

Entre la lógica y los lenguajes. ¿Qué favorece más el aprendizaje de la programación?

Omar Iván Trejos Buriticá & Luis Eduardo Muñoz Guerrero

Facultad de Ingenierías, Universidad Tecnológica de Pereira, Pereira, Colombia.
omartrejos@utp.edu.co, lemunozg@utp.edu.co

Resumen— El área que tiene más relación con la Ingeniería de Sistemas es la programación. Aprender a programar implica apropiarse de un conjunto de reglas lógicas y las posibilidades de unos lenguajes para implementar soluciones a determinados problemas en el computador. La búsqueda de caminos para que esta lógica de programación, que es diferente a la lógica deliberativa, sea asimilada es uno de los retos que tienen los ingenieros docentes que se desempeñan en esa área. El objetivo de este artículo es aportar una experiencia de investigación en el aula que permitió establecer un comparativo entre dos tendencias: la enseñanza de la lógica de programación o la enseñanza de los lenguajes de programación, como caminos diferentes para aprender a programar. La metodología utilizada implicó la designación de dos cursos paralelos con los cuales se desarrolló todo el curso, por los dos caminos planteados. Los resultados dejan entrever ventajas y desventajas que tienen cada uno de los dos caminos como preparación para los futuros ingenieros. Se concluye que en la medida en que se exploren caminos diferentes para la enseñanza de la programación, en esa misma medida se va a posibilitar el hallazgo de formas efectivas y eficientes de aprenderla.

Palabras Clave— Aprendizaje, ingeniería, lenguajes, lógica, programación.

Recibido: 28 de julio de 2022. Revisado: 5 de noviembre de 2022. Aceptado: 28 de febrero de 2023.

Between logic and languages. What favors the learning of programming the most?

Abstract— The area that is most closely related to Systems Engineering is programming. Learning to program implies appropriating a set of logical rules and the possibilities of some languages to implement solutions to certain problems in the computer. The search for ways for this programming logic, which is different from deliberative logic, to be assimilated, is one of the challenges facing teaching engineers who work in this area. The objective of this article is to provide a research experience in the classroom that allowed establishing a comparison between two trends: the teaching of programming logic or the teaching of programming languages, as different ways to learn to program. The methodology used involved the designation of two parallel courses with which the entire course was developed, along the two paths proposed. The results suggest advantages and disadvantages that each of the two paths have as a preparation for future engineers. It is concluded that to the extent that different paths are explored for teaching programming, to that same extent it will be possible to find effective and efficient ways to learn it.

Keywords— Engineering, languages, learning, logic, programming.

1 Introducción

La temática por tratar consiste en la búsqueda de caminos que permitan la apropiación de la lógica computacional, y su

articulación con la lógica deliberativa humana, en la implementación de soluciones a problemas computables [1], labor que permanentemente deben hacer los docentes y los ingenieros docentes que atienden las asignaturas de programación en las ingenierías, especialmente ingeniería de sistemas, y particularmente en los primeros semestres. Diversos estudios realizados hasta el momento acerca de las concepciones y creencias sobre la enseñanza han mostrado que los profesores poseen mayoritariamente representaciones que no se corresponden con las teorías actuales centradas en el estudiante [2] por lo que el objetivo de este artículo consiste en presentar los resultados de una investigación realizada en el aula en la cual se pretende comparar la efectividad tanto de la lógica de programación como de los lenguajes de programación, por caminos separados, en el aprendizaje, apropiación, asimilación, retroalimentación y evaluación de la programación en un paradigma específico dentro del contexto del I semestre de Ingeniería de Sistemas en una universidad pública. Lo que se busca es aportar a la discusión permanente en cuanto a dos de las formas más sugeridas para la enseñanza de la programación, viéndola desde una reflexión matemática y lógica o desde las posibilidades directas y puntuales que ofrecen los lenguajes de programación ya que es conveniente que el profesor de ingeniería sea diestro en la enseñanza y también en la investigación [3].

Si bien no ha de desconocerse que se tiende a pensar que los estudiantes de Ingeniería de Sistemas primero debieran conocer el fundamento matemático y lógico de la programación, como objeto de estudio, y después aplicar dicho conocimiento a través de las posibilidades que ofrezca un lenguaje de programación específico dentro del marco de un paradigma definido [4], no puede descartarse como objeto de investigación que el camino directo a través de los lenguajes, sin pasar por el fundamento matemático lógico, puede ser otra forma de aprender la programación toda vez que los jóvenes de hoy han tenido tanto contacto con la tecnología que mucho del pensamiento computacional forma parte de su lenguaje natural y que, para los efectos de las aplicaciones que se derivan de los lenguajes de programación, resultan ser ventajas comparativas con otras generaciones que bien vale la pena explorar pues el mundo de hoy exige que un buen profesional emplee lo aprendido para



atender las situaciones en su diario vivir interactuando la mayoría de las veces en un equipo de trabajo en donde tendrá la oportunidad de plantear sus iniciativas respetando las ideas de los demás [5]. La escolástica tradicional plantea la necesidad de conocer los fundamentos epistemológicos de un saber para acudir a las aplicaciones de dicho saber pero, en tiempos de hoy, es posible que la academia se encuentre ante una nueva forma de ver el mundo, en relación con la tecnología, y es que para llegar a sus aplicaciones no necesariamente tenga que conocerse sus fundamentos y que, precisamente, es lo que se quiere indagar en esta investigación [6].

