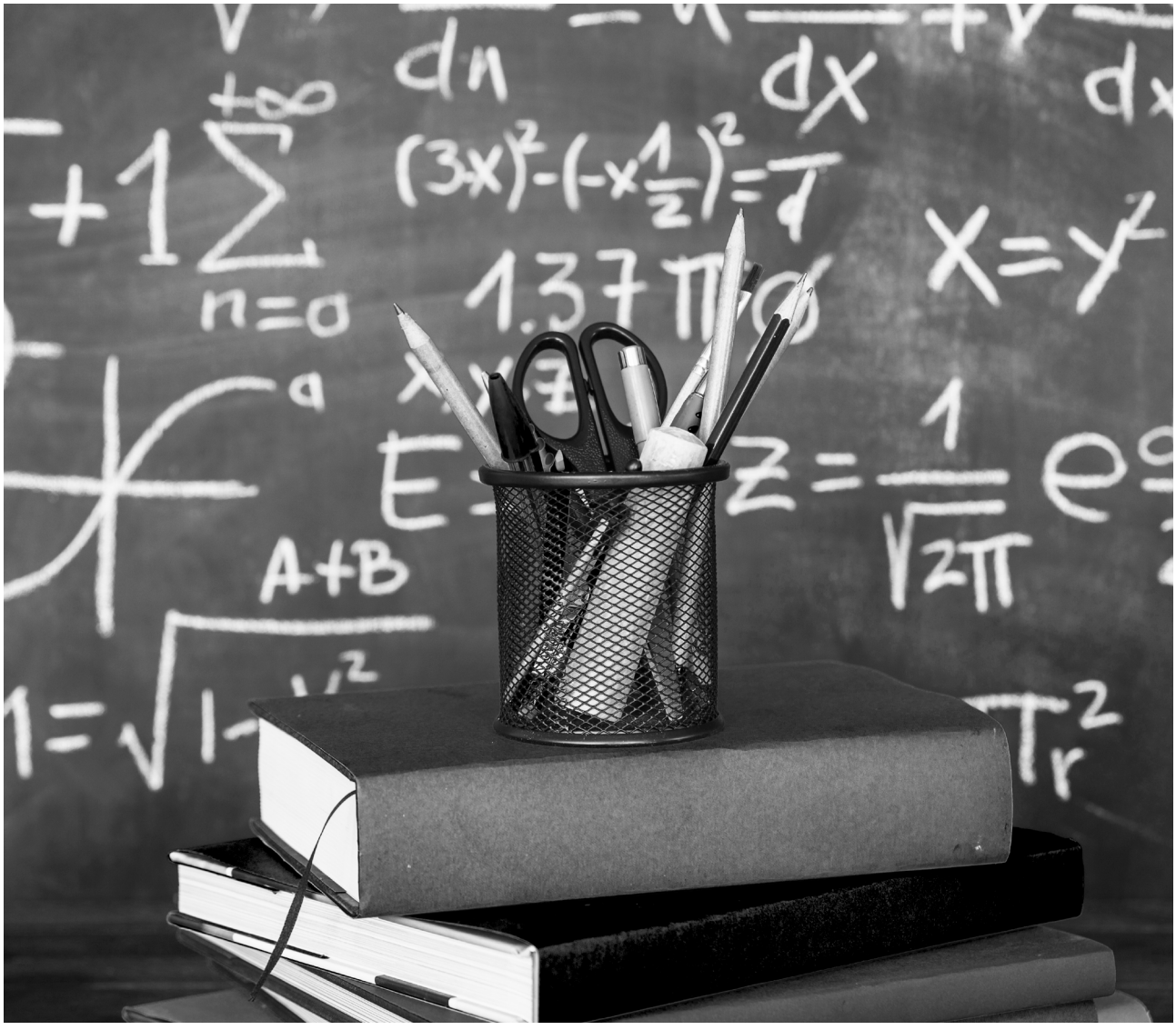
Metodología de aprendizaje para descomponer números naturales en factores primos usando programación funcional

Methodology to learn how to decompose a number into prime factors using functional programming



Fuente: freepik.com. Licencia Creative Commons.

UNIVERSIDAD DE INVESTIGACIÓN Y DESARROLLO

I+D Revista de Investigaciones
ISSN 2256-1676 / ISSN en línea 2539-519X
Volumen 13 Número 1 Enero-Junio de 2019 pp. 32-44

Metodología de aprendizaje para descomponer números naturales en factores primos usando programación funcional[1]

Methodology to learn how to decompose a number into prime factors using functional programming

Omar Iván Trejos Buriticá[2]

Artículo recibido en junio de 2017; artículo aceptado en septiembre de 2018.

y se referencia usando el siguiente formato: Trejos, O. I. (2019). Metodología de aprendizaje para descomponer números naturales en factores primos usando programación funcional. *I+D Revista de Investigaciones, 13* (1), 32-44.

DOI: https://doi.org/10.33304/revinv.v13n1-2019003

Resumen

El presente artículo propone una forma metodológica para enseñar y aprender a resolver el problema de descomponer, algorítmicamente, un número natural en sus factores primos valiéndose de los principios y recursos de la programación funcional dentro de un curso de programación de computadores de un programa de ingeniería. La metodología propuesta ha permitido exponer una manera conjunta de construir soluciones a problemas comunes y cotidianos de forma interdisciplinaria, pues vincula la programación, la lógica y las matemáticas. Los resultados indican que a partir de problemas heredados de las Matemáticas se pueden encontrar soluciones eficientes valiéndose de la programación de computadores y que estos problemas pueden ser familiares para los estudiantes. Se concluye que la metodología propuesta es efectiva y puede extrapolarse a otros temas y a otros cursos en diferentes programas de formación en ingeniería.

Palabras clave: algoritmo, aprendizaje, número natural, números primos, programación funcional, resolución de problemas.

