Escribe, compila y ejecuta programas que muestran mensajes que se desplazan por la terminal utilizando la biblioteca PR1-ULA.
- **Sesión 3:** Escribe, compila y ejecuta programas que utilizan variables de tipo entero y muestra su valor por la terminal.
- **Sesión 4:** Escribe, compila y ejecuta programas que utilizan decisión simple para condicionar la ejecución de instrucciones.
- **Sesión 5:** Escribe, compila y ejecuta programas que utilizan la estructura repita para en la ejecución de instrucciones que se repiten.
- **Sesión 6:** Escribe, compila y ejecuta programas que utilizan procedimientos para agrupar las instrucciones que se utilizan repetidamente en distintas partes del programa.
- **Sesión 7:** Escribe, compila y ejecuta programas que utilizan procedimientos que se ejecutan según la cantidad de veces indicada en el parámetro.
- **Sesión 8:** Escribe, compila y ejecuta programas que muestran movimientos, utilizando la estructura de

Tabla 3.
Conceptos que abarca la biblioteca PR1-ULA.

Concepto	Asignatura	PR1-ULA
Decisiones	Decisión simple, decisión compuesta y decisión múltiple.	Decisión simple.
Repeticiones	Repita para, repita mientras y hacer mientras.	Repita para.
Subprogramas	Procedimientos y funciones.	Procedimientos.
Paso de parámetros	Valor y referencia.	Parámetros por valor.
Estructuras de datos	Variables, vectores, matrices, cubos y registros.	Variables.
Tipos de datos	Entero, real, carácter y apuntador.	Entero.

Fuente: Los autores.

Tabla 4.
Estructura de los videos.

Conceptos	Ejemplos	Actividades
1) Primer programa: Descarga de la biblioteca, partes de un programa, compilación y ejecución.	Programa que muestra un mensaje por la terminal de Linux.	Ejecutar el primer programa mostrando el mensaje de su preferencia y compartir la captura de pantalla.
2) Instrucciones: Explica las instrucciones: *imprimir, limpiar y esperar*.	Programa que muestra una frase que se desplaza por la terminal de Linux.	Ejecutar un programa que muestre una animación, combinando las instrucciones: *imprimir, limpiar y esperar*; y compartir un video.
3) Variable: Abarca el concepto de variable y presentación de *imprimirEntero.*	Programa que declara un entero, le asigna un valor, lo imprime, lo actualiza y lo vuelve a imprimir.	Ejecutar un programa que declare un entero, le asigne valores e imprima. Compartir una captura de pantalla.
4) Decisión simple: Abarca el concepto y sintaxis de la decisión simple.	Programa que muestra un mensaje que depende del valor de una variable de tipo entero.	Ejecutar un programa que utilice al menos dos decisiones simples con operadores relacionales diferentes y compartir la captura de pantalla.
5) Repita para: Abarca concepto y sintaxis del repita para y explica las expresiones que lo componen.	Programa que muestra un mensaje que parpadea cinco veces.	Ejecutar un programa que utilice al menos un repita para y compartir una captura de pantalla.
6) Procedimientos: Abarca concepto y sintaxis del prototipo y declaración de procedimientos.	Programa que invoca un procedimiento.	Ejecutar un programa que utilice al menos dos procedimientos; y compartir la captura de pantalla.
7) Parámetros: Abarca conceptos de sintaxis, parámetros actuales y formales.	Programa que ejecuta un procedimiento según la cantidad de veces indicada en su parámetro.	Ejecutar un programa que utilice al menos dos procedimientos que reciban parámetros de tipo entero y compartir la captura de pantalla.
8) Saltos: Explica las instrucciones: *saltar y espaciar.*	Programa que introduce la noción de movimiento.	Ejecutar un programa para que un número se mueva por toda la pantalla, incrementando su valor en uno en cada movimiento y compartir un video de la ejecución.
9) Audio: Explica la instalación de paquetes requeridos y la instrucción *reproducirAudio.*	Programa que reproduce un archivo de audio.	Ejecutar un programa que reproduzca el audio de su preferencia y compartir un video.
10) Color: Explica la instrucción *imprimirColor.*	Programa que imprime dos mensajes con distintos colores de letra y fondo.	A modo de reto, ejecutar un programa que muestre una historia animada, utilizando todas las instrucciones de la biblioteca PR1-ULA; y compartir un video de la ejecución.

Fuente: Los autores.

repetición y las instrucciones: saltar y espaciar.

- **Sesión 9:** Escribe, compila y ejecuta programas que reproducen audios.
- **Sesión 10:** Escribe, compila y ejecuta programas que muestran colores en la terminal.