El documento está estructurado según la forma estándar IMRYD [7] según la cual se plantea una introducción que se sigue de la exposición de la metodología como se realizó la investigación, de allí se derivan unos resultados que se discuten desde una perspectiva crítica y analítica para finalizar con unas conclusiones y las referencias bibliográficas que han servido de soporte para el desarrollo de este. En cuanto a su contenido, el artículo expone las necesidades que llevan a tener esta inquietud investigativa. Posteriormente presenta una metodología en paralelo que permite realizar los comparativos que invitan a las reflexiones que de allí se derivan. Es importante tener en cuenta que se busca aportar a la discusión en cuanto a la estructura y contenido de las asignaturas de Programación en los programas de Ingeniería de Sistemas de forma que se tenga en cuenta esta experiencia especialmente en el diseño de los cursos de primeros semestres pues se supone que un mayor conocimiento de la brecha que distancia -o que aproxima- las expectativas iniciales sobre el aprendizaje y las valoraciones posteriores que los mismos estudiantes efectúan, permitirá brindar herramientas conceptuales y metodológicas más sensibles para promover mejoras en las propuestas de enseñanza [8].

Los criterios que se han utilizado para el desarrollo de esta investigación y la confección del presente artículo están basados en a) la experiencia de los autores investigadores e ingenieros docentes como profesores de programación durante más de dos décadas, b) la relación con los estudiantes de primeros semestres y el impacto de la articulación de la lógica deliberativa humana con la lógica computacional para la construcción de soluciones que pone a los alumnos ante un mundo académico que, igualmente, deben prepararse para que estén preparados para articularse y afrontar una sociedad que los está requiriendo con gran necesidad [9] y c) la apropiación del pensamiento computacional como camino para aprender a pensar en términos en que el vínculo entre la tecnología y la sociedad de hoy lo exige [10]. Estos criterios han sido aplicados a la redacción del presente artículo y al diseño de la investigación sobre el principio investigativo de dejar que los datos hablen y siempre con una mentalidad proactiva en favor del aprendizaje de la programación.

El problema por abordar consiste en: ¿Qué favorece más en el aprendizaje de la programación: los fundamentos matemáticos y lógicos o los lenguajes de programación? Definirlo no es tan simple pues precisamente este ha sido tema de muchas salas de profesores a lo largo del tiempo que se ha invertido en el diseño de currículos y contenidos de las asignaturas de programación en los programas de Ingeniería de Sistemas de todo el mundo. Enfrentarse a esta pregunta implica

pensar en que vale la pena aferrarse menos a lo tradicional, incluso dejarlo ir, y permitir que la actualidad, a través de sus datos, cuente realidades que podrían ser insospechadas en un mundo que, hoy más que nunca, está atravesado por tecnología, pantallas y dispositivos electrónicos que posibilitan acceso a la información y a canales de comunicación [11].

Por citar un simple dato, en el año 2021 en Colombia la población era cercana a los 47 millones de habitantes y existían aproximadamente 60 millones de aparatos celulares activos [12]. Eso demuestra la gran penetración que tiene la tecnología en el mundo actual y, sin citar otros datos de computadores, PCs, portátiles y tabletas, deja entrever que el tema del aprendizaje de la programación sigue hoy más vigente que nunca luego pensar en encontrar caminos que permitan dicho aprendizaje de forma más efectiva en términos del tiempo [13] es un deber de los investigadores de las universidades.

Para el desarrollo de esta investigación se ha acudido a diferentes teorías y modelos que permiten tener un fundamento teórico para llegar a resultados confiables. Se consultó los aportes de la teoría del aprendizaje significativo [14], el aprendizaje por descubrimiento [15], el conectivismo [16], el aprendizaje invisible [17] y la teoría del pensamiento computacional [18]. Además se conocieron investigaciones de otros autores que han aportado a la discusión acerca de la manera como se sugiere enseñar la programación y su impacto en los caminos para que el aprendizaje suceda de la mejor de las formas, es decir, de manera efectiva y eficiente.

