



Improving Computational Thinking in Secondary Students with Unplugged Tasks

Mejora del Pensamiento Computacional en Estudiantes de Secundaria con Tareas Unplugged

Hernán Montes-León^a, Raquel Hijón-Neira^b, Diana Pérez-Marín^c, Sergio Raúl Montes-León^d

^a Universidad Rey Juan Carlos, Madrid, España
<https://orcid.org/0000-0001-9309-1252> h.montes.2016@alumnos.urjc.es

^b Computer Science Department, Universidad Rey Juan Carlos, Madrid, España
<https://orcid.org/0000-0003-3833-4228> raquel.hijon@urjc.es

^c Computer Science Department, Universidad Rey Juan Carlos, Madrid, España
<https://orcid.org/0000-0003-3390-0251> diana.perez@urjc.es

^d Universidad Rey Juan Carlos, Madrid, España
<https://orcid.org/0000-0002-2564-4465> s.montesi@alumnos.urjc.es

ARTICLE INFO

Key words:

Computational thinking
Secondary education
Programming fundamentals
Unplugged approach

Palabras clave:

Pensamiento computacional
Educación secundaria
Fundamentos de programación
Enfoque unplugged

ABSTRACT

The teaching-learning of programming fundamentals is increasing in secondary schools, however, it is a very difficult task for the teacher because the student has not yet achieved full development of their computational thinking, so they have difficulty in learning programming fundamentals. In this article, we describe the development of computational thinking activities prior to the teaching of programming fundamentals and we analyze the data obtained from a pre- and post-test of computational thinking applied to a control group and an experimental group. Afterwards, we analyze the results obtained from a test of programming fundamentals to both the control group and the experimental group.

The activity has been evaluated during the academic year 2017-2018 by 80 high school students from K-10 (15 and 16 years old), our results indicate that the development of activities of Computational Thinking previous to the teaching of programming fundamentals has allowed the students to assimilate in a better way the learning in the mentioned subject, that is to say, they extended their mathematical logical reasoning to develop the respective flow charts without any difficulty. That is, they expanded their mathematical logical reasoning to develop the respective flow diagrams without any difficulty.

RESUMEN

La enseñanza-aprendizaje de fundamentos de programación se va incrementando en las escuelas de secundaria, sin embargo, es una tarea muy difícil para el docente debido a que el estudiante aún no ha desarrollado por completo su pensamiento computacional, lo que dificulta el aprendizaje de fundamentos de programación. En este artículo se describen el desarrollo de actividades de pensamiento computacional previo a la enseñanza de fundamentos de programación y se analizan los datos obtenidos de un pre y post Test de pensamiento computacional aplicado a un grupo control y un grupo experimental. Luego se analizan los resultados obtenidos de un test de fundamentos de programación tanto al grupo de control como al experimental.

La actividad ha sido evaluada durante el curso académico 2017-2018 por 80 estudiantes de secundaria de K-10 (15 y 16 años). Nuestros resultados indican que el desarrollo de actividades de pensamiento computacional previo a la enseñanza de fundamentos de programación ha permitido que los estudiantes asimilen de mejor manera el aprendizaje en la mencionada materia, es decir ampliaron su razonamiento lógico-matemático para desarrollar los respectivos diagramas de flujo sin ninguna dificultad.

1. Introducción

Hoy en día la enseñanza de la programación tanto en la educación primaria, secundaria y universidades es muy importante, en especial en las carreras de Ingeniería ligadas a las tecnologías (Blake, 2011). Por tal motivo es muy importante que los estudiantes logren desarrollar habilidad de escribir programas correctamente, eficientes, bien organizados y adecuadamente documentados para ser un buen desarrollador (Compañ-Rosique et al., 2015). El aprendizaje de la programación no es como aprender un procedimiento o fórmula, la cual se aplica en un cálculo diferencial, tampoco es memorizarse una lista de fechas importantes y luego repetirlas, aprender a programar no es solo aprender palabras reservadas de un determinado lenguaje y luego aplicarlo. Aprender a programar es dar solución a un problema, cada problema se soluciona de manera diferente y un programador lo resuelva de diferente forma, es aquí donde se encuentra la dificultad de aprender a programar (Fuentes-Rosado & Moo-Medina, 2017). Estas habilidades se pueden desarrollar con un pensamiento específico, es decir un pensamiento computacional (Zapata-Ros, 2015). El pensamiento computacional y la programación empieza a formar parte del currículo oficial en los sistemas educativos formales (Valverde Berrocoso et al., 2015). Al igual que se inicia con proyectos de capacitación para incorporar el pensamiento computacional en las escuelas (Dapozo et al., 2016)

En un estudio previo (Montes-León et al., 2020) se realizó una experiencia de uso y aplicación para el aprendizaje de habilidades de programación a través de la gamificación, llamada FUNJAVA, desarrollada para los jóvenes estudiantes que disfrutaban el uso de teléfonos inteligentes en el aula. La experiencia se realizó con 144 estudiantes de primero y segundo año de secundaria y cada curso fue dividido al azar en grupos de prueba y control, formando 2 grupos de prueba y 2 grupos de control de primer y segundo año de secundaria. Así pues, del análisis cuantitativo se extrajeron interesantes conclusiones: una medida de la eficacia educativa del uso de la aplicación de juegos serios y una medida de la eficacia de la metodología tradicional, es decir introducción a la programación donde se resuelven unos pocos problemas matemáticos y luego se procede al desarrollo de diagramas de flujo.