Adicionalmente, para cada sesión de clases se realiza un video complementario con el propósito de explicar cada concepto de la manera más sencilla posible, mostrar al menos un ejemplo práctico y asignar al menos una actividad evaluada (cuya filosofía es proporcionar libertad de elección del tema de cada actividad, es decir, los estudiantes entregan soluciones distintas). Los videos utilizan diapositivas para explicar los conceptos, y para mostrar los ejemplos usan los mismos recursos que deben utilizar los estudiantes: sistema operativo Linux, editor de texto "gedit" y compilador "gcc". En la Tabla 4 se muestra la estructura de cada video.

4. Experimento

Esta investigación tiene enfoque cuantitativo con un diseño cuasi experimental porque los grupos de estudio (estudiantes de nuevo ingreso) estaban previamente formados. La variable independiente es la incorporación de la estrategia LE1, y la variable dependiente es el rendimiento en la primera evaluación de la asignatura, el cual se mide con las calificaciones obtenidas. El estudio se realiza con los estudiantes de los semestres A-2017 y A-2018, y el experimento consiste en proporcionar las diez sesiones en las primeras tres semanas (tres clases por semana y una sesión por cada clase), finalizando con la primera evaluación de la asignatura.

En cada sesión, participan: el profesor, un preparador (estudiante que recibe remuneración monetaria mensualmente para colaborar con las actividades docentes) y los estudiantes. Las sesiones se realizan en un laboratorio con computadores, un pizarrón acrílico y un proyector, según la siguiente planificación:

- El profesor explica el concepto asociado a la clase en el pizarrón.
- El preparador codifica el ejemplo práctico en un computador conectado a un proyector mientras el profesor explica cada línea codificada.
- Los estudiantes replican en el computador el ejemplo práctico explicado.
- En caso de inconvenientes, los estudiantes deben preguntar al preparador y al profesor.
- El profesor asigna al menos un ejercicio.
- El profesor y el preparador supervisan la realización de los ejercicios.

Al finalizar cada sesión de clases, se proporciona mediante la plataforma Piazza, el enlace del video complementario que está alojado en la plataforma YouTube, indicando la fecha de vencimiento de la entrega de la asignación, y adicionalmente, se envía un recordatorio por dos canales: correo electrónico y grupo de Facebook de la asignatura.

Las asignaciones son recibidas mediante la plataforma Piazza, y los resultados de las evaluaciones son publicados en un documento en línea para que todos los estudiantes puedan ver el progreso del curso. Las asignaciones son las actividades relacionadas con cada video, por lo tanto, se tiene un total de diez asignaciones. La evaluación se realiza de manera binaria, es decir, se asigna 1 si el estudiante logró el objetivo de la actividad y 0 en caso contrario.

En la semana cinco se realiza la primera evaluación de la asignatura, que consta de tres enunciados con orden de complejidad creciente. El primer enunciado tiene como objetivo verificar que el estudiante puede escribir un programa sencillo; el segundo enunciado busca comprobar si el estudiante entiende las estructuras de repetición; y el tercer enunciado persigue determinar si el estudiante es capaz de programar una de las figuras clásicas utilizando asteriscos. La evaluación de los enunciados se realiza de manera binaria, es decir, todo o nada, por lo tanto, el primer enunciado vale 5, el segundo 7 y el tercero 8; en caso de responder incorrectamente todas las preguntas se califica con el mínimo valor, el cual es 1.

El análisis de los resultados se realiza por separado a los semestres A-2017 y A-2018, según las siguientes dimensiones: diagrama de barras de las asignaciones realizadas, estadísticas de la primera evaluación de la asignatura, relación entre cantidad de asignaciones realizadas y la primera evaluación, y relación entre la primera evaluación y los aprobados de la asignatura. Adicionalmente, considerando que los resultados fueron mejores de lo esperado, se hace un análisis de las tres mejores actividades para conocer en profundidad el uso (por parte de los estudiantes) de las instrucciones de la biblioteca PR1-ULA y los conceptos asociados a la asignatura, tales como: decisión simple, repita para, procedimientos y parámetros.

5. Resultados

5.1. Semestre A-2017

El estudio se realiza a 31 estudiantes del semestre A-2017 y 15 de ellos realizaron todas las actividades, los cuales representan el 48.39% del total. En la Fig. 1 se presenta un diagrama de barras que indica la cantidad de veces que se realizó cada actividad, donde en el eje horizontal, se representan los 10 videos, y en el eje vertical, la cantidad de estudiantes que realizaron cada actividad, por lo tanto, el valor máximo del eje vertical es 31.