Abstract

This article proposes a methodological way to teach and learn to solve the problem of decomposing, algorithmically, a natural number in its prime factors using the principles and resources of functional programming within a course of computer programming of an engineering program. The proposed methodology has allowed exposing a joint way of building solutions to common and everyday problems in an interdisciplinary way since it links programming, logic and mathematics. The results indicate that from problems inherited from mathematics, efficient solutions can be found using computer programming and that these problems can be familiar to students. It is concluded that the proposed methodology is effective and can be extrapolated to other subjects and to other courses in different engineering formation programs.

1. Artículo de investigación científica y tecnológica, de enfoque cualitativo, producto de un proyecto de investigación culminado: código 6-16-13 titulado "Desarrollo de un modelo metodológico para el aprendizaje de la programación imperativa en Ingeniería de Sistemas basado en aprendizaje significativo, aprendizaje por descubrimiento y el modelo 4Q de preferencias de pensamiento". Perteneciente al área de Ingeniería de sistemas. Desarrollado en la Universidad Tecnológica de Pereira (Pereira, Colombia). Dirección: Carrera 27 #10-02. PBX: PBX: +57 63137300 Fecha de inicio: enero de 2015 - Fecha de finalización: junio de 2017.

2. Ingeniero de Sistemas, especialista en Instrumentación Física, MSc en Comunicación Educativa, PhD en Ciencias de la Educación. Docente de planta, investigador, Ingeniería de Sistemas y Computación, Universidad Tecnológica de Pereira (Pereira, Colombia). Dirección: Carrera 27 #10-02. PBX: PBX: +57 63137300 ORCID ID: http://orcid.org/0000-0002-3751-6014. Correo electrónico institucional: omartrejos@utp.edu.co.

Keywords: algorithm, learning, natural number, prime numbers, functional programming, problem solving.

Introducción

Una de las competencias que necesitan desarrollar los estudiantes de Ingeniería de Sistemas, en tiempos modernos, es la capacidad de plantear problemas desde el formalismo que proveen las Matemáticas (Attard, Di Ioio & Geven, 2010), de plantear soluciones desde la perspectiva de la Lógica (Brassard & Bratley, 2006) y de resolverlos aprovechando las potencialidades que brinda la tecnología computacional moderna (Trejos, 2009), toda vez que dicho problema sea solucionable con tecnología. Si bien es cierto que un estudiante de Ingeniería de Sistemas ha de enfrentarse permanentemente a situaciones problema, también lo es que se genera una gran motivación cuando se le estimula a encontrar soluciones a problemas inherentes a su formación curricular y a resolverlos con saberes idóneos para su preparación universitaria (Fríes, Monzón & Di Paolo, 2014).

Es poco común encontrar el panorama completo en el primer semestre de Ingeniería de Sistemas, por una parte, debido a que los docentes solo algunas veces se dan a la tarea de empalmar las tres vertientes necesarias para establecer relaciones entre Matemáticas, Lógica y programación y, por otra a que no es usual que los docentes en programación de computadores, además de conocer los fundamentos propios de la programación, conozcan y apliquen teorías de aprendizaje que respalden su labor docente (Díaz, 2005).

En este artículo se expone una metodología completa, aplicada en el desarrollo de una posible solución al problema de la descomposición de un número natural en factores primos, usando programación funcional a través del lenguaje de programación DrScheme (Güiza, 2014). Se presenta la manera como se capitalizan las Matemáticas, la Lógica y la programación en favor de la formulación, búsqueda y solución de un problema computable de forma que el estudiante no solo pueda palpar la transversalidad del conocimiento y la interdisciplinariedad de la solución, sino que, además, pueda apropiarse de la forma como se implementa la solución desde el buen uso de la tecnología. De la misma manera, se presentan los resultados obtenidos después de socializar la metodología a utilizar en el aula, establecer unos objetivos y compartirlos con los estudiantes, y tras abrir unos espacios para que el mismo estudiante, protagonista de la escena educativa, plantee sus inquietudes, críticas, observaciones y aportes tanto desde lo cuantitativo como desde lo cualitativo.

De los diferentes modelos didácticos que se han sugerido en la enseñanza de la programación, en esta investigación se ha adoptado el modelo de resolución de problemas (Kaasboll, 1999), dadas sus ventajas y proximidades con los objetivos de aprendizaje a alcanzar, aunque debe aceptarse que este modelo puede considerarse más como un modelo de aprendizaje que como una estrategia de enseñanza (Romero & Rosero, 2014). La solución de problemas en Matemáticas siempre ha sido un tema recurrente en las investigaciones educativas en procesos de formación tecnológica, así como la ejecución de su solución. No existe una metodología única, pero sí es posible encontrar un camino de solución, tal como lo proponen Polya (1989) y Scott, Le Blanc y Rizzo (2014), desde el cual se puede establecer una ruta sistemática para llegar a una solución que no solo satisfaga plenamente el problema planteado, sino que, además, pueda ser implementada desde algún mecanismo. En este caso, se acude a la tecnología computacional para implementar la solución.

Cuando el estudiante se enfrenta a la resolución de problemas partiendo de la convergencia entre Matemáticas, Lógica y programación, el conocimiento adquiere sentido y significado. El sentido lo da la razón de ser de dicho documento, al verse necesario como sustento para fortalecer ese punto de convergencia; el significado lo confiere el hecho de convertirse en solución para un problema y, de paso, poder ver resuelto el problema con dicha solución. Adicionalmente, una vez conocida esta metodología cuyas bases son interdisciplinarias, la motivación para continuar aprendiendo, se genera automáticamente según Ausubel (1986), y eso favorece el aprendizaje autónomo y la participación activa del estudiante en sus procesos de aprendizaje, tal como lo sugiere el active learning con Brown (2011).

De acuerdo con esto, pareciera ser evidente que, tratándose de enseñanza y aprendizaje de la programación, no es suficiente (aunque es muy útil) que el estudiante llegue a niveles altos de conocimiento del nivel sintáctico del lenguaje de programación, de acuerdo con Jones (2007). Las tareas previas que se involucran en la solución de un problema implican conocimientos adicionales que exigen la fundamentación necesaria para que se fortalezca también el conocimiento conceptual, además del desarrollo de las correspondientes habilidades cognitivas básicas y la motivación necesaria para conjugar esas partes. Este tipo de investigaciones deja presente la necesidad de que el conocimiento que le queda al estudiante debe servir como base de competencia para que dicho conocimiento se aplique en otras situaciones, según Trejos en *Significado y Competencias* (2013), con otros temas y en relación con otras asignaturas.