La importancia de esta temática radica en que poco a poco se ha ido popularizando la necesidad de aprender a programar como parte de las competencias básicas en el mundo de hoy, mucho más cuando se tienen iniciativas como Code.Org que invita a que la programación sea tema de los currículos de básica secundaria y de nivel superior de todo el mundo pues, incluso, se está comenzando a considerar esta temática como parte de las ciencias básicas dada las necesidades del mundo de hoy y la interacción permanente con la tecnología, sus herramientas, sus facilidades y sus servicios. El artículo se justifica porque a) es muy importante estar buscando, por el camino de la investigación científica y tecnológica, nuevos y mejores caminos para mejorar el aprendizaje en diferentes materias del saber, b) programas como ingeniería de sistemas han tenido la mayor preferencia en los últimos diez años en toda Latinoamérica, c) la programación está comenzando a ser parte de los currículos de los programas de formación de básica secundaria y educación superior de forma masiva y d) toda revisión metodológica, conceptual y epistemológica es necesaria cuando se trata de temas que tienen tanto impacto en la sociedad de hoy.

Como hipótesis para esta investigación se puede considerar la siguiente: ¿qué facilita más el aprendizaje de la programación: la enseñanza de sus fundamentos matemáticos y lógicos o la utilización de lenguajes de programación directamente para el desarrollo de programas solución a problemas computables? Este artículo es un subproducto del proyecto de investigación 6-19-11 “Desarrollo de un modelo de enseñanza y aprendizaje que transversalise del conocimiento derivado de las Ciencias Básicas aprovechando la programación de computadores en Ingeniería de Sistemas

basado en *Brain Based Learning* y Pensamiento Computacional” tramitado y aprobado por la Vicerrectoría de Investigaciones, Innovación y Extensión de la Universidad Tecnológica de Pereira.

2 Metodología

Para el desarrollo de la presente investigación se trabajó con grupos de I semestre de Ingeniería de Sistemas y Computación de la Universidad Tecnológica de Pereira en la asignatura Programación I cuyo contenido corresponde al paradigma de programación funcional, durante el período comprendido entre el I semestre 2019 hasta el I semestre 2022, incluyéndolo. En este período, en todos los semestres de estudio, se seleccionaron dos grupos paralelos de la misma asignatura. Con uno de ellos se trabajó el contenido de la asignatura Programación I a partir de los elementos que provee el modelo matemático y la lógica, en una mirada teórica y acudiendo a los fundamentos que corresponden al paradigma de programación funcional, sin utilizar en ninguna herramienta computacional hasta llegar hasta la semana 12. Con el otro grupo se trabajó la misma asignatura y los mismos contenidos desde la perspectiva del lenguaje de programación DrRacket sin acudir en ningún momento al modelo matemático que subyace a dicho paradigma ni a la lógica de programación como base para el desarrollo de soluciones a problemas computables, hasta llegar a la semana 12.

En la semana 12, fueron unificados los contenidos, es decir, en el grupo donde se adoptó solamente el modelo matemático y la lógica de programación para cubrir el contenido, se empezó a utilizar el lenguaje de programación DrRacket. Por su parte, en el grupo con el cual se había trabajado durante el semestre el contenido de la asignatura Programación I a partir del uso exclusivo del lenguaje de programación DrRacket, y sin incluir el modelo matemático que le subyace al paradigma funcional ni a la lógica de programación, se empezó a presentar la parte puramente teórica durante las 4 semanas que restaban para finalizar el semestre, lo mismo que en el otro grupo.

Durante el semestre el contenido de la asignatura se segmentó de la siguiente forma para los dos grupos: a) de la semana 1 a la semana 4 se cubrió el tema de las características del paradigma funcional, la notación (bien matemática y lógica para un grupo o la sintáctica para el otro grupo) y el concepto de función; b) de la semana 5 a la 8, la estrategia “divide y vencerás”, los operadores y los condicionales (función a trozos desde lo matemático y decisiones desde lo lógico para un grupo y la instrucción *if* para el otro grupo); c) de la semana 9 a la 12, el concepto de recursividad (funciones recursivas y su notación desde lo matemático y la auto invocación y la invocación mutua desde el lenguaje de programación); de la semana 13 a la semana 16 (que es la última semana del período académico semestral) se revisaron los conjuntos de datos (matrices desde la perspectiva matemática y vectores desde lo lógico y el concepto de listas vectoriales en el lenguaje de programación DrRacket. Es en este último período en donde se unifican los contenidos tanto en su fondo como en su forma pues ambos grupos comparten tanto lo matemático, lo lógico y lo sintáctico.

Durante el semestre, en ambos cursos, se realizaron tres evaluaciones parciales y una evaluación final. Las evaluaciones parciales se hicieron en las semanas 4, 8 y 12 y la evaluación final se hizo en la semana 16. Las evaluaciones parciales se diseñaron en consonancia con el estilo temático que se había adoptado para la asignatura pero con el mismo tema para ambos grupos, cada uno desde su respectiva arista fuera esta conceptual o práctica. La evaluación final fue la misma para ambos grupos y se realizó conjuntamente con ambos grupos, en el mismo horario y en las mismas condiciones, y su diseño fue el mismo en lo temático y en lo tecnológico.