En el presente estudio son 80 estudiantes de primer año de bachillerato divididos en dos grupos (paralelo A y B). El grupo de experimental con 40 estudiantes que corresponde al paralelo "A" y el grupo control que corresponde al paralelo "B" con la finalidad de incorporar actividades de pensamiento computacional para la enseñanza-aprendizaje en fundamentos de programación, entre las principales actividades que se realizó tenemos: ejercicios tomados del concurso internacional de Bebras, ejercicios matemáticos, ejercicios de razonamiento lógico matemático tomado de las pruebas "Ser Bachiller 2017", Juegos mentales. Es así que después de un análisis, se logró obtener conclusiones interesantes con respecto a la propuesta de que el pensamiento computacional influye en la mejora del aprendizaje de programación en estudiantes de bachillerato (K-10).

El artículo consta de cinco secciones: en la sección 2 se habla sobre qué es el pensamiento computacional, el pensamiento computacional en la educación y recursos utilizados para desarrollar el pensamiento computacional. En la sección 3 se explica el proceso como se llevó a cabo el presente estudio y sus respectivos resultados. En la sección 4 tenemos la discusión acerca de los resultados. Por último, en la sección 5 se encuentran las conclusiones.

2. Marco Teórico

2.1. Pensamiento computacional

La evolución, distribución y utilización de los dispositivos electrónicos en nuestra sociedad, ha permitido que estos dispositivos se utilicen en diferentes campos, como la medicina, industria, educación, etc. Por lo que, es necesario contar con aplicaciones que puedan ser instaladas en los dispositivos y utilizadas para las actividades específicas. Es por ello que hoy en día se hace muy necesario la enseñanza de la programación desde primaria. Sin embargo, la enseñanza aprendizaje de la programación en secundaria es muy compleja. En (Insuasti, 2016) se menciona que la dificultad para el aprendizaje de la programación se debe a la complejidad de la sintaxis del lenguaje, conceptos de programación, la carga cognitiva implicada en el aprendizaje de programación, el mal diseño de los objetos de saber y la falta de habilidades cognitivas propias para la solución de problemas.

Con la finalidad de eliminar las dificultades en el aprendizaje de programación, varios centros de educación tanto de primaria como de secundaria, han implementado en su currículo, la enseñanza del "pensamiento computacional", (Queiruga et al., 2014) en su artículo "El juego como estrategia didáctica para acercar

la programación a la escuela secundaria” cita a Resnick M. et al, 2009 quien dice que “el pensamiento computacional promueve el pensamiento analítico, sistemático, fomenta la creatividad y el trabajo colaborativo, todas ellas habilidades consideradas fundamentales para la sociedad del siglo 21” .

Wing (2006) indica que “El pensamiento computacional implica la resolución de problemas, el diseño de sistemas y la comprensión del comportamiento humano, basándose en los conceptos fundamentales de la informática.”.

Zapata-Ros (2015) en su artículo “Una nueva alfabetización digital”, explica los componentes del pensamiento computacional, entre los principales tenemos: Pensamiento algorítmico, descomposición, patrones, abstracción, evaluación, además hace referencia a técnicas que se encuentran relacionadas con el pensamiento computacional como son: reflexionar, codificar, diseñar, analizar y aplicar.

2.2. El pensamiento computacional en la enseñanza

Según (Bocconi et al., 2016; González-González, 2019) en varias partes del mundo se está incorporando el pensamiento computacional en los currículos de la enseñanza como un medio para desarrollar las habilidades de resolución de problemas. Las dos principales justificaciones son:

1. El desarrollo de habilidades de Pensamiento Computacional en niños y jóvenes para que puedan pensar de manera diferente, expresarse a través de una variedad de medios, resolver problemas del mundo real y analizar temas cotidianos desde una perspectiva diferente (García-Peñalvo & Mendes, 2018).
2. El fomento del Pensamiento Computacional para impulsar el crecimiento económico, cubrir puestos de trabajo TIC y prepararse para futuros empleos.

Zúñiga et al. (2014) indica que los estudiantes deben desarrollar una gran variedad de habilidades que van más allá de la simple codificación de un programa, pues implica aprender a entender un problema (abstraer, modelar, analizar), plantear soluciones efectivas (reflexionar sobre una abstracción, definir estrategias, seguir un proceso, aplicar una metodología, descomponer en problemas más simples), manejar lenguajes para expresar una solución (codificar, entender y respetar una sintaxis), utilizar herramientas que entiendan esos lenguajes (programar, compilar, ejecutar, depurar), probar que la solución sea válida (entender el concepto de corrección y de prueba), justificar las decisiones tomadas (medir, argumentar), entre otras.

La complejidad, entonces, no está en aprender la rigurosidad de un lenguaje de programación y expresar un algoritmo con ese lenguaje, sino en poder definir un algoritmo que resuelva el problema inicial. Es decir, el desafío está en desarrollar el pensamiento computacional para resolver problemas utilizando un ordenador.

La programación permite materializar la idea de abstracción, uno de los procesos claves del pensamiento computacional, e incluso dentro de esta actividad se demuestra cómo de útil es dominar esta idea. El pensamiento computacional, entonces, también se hace concreto cuando aprendemos a programar.