De la misma manera, este tipo de estrategias de aprendizaje refuerza en los estudiantes la necesidad de fortalecer la capacidad de plantear soluciones a partir de modelos de representación formales, como los que proveen las Matemáticas, la facilidad de diseñar algoritmos basados en funciones y la posibilidad de construir soluciones a los problemas que se planteen, de forma que sean verificables (Acevedo & Ambrosio, 2014).

Marco teórico

En primera instancia, debe señalarse que la palabra "algoritmo" (que corresponde a la latinización del nombre del matemático árabe Al-Juarhizmi), de acuerdo con Kline (2012), ha adquirido en estos tiempos un significado muy importante, debido a que, en términos generales, un algoritmo se puede definir como el conjunto de pasos ordenados y secuenciales que, de una forma sistemática, permiten alcanzar plenamente un objetivo, según Trejos en Algoritmos *Problemas básicos* (2008). Son pasos ordenados, porque solo en determinado orden pueden realizarse para que se alcance el objetivo, y son secuenciales, debido a que cada paso se da uno detrás del otro; es decir, cada paso se ejecuta antes del siguiente y solo después del anterior. Esto implica que el algoritmo puede considerarse como el conjunto predefinido tanto de instrucciones como de reglas que, siendo ordenadas y finitas, permite que se realice una actividad a través de pasos sucesivos que, por su naturaleza y dentro de un determinado contexto, no generan dudas para quien deba realizarlos, sea una persona o una máquina (Deitel & Deitel, 2013).

Teniendo en cuenta que proviene del latín *numĕrus,* el concepto número se refiere a cada uno de los signos (o también el conjunto de signos) que posibilitan la expresión de una determinada cantidad con relación a lo que se considere como su unidad. Entre los grupos que pueden considerarse como número están los números reales (aquellos que permiten describir magnitudes continuas) y los números enteros (que son aquellos con los que se pueden describir magnitudes discretas no divisibles). Los números naturales se definen como todos aquellos números que se pueden utilizar para contar los elementos que forman parte de un conjunto y entre los cuales se pueden realizar operaciones de cálculo elemental, de acuerdo con Meavilla (2012).

Matemáticamente se ha establecido que cualquier miembro del conjunto N={1,2,3,4,…} es considerado como un número natural, lo cual posibilita que, a través de estos números, el ser humano no solo pueda determinar cuántos elementos tiene un conjunto y saber qué posición ocupa cada uno dentro del conjunto –en caso de que este sea ordenado– sino que, además, pueda establecer relaciones entre ellos (Boyer, 2010). Vale la pena anotar que, dado que el conjunto de números naturales es un conjunto infinito en el cual cada elemento se encuentra entre dos números, el número que se encuentra a la derecha se llamará sucesivo o también elemento siguiente (Chabert, 2005). Nótese que el conjunto de números naturales no comienza en el número 0 sino en el número 1, debido a que el número cero no es considerado –matemáticamente hablando– un número natural bajo el precepto de que, si los números naturales se usan para contar, el cero en realidad no cuenta nada. En este sentido, la teoría de conjuntos admite el cero como parte del conjunto de los números naturales, mientras que la teoría de números lo excluye (Jiménez, 2014).

Un número primo se ha definido como un número natural mayor que 1, cuyos dos únicos divisores exactos son el número 1 y el mismo número. Los números que no cumplen con esa condición se conocen como números compuestos, debido a que tienen algún divisor, número natural, diferente de ellos mismos y de la unidad (el valor 1). De acuerdo con Rey y Babini (2005), el número 1 no se considera –matemáticamente– como un número primo. La característica –elevada a la categoría de propiedad– de que un número sea un número primo se conoce como primalidad. El único número primo par es el número 2 y por eso los números primos también se conocen como números primos impares, según lo afirma Rooney (2009).

La teoría de números se ocupa de estudiar las propiedades, características y aplicaciones de los números primos desde una óptica fundamentalmente aritmética y dentro del marco de los números enteros (Shoup, 2008). Aunque la distribución de los números primos se ha convertido en un tema frecuente de investigación, los resultados obtenidos hasta el momento indican que los números primos parecieran estar distribuidos aleatoriamente, aunque, lo que pudiera llamarse como la distribución general (o global) pareciera seguir leyes claramente establecidas (Crilly, 2011).

Los números primos tienen gran importancia en los estudios que, en la actualidad, se hacen en relación con la seguridad informática, ya que ellos son los que posibilitan la construcción de las claves públicas que se utilizan para descifrar mensajes. Descomponer un número natural en sus factores primos consiste en encontrar el producto (coeficientes y números primos) para el cual dicho número es el resultado, dado que todo número natural puede escribirse como la multiplicación de dos o más factores primos (Crandall & Pomerance, 2005). De esta forma podemos encontrar que un número, por ejemplo el 24, es igual a la multiplicación de 2x2x2x3 o sea 2x3 en donde 2 y 3 son números primos.

Por su parte, un paradigma de programación puede considerarse como un conjunto específico de conceptos,

reglas, técnicas, metodologías y tecnologías que son adoptados, aceptados y aplicados por una determinada comunidad de desarrolladores de *software,* que posibilita el alcance de objetivos de programación como solución a problemas que sean computables y que estén claramente definidos, tal como lo afirma Trejos, en *La Esencia de la Lógica de Programación* (2000). Estos paradigmas tienen una estrecha relación con la ingeniería del *software*.

Los parámetros que caracterizan un paradigma de programación representan un enfoque específico o una filosofía que permite diseñar soluciones que resuelvan problemas. La diferencia entre los paradigmas se basa en la forma de concebir, de manera abstracta, los elementos que se involucran en el problema, de aplicar y plantear los conceptos que le subyacen y de establecer los pasos que determinan la posible solución que un problema requiera (Van Roy, 2008).