Al finaliza cada semestre se solicitó a los alumnos de cada grupo, manifestar la experiencia a lo largo del semestre planteando de forma libre y espontánea tanto las ventajas como las desventajas y respondiendo a la pregunta ¿aprendió a programar? Para el tratamiento de la información cualitativa recogida en estas opiniones se acudió a la técnica de nube de palabras y análisis semántico para decantar y poder seleccionar las respuestas más representativas desde la perspectiva de que fueran las que recogieran el sentido de la mayoría de dichas respuestas. Los datos cuantitativos respecto de las notas de las evaluaciones tanto parciales como finales fueron tratados adoptando como medida central la media aritmética una vez se pudo comprobar que las conclusiones e inferencias no se afectaran por los valores extremos, pero sin excluir ningún valor.

3 Resultados

En la tabla 1 se presenta la cuantificación de los estudiantes que participaron en la investigación distribuidos según año y semestre.

Para todos los efectos en la presentación de las tablas de resultados, la sigla MyL significa “Matemáticas y Lógica” y la sigla LP significa “Lenguaje de Programación”.

En la tabla 2 son presentados los resultados de las evaluaciones parciales en cada uno de los grupos, simplificados a través de la media aritmética para facilitar su presentación y análisis.

Tabla 1
Estudiantes participantes

Año	Sem	Grupo con MyL	Grupo con LP	Total
2019	I	24	22	46
2019	II	22	21	43
2020	I	21	23	44
2020	II	23	21	44
2021	I	22	21	43
2021	II	20	22	42
2022	I	21	22	43
Total		153	152	205

Fuente: Elaboración propia.

Tabla 2
Resultados evaluaciones parciales

Año	Sem	Grupo con MyL			Grupo con LP		
		IP	IIP	IIIP	IP	IIP	IIIP
2019	I	3,7	4,2	4,1	3,6	4,1	3,4
2019	II	3,6	4,4	4,0	3,7	4,5	3,2
2020	I	3,8	4,3	3,9	3,7	4,2	3,3
2020	II	3,9	4,4	3,9	3,8	4,4	3,4
2021	I	3,5	4,4	4,0	3,6	4,3	3,4
2021	II	3,8	4,3	4,1	3,7	4,4	3,3
2022	I	3,6	4,2	4,0	3,7	4,3	3,4
Promedio		3,7	4,3	4,0	3,7	4,3	3,3

Fuente: Elaboración propia.

En la tabla 3 son presentados los resultados de las evaluaciones finales en cada uno de los grupos igualmente facilitados con la adopción de la media aritmética como camino para facilitar la presentación y el análisis.

Tabla 3
Resultados evaluaciones finales

Año	Sem	Grupo con MyL	Grupo con LP	Dif
2019	I	3,2	4,1	0,9
2019	II	3,3	4,3	1,0
2020	I	3,2	4,5	1,3
2020	II	3,1	4,2	1,1
2021	I	3,2	4,3	1,1
2021	II	3,3	4,4	1,1
2022	I	3,2	4,3	1,1

Fuente: Elaboración propia.

En la tabla 4 son presentadas las opiniones libres de los estudiantes, condensadas a través de la técnica de nube de palabras y análisis semántico de forma que se pudieran exponer las respuestas que mejor interpreten a las demás sin quitarles sentido ni intención. Se han considerado como opiniones positivas, aquellas que motivan al estudiante a continuar en el curso dentro de la modalidad investigativa que le correspondió y como opiniones negativas aquellas que tienen destellos de desánimo o que desmotivan al estudiante en la forma de abordar la asignatura.

Tabla 4
Opiniones de los estudiantes

Ítem	Grupo con MyL	Grupos con LP
Opiniones Positivas	La concepción es más general	La solución se vé (sic) literalmente
	La representación es más científica	El computador facilita todo
	Se aleja uno de los computadores	Uno programa con el computador
	Las soluciones son genéricas	Se puede ver si la solución funciona
	Uno no depende de un computador	Los lenguajes son más fáciles
Opiniones Negativas	La notación matemática es aburridora	Todo depende del lenguaje de programación
	Si no hay computador, no hay programación	¿...y si a uno le cambian el lenguaje de programación?
	Esta clase parece de matemáticas	Un simple error de sintaxis lo frena a uno
	¿Cuándo habrá computador?	Las soluciones son muy específicas
	Las soluciones no siempre son tan entendibles	El computador ayuda pero no piensa por uno
¿Aprendió a programar?	Si.....72%	Si.....87%
	No.....28%	No.....13%

Fuente: Elaboración propia

Para los efectos de los objetivos trazados en esta investigación, se les hizo seguimiento a los estudiantes, de forma individual, en las dos asignaturas que seguían en su plan de estudios (Programación II y Programación III). Se pudieron recoger datos de más del 50% de los estudiantes y se pudo observar que los estudiantes que en I semestre asignatura Programación I estuvieron en los cursos MyL tuvieron un rendimiento académico tanto cuantitativo como cualitativo notablemente mejor que los grupos LP. Este patrón de comportamiento se mantuvo en todos los estudiantes a quienes se les hizo seguimiento y es motivo de análisis en el aparte destinado para tal fin.