2.3. Recursos para la enseñanza del pensamiento computacional

Hoy en día existen varios recursos utilizados para la enseñanza del Pensamiento Computacional, entre los principales encontramos actividades que se realizan sin necesidad de utilizar un ordenador (*Unplugged*) (Zapata-Ros, 2019), es decir, estando desconectado, como en el Proyecto TACCLE 3 (García-Peñalvo, 2016) también encontramos actividades mediante el uso de ordenador como juegos y programación visual (Silva et al., 2017).

Actividades Desconectadas

- Proyecto CS Unplugged (Computer Science Education Research Group, 2017): “Este proyecto tiene como objetivo principal el de promocionar la Informática entre los jóvenes como una disciplina interesante, fascinante e intelectualmente estimulante, para ello se basa en una serie actividades de aprendizaje de forma gratuita, estas actividades son a base de juegos, puzzles con la ayuda de cartas, cuerdas, lápices de colores y mucha actividad física”.

Las actividades permiten explorar mapas, problemas de ordenación, problemas de patrones y criptografías, además permite que los estudiantes desarrollen la creatividad y se involucren de forma activa en la solución de problemas.

- Bebras es una iniciativa internacional que tiene como objetivo promover la informática (informática o computación) (ver Figura. 1) y el pensamiento computacional entre los estudiantes de todas las edades (Bebras, 2017).



PENSAMIENTO COMPUTACIONAL

Nombre:.....

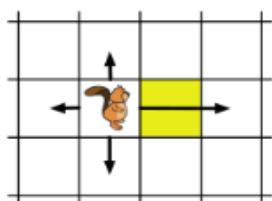
Fecha:.....

TABLERO DE CASTOR

Dificultad: Nivel 8-10: Medio
Nivel 11-13: Fácil

En una tabla parecida a un tablero de ajedrez con campos blancos y amarillos, un castor sirve como un personaje del juego.

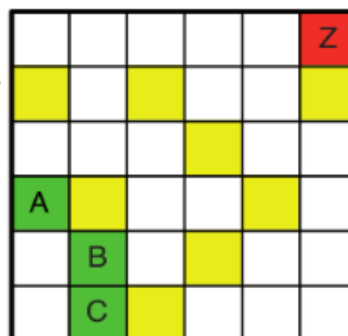
Él puede hacer los siguientes movimientos: (arriba, abajo o al costado) si el campo es amarillo, debe omitirlo; de lo contrario, puede moverse al campo.



Ejemplo: en la situación anterior, son posibles cuatro movimientos.

En el tablero de juego que se muestra a la derecha, el castor debe llegar al campo objetivo (Z).

Él puede elegir desde cuál de los campos A, B o C para iniciar..



¿Cuál de las siguientes afirmaciones es verdadera?

- A) El castor logra Z con el menor número de movimientos si elige A como el campo de inicio.
- B) El castor llega a Z con el menor número de movimientos, si elige C como el campo de inicio.
- C) El castor logra Z con el menor número de movimientos si elige B como campo de inicio.
- D) De los tres campos iniciales, el castor puede alcanzar Z con el mismo número de movimientos.

Figura 1. Ejercicio de Bebras

(tomado de algunos países que han preparado folletos anuales de tareas y soluciones nacionales de Bebras)

Actividades mediante el uso del computador:

- Lightbot (LightBot Inc, 2018): es un juego de rompecabezas basado en la codificación; mientras el niño juega, secretamente le enseña lógica de programación (Secuencia, Sobrecarga, Procedimientos, Bucles recursivos).
- The Foss (codeSpark, 2017): es un juego para los niños más pequeños, el cual introduce conceptos de programación, el juego se lo encuentra de forma gratuita para Android e iOS, aunque también puede accederse desde la web.

- Robot School (Robot School, 2017): es un juego de programación adecuado para niños a partir de 7 años, en el cual aprenderá procedimientos, bucles e instrucciones condicionales, en la resolución creativa de problemas.
- Scratch (Massachusetts Institute of Technology, 2017): permite crear historias interactivas, juegos, y animaciones, así como también compartir las creaciones con otros miembros de la comunidad en línea.
- The Hour of Code (Hour of Code, 2017): la Hora del Código es una introducción de una hora de duración a las Ciencias de la Computación, diseñada para mostrar que todo el mundo puede aprender a programar y así comprender los fundamentos básicos de la disciplina.

Basándonos en los componentes del pensamiento computacional que explica Zapata-Ros (2015), se procedió a seleccionar para nuestro estudio las siguientes actividades:

- Bebras, que es un desafío que promueve habilidades de resolución de problemas y conceptos informáticos, incluida la capacidad de dividir tareas complejas en componentes más simples, diseño de algoritmos, reconocimiento de patrones, generalización de patrones y abstracción (Bebras, 2017).
- Ejercicios de analogías, relación entre figuras, secuencias horizontales gráficas, armar cubos de las pruebas de ingreso a la universidad "SER BACHILLER".
- Aprovechando que en la institución el director general promueve los juegos mentales, se incentivó a los estudiantes de primer año a que participen de estas actividades en los descansos y en el mes de febrero en las horas clases de fundamentos de programación.

3. Experimentación en pensamiento computacional

En el presente estudio se desea incorporar actividades de pensamiento computacional para mejorar la enseñanza-aprendizaje de la asignatura fundamentos de programación de primer año de bachillerato.