De acuerdo con Van Roy en *Techniques and methods in programming computer* (2008, el paradigma de programación funcional se basa en la aplicación del cálculo Lambda para proponer soluciones computables a problemas planteados, fundamentándose en el formalismo matemático que proveen las matemáticas en relación con el concepto de funciones, sus interrelaciones y sus interacciones. Al mismo tiempo, la programación funcional aprovecha conceptos como la recursividad para poder desarrollar procesos iterativos que, con un consumo determinado de máquina, posibilita el cálculo eficiente y eficaz de expresiones y la obtención de valores solución. Entre los diferentes lenguajes de programación con los cuales se puede aplicar el paradigma funcional está el lenguaje DrScheme, cuyas características de aproximación a la notación matemática y su entorno DrRacket facilitan la notación matemática, el planteamiento de soluciones y su implementación (Felleisen, 2006).

De acuerdo con Ausubel en *Psychology of Meaningful Verbal Learning: An Introduction to School Learning* (1963), la teoría de aprendizaje significativo establece que el aprendizaje se produce cuando el conocimiento adquirido tiene significado y sentido para el aprendiz, es decir, para el estudiante. El significado implica la respuesta a la pregunta ¿para qué sirve determinado conocimiento?, y el sentido implica la aplicación de dicho conocimiento en situaciones que se muevan dentro de un contexto cercano o alcanzable por el estudiante. De la misma manera, el aprendizaje significativo se sirve de tres elementos que constituyen su base: el conocimiento previo, el nuevo conocimiento y la actitud del estudiante. Como conocimiento previo se puede considerar todo lo que el alumno ya sabe; como nuevo conocimiento se define aquello que aún no sabe o a lo cual el alumno aún no ha accedido. La actitud del estudiante es la voluntad para aprender que está fuertemente mediada por la motivación, factor definitivo en el desarrollo de los procesos de formación, aprendizaje y adquisición de habilidades y destrezas (Bruner, 2009).

En el sentido de la motivación del estudiante por aprender, dos factores influyen en su proceso y son la motivación que el docente, guía o acompañante pueda brindarle y la manera como dicho estudiante se involucre en su proceso de formación y aprendizaje, de forma que sea él quien establezca sus propias metas, se exija y, además, pueda establecer mecanismos autónomos para determinar si está cumpliendo con los objetivos propuestos o no (Gomez, M., Gomez, P. & Gonzalez, 2007).

Metodología

Para desarrollar una solución sistemática que permitiera resolver el problema planteado se dividió el grupo de estudiantes en dos subgrupos. Los estudiantes se categorizaron en tres niveles de manera que la división en subgrupos fuera lo más balanceada posible. La metodología se aplicó en los dos semestres de los años 2014, 2015 y 2016. Debe aclararse que cuando se utiliza el término "metodología tradicional" se hace referencia a una metodología en la cual la exposición magistral es la protagonista, pero sin una estructura prevista ni una estrategia que haga partícipe al estudiante de su propio proceso de aprendizaje. La metodología propuesta en este artículo acude a una exposición magistral con una estructura prevista, buscando que el estudiante se involucre conscientemente en el proceso de aprendizaje. La Tabla 1 expone las fases de la metodología aplicada en esta investigación.

Como se puede observar, la metodología está orientada a que el estudiante no solo reciba el conocimiento que se le quiere brindar, sino que, además, se le haga partícipe en todo momento de los avances en cada fase de dicha metodología. De la misma manera, se pretende realizar una observación cualitativa en relación con la participación de los estudiantes en la implementación de la solución del problema para observar el impacto de la metodología en los estudiantes que fueron guiados con la metodología tradicional frente a los que fueron guiados con la metodología que se propone en este artículo. Cabe anotar que, cuando se habla de la metodología tradicional se refiere a la exposición magistral, en la cual no se destaca la relación entre las matemáticas y la programación, la notación formal que proveen las matemáticas para resolver un problema, ni se establecen nexos con la lógica, de manera que matemáticas y programación tengan un puente directo de comunicación. En el desarrollo de la fase 1 se les presenta a los estudiantes la metodología que se va a implementar y se comparten con ellos los propósitos y objetivos de esta. En cumplimiento de las fases 2, 3, 4 y 5 se distribuye el instrumento para categorizar a los estudiantes y se obtienen los resultados que se presentan en la Tabla 2.

Tabla 1
Metodología utilizada

Fase	Descripción	Objetivo
1	Se describe a los estudiantes los propósitos de la investigación y se expone la metodología que se va a utilizar.	Se pretende que los estudiantes, conscientes de lo que se va a realizar, se involucren y puedan capitalizar cada avance que se presente en las respectivas fases.
2	Se distribuye un instrumento que permite aproximarse al perfil de cada estudiante en tres niveles: altamente talentosos (AT), medianamente talentosos (MT) y bajamente talentosos (BT).	Aproximarse a las características de los alumnos de forma que la distribución en subgrupos se pueda realizar de manera más balanceada y equitativa.
3	Se obtienen los resultados de la categorización de los alumnos y se organizan los subgrupos de forma que la distribución sea equitativa.	Se pretende obtener unos resultados de gran objetividad de manera que, por azar, no queden sesgados los subgrupos y, por tanto, los resultados obtenidos.
4	Se distribuye la sesión de exposición de la siguiente forma: Grupo 1 con exposición tradicional y Grupo 2 con exposición aplicando la metodología.	Se busca que los dos subgrupos de estudiantes tengan las mismas condiciones generales para recibir la clase, de manera que se reduzcan los factores externos que puedan influir en los resultados.
5	En el subgrupo 1 se realiza la exposición magistral de forma tradicional durante una sesión completa. En el subgrupo 2 se realiza la exposición magistral partiendo del formalismo matemático y estableciendo un enlace entre matemáticas y programación de computadores.	Permitir que los dos subgrupos reciban el conocimiento por caminos diferentes para poder establecer diferencias y, de esa forma, poder determinar el impacto de la metodología utilizada.
6	Se abre un espacio para que los estudiantes propongan diferentes caminos algorítmicos como solución al problema matemático planteado.	Se busca que los estudiantes se articulen con la metodología y participen activamente en el desarrollo, implementación de la solución de un determinado problema.
7	Se implementa en un lenguaje de programación, a la vista de todos, la solución escogida como óptima por parte de los estudiantes.	Se pretende que los estudiantes sean partícipes de la construcción de un programa a nivel de código.
8	Se realizan pruebas y se pone a funcionar el programa atendiendo sugerencias, observaciones y gustos de los estudiantes.	Se busca motivar la participación activa de los estudiantes en la solución.
9	Se abre un espacio de opinión para que los alumnos, libremente, manifiesten sus inquietudes acerca de la metodología utilizada.	Se pretende retroalimentar el proceso a partir de las opiniones de los estudiantes.
10	Se realiza una prueba evaluativa que, partiendo del enunciado revisado, permita que el estudiante avance un poco más allá.	Verificar el impacto de la metodología en ambos grupos y constatar si existe diferencia significativa en los resultados cuantitativos y cualitativos.