Las calificaciones de la primera evaluación de la asignatura tienen un promedio de 5.68, lo cual indica que la mayoría de los estudiantes reprobó la evaluación. La mediana es 5 y la moda es 1, lo cual justifica que la desviación estándar sea 5.20. En este semestre sólo un estudiante tuvo calificación de 20 puntos, el cual representa el 3.22% y 11 estudiantes obtuvieron 1 punto, los cuales representan el 35.48%. En la Tabla 5 se presentan los parámetros estadísticos de la primera evaluación.

El coeficiente de correlación biserial puntual es calculado para: primero, determinar la relación entre los estudiantes que realizaron todas las actividades de las sesiones y las calificaciones obtenidas en la primera evaluación de la asignatura; y segundo, para determinar la relación entre las

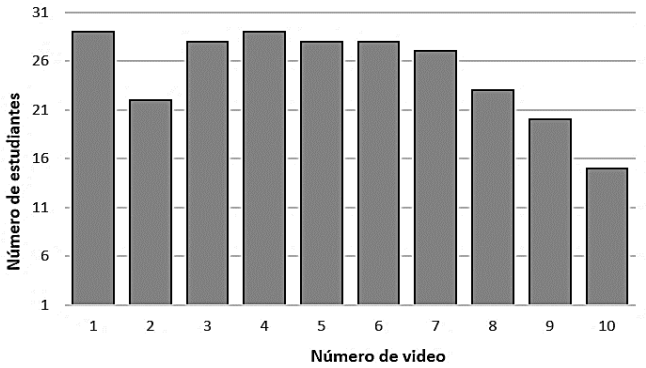

Figura 1. Actividades realizadas en el semestre A-2017.
Fuente: Los autores.

Tabla 5.
Parámetros estadísticos de la primera evaluación de la asignatura (A-2017).

Parámetro	Valor
Media	5.68
Mediana	5
Moda	1
Desviación estándar	5.20
Calificación mínima	1
Calificación máxima	20

Fuente: Los autores.

calificaciones finales de la asignatura con respecto a los que aprobaron la primera evaluación. En el primer caso, la variable que ha sido medida en una escala de intervalos es el rendimiento en la primera evaluación, representado por las calificaciones, y la variable dicotómica está relacionada con la realización de todas las actividades, de manera que, si el estudiante no realizó todas las actividades se le asigna 1 y en caso contrario se le asigna 2. El resultado obtenido es 0.38924, lo cual se interpreta de la siguiente manera: las calificaciones altas de la primera evaluación, están asociadas a los estudiantes que hicieron todas las actividades.

En el segundo caso, la variable medida en una escala de intervalos son las calificaciones finales de la asignatura y la variable dicotómica está relacionada con la condición de la primera evaluación (aprobado o reprobado), obteniendo como resultado 0.4887, lo cual indica que las calificaciones altas de las notas definitivas están asociadas a los estudiantes que aprobaron la primera evaluación.

5.2. Semestre A-2018

Los estudiantes que participaron en el semestre A-2018 fueron 20, y 6 de ellos realizaron todas las actividades, los cuales representan el 30% del total. En la Fig. 2 se presenta un diagrama de barras que indica la cantidad de veces que se realizó cada actividad. En el eje horizontal, se representan los 10 videos, y en el eje vertical, la cantidad de estudiantes que realizaron cada video, por lo tanto, el valor máximo del eje vertical es 20. Las actividades más frecuentes fueron 3, 4 y 5, realizadas por 16 estudiantes, los cuales representan el 80% del total.

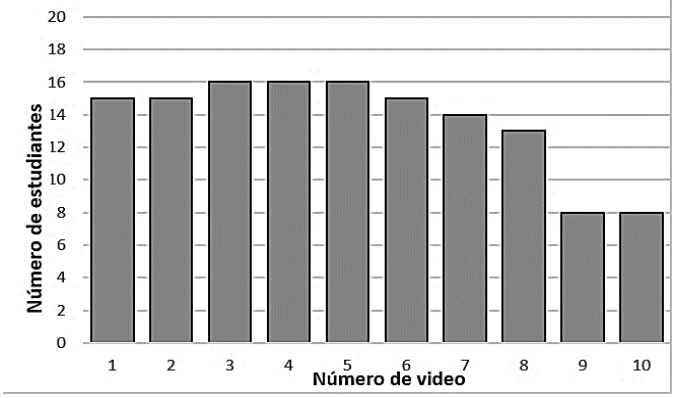

Figura 2. Actividades realizadas en el semestre A-2018.
Fuente: Los autores.