Se desea probar dos hipótesis:

- H1: La metodología propuesta desarrolla el pensamiento computacional en estudiantes de bachillerato (K-10).
- H2: La metodología propuesta de pensamiento computacional influye en la mejora del aprendizaje de fundamentos de programación en estudiantes de bachillerato (K-10)

3.1. Participantes

La experiencia se llevó a cabo en el Colegio Hermano Miguel de la ciudad de Latacunga (Ecuador), del 6 de noviembre del 2017 al 8 de abril del 2018, con un total de 80 alumnos de dos cursos de primer año de bachillerato técnico con especialización en Informática divididos en dos grupos (A y B) con el mismo profesor. En el grupo A se encuentran 40 alumnos, de ellos 21 son hombres (52.5%) y 19 son mujeres (47.5%), en el grupo B se encuentran otros 40 alumnos, de los cuales 22 son hombres (55%) y 18 son mujeres (45%).

3.2. Procedimiento

El presente trabajo es un estudio experimental que se aplicó un grupo de estudiantes que se asignaron de forma aleatoria a el grupo experimental y al grupo de control. El grupo experimental con 40 estudiantes que corresponde al grupo "A" y el grupo control también con 40 estudiantes que corresponde al grupo "B". En esta investigación se pretende evaluar, con esta metodología propuesta, la ganancia de pensamiento computacional y la mejora del aprendizaje de fundamentos de programación. El experimento se llevó a cabo durante 5 meses, con 2 sesiones semanales de 90 minutos cada una, un total de 40 sesiones.

La primera semana de 6 al 10 de noviembre el grupo experimental y grupo control completaron el test de pensamiento computacional (Román-González et al., 2015).

Con el grupo experimental se realizaron las actividades como se ven en la Tabla 1 desde el 13 de noviembre del 2017 hasta el 23 de febrero del 2018. Las actividades fueron seleccionadas según los conceptos de pensamiento algorítmico, descomposición, patrones, abstracción y evaluación ya que permiten que los estudiantes tengan creatividad y se involucren de forma activa en la solución de problemas.

Actividades	Descripción
Ejercicios concurso Internacional de Bebras	Castores en el río, castores y bisontes, falso operación grúa, buscando el tesoro, registro de castores, dibujo de estrellas, fabricación de tazones, familia súper power, etc.
Ejercicios del Libro “100 problemas matemáticos” (Berabeu Soria, 2017)	La habitación de mi casa, blancas y negras, escaleras, el gato y el ratón, el banco, cuadrados mágicos I y II.
Ejercicios de las Pruebas de ingreso a la universidad “SER BACHILLER” (ejercicio3.pdf, ejercicio4.pdf) (FORMAS - PRUEBAS SER BACHILLER, 2017)	Ejercicios analogías, relación entre figuras, secuencias horizontales gráficas, armar cubos
Juegos Mentales	Juegos de mesa: Damas chinas, tres en raya, juego del molino, Pitarra, esquinas chinas, alquerque, Zorro y cordero, Ajedrez de mesa y ajedrez gigante, Apps: 4 en raya, laberintos, dominó, damas chinas

Tabla 1. Actividades realizadas con el grupo experimental (K-10).

Con el grupo control se trabajó desde el 13 de noviembre del 2017 hasta el 23 de febrero del 2018 desarrollando los ejercicios del documento digital de Razonamiento: Lógico, Matemático, Inductivo, Deductivo, Abstracto de Orlando Ayala (Ayala, 2017) de la siguiente manera:

Razonamiento matemático (suma, resta, multiplicación, regla de tres simple, inversa y compuesta, tanto por ciento) hasta el 19 de enero del 2018

A continuación, se trabajó con los estudiantes (ver Figura. 2) con juegos mentales hasta el 23 de febrero del 2018, los juegos de mesa que se realizaron fueron: damas chinas, tres en raya, juego del molino, pitarra, esquinas chinas, alquerque, zorro y cordero, ajedrez de mesa y ajedrez gigante, además se trabajó con apps: 4 en raya, laberintos, dominó, damas chinas.



Figura 2. Estudiantes realizando actividades mentales en juegos de mesa en la institución “Hermano Miguel”.

Las semanas del 26 al 2 de marzo se recibió el test de pensamiento computacional tanto al grupo prueba como al grupo control.

Del 5 al 30 de marzo del 2018, con la finalidad de comprobar si estos resultados obtenidos influyen positivamente en el aprendizaje de la asignatura de fundamentos de programación, se procedió a impartir clase a los dos grupos (control y experimental) con la planificación establecida para el presente año, mismo profesor, los principales temas fueron los siguientes:

Conceptos básicos, metodología para la solución de problemas por medio de computadora, tipos de datos, expresiones, operadores y operandos, técnicas para la formulación de algoritmos: diagrama de flujo, pseudocódigo, estructuras algorítmicas secuenciales y ejercicios.

Una vez finalizados estos temas, en la semana de 2 al 6 de abril se procedió a elaborar y pasar un test sobre la materia de fundamentos de programación: sobre conceptos básicos, metodología para la solución de problemas, operadores lógicos, aritméticos y relacionales. Como práctica se les pidió realizar las siguientes tareas: diseño de un diagrama de flujo de estructura secuencial, prueba de escritorio y corrección de errores de una codificación en lenguajes C.

El test se aplicó tanto al grupo control como al grupo experimental, con la finalidad de evaluar si la metodología explicada previamente influyó positivamente en el aprendizaje de la materia de fundamentos de programación en el grupo experimental.