Fuente: Autor.

Tabla 2
Categorización y distribución de estudiantes

Sem.	Grupo	Categorización de estudiantes			SubG	Distribución de estudiantes			Total Estudiantes
		AT	MT	BT		AT	MT	BT	
I 2014	1	4	8	7	1 2	2 2	4 4	4 3	19
II 2014	3	3	10	9	1 2	2 1	5 5	4 5	22
I 2015	1	4	11	9	1 2	2 2	5 6	4 5	23
II 2015	2	2	8	10	1 2	1 1	4 4	5 5	20
I 2016	1	3	10	6	1 2	2 1	5 5	3 3	19
II 2016	3	4	8	8	1 2	2 2	4 4	4 4	20

Nota: SubG = Subgrupo (1→Metodología tradicional, 2→Metodología propuesta). Fuente: Autor.

Resultados

La Tabla 3 muestra algunas opiniones de estudiantes que han estado en alguno de los dos grupos y también muestra el concepto del docente a partir de un proceso de observación.

Tabla 3
Opiniones de estudiante y docente

	Algunas opiniones de estudiantes	Opinión del profesor
METODOLOGÍA TRADICIONAL	• La explicación es muy normal • Es entendible el tema • Me parece que la metodología puede ser mejor • Un programa requiere de algo más • No veo lo nuevo • Me parece que el profe explica bien • No entendí nada • Eso yo ya lo sabía • Qué va a preguntar uno cuando no entiende • No veo para qué sirven las matemáticas • No me ha podido gustar la programación • Me siento bien • Me gusta la metodología • Programar es difícil • No he podido aprender a programar	Los estudiantes son muy pasivos, participan muy poco y pareciera que se molestaran si uno les pregunta algo. Son bastante dispersos y se entretienen con cualquier cosa. A pesar de los intentos pareciera que la programación no los sedujera mucho. Los estudiantes son más pasivos que los del semestre anterior, prestan atención (o por lo menos eso parece). Son más proclives a la programación, al punto que algunos han sugerido que los pase al otro grupo. La relación entre las diferentes categorías de estudiantes no es muy buena.
METODOLOGÍA SUGERIDA	• Muy entendible todo desde el principio • Ya sé para qué sirven las matemáticas • Programar es muy fácil • El sistema de fases es muy motivador • Todo el proceso me parece muy bueno • Me gusta que me involucren en lo que hacen • Programar es delicioso • La metodología es muy motivadora • Me gusta mucho esa relación entre matemáticas y programación • Las fases son muy claras • No creo que se pueda mejorar • El profe es un verraco (sic) • El profe es muy metódico	Los estudiantes son más activos, se motivan a preguntar, interrumpen cuando no entienden algo y dinamizan la clase. Son mucho más activos, preguntan y preguntan, sugieren cosas nuevas e, incluso, sugieren algún tipo de locuras como solución al problema; a la hora de hacer críticas son mordaces pero muy objetivos. En algunos casos son demasiado activos, desde el principio quieren proponer nuevas formas de resolver el problema, se atreven a opinar, a criticar, a sugerir y a hacer observaciones.

Fuente: Autor.