Tabla 6.
Parámetros estadísticos de la primera evaluación de la asignatura (A-2018)

Parámetro	Valor
Media	8.85
Mediana	7
Moda	1
Desviación estándar	7.83
Calificación mínima	1
Calificación máxima	20

Fuente: Los autores.

Las actividades menos frecuentes fueron la 9 y 10, realizada por 8 estudiantes, que representan el 40% del total. El diagrama de barras tiene comportamiento similar al de una campana, comenzando en un valor para las actividades 1 y 2, incrementando y manteniéndose desde la actividad 3 hasta la 5, y disminuyendo progresivamente hasta la actividad 10. En la semana cinco se realizó la primera evaluación parcial, y los parámetros estadísticos de las calificaciones se muestran en la Tabla 6. Las calificaciones tienen un promedio de 8.85, indicando que la mayoría de los estudiantes reprobó la evaluación. La mediana es 7, la moda es 1 y la desviación estándar es 7.83. En este semestre cuatro estudiantes obtuvieron la calificación máxima (20 puntos), los cuales representan el 20% y 7 estudiantes obtuvieron 1 punto, los cuales representan el 35%.

El coeficiente de correlación biserial puntual entre los estudiantes que realizaron todas las actividades de las sesiones y las calificaciones de la primera evaluación es 0.24144, lo cual indica que las calificaciones altas de la primera evaluación parcial se relacionan con los estudiantes que hicieron todas las actividades. Por otra parte, el coeficiente de correlación biserial puntual entre las calificaciones finales de la asignatura con respecto a los estudiantes que aprobaron la primera evaluación es 0.5158, lo cual indica que las calificaciones altas de las notas definitivas están asociadas a los estudiantes que aprobaron la primera evaluación.

5.3. Mejores actividades

Las actividades realizadas por los estudiantes permitieron descubrir el potencial de la biblioteca PR1-ULA, el cual fue mejor de lo esperado. De hecho, en principio, la evaluación estaba planteada de manera dicotómica mediante la entrega de un video de la ejecución del programa, y sin embargo, motivado a que la actividad 10 (consistió en realizar una historia animada), al permitir la utilización de todos los elementos de la biblioteca, generó tan buenos resultados, que se solicitaron los códigos de las tres mejores actividades para realizar su respectivo análisis, cuyo criterio de selección se basó en la votación por parte de los estudiantes.

En la Fig. 3 se presenta Kirby, una animación alusiva al videojuego denominado con el mismo nombre, que muestra una representación donde el personaje principal avanza horizontalmente, y al igual que en el videojuego, puede conseguir comida que aumenta su vitalidad, absorber a sus enemigos y copiar sus ataques. En la parte inferior de la imagen se puede observar una barra de vida que disminuye si Kirby es atacado o aumenta si éste realiza un ataque exitoso. La codificación del programa tiene 2165 líneas y la ejecución tiene una duración de 1 minuto y 37 segundos.

En la Fig. 4 se muestra una animación que también es alusiva a un videojuego, en este caso a Megaman, donde se presenta una escena de batalla con uno de sus enemigos, el cual es conocido como Cutman. En la imagen se puede ver cómo Megaman le lanza poderes a Cutman y también se puede apreciar un fondo degradado que no puede ser realizado con la biblioteca PR1-ULA, lo cual indica que el estudiante hace uso de la creatividad para modificar el fondo de su terminal de Linux y usarlo a su conveniencia. La codificación de este programa tiene 27143 líneas y su duración es de 45 segundos.

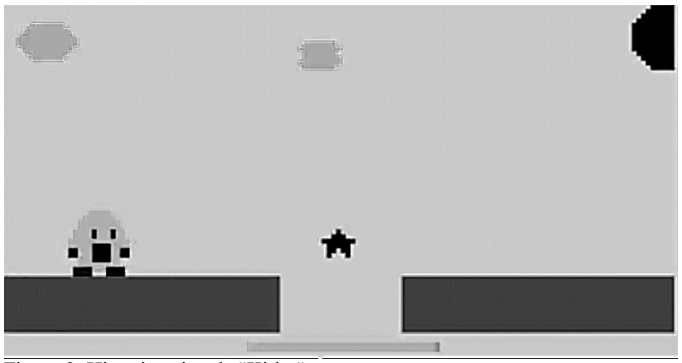

Figura 3. Historia animada "Kirby".
Fuente: Los autores.