Teniendo en cuenta que la investigación cuantitativa busca describir un fenómeno objeto de investigación mientras que la investigación cualitativa posibilita comprender dicho fenómeno, puede decirse que este artículo presenta una investigación cualitativa que busca comprender dos de los caminos posibles a través de los cuales se puede aprender a programar asimilando sus conceptos, sus modelos y sus posibilidades tecnológicas, dentro de un marco descriptivo y experimental con cursos en paralelo. Las técnicas de recolección de información que se adoptaron incluyeron observación directa, evaluaciones escritas tanto parciales como finales y recepción de opiniones conceptuales sobre el resultado del curso como camino para aprender a programar, por parte de los estudiantes.

Los parciales, como instrumentos de recolección de información, se diseñaron de forma similar para los dos tipos de cursos y se buscaron horarios y condiciones similares para su presentación. Fueron diferentes en cuanto al contenido como tal por las razones académicas que implican la posibilidad de copia y fraude. Los datos cuantitativos fueron tratados usando una hoja electrónica y los datos cualitativos fueron abordados desde la perspectiva de la técnica de nube de palabras y análisis semántico. Todos los datos son fiables y se garantiza la confiabilidad de su recolección, de su procesamiento y de las conclusiones a las cuales se ha llegado. La interpretación de los datos se realizó sobre la base del aprendizaje de la programación y su relación con una metodología basada en los fundamentos matemáticos y lógicos como un camino posible o con otro camino representado por el uso directo de los lenguajes de programación de forma directa y sin acudir a los fundamentos epistemológicos de la programación y su respectivo paradigma.

4 Discusión y Análisis de Resultados

Lo primero que debe aclararse es el significado de la palabra Lógica pues desde la filosofía se puede describir como aquella parte que se encarga de estudiar lo que se refiere a formas y principios que, desde una perspectiva general, rigen el corpus del conocimiento y los caminos que recorre el pensamiento humano sin depender o referirse a los objetos de forma particular. También es concebida la Lógica como aquella forma de razonar en donde una sucesión de hechos o de ideas se van desarrollando con una coherencia que no permite contradicción entre ellas.

La lógica que nos compete en el sentido del contenido del presente artículo tiene que ver con dos aristas: la lógica deliberativa humana y la lógica binaria computacional. La lógica deliberativa humana es aquel conjunto de razonamientos que permite que una persona haga algo por el camino que, a su juicio, es el más apropiado, el más sencillo y el más fácil. Podría decirse que la lógica deliberativa es la mirada personal desde la cual se define el camino más fácil para lograr un objetivo. Por su parte la lógica binaria computacional implica la combinación de la tecnología binaria que se representa sobre 1s y 0s, es decir, que solamente tiene dos estados y unas reglas que los dinamizan y que han servido como base para el desarrollo de los dispositivos, servicios y facilidades tecnológicas de la actualidad.

La lógica computacional implica computadores, es decir, dispositivos que tengan capacidad de recibir información, de almacenarla, de recuperarla a alta velocidad y de generar respuestas oportunas y veraces. La lógica deliberativa humana es la que se usa cuando se deben tomar decisiones en medio de estructuras, sistemas o interacciones humanas y la lógica binaria computacional es la que se usa para que las máquinas (computadores) realicen operaciones instruidas y programadas por los seres humanos pero en las cuales se puede aprovechar su alta capacidad de almacenamiento y recuperación de información, su inmensa capacidad de procesamiento y todo el potencial de generar respuestas basadas en la información almacenada y en el momento en donde éstas se requieren.

Las matemáticas, por su parte, abren el espacio para la lógica matemática que es una forma de representar las reglas, los postulados y las interacciones entre planteamientos, enunciados y proposiciones que se enmarcan en un contexto de forma que se pueda verificar, comprobar, evaluar y retroalimentar su validez. Si bien la lógica deliberativa humana ha acompañado al ser humano desde sus primeras instancias de alta evolución, la lógica matemática ha permitido formalizar muchos de sus razonamientos, reflexiones y planteamientos al punto que uno de sus resultados más significativos en la actualidad es la lógica de programación que puede definirse, en palabras simples, como el conjunto de reglas, interacciones y símbolos que permite lograr un objetivo expresándolo de manera formal y que se puede implementar a través de un lenguaje de programación. Esta es la razón por la cual, a cada paradigma de programación subyace un modelo matemático que lo formaliza y lo hace efectivo desde la mirada simbólica y científica.