A continuación, se presenta el algoritmo propuesto como solución al problema matemático y su respectivo código en lenguaje DrScheme, cuyo enunciado es el siguiente: Construir un programa que permita descomponer un número natural en sus factores primos, apoyándose–por el contenido de la asignatura-en programación funcional.

```
;; ================================================================
;; DESCOMPOSICION DE UN NUMERO EN SUS FACTORES PRIMOS
;; Programa que recibe un entero y lo descompone en sus factores primos
;; realizando la multiplicación de comprobación
;; ================================================================

;; Función que determina si un num es múltiplo de otro
;; ------------------------------------------------------------------------------------------------
(define (divisor a b)                    ; definición de función
(if ( = (remainder a b) 0)      ; si a es multiplo de b
  1                  ; retorne valor 1
  0                  ; sino retorne valor 0
))                   ; fin función
```

```
;; Función que cuenta los div exactos de un num
;; en el rango (2,raiz cuad num)
(define (cuentadivisores num div)     ; definición de función
  (if (= div (floor ( / (sqrt num) 1)))      ; si se llegó a la raiz cuad del num
    (divisor num (floor ( / (sqrt num) 1)))          ; verifique si tiene un múltiplo
    ( + (divisor num div)                            ; sino, sume el retorno de divisor
      (cuentadivisores num (+ div 1)))  ; con lo que retorne el llamado recursivo
  ))                                                 ; fin función

;; Función que determina si un num es primo (optimizada)
;; ------------------------------------------------------------------------------------------------------
(define (esprimo n)                                  ; definición de función
(if (= (cuentadivisores n 2) 0)                      ; si tiene 2 divisores exactos
   1                  ; retorne 1
   0                  ; sino, retorne 0
))                    ; fin función

;; Función que carga el vector con los primeros N números
;; primos para un valor N definido por el usuario
;; ------------------------------------------------------------------------------------------------------
(define (cargarvp v posvec tamvec cp num) ; definición de función
  (if (> cp (- tamvec 1))                 ; Si se alcanzó el tamaño del vector
     0                                              ; retorne 0
    (begin                                          ; sino, entonces
     (if (= (esprimo num) 1)              ; si es un num primo
        (begin                                      ; entonces
         (vector-set! v posvec num)       ; Almacene el num primo
         (cargarvp v (+ posvec 1) tamvec (+ cp 1) (+ 1 num))) ; cargue el vector
        (begin                                      ; sino
         (cargarvp v posvec tamvec cp (+ 1 num)))  ; proceda con el sigte valor
        ))))                                        ; fin función

;; Función que muestra el contenido del vector
;; ------------------------------------------------------------------------------------------------------
(define (mostrarvp vec pos tamvec)  ; definición de vector
  (if (> pos (- tamvec 1))                  ; si se alcanzó el tamaño del vector
     (newline)                              ; deje una línea en blanco
   (begin                                           ; sino
     (newline)                              ; deje una nueva linea
     (display (+ pos 1))                            ; muestre un valor
     (display "o primo...")                 ; n-ésimo primo
     (display (vector-ref vec pos))         ; muestre valor almacenado en vector
     (mostrarvp vec (+ 1 pos) tamvec) ; llamado recursivo
     )))                                            ; fin función

;; Función que muestra los operandos de la multiplicación de primos
;; ------------------------------------------------------------------------------------------------------
(define (operandos vec pos tamvec)  ; definición de función
  (if (= pos (- tamvec 1))                  ; si se alcanzó el tamaño del vector
     (display (vector-ref vec pos))         ; muestre valor almacenado en vector
     (begin                                 ; sino
      (display (vector-ref vec pos))        ; muestre valor del vector
      (display " x ")                       ; despliegue signo de multiplicación
      (operandos vec (+ 1 pos) tamvec)))  ; llamado recursivo
```

```
 )                                          ; fin función
;; Función que muestra el resultado de multiplicar los primos
;; ---------------------------------------------------------------------------------------------------
(define (prodvp vec pos tamvec)      ; definición de función
 (if (= pos (- tamvec 1))            ; si se alcanza el tamaño del vector
   (vector-ref vec pos)              ; muestre valor almacenado en vector
   (* (vector-ref vec pos)           ; multiplique valor del vector
     (prodvp vec (+ pos 1) tamvec))) ; con resultado del llamado recursivo
 )                                          ; fin función

;; Función que realiza el proceso de descomposición de un
;; número en sus factores primos
;; ---------------------------------------------------------------------------------------------------
(define (descomp v pos num v2 k)     ; definición del función
 (if (= num 1)                              ; si se llegó al 1o valor
   (begin                                   ; entonces
     (newline)                              ; deje linea en blanco
     (display "1"))                         ; muestre el número 1
   (begin                                   ; sino
     (if (= (remainder num (vector-ref v pos)) 0) ; si se encuentra un múltiplo
       (begin                               ; entonces
         (newline)                          ; deje línea en blanco
         (display num)                      ; muestre argumento num
         (display " | ")                    ; muestre linea divisoria
         (display (vector-ref v pos))       ; muestre valor del vector
         (vector-set! v2 k (vector-ref v pos)) ; almacene valor del vector
         (descomp v pos (quotient num (vector-ref v pos)) v2 (+ k 1))) ; llamado recursivo
       (begin                               ; sino
         (descomp v (+ pos 1) num v2 k)))   ; llamado recursivo
       ))                                   ; fin condicional
 )                                          ; fin función

;; Muestra los factores en formato de multiplicación
;; ---------------------------------------------------------------------------------------------------
(define (muestraf v2 pos)            ; definición de función
 (if (= (vector-ref v2 (+ pos 1)) 0) ; si se llega a la posición 0
   (display (vector-ref v2 pos))     ; muestre un valor del vector
  (begin                                    ; sino
   (display (vector-ref v2 pos))            ; muestre valor del vector
   (display " x ")                          ; muestre signo de multiplicación
   (muestraf v2 (+ pos 1))))                ; llamado recursivo
 )                                          ; fin función

;; Multiplica los factores para comprobar el resultado
;; ---------------------------------------------------------------------------------------------------
(define (prueba v2 pos)                     ; definición función
 (if (= (vector-ref v2 (+ pos 1)) 0)        ; si se llega a la posición 0
   (vector-ref v2 pos)                      ; referencia un valor del vector
  (* (vector-ref v2 pos)                    ; multiplique el valor referenciado
    (prueba v2 (+ pos 1)))                  ; con el resultado del llamado recursivo
  ))                                        ; fin función

;; Función que dimensiona el vector con el tamaño definido
;; por el usuario y activa el proceso de carga
```

```
;; ---------------------------------------------------------------------------------------------------
(define (vp tamvec num)              ; definición de función
 (define v (make-vector tamvec 0))   ; construcción de un vector
 (define v2 (make-vector tamvec 0)) ; construcción de otro vector
 (vector-set! v 0 2)                        ; almacene un valor en un vector
 (vector-set! v 1 3)                        ; almacene un valor en el otro vector
 ;; v vector, 2 posvec, n tamvec, 2 contprim, 4 1er num a eval
 (cargarvp v 2 tamvec 2 4)          ; cargue el vector
 (descomp v 0 num v2 0)             ; descomponga el número recibido
 (newline)                                  ; deje una linea en blanco
 (display "Prueba")                         ; titulo
 (newline)                                  ; linea en blanco
 (display "======")                         ; titulo
 (newline)                                  ; linea en blanco
 (muestraf v2 0)                    ; muestre valor en pantalla
 (display " = ")                            ; muestre signo igual
 (prueba v2 0)                              ; llamado a la función prueba
 )                                          ; fin función

;; Función que inicia el proceso de descomposición de un número
;; en sus factores primos (dnfp)
;; ---------------------------------------------------------------------------------------------------
(define (dnfp n)                            ; definición función
 (if (< n 4)                                ; si el valor es menor que 4
    (display "El numero debe ser positivo mayor que 4") ; avise
    (vp 1000 n))                    ; sino, llame a la función vp
)                                           ; fin función
```

Al ejecutar este programa con dos valores, por ejemplo 20, 120 y 720, el resultado que arroja el programa se muestra en la Tabla 4.