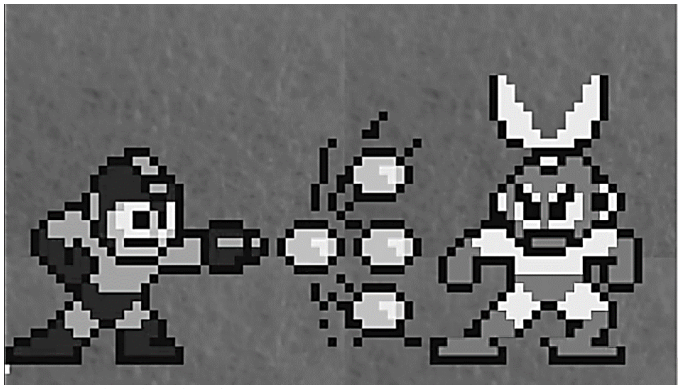

Figura 4. Historia animada "Megaman".
Fuente: Los autores.

La Fig. 5 presenta una animación tipo "video musical" de la banda Pink Floyd que muestra imágenes del prisma y la descomposición de la luz, representativa del albúm "The Dark Side of the Moon", y la pared de ladrillos que hace alusión a la canción "Another Brick in the Wall", la cual se reproduce durante toda la animación. Esta secuencia animada cambia los patrones de los ladrillos y agrega o quita elementos a conveniencia según el ritmo de la música. El código fuente de este programa tiene 3655 líneas y la ejecución tiene una duración de 51 segundos.

Los códigos fueron revisados para verificar la utilización de la biblioteca PR1-ULA, determinando la frecuencia de uso de las nueve instrucciones y de los conceptos asociados (decisión simple, repita para, procedimientos y parámetros), los cuales se muestran en la Tabla 7. Por una parte, las tres animaciones coinciden en que la instrucción más utilizada fue imprimirColor y en el poco uso de las instrucciones imprimir, imprimiEntero y reproducirAudio. Aunque Kirby tiene la mayor duración (1 minuto y 37 segundos), Megaman es quien tiene la mayor cantidad de líneas (27143), cuyo valor se separa significativamente de las otras animaciones (Kirby tiene 2165 y Floyd tiene 3655 líneas). La explicación a esta diferencia se observa en la cantidad de veces que en Megaman se utilizó la estructura de repetición "repita para" y la de decisión simple, siendo 0 y 1602 respectivamente, lo cual indica debilidades en la utilización adecuada de estas estructuras. Finalmente, el uso de parámetros en los procedimientos permite reutilizar de una mejor manera las instrucciones, tal como sucede en Kirby, donde la duración de la animación es la mayor y la cantidad de líneas es la menor.

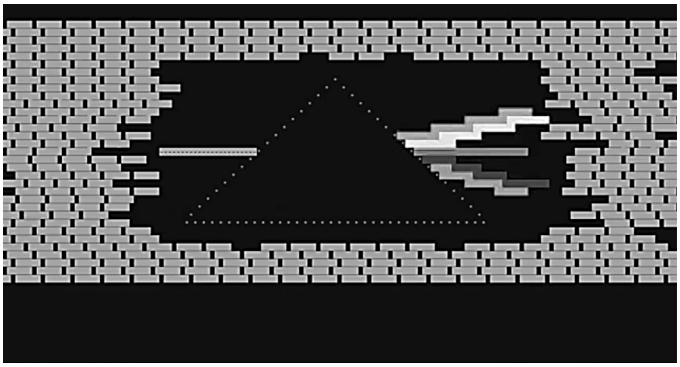

Figura 5. Historia animada "Floyd".
Fuente: Los autores.

Tabla 7.
Frecuencia de uso de elementos asociados a PR1-ULA.

Elemento	Kirby	Megaman	Floyd
void limpiar();	44	62	13
void saltar();	88	1681	253
void espaciar(int);	71	2202	19
void imprimir(char *);	2	1	0
void imprimirColor(char*,char*,char*);	368	11675	1578
void imprimirEntero(int);	0	0	0
void esperar(int);	1	17	4
void esperarMilisegundos(int);	3	43	19
void reproducirAudio(char *);	2	4	1
Decisión simple	293	1602	175
Repita para	86	0	270
Procedimientos	21	7	6
Parámetros	19	0	2

Fuente: Los autores

6. Discusión

En el semestre A-2017 la cantidad de estudiantes que participaron en el experimento fue mayor, sin embargo, aunque sus porcentajes siempre son mayores, se pueden observar comportamientos similares con el semestre A-2018 en la cantidad de estudiantes que realizaron: primero, todas las actividades; segundo, las actividades más frecuentes; y tercero, las actividades menos frecuentes. En ese sentido, se puede decir que ninguna actividad fue realizada por todos los estudiantes, que la actividad 4 coincide en ambos semestres como más frecuente y que la actividad 10 coincide como menos frecuente. En general, se observa en los estudiantes una tendencia a entregar las actividades centrales y disminuir las entregas en las actividades finales (en ambos casos menos del 50% de los estudiantes entregaron la última actividad).