3.3. Resultados

Dado que el objetivo de este estudio era desarrollar el pensamiento computacional en el estudiante de bachillerato(K-10) y evaluar si el empleo de actividades de pensamiento computacional ayuda a mejorar el aprendizaje o no de fundamentos de programación, los análisis de los resultados se orientaron a realizar un análisis descriptivo de los datos presentados y, a continuación, a deducir si había habido o no una mejora significativa entre el grupo control y el grupo experimental de la variable evaluada en primero de bachillerato. El análisis se realizó utilizando el programa RKWard (Proyecto KDE y la comunidad, 2018).

Resultados del pre y post test para la mejorar en pensamiento computacional

En la Figura 3 se muestra un boxplot para el pretest y posttest de pensamiento computacional aplicado tanto al grupo control como al grupo experimental. Se puede observar que los resultados del pre-test no tienen valores similares entre el grupo experimental y control, esto posiblemente se debe a que el momento que se aplicó el test a los dos grupos se les indicó que no tiene una calificación, en el grupo control se pudo observar un desinterés muy grande al contestar el pre test debido a que no tenía ninguna calificación, sin embargo, en el grupo experimental se observó, por alguna razón desconocida, una mayor motivación.

Las cajas están delimitadas por los valores Q1 (primer cuartil) y Q3 (tercer cuartil). Cada cuadro contiene el 50% de los casos correspondientes, destacando la mediana. Los valores mínimo y máximo en los extremos del diagrama corresponden a valores que no son inferiores a $Q1 - 1.5 \cdot (Q3 - Q1)$ y no superiores a $Q3 + 1.5 \cdot (Q3 - Q1)$.

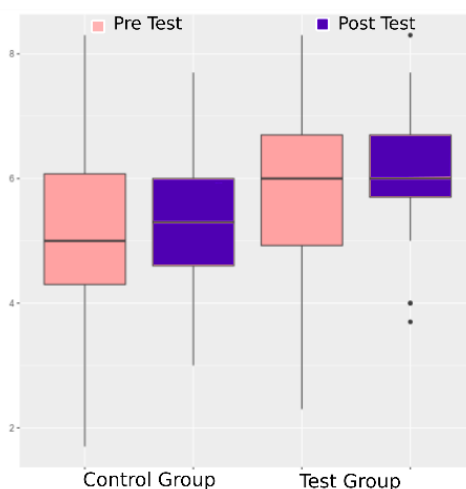


Figura 3. Boxplot para el grupo de control y experimental en las fases pre y post de la prueba de pensamiento computacional.

Puede observarse que los datos del grupo de control mantienen una tendencia central en torno al mismo valor mediano, que ha sufrido una ligera variación en la fase previa y posterior a la prueba. Sin embargo, la dispersión de los datos es muy grande, aunque desaparece en la fase posterior a la prueba en comparación con la fase anterior. Hay una ligera asimetría en ambos casos. Sin embargo, para el grupo experimental, se observa de igual manera una tendencia central en torno al mismo valor mediano, que ha sufrido una gran variación en la fase previa y posterior a la prueba debido a que en la fase de prueba se agrupan la mayoría de datos sobre el valor de la media. Sin embargo, existe una dispersión pequeña de los datos en la fase posterior a la prueba. En la Tabla 2 se muestran los valores de la mediana, la media, la desviación típica y el coeficiente de asimetría de los grupos control y experimentación.

Group	Median	Mean	Standard Deviation	Asymmetry coefficient
Control Pre-test	5	5,15	1,44	-0,13
Control Post-test	5,3	5,33	1,12	0,15
Test Pre-test	6	5,76	1,48	-0,38
Test Post-test	6	6,06	0,96	-0,28

Tabla 2. Resultados pre-test y post-test del grupo control y experimental en el primer año (K-10).

Estos valores corroboran las afirmaciones anteriores referidas a la figura 2, ya que en el grupo de control la media presenta una ligera mejora (de 5,15 a 5,3) que es casi imperceptible en el caso de la mediana. La desviación estándar, ligeramente baja, con respecto a su asimetría podemos decir que la mayoría de las calificaciones están cercanas a la media. Por el contrario, el grupo experimental mostró una evidente mejora (de 5,76 a 6,06 en el caso promedio), con datos ligeramente dispersos, con respecto a la desviación típica se nota una considerable baja; respecto de su asimetría podemos decir que hay un porcentaje promedio de notas por encima de la media.

El análisis descriptivo da una primera idea de la distribución de la muestra. Ahora, ofrecemos un estudio más detallado que compara los resultados de las fases previas y posteriores a la prueba en ambos grupos para deducir si hubo alguna mejora significativa.

En primer lugar, estudiamos la fase previa al test para determinar si había diferencias entre el grupo experimental y el de control. Debido al tamaño relativamente grande de la muestra, utilizamos la prueba T-Student para muestras independientes y llegamos a la conclusión de que hay diferencias significativas entre ellos, con un significado $p < 0,05$. Por lo tanto, podemos asumir que ambos grupos no son homogéneos en la fase de pre-test con respecto a las variables en estudio,

Sin embargo, observamos diferencias significativas entre ambos grupos en la fase posterior a la prueba, con una significación $p < 0,05$.

Esto indica que ha habido una mejora significativa en el grupo experimental; sin embargo, los grupos siguen siendo no homogéneos.

Resultados del test para la mejora de fundamentos de programación

En la Figura 4 se muestra un boxplot para el pretest y el posttest de fundamentos de programación. Las cajas están delimitadas por los valores Q1 (primer cuartil) y Q3 (tercer cuartil). Cada cuadro contiene el 50% de los casos correspondientes, destacando la mediana. Los valores mínimo y máximo en los extremos del diagrama corresponden a valores que no son inferiores a $Q1 - 1.5 \cdot (Q3 - Q1)$ y no superiores a $Q3 + 1.5 \cdot (Q3 - Q1)$.