Después de realizar la prueba evaluativa que debe corroborar lo aprendido por los estudiantes con ejercicios y enunciados que se basen en lo visto, se obtienen los resultados que se muestran en la Tabla 5.

Tabla 4
Resultados obtenidos con el programa

Valor de prueba = 20	**Valor de prueba = 120**	**Valor de prueba = 720**
> (dnfp 20)	*> (dnfp 120)*	*> (dnfp 720)*
20 | 2	*120 | 2*	*720 | 2*
10 | 2	*60 | 2*	*360 | 2*
5 | 5	*30 | 2*	*180 | 2*
1	*15 | 3*	*90 | 2*
Prueba	*5 | 5*	*45 | 3*
======	*1*	*15 | 3*
2 x 2 x 5 = 20	*Prueba*	*5 | 5*
>	*======*	*1*
	2 x 2 x 2 x 3 x 5 = 120	*Prueba*
	>	*======*
		2 x 2 x 2 x 2 x 3 x 3 x 5 = 720
		>

Fuente: Autor.

Tabla 5
Resultados cuantitativos

Sem.	Subgrupo 1 Metodología tradicional			Subgrupo 2 Metodología sugerida		
	Nota más baja	Nota más alta	Promedio del grupo	Nota más baja	Nota más alta	Promedio del grupo
I 2014	3.1	3.8	3.46	4.0	4.4	4.31
II 2014	3.2	3.6	3.42	4.2	4.6	4.38
I 2015	3.4	3.7	3.38	4.4	4.8	4.52
II 2015	3.5	3.8	3.62	4.4	4.7	4.61
I 2016	3.2	3.6	3.34	4.6	4.9	4.83
II 2016	3.1	3.4	3.33	4.5	4.9	4.78

Fuente: Autor.

Discusión

La metodología utilizada buscaba que el estudiante conociera lo que se iba a hacer en el marco de la presente investigación, lo que se quería lograr y cómo se quería lograr, con el ánimo de que se involucrara plenamente tanto en los logros como en la forma, de acuerdo con el avance de dicha metodología. Aunque no se socializó públicamente la categorización de estudiantes, los mismos alumnos concluyeron que el instrumento los ubicaba a cada uno en una determinada categoría que, por momentos, podría ser desmotivante si no se maneja apropiadamente. De allí que los resultados del instrumento categorizador deben ser manejados con muchísima confiabilidad y reserva.

Gracias al instrumento utilizado, se logró una muy buena categorización, dado que el desempeño de los alumnos hasta el momento de la aplicación de este instrumento confirmó el perfil de cada uno. De pronto pudieron existir no más de dos casos especiales cuya ubicación en una de las categorías pudiera generar algo de duda, pero todos los demás fueron exitosos. La distribución fue acertada, pues se logró que los dos grupos quedaran balanceados en relación con la cantidad de los alumnos en cada una de las categorías. La interacción fue muy buena tanto entre alumnos como entre ellos y el docente autor de esta investigación.

La opinión de los estudiantes que trabajaron bajo la metodología sugerida siempre fue favorable, pues destacan la metodología como tal. En cuanto a los estudiantes que trabajaron bajo metodología tradicional las opiniones fueron disímiles y estaban más orientadas a destacar las cualidades histriónicas y didácticas del docente que la metodología como tal. La actitud de los estudiantes que trabajaron bajo la metodología sugerida es notoriamente diferente a la de los estudiantes que trabajaron bajo metodología tradicional, en el sentido de que son más proactivos, buscan más las soluciones y están atentos a hacer ajustes y correcciones oportunamente.

El programa se desarrolló teniendo en cuenta las características fundamentales de la programación funcional, y para ello el diseño se basó en la interacción entre funciones y el concepto de recursividad. Asimismo, se acudió a un algoritmo optimizado de búsqueda de números primos que también es producto de las investigaciones del autor de este artículo. Se codificó de la manera más directa en DrScheme y se logró que la solución obedeciera a un algoritmo con altos niveles de eficiencia, tema que se sale de las fronteras de este artículo.

Sabiendo que la prueba evaluativa se basó en lo expuesto durante la exposición magistral (fuera esta con metodología sugerida o sin ella), se encuentra una diferencia cuantitativa muy notoria en la cual los resultados más favorables los obtiene el subgrupo al que se aplicó la metodología. Incluso, si se hace un análisis más detallado, se puede observar que el promedio de los grupos que recibieron la exposición magistral aplicando la metodología sugerida tiene tendencias a aproximarse a la nota más alta, lo cual indica que fueron más las personas que lograron notas altas a diferencia del grupo en el cual no se aplicó esta metodología. La desviación estándar podría ser una medida de tendencia central que facilite este análisis, pero está por fuera de los límites temáticos de este artículo.

Conclusiones

Es posible lograr muy buenos resultados en la aplicación de metodologías de enseñanza y aprendizaje que difieran de las formas tradicionales en la medida en que se socialicen con los estudiantes los alcances, los objetivos a lograr y los propósitos dentro del contexto de las asignaturas, en cualquier investigación que se vaya a realizar. En este tipo de investigaciones conviene categorizar a los estudiantes de acuerdo a modelos que

sean coherentes y apropiados para el perfil mismo de la investigación.

En estos procesos conviene dejar que los datos hablen por sí solos y que cada subgrupo con el cual se quiera experimentar reciba una formación que no lo perjudique, sino que, por el contrario, lo favorezca, sea cual fuere el propósito investigativo. Si el estudiante de programación de computadores percibe la íntima relación que existe entre las matemáticas y la programación, podrá capitalizar más fácilmente los recursos que provee la Matemática para describir formalmente un problema y los recursos que provee la programación para resolverlo desde una óptica tecnológica, todo ello mediado por la lógica como puente de comunicación entre ambas.