Ahora bien, la lógica de programación se cristaliza a través de los lenguajes de programación (valga esta redundancia) pues éstos son conjuntos de instrucciones que pueden ser entendidos por los computadores para realizar tareas específicas que el ser humano (programador) necesita que se realicen aprovechando sus potencialidades. La inquietud investigativa de este artículo radica en preguntarse qué es más enriquecedor para los estudiantes de programación de I semestre: el aprendizaje e interiorización de los fundamentos de la lógica desde una perspectiva formal como las matemáticas o el aprendizaje de los lenguajes de programación a través de sus instrucciones y sus propósitos de forma directa, como caminos posibles para aprender a programar.

Los lenguajes de programación se enmarcan dentro de paradigmas de programación que son modelos que posibilitan una mirada específica de la manera como debe entenderse el mundo para, asimismo, incorporarlo a un proceso computacional. De esta forma, el paradigma de programación funcional concibe el mundo desde el uso y utilidad de las cosas privilegiando el concepto de función como la base para el desarrollo de cualquier programa pues en ella se condensa la razón de ser, la utilidad y los resultados, de un ente informático cuando se interactúa con éste. Un programa a la luz de este paradigma no es más que un conjunto de instrucciones que permite la interacción (invocación) entre funciones.

Por su parte, el paradigma de programación imperativo visualiza el mundo a través de los cambios, de la transformación de estados y de la manera formal como éstos pueden describirse a través de los sistemas que para tal fin el ser humano ha desarrollado. De allí se colige que este paradigma privilegia el concepto de variable que puede definirse como un espacio de memoria en donde se puede registrar, confiablemente y con la perdurabilidad requerida, los cambios que suceden en un entorno del mundo real. Un programa bajo este paradigma es un conjunto de instrucciones que permiten la interacción entre variables y sus consecuentes cambios.

El paradigma de programación orientado a objetos visualiza el mundo como un conjunto de objetos, es decir, como una serie de elementos independientes e interdependientes que son el resultado de sus características y sus usos. Las características se llaman atributos y los usos se conocen como métodos. Un ente que puede describirse a través de sus atributos y sus métodos es lo que se conoce como un objeto. El ser humano ha llegado a ser capaz de describir científicamente objetos que no se pueden tocar, ni ver, ni palpar y aun así ha podido lograrlo. Un programa orientado a objetos consiste en un conjunto de instrucciones que permiten la interacción entre objetos tal como sucede en el mundo real, razón por la cual en la mayoría de los casos, es el paradigma que mejor simula cualquier entorno real dentro del computador, como es el caso de los juegos.

Como todos obedecen a un modelo matemático que formalmente se conoce como paradigma de programación y que, en lo práctico, se hace realidad a través de un lenguaje de programación, la investigación va en el sentido de pensar cuál es el camino más efectivo para aprender la programación, si lo es a partir del modelo matemático que subyace a su paradigma de programación o lo es a partir del lenguaje que lo cristaliza, ya que son dos aristas diferentes. El modelo y paradigma puede considerarse como sus fundamentos y los lenguajes de programación puede considerarse como su puesta práctica en escena. Todo esto considerando la naturaleza misma de los jóvenes de hoy que por la alta penetración de la tecnología en el mundo actual, la han convertido en parte de su lenguaje natural.

Sobre esta base se analizarán los resultados obtenidos en la investigación. En la tabla 1 se presenta la cuantificación de los estudiantes que participaron en la investigación. Fueron 205 en total a lo largo de 7 semestres de estudio y que corresponden al 35% aproximadamente de la población de nuevos estudiantes que ingresaran al programa Ingeniería de Sistemas y Computación de la Universidad Tecnológica de Pereira. Esa

misma proporción se mantuvo durante cada semestre, si se toma como base los 120 estudiantes que ingresan en cada período académico. Para las explicaciones que prosiguen debe tenerse en cuenta que la sigla MyL se refiere a los grupos en los cuales se adoptó la matemática y la lógica como camino de aproximación a la programación. La sigla LP se refiere a los grupos en los cuales se hizo referencia a los Lenguajes de Programación.

La tabla 2 presenta los resultados de las evaluaciones parciales a la luz de la metodología explicada en el aparte destinado para ello. Se puede observar una gran similitud en los resultados, de forma comparativa, en lo que se refiere al parcial I y al parcial II. Debe anotarse que por facilidad de la presentación de los resultados, éstos se promediaron en cada curso una vez que se comprobó que los valores extremos no afectan ni la interpretación ni las conclusiones que al respecto puedan tomarse. Las temáticas cubiertas en el 1er bloque evaluativo (de la semana 1 a la semana 4) y el 2º bloque (de la 5ª a la 8ª semana) parecieran tener un mismo impacto en el aprendizaje toda vez que los resultados de las evaluaciones parciales son similares tanto en el promedio de cada curso como en el promedio de los promedios que se calcula al final.