Se puede observar que los datos del grupo control mantienen una tendencia central en torno al valor mediano, agrupándose la mayoría de datos bajo el valor mediano, además existe una dispersión de datos muy grande. Sin embargo, para el grupo experimental se puede observar el valor mediano es superior al grupo control, los datos mantienen una tendencia central entorno al valor mediano. En la tabla 3 se muestran los valores de la mediana, la media, la desviación típica y el coeficiente de asimetría de los grupos control y experimentación de muestra.

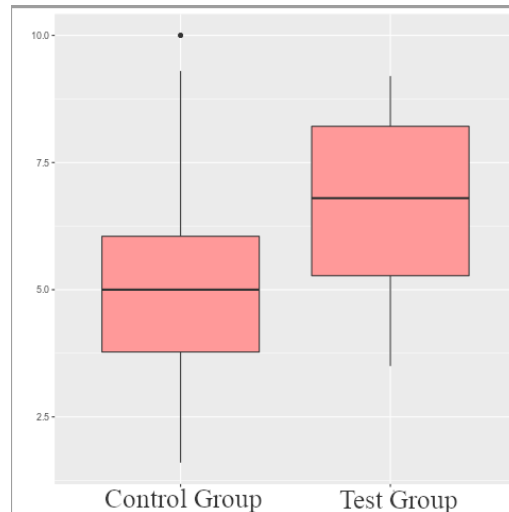


Figura 4. Boxplot para el grupo de control y experimental del post test de fundamentos de programación.

Group	Median	Mean	Standard Deviation	Asymmetry coefficient
Control	5	5,17	2,19	0,68
Test	6,8	6,74	1,69	-0,27

Tabla 3. Resultados de la prueba de fundamentos de programación en el grupo de Control y Prueba en el primer año (K-10).

El análisis descriptivo da una primera idea de la distribución de la muestra. Ahora, ofrecemos un estudio más detallado que compara los resultados de la prueba en ambos grupos para deducir si hubo alguna mejora significativa.

En primer lugar, estudiamos la fase previa al test para determinar si había diferencias entre el grupo experimental y el de control. Debido al tamaño relativamente grande de la muestra, utilizamos la prueba T-Student para muestras independientes y llegamos a la conclusión de que hay diferencias significativas entre ellos, con un significado $p < 0,05$ siendo $p = 0,0007$. Esto indica que ha habido una mejora significativa con el grupo experimental; sin embargo, siguen siendo no homogéneos.

Como puede observarse, se puede evidenciar que el grupo experimental ha obtenido un mejor aprendizaje con respecto al grupo control, en la materia de Fundamentos de Programación, previo a realizar las actividades de pensamiento computacional detalladas anteriormente en el apartado procedimiento. Es decir, en el grupo experimental en base a un enunciado (problema) que se les indique a los estudiantes, estos pueden realizar el diagrama de flujo, prueba de escritorio y su respectiva codificación sin ningún inconveniente, encontrando mayores dificultades, en general, en el grupo control.

4. Discusión

En Ecuador en las instituciones educativas de primaria y secundaria aún no se ha incorporado un plan académico para el aprendizaje de ciencias de la computación con las habilidades de pensamiento computacional (Ministerio de Educación, 2020) como lo tiene Australia, Inglaterra, Estonia, Finlandia, Nueva Zelanda, Noruega, Suecia, Corea del Sur, Polonia y los EE.UU., entre los principales países que han incorporado las ciencias de la informática en el plan de estudios oficial (Heintz et al., 2016).

El no realizar actividades que promuevan el pensamiento computacional en los estudiantes, conlleva a que tendrán mayores dificultades al resolver problemas de razonamiento lógico- matemáticos. Así se evidencia en el presente estudio, en el que existió una marcada diferencia entre grupo control y experimental, así, en el momento que se entregaba un problema y se solicitaba que desarrollaran el diagrama de flujo y su respectiva codificación en lenguaje C, el grupo control encontraba dificultades en entender el problema, además realizaba una lectura

superficial y pasaba a elaborar el diagrama de flujo y su respectiva codificación. Sin embargo, en el grupo experimental se realizaba una lectura cuidadosa del problema e iba sacando datos para realizar primero un análisis y después realizaba el diagrama de flujo, posteriormente realizaban la prueba de escritorio para ver si estaba correctamente realizado el diagrama, una vez realizado este proceso pasaban a realizar la respectiva codificación. Es en la resolución de estos pasos estructurados donde se puede evidenciar que la metodología aplicada del pensamiento computacional ayudó a los estudiantes del grupo experimental a que desarrollaran la capacidad de analizar y resolver problemas sin ningún inconveniente y los apliquen en el desarrollo del diagrama de flujo, prueba de escritorio y la codificación en lenguaje C, lo que no sucedió con el grupo control. Este comportamiento que hemos visto en este estudio es similar a la que concluye Ying Li (2016) "El pensamiento computacional tuvo un profundo efecto en la ciencia de la computación. La enseñanza basada en el Pensamiento Computacional era diferente de la enseñanza ordinaria, que prestaba más atención a explicar el proceso de resolución de problemas y la formación del pensamiento".