Estos procesos investigativos se retroalimentan con la misma opinión de los estudiantes, abriéndoles espacios para que ellos manifiesten libremente sus críticas, sus observaciones, sus sugerencias y, por qué no, sus correcciones. Al fin y al cabo, son procesos en los cuales el protagonista y la razón de ser de estas investigaciones es el mismo alumno. La programación funcional es un buen recurso para aplicar lo que se puede describir formalmente a través de los recursos que la matemática provee, dada su cercanía con la notación matemática y la eficiencia computacional que involucra.

Pareciera ser muy importante que, además de las opiniones cualitativas de los estudiantes en relación con estos procesos, se realicen pruebas evaluativas que posibiliten la verificación de lo aprendido a partir de la metodología utilizada. El docente deberá ser cuidadoso de que la prueba evaluativa tenga una relación íntima con el tema abordado y represente, para el estudiante, la continuación del tema a partir del conocimiento adquirido, generando, por sus propios medios, nuevo conocimiento.

A partir de los resultados obtenidos se puede confirmar que si se aplica la metodología propuesta será muy sencillo para los estudiantes aprender, desde lo puramente matemático, el proceso y el sentido de descomposición de números naturales en factores primos y, desde lo tecnológico a nivel de programación, la implementación de dicha solución en un lenguaje de programación funcional.

Referencias

Acevedo, G. A. & Ambrosio, J. E. (2014). Utilización del LEGO serios Play como herramienta para la solución de problemas. *I+D Revista de Investigaciones, 3*(1), 51–59. https://doi.org/10.33304/revinv.v03n1-2014006

Attard, A., Di Ioio, E. & Geven, K. (2010). *Student Centered Learning. An insight into theory and practice.* Bucarest: Lifelong learning programme - European Community.

Ausubel, D. (1963). *Psychology of Meaningful Verbal Learning: An Introduction to School Learning.* New York: Grune & Stratton.

Ausubel, D. (1986). *Sicología educativa: un punto de vista cognoscitivo.* Ciudad de México: Trillas.

Boyer, C. (2010). *Historia de la Matemática.* Madrid: Alianza Editorial.

Brassard, G. & Bratley, P. (2006). *Fundamentos de algoritmia.* Madrid: Prentice Hall.

Brown Wright, G. (2011). Studen centered learning in Higher Education. *International Journal of Teaching and Learning in Higher Education, 23*(3), 92-97.

Bruner, J. S. (2009). *Actos de significado: más allá de la revolución cognitiva.* Madrid: Alianza Editorial.

Chabert, J. (2005). *A history of algorithms.* Berlin, Germany: Springer.

Crandall, R. & Pomerance, C. (2005). *Prime Numbers. A computational perspective.* New York: Springer.

Crilly, T. (2011). *Grandes cuestiones matemáticas.* Barcelona: Ariel Editorial.

Deitel. & Deitel. (2013). C++ *Programming.* New York: Prentice Hall.

Díaz Barriga, F. (2005). *Estrategias docentes para un aprendizaje significativo.* México: McGraw Hill.

Felleisen, M. e. (2006). *How to design Programs.* Boston: MIT Press.

Fríes, E., Monzón, G. & Di Paolo, J. (2014). Resolución de una situación problemática mediante la utilización de TIC. A. C. ACOFI (Ed.). *Revista Educación en Ingeniería, 9*(17), 45-52.

Gomez Martin, M., Gomez Martin, P. & Gonzalez Calero, P. (2007). El estilo de aprendizaje y la relación con el desempeño académico de los estudiantes. *Revista Iberoamericana de Inteligencia Artificial* (33), 25-36.

Güiza, R. R. (2014). Una mirada entre programación paralela ya tradicional práctica educativa. I+D *Revista de Investigaciones, 4*(2), 23–33. https://doi.org/10.33304/revinv.v04n2-2014003

Jiménez Murillo, J. (2014). *Matemáticas para la computación.* Ciudad de México: Alfaomega.

Jones, L. (2007). *Student Centered Learning.* Cambridge: Cambridge University Press.

Kaasboll, J. (1999). *Exploring didactic models for programming.* Oslo: Universidad de Oslo.

Kline, M. (2012). *El pensamiento matemático de la antigüedad a nuestros días*. Madrid: Alianza Editorial.

Meavilla, V. (2012). *Eso no estaba en mi libro de Matemáticas.* Barcelona: Almuzara.

Polya, G. (1989). *Cómo plantear y resolver problemas.* México D. F.: Editorial Trillas.

Rey Pastor, J. & Babini, J. (2005). *Historia de la Matemática.* Barcelona, España: Gedisa.

Romero Chaves, C. & Rosero Sosa, M. (2014). Modelo

de enseñanza y su relación con los procesos metacognitivos en programación de sistemas. A. C. Acofi (Ed.). *Revista Educación en Ingeniería, 3*.

Rooney, A. (2009). *Historia de las Matemáticas*. Barcelona: La Biblioteca del Saber.

Scott Fogler, H., Le Blanc, S. & Rizzo, B. (2014). *Strategies for creative solving problem*. Boston: Prentice Hall.

Shoup, V. (2008). *A computational introduction to number theory and algebra*. Cambridge: Cambridge University Press.

Trejos Buriticá, O. (2000). *La esencia de la lógica de programación*. Manizales: Centro Editorial Universidad de Caldas.

Trejos Buriticá, O. (2008). *Algoritmos. Problemas básicos*. Pereira, Colombia: Papiro.

Trejos Buriticá, O. (2009). *Fundamentos de programación*. Pereira: Papiro.

Trejos Buriticá, O. (2013). *Significado y competencias*. Pereira: Papiro.

Van Roy, P. (2008). *Techniques and methods in programming computer. Louvain*e: University Press.