En ambos semestres la mayoría de los estudiantes reprobó la primera evaluación, obteniéndose el mayor promedio (8.85) y mediana (7) en el semestre A-2018. La calificación más repetida fue 1 para ambos casos y la mayor calificación fue 20, obtenida por 1 y 4 estudiantes para los semestres A-2017 y A-2018 respectivamente. La mayor dispersión de las notas se presenta en el semestre A-2018 motivado a que el 20% de los estudiantes obtuvo 20 puntos y el 35% obtuvo 1 punto.

En ambos semestres el coeficiente de relación biserial puntual que relaciona las calificaciones de la primera evaluación parcial con la realización de todas las actividades fue favorable, es decir, las calificaciones altas de la primera evaluación parcial, están relacionadas con los estudiantes que hicieron todas las actividades de los videos. En el semestre A-2017 la correlación fue mayor, sin embargo, la diferencia entre ambos coeficientes no es muy distante (se encuentra alrededor de una décima).

En las secciones analizadas de los semestres anteriores (B-2015 y A-2016), la correlación entre las calificaciones finales de la asignatura y la aprobación de la primera evaluación parcial es alta, en promedio 0.67. En ese sentido, en los semestres A-2017 y A-2018, se verifica esa relación, y aunque el coeficiente de correlación acusa menor relación, ésta sigue siendo alta.

El análisis de los códigos de las actividades es importante porque permite evaluar la comprensión de los conceptos asociados a la programación que fueron explicados en las sesiones. Por ejemplo, la cantidad de líneas de los programas de las tres mejores actividades está relacionada con la comprensión de algunos de los conceptos explicados, donde la comprensión de esos conceptos es inversamente proporcional a la cantidad de líneas. En ese sentido, en Megaman no se utilizaron parámetros en los procedimientos y tampoco se usó la estructura de repetición "repita para", indicando que el estudiante no comprendió esos conceptos; en Floyd se utilizaron muchas estructuras de repetición y pocos parámetros, evidenciando una comprensión parcial del concepto de parámetros; y en Kirby se observan frecuencias de uso proporcionales de estructuras de repetición, parámetros y procedimientos, lo cual indica una comprensión total de esos conceptos.

7. Conclusiones

La revisión de investigaciones recientes relacionadas con cursos introductorios de programación mostró que la alta tasa de reprobados es un factor común en diferentes universidades. La estrategia LE1 proporciona un primer acercamiento amistoso a la programación mediante 10 sesiones de clases que están basadas en la biblioteca PR1-ULA, la cual permite realizar animaciones con colores y reproducir audios, utilizando sólo nueve instrucciones, en concordancia con los requerimientos de la asignatura "Programación 1": utilización de la terminal de Linux y programación en lenguaje C.

Los resultados de los experimentos aplicados en los semestres A-2017 y A-2018, permitieron demostrar que la incorporación de la estrategia LE1, para la realización de los primeros programas de los estudiantes, tiene incidencia positiva en las calificaciones de la primera evaluación, es decir, las calificaciones altas de esta evaluación están relacionadas con los estudiantes que realizaron las diez actividades asignadas en las diez sesiones de clases de la estrategia LE1. Al mismo tiempo, los resultados permitieron verificar la alta relación que existe entre las calificaciones definitivas y la condición de la primera evaluación (aprobado o reprobado).

Adicionalmente, la inclusión de la estrategia LE1 en las primeras semanas de clases, permitió descubrir el potencial de programación de algunos estudiantes en un corto periodo de tiempo, el cual se ve reflejado en las mejores actividades presentadas (Kirby, Megaman y Floyd). En ese sentido, es conveniente que estudiantes de próximos semestres conozcan estas actividades, en aras de fomentar su creatividad. Por otra parte, es importante incorporar una estrategia para analizar todos los códigos de cada actividad con el propósito de detectar las debilidades de los estudiantes y aplicar las correcciones pertinentes.