Además, en el aprendizaje de fundamentos de programación, podemos comprobar que la media del grupo control es de 5/10; en cambio, la del grupo experimental es de 6,8/10, siendo una mejora considerable con respecto al grupo control. Los resultados obtenidos muestran que las tareas de pensamiento computacional aplicadas en esta experiencia incrementaron los resultados esperados, al igual que lo demostró R. J. Haddad & Y. Kalaani (2015) en su estudio en donde concluyen "que las evaluaciones de las habilidades de Pensamiento Computacional de los estudiantes están fuertemente correlacionadas con su futuro rendimiento académico y por lo tanto pueden servir como un predictor bastante preciso"; Sin embargo, se observa que la media es inferior a 7/10 que es una nota aceptable en la institución.

Por lo tanto, se va a estudiar cómo introducir otras actividades adicionales que permitan que mejore su pensamiento computacional y a su vez su media, estas podrían ser: la utilización de Sistemas Inteligentes de Tutoría como tenemos CPP-TUTOR (Ademuwagun, 2013), herramientas de aprendizaje visual como JkAREL ROBOT (Pattis, 2000), Scratch (Massachusetts Institute of Technology, 2017), etc.

Además, se podría utilizar juegos serios como FUNJAVA, el cual dio muy buenos resultados para el aprendizaje de fundamentos de programación en un estudio previo (Montes-León et al., 2020).

Adicionalmente, se pretende desarrollar un juego serio con los contenidos del curso de fundamentos de programación de primer año de bachillerato, y evaluar si es posible mejorar la calidad del aprendizaje y la motivación de los estudiantes en este curso gracias a su uso y, por ende, la mejora en su pensamiento computacional.

5. Conclusiones

Este artículo presenta un análisis sobre la aplicación de actividades de pensamiento computacional para mejorar el aprendizaje de la materia de Fundamentos de Programación en los estudiantes de primer curso de secundaria en la especialidad Informática. Se realizó una evaluación cuantitativa entre 80 estudiantes K-10 (15 y 16 años) en los que se observa que al aplicar actividades de pensamiento computacional se influye positivamente en su aprendizaje de la materia de Fundamentos de Programación en los estudiantes

Ambas hipótesis planteadas en el estudio se validaron:

- H1: La metodología propuesta de pensamiento computacional mejora el pensamiento computacional en estudiantes de bachillerato (K-10).
- H2: La metodología propuesta de pensamiento computacional influye en la mejora del aprendizaje de programación en estudiantes de bachillerato (K-10).


Por lo que puede decirse que, al aplicar actividades de pensamiento computacional como recurso en el aula, ayuda a los estudiantes a mejorar el pensamiento computacional en bachillerato, estas actividades a su vez influyen positivamente en aumentar sus conocimientos de fundamentos de la programación, como se ha observado experimentalmente. Por lo tanto, recomendamos como actividades previas a la enseñanza de la materia de Fundamentos de Programación que se realicen actividades de pensamiento computacional ya que mejora su pensamiento lógico-matemático en el proceso de enseñanza y aprendizaje.

Agradecimientos

Esta investigación se ha realizado dentro del Proyecto del Ministerio y Competitividad TIN2015-66731-C2-1-R y del proyecto de la Comunidad Autónoma de Madrid S2018/TCS-4307.