El parcial III requiere un análisis diferente pues puede observarse que en los grupos MyL el promedio de los promedios es igual a 4,0 mientras que en los grupos LP dicho resultado es igual a 3,3. Debe anotarse que durante este período, la temática vista era el concepto de recursividad que consiste en la posibilidad de que una función pueda invocarse a sí misma (si se mira desde la perspectiva del lenguaje de programación) o que puede definirse en términos de sí misma (se es vista desde las matemáticas y la lógica). Esto confiere un sentido muy especial a la programación funcional pues la recursividad exige que las funciones se deban concebir desde una estructura claramente definida que facilita tanto su diseño como su implementación y que comprende a) definición de la función, b) condición de parada y c) llamado recursivo. Los resultados parecieran indicar que es mucho más fácil, para los alumnos, concebir la recursividad desde lo formal que desde lo práctico, es decir, mirar los procesos recursivos desde sus formalismos matemáticos es más sencillo, en términos del aprendizaje, que desde la aplicación a través de los lenguajes de programación.

La tabla III presenta los resultados de las evaluaciones finales que si bien cubren toda la temática vista durante el semestre y se han diseñado de forma similar en ambos tipos de cursos, se hace hincapié en el concepto de conjuntos de datos (listas y vectores) que, a juzgar por los resultados, parecieran ser mucho más digeribles en términos de aprendizaje, desde su aplicación a través de los lenguajes de programación que desde sus formalismos matemáticos y lógicos a un punto en que la diferencia entre un promedio y otros es de 1,1 que representa más del 20% del valor máximo a alcanzar (5,0) y que bien puede endilgarse a la ganancia en términos de aprendizaje.

En la tabla 4 se presenta la opinión de los estudiantes discriminando las respuestas entre opiniones positivas (como aquellas que destacan bondades de la metodología usada) y opiniones negativas (como aquellas que critican alguna arista de la metodología utilizada). Debe anotarse que usando las técnicas de nube de palabras y análisis semántico, se decantaron

las respuestas para poder seleccionar las que interpretan mejor a las demás o que, en algún sentido positivo o negativo, encarnan lo que quisieron manifestar todos los estudiantes. Se destaca la generalidad de la representación de la lógica a través de sus formalismos matemáticos lo cual inhibe la dependencia de los lenguajes de programación en lo que se refiere a los grupos MyL. En cuanto a los grupos LP, en lo positivo destacan la posibilidad de ver la solución funcionando y de esta forma acudiendo a un camino que comprueba si se resolvió el problema o no, desde lo puramente práctico además que, por la misma naturaleza de los jóvenes de hoy, encuentran en estos lenguajes una forma más de seguir inmersos en ese mundo de tecnología que forma parte de su lenguaje natural.

En cuanto a lo negativo, se destaca por parte de los grupos MyL que por momentos todo pareciera indicar que es una clase de matemáticas y no de programación, lo cual es cierto pero de donde debe colegirse que son esas matemáticas las que proveen una generalidad que permitirá después que los estudiantes pueden entender y comprender fácilmente cualquier lenguaje de programación que se enmarque dentro de este paradigma o dentro de otro. En cuanto a los grupos LP admiten la dependencia tan alta de un lenguaje de programación pues están sujeto a sus posibilidades, a su sintaxis y a las instrucciones que éste provea además de saber que finalmente es el programador el que termina endilgando al lenguaje responsabilidades que no tiene cuando la solución no funciona pues se tiende a pensar, según sus propias palabras, que el computado ayuda pero no es el que piensa. La respuesta a la pregunta ¿aprendió a programar? tiene dos matices muy interesantes. De una parte, el 72% de los estudiantes de los grupos MyL consideran que han aprendido a programar desde una perspectiva general, formal y científica pues su base han sido la lógica y las matemáticas. Por su parte los grupos LP aceptan haber aprendido a programar en un 87% a lo cual debe acotarse que esta es una sensación que pueden tener pero con la restricción que la lógica de programación se ha formado desde la mirada de un lenguaje específico de programación que no impide una mirada general pero que tampoco la promueve.

5 Conclusiones

Teniendo en cuenta que el objetivo de este artículo consistió en presentar los resultados de una investigación realizada en el aula con la cual se pretendía comparar la efectividad tanto de la lógica de programación como de los lenguajes de programación, por caminos separados, en el aprendizaje, apropiación, asimilación, retroalimentación y evaluación de la programación en un paradigma específico dentro del contexto del I semestre de Ingeniería de Sistemas en una universidad pública buscando aportar a la discusión permanente en cuanto a dos de las formas más sugeridas para la enseñanza de la programación sea esta desde una reflexión matemática y lógica o desde las posibilidades directas y puntuales que ofrecen los lenguajes de programación, puede decirse que éste objetivo se ha cumplido toda vez que puede pensarse en que el aprendizaje de la programación puede mejorarse notoriamente, en el marco de un curso de programación funcional en I semestre de Ingeniería de Sistemas con lenguaje DrRacket (tal como se

especificó en la investigación), si se tienen en cuenta los siguientes factores: en primer lugar, dividir la temática de las 16 semanas en cuatro bloques de 4 semanas.