Referencias

[1] Timmermann, D., Kautz, C. and Skwarek, V., Evidence-based re-design of an introductory course "Programming in C", in: 2017 IEEE Frontiers in Education Conference (FIE), pp. 1-5, 2016. DOI: 10.1109/FIE.2016.7757492

[2] Hegazi, M.O. and Alhawarat, M., The challenges and the opportunities of teaching the introductory computer programming course: case study, in: 2015 Fifth International Conference on e-Learning (econf), pp. 324-330, 2015. DOI: 10.1109/ECONF.2015.61

[3] Blas, M.J., Hauque, F., Re, S. and Castellaro, M., A support tool designed as didactic material for teaching and learning programming, in: 2017 XLIII Latin American Computer Conference (CLEI), pp. 1-10, 2017. DOI: 10.1109/CLEI.2017.8226382

[4] Dapozo, G.N., Greiner, C.L., Petris, R.H., Espíndola, M.C. and Company, A.M., Introduction to programming based on playful activities in the university, in: 2017 XLIII Latin American Computer Conference (CLEI), pp. 1-8, 2017. DOI: 10.1109/CLEI.2017.8226437

[5] López, P.E.M., Ciolek, D., Arévalo, G. and Pari, D., The GOBSTONES method for teaching computer programming, in: 2017 XLIII Latin American Computer Conference (CLEI), pp. 1-9, 2017. DOI: 10.1109/CLEI.2017.8226428

[6] Albarracín, M., Ferrigno, L. and Wachenchauzer, R., Adopt a language: a step towards the education of a reflective technology practitioner, in: 2017 XLIII Latin American Computer Conference (CLEI), pp. 1-7, 2017. DOI: 10.1109/CLEI.2017.8226445

[7] Araujo, A.L., Santos, J.S., Andrade, W.L., Guerrero, D.D. and Dagienė, V., Exploring computational thinking assessment in introductory programming courses, in: 2017 IEEE Frontiers in Education Conference (FIE), pp. 1-9, 2017. DOI: 10.1109/FIE.2017.8190652

[8] Gómez-Álvarez, M.C, Sánchez-Dams, R. and Barón-Salazar, A., Trouble hunters: a game for introductory subjects to computer engineering, in: 2016 XLII Latin American Computing Conference (CLEI), pp. 1-8, 2016. DOI: 10.1109/CLEI.2016.7833398

[9] González, L., Gómez, M.C. and Echeverri, J.A., Motivation and virtual education in computer science: case Universidad de Medellín-Colombia, in: 2017 IEEE Latin America Transactions, 15(6), pp. 1176-1181, 2017. DOI: 10.1109/TLA.2017.7932706

[10] Castellanos, H., Restrepo-Calle, F., González, F.A. and Echeverry, J.J.R., Understanding the relationships between self-regulated learning and students source code in a computer programming course, in: 2017 IEEE Frontiers in Education Conference (FIE), pp. 1-9, 2017. DOI: 10.1109/FIE.2017.8190467

[11] Hidalgo-Céspedes, J., Marín-Raventós, G. and Lara-Villagrán, V., Student understanding of the C++ notional machine through traditional teaching with conceptual contraposition and program memory tracing, in: 2015 Latin American Computing Conference (CLEI), pp. 1-8, 2015. DOI: 10.1109/CLEI.2015.7360049

[12] Muñoz, R., Barcelos, T., Villarroel, R. and Frango, I., Using scratch to support programming fundamentals, International Journal on Computational Thinking (JCThink), 1(1), pp. 68-78, 2017. DOI: 10.14210/ijcthink.v1.n1.p68

[13] Abidin, H.Z. and Zaman, F.H.K., Students' perceptions on game-based classroom response system in a computer programming course, in: 2017 IEEE 9th International Conference on Engineering Education (ICEED), pp. 254-259, 2017. DOI: 10.1109/ICEED.2017.8251203

[14] Khaleel, F.L., Ashaari, N.S., Wook, T.S. and Ismail, A., Gamification-based learning framework for a programming course, in: 2017 6th International Conference on Electrical Engineering and Informatics (ICEEI), pp. 1-6, 2017. DOI: 10.1109/ICEEI.2017.8312377

[15] Azmi, S., Iahad, N.A. and Ahmad, N., Attracting students' engagement in programming courses with gamification, in: 2016 IEEE Conference on e-Learning, e-Management and e-Services (IC3e), pp. 112-115, 2016. DOI: 10.1109/IC3e.2016.8009050

[16] Fuentes-Rosado, J.I. y Moo-Medina, M., Dificultades de aprender a programar, Revista Educación en Ingeniería, 12(24), pp. 76-82, 2017. DOI: 10.26507/rei.v12n24.728

[17] Delgado, A. and Paragulla, J.V., Improving teaching and learning in systems programming courses using participatory action research, in: 2016 IEEE Congreso Argentino de Ciencias de la Informática y Desarrollos de Investigación (CACIDI), pp. 1-5, 2016. DOI: 10.1109/CACIDI.2016.7786000

[18] Almeida, R. and Pessoa, T., Scratch software in higher education: pedagogical experience in educational science, in: 2017 International Symposium on Computers in Education (SIIE), pp. 1-5, 2017. DOI: 10.1109/SIIE.2017.8259653