Referencias

- Ademuwagun, O. (2013, marzo). *GitHub—Oademuwagun/cpp-tutor: C++ equivalent of the Online Python Tutor*. <https://bit.ly/2ZtVmA5>
- Ayala, O. (2017). *Razonamiento: Lógico, Matemático, Inductivo, Deductivo, Abstracto*. <https://bit.ly/3me8d2Y>
- bebras.org. (2017, October). *What is Bebras* | www.bebas.org. <https://www.bebas.org/>
- Berabeu Soria, G. (2017). (PDF) *RECURSOS PARA EL AULA 100 problemas matemáticos*. <https://bit.ly/2Rv7uMR>
- Blake, J. D. (2011). Language considerations in the first-year CS curriculum. *Journal of Computer Science*, 26(6), 124–129–124–129.
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Developing computational thinking in compulsory education. *European Commission, JRC Science for Policy Report*. doi:<https://doi.org/10.2791/792158>
- codeSpark. (2017, October). *Coding App for Kids* | [codeSpark Academy](http://codeSparkAcademy.com). <https://codespark.com/>
- Compañ-Rosique, P., Satorre-Cuerda, R., Llorens-Largo, F., & Molina-Carmona, R. (2015). Enseñando a programar: Un camino directo para desarrollar el pensamiento computacional. *Revista de Educación a Distancia*, 46. doi:<https://doi.org/10.6018/red/46/11>
- Computer Science Education Research Group. (2017, October). *CS Unplugged*. *Informática Sin Un Ordenador*. <https://csunplugged.org/>
- Dapozo, G. N., Petris, R. H., Greiner, C. L., Espíndola, M. C., & López, M. (2016). Capacitación en programación para incorporar el pensamiento computacional en las escuelas. *Actas del XI Congreso de Tecnología en Educación y Educación en Tecnología (TE&ET 2016)* (pp. 194-203). Red de Universidades con Carreras en Informática (RedUNCI).
- Formas - pruebas ser bachiller. (2017). *Razonamiento abstracto*. <https://bit.ly/2Zufjqc>
- Fuentes-Rosado, J. I., & Moo-Medina, M. (2017). Dificultades de aprender a programar. *Revista Educación En Ingeniería*, 12(24), 76–82–76–82. doi:<https://doi.org/10.26507/rei.v12n24.728>
- García-Peñalvo, F. J. (2016). A brief introduction to TACCLE 3 – Coding European Project. In F. J. García-Peñalvo & J. A. Mendes (Eds.), *2016 International Symposium on Computers in Education (SIIE 16)*. USA: IEEE. doi:<https://doi.org/10.1109/SIIE.2016.7751876>
- García-Peñalvo, F. J., & Mendes, J. A. (2018). Exploring the computational thinking effects in pre-university education. *Computers in Human Behavior*, 80, 407-411. doi:<https://doi.org/10.1016/j.chb.2017.12.005>
- González-González, C. S. (2019). State of the art in the teaching of computational thinking and programming in childhood education. *Education in the Knowledge Society*, 20, 17. doi:https://doi.org/10.14201/eks2019_20_a17
- Haddad, R. J., & Kalaani, Y. (2015). Can computational thinking predict academic performance? *2015 IEEE Integrated STEM Education Conference* (pp. 225–229). doi:<https://doi.org/10.1109/ISECon.2015.7119929>
- Heintz, F., Mannila, L., & Farnqvist, T. (2016). A Review of Models for Introducing Computational Thinking, Computer Science and Computing in K–12 Education. In *Proceedings of the 2016 IEEE Frontiers in Education Conference (FIE) (Erie, PA, USA, USA, 12-15 Oct. 2016)*. USA: IEEE. doi:<https://doi.org/10.1109/FIE.2016.7757410>
- Hour of Code. (2017, October). *Hour of Code: Join the Movement*. Code.Org. <https://hourofcode.com/>
- Insuasti, J. (2016). Problemas de enseñanza y aprendizaje de los fundamentos de programación. *Revista Educación y Desarrollo Social*, 10(2), 234–246. doi: <https://doi.org/10.18359/reds.1966>
- LightBot Inc. (2018, February). *LightBot*. <https://lightbot.com/>
- Massachusetts Institute of Technology. (2017, October). *Scratch—Imagine, Program, Share*. <https://scratch.mit.edu/>
- Ministerio de Educación. (2020). *Malla Curricular educación general básica—Ministerio de Educación*. <https://bit.ly/2FozzTg>
- Montes-León, H., Hijón-Neira, R., Pérez-Marín, D., & Montes León, S. R. (2020). Improving Programming Learning on High School Students through Educative Apps. *Proceedings of the 19 International Symposium on*

- Computers in Education (SIIE) (Tomar, Portugal, Portugal, 21-23 Nov. 2019)*. USA: IEEE. doi:<https://doi.org/10.1109/SIIE48397.2019.8970112>
- Pattis, R. E. (2000, diciembre). *JKarel*. <https://bit.ly/35tvKXy>
- Proyecto KDE y la comunidad. (2018, February). *RKward*. <https://rkward.kde.org/>
- Queiruga, C. A., Fava, L. A., Gómez, N. S., Kimura, I., & Brown Bartneche, M. (2014). El juego como estrategia didáctica para acercar la programación a la escuela secundaria. *Actas del XVI Workshop de Investigadores en Ciencias de la Computación* (pp. 358-362). Red de Universidades con Carreras en Informática (RedUNCI).
- Robot School. (2017).  Robot School. Programming For Kids. *App Store*.
- Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2015). Test de Pensamiento Computacional: Diseño y psicometría general. In Á. Fidalgo Blanco, M. L. Sein-Echaluze Lacleta, & F. J. García-Peñalvo (Eds.), *La Sociedad del Aprendizaje. Actas del III Congreso Internacional sobre Aprendizaje, Innovación y Competitividad. CINAIC 2015 (14-16 de Octubre de 2015, Madrid, España)* (pp. 353-358). Madrid, Spain: Fundación General de la Universidad Politécnica de Madrid.
- Silva, V., da Silva, L. L., & França, R. (2017). Pensamento computacional na formação de professores: Experiências e desafios encontrados no ensino da computação em escolas públicas. *Anais do XXIII Workshop de Informática na Escola (WIE 2017)*. (805-814). doi:<https://doi.org/10.5753/cbie.wie.2017.805>
- Valverde Berrocoso, J., Fernández Sánchez, M. R., & Garrido Arroyo, M. C. (2015). El pensamiento computacional y las nuevas ecologías del aprendizaje. *Revista de Educación a Distancia*, 46. doi:<https://doi.org/10.6018/red/46/3>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35–33–35. doi:<https://doi.org/10.1145/1118178.1118215>
- Ying Li. (2016). Teaching programming based on Computational Thinking. In *Proceedings of the 2016 IEEE Frontiers in Education Conference (FIE) (Erie, PA, USA, USA, 12-15 Oct. 2016)*. USA: IEEE. doi:
- Zapata-Ros, M. (2015). Pensamiento computacional: Una nueva alfabetización digital. *Revista de Educación a Distancia*, 46. doi:<https://doi.org/10.6018/red/46/4>
- Zapata-Ros, M. (2019). Computational Thinking Unplugged. *Education in the Knowledge Society*, 20, 18. doi:https://doi.org/10.14201/eks2019_20_a18
- Zúñiga, M. E., Rosas, M. V., Fernández, J., & Guerrero, R. A. (2014). El desarrollo del pensamiento computacional para la resolución de problemas en la enseñanza inicial de la programación. *Actas del XVI Workshop de Investigadores en Ciencias de la Computación* (pp. 340-343). Red de Universidades con Carreras en Informática (RedUNCI).