En el 1° y 2° bloque se exponen el concepto de función y los condicionales (junto con todos sus recursos y variantes), respectivamente. En estos bloques pareciera ser indiferente, por los resultados obtenidos, si se acude a la explicación de la programación a partir de la modelación matemática y su respectivo paradigma o bien si se acude a los lenguajes de programación directamente. El 3er bloque (de la semana 9 a la semana 12) se debe ocupar del concepto de recursividad y, por lo obtenido en los resultados, pareciera que es mucho más efectivo en términos de aprendizaje si se acude al modelo matemático, a sus procesos formales y a la lógica como área específica de representación. El 4° bloque (cuya evaluación corresponderá a la evaluación final y que va de la semana 13 a la semana 16), deberá ocuparse de los conjuntos de datos pero desde la perspectiva que ofrecen los lenguajes de programación pues los resultados así lo evidencian.

Referencias

- [1] E. Serna-Montoya, «La Ingeniería», *Revista Digital Lamsakos*, n° 1, pp. 12-21, 2009.
- [2] F.A. Buffa, M.B. García, y otros. «Concepciones acerca de la enseñanza en docentes de ingeniería», *Educación en Ingeniería*, vol. 15, n° 30, pp. 18-25, 2020.
- [3] M.E. Guerrero, «Formación de investigadores para la educación en ingeniería», *Educación en Ingeniería*, vol. 17, n° 33, pp. 1-2, Febrero 2022.
- [4] O. Trejos, Programación Imperativa con Lenguaje C, Bogotá: ECOE Ediciones, 2017.
- [5] M. F. Serrano Guzmán y D. D. Pérez Ruiz, «Aprendizaje basado en Investigación», *Educación en Ingeniería*, vol. 17, n° 34, pp. 1-7, Julio 2022.
- [6] J. Alves, N. Lima y G. Alves, «Adjusting higher education competences to companies professional needs», *International Journal of human capital and information technology professionals*, vol. 8, n° 1, pp. 66-77, 2017.
- [7] R. Day, How to write and publish scientific works, Washington: The Oryx Press, 2005.
- [8] D. Schlegel, P. V. Paoloni, D. Donolo, «Expectativas y valoraciones sobre el contexto de aprendizaje», *Educación en Ingeniería*, vol. 16, n° 31, pp. 35-48, Febrero 2021.
- [9] O. M. Trejos, «Programación + Física: estrategia motivadora de aprendizaje en Ingeniería de Sistemas», *Educación en Ingeniería*, vol. 17, n° 33, pp. 63-67, Febrero 2022.
- [10] R. Bello, La educación en la sociedad del conocimiento, N.Y.: Kindle Unlimited, 2018.
- [11] B. Díaz, Estrategias docentes para un aprendizaje significativo, Buenos Aires: McGraw Hill Interamericana, 2010.
- [12] MinTIC, «Estadísticas de consumo» junio 2021. [En línea]. Available: <http://colombiatic.mintic.gov.co>. [Último acceso: junio 2021].
- [13] M. I. Cabrera, L. E. Nieto-Gómez y R. Giraldo-Díaz, «La investigación desde un referente epistemológico como aporte para la reflexión misional de las IES», *Entramado*, vol. 12, n° 2, pp. 188-202, 2016.
- [14] D. Ausubel, The acquisition and retention of knowledge, N.Y.: Springer, 2012.
- [15] J. Bruner, Actos de significado, Buenos Aires: Gedisa, 2010.
- [16] G. Siemens, Knowing Knowledges, Morrisville (NC): Lulu.com, 2006.
- [17] C. Cobo y J. Moravec, Aprendizaje invisible: hacia una nueva ecología de la educación, Barcelona: Publicacions i Edicions Universitat de Barcelona, 2011.
- [18] J. Wing, Computational thinking, Boston: Oreilly Publishing, 2017.

O. I. Trejos Buriticá. Ingeniero de Sistemas. Especialista en Instrumentación Física. MSc en Comunicación Educativa. PhD en Ciencias de la Educación. Docente de planta categoría Titular, Programa Ingeniería de Sistemas y Computación, Facultad de Ingenierías, Universidad Tecnológica de Pereira. Autor de varios libros de programación y de una buena cantidad de artículos de investigación científica educativa en el área de la programación de computadores sobre aproximación a la optimización de procesos de enseñanza y aprendizaje dentro del contexto de la formación de ingenieros con perfil tecnológico.
<https://orcid.org/0000-0002-3751-6014>

L. E. Muñoz Guerrero. Ingeniero de Sistemas. MSc en Ingeniería de Sistemas. Docente Titular de Planta Universidad Tecnológica de Pereira, con 15 años de experiencia en el campo de la formación universitaria. Autor de libros académicos y de investigación. Ha publicado artículos en revistas especializadas nacionales e internacionales. Su área de Investigación se centra en los procesos de enseñanza y aprendizaje de la Programación.
<http://orcid.org/0000-0002-9414-6187>