[19] Hsu, W.C. and Lin, H.C.K., Impact of applying WebGL technology to develop a Web digital game-based learning system for computer programming course in flipped classroom, in: 2016 International Conference on Educational Innovation through Technology (EITT), pp. 64-69, 2016. DOI: 10.1109/EITT.2016.20

[20] Su, J.M. and Wang, S.J., A Web-based learning activity integrated with scratch tool to support programming learning, in: 2017 10th International Conference on Ubi-media Computing and Workshops (Ubi-Media), pp. 1-4, 2017. DOI: 10.1109/UMEDIA.2017.8074137

[21] Tek, F.B., Benli, K.S. and Deveci, E., Implicit theories and self-efficacy in an introductory programming course, in: 2018 IEEE Transactions on Education, pp. 1-8, 2018. DOI: 10.1109/TE.2017.2789183

[22] Topalli, D. and Cagiltay, N., Improving programming skills in engineering education through problem-based game projects with Scratch, Computers & Education, 120, pp. 64-74, 2018. DOI: 10.1016/j.compedu.2018.01.011

[23] Erol, O. and Kurt, A.A., The effects of teaching programming with scratch on pre-service information technology teachers' motivation and achievement, Computers in Human Behavior, 77, pp. 11-18, 2017. DOI: 10.1016/j.chb.2017.08.017

[24] Adorjan, A. and Kereki, I.F., Academic misconduct in projects: perspective of students and teachers of introductory computer science courses, in: 2017 XLIII Latin American Computer Conference (CLEI), pp. 1-7, 2017. DOI: 10.1109/CLEI.2017.8226375

[25] Barrera, Y.C., Skills and strategies to promote learning algorithms and programming. evolution from learning objectives, in: 2017 XLIII Latin American Computer Conference (CLEI), pp. 1-10, 2017. DOI: 10.1109/CLEI.2017.8226464

[26] Pérez, J. y Castro, J., Estímulo del razonamiento lógico mediante el juego Millonario en C para la asignatura "Programación 1"., Tekhné, 21(3), pp. 8-14, 2018.

[27] Pérez, J. y Castro, J., LRS1: un robot social de bajo costo para la asignatura "Programación 1", Revista Colombiana de Tecnologías de Avanzada, 2(32), pp. 68-77, 2018.

[28] Pérez, J. and Pedroza, O., LM1: una metodología de estudio para la asignatura "Programación 1", Revista Educere, 22(73), pp. 635-648, 2018.

[29] Pinto, A. and Escudeiro, P., The promotion of the century learning skills through the development of games using scratch, in: 2017 12th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1-6, 2017. DOI: 10.23919/CISTI.2017.7975870

[30] Chang C. K., Using Computational Thinking Patterns to Scaffold Program Design in Introductory Programming Course, 2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), pp. 397-400, 2016. DOI: 10.1109/IIAI-AAI.2016.27

[31] Bauer, A., Butler, E. and Popović, Z., Dragon architect: open design problems for guided learning in a creative computational thinking sandbox game, in: Proceedings of the 12th International Conference on the Foundations of Digital Games. ACM, 2017, 26 P.

[32] Cernuda, A., Hevia, S., Suárez, M. y Gayo, D., Un estudio sobre el absentismo y el abandono en asignaturas de programación. ReVisión, 6(1), pp. 30-38, 2013.

[33] Deitel, H. y Deitel, P., Cómo programar en C/C++ y Java, Pearson Educación, México, 2004.

J.A. Pérez, es Ing. Electrónico en 2012, del Instituto Universitario Politécnico "Santiago Mariño", Mérida, Venezuela, Ing. de Sistemas en 2014 de la Universidad de Los Andes, Venezuela y MSc. en Educación Superior Mención Docencia Universitaria, en 2015 de la Universidad Fermín Toro, Mérida, Venezuela. Actualmente es profesor del Departamento de Computación de la Escuela de Ingeniería de Sistemas de la Universidad de Los Andes, Venezuela e integrante del Laboratorio de Sistemas Discretos, Automatización e Integración. Sus líneas de investigación incluyen: interacción humano–robot y enseñanza de la ingeniería.
ORCID: 0000-0002-6585-2648

M.A. Azuaje, es Ing. de Sistemas en 2018 de la Universidad de Los Andes, Mérida, Venezuela, e integrante del Laboratorio de Sistemas Discretos, Automatización e Integración. Sus líneas de investigación son: robótica y enseñanza de la programación de computadoras.
ORCID: 0000-0002-4370-0985