Evaluación de habilidades del pensamiento computacional para predecir el aprendizaje y retención de estudiantes en la asignatura de programación de computadoras en educación superior

Assessment of computational thinking skills to predict student learning and retention in the subject programming computer in higher education

Arturo Rojas-López
Universidad Tecnológica de Puebla. Puebla, México
arturo.rojas@utpuebla.edu.mx

Francisco José García-Peñalvo
Universidad de Salamanca. Salamanca, España
fgarcía@usal.es

RESUMEN
El objetivo del artículo es presentar el conocimiento generado al realizar una evaluación de habilidades específicas del pensamiento computacional del 2016 al 2019. Lo anterior para determinar el estado inicial de los estudiantes de nuevo ingreso de la carrera de tecnologías de la información en la Universidad Tecnológica de Puebla – México y favorecer el desempeño académico en el curso Metodología de la programación en educación superior. Por medio de la selección de cinco reactivos en correspondencia a cinco habilidades del pensamiento computacional (abstracción, generalización, descomposición, diseño algorítmico y evaluación) se establece una relación con los contenidos temáticos del curso, por lo anterior, se ha podido determinar cuales serían las fortalezas y debilidades de los estudiantes con el objetivo de apoyar su desempeño académico. Dos resultados destacan del trabajo realizado. Primero, el impacto favorable en los estudiantes para la adquisición de las competencias a través de encuestas solicitadas durante cada intervención. Segundo, a partir de la tercera intervención, obtener una predicción de la retención de matrícula al menos para el primer cuatrimestre. Tomando como base el diseño experimental realizado en México, también se comenta la propuesta de diseño para la Universidad Tecnológica de Chile.

Palabras clave: Pensamiento computacional, educación superior, enseñanza-aprendizaje programación.

ABSTRACT
The objective of the article is to present the knowledge generated by carrying out an evaluation of specific computational thinking skills from 2016 to 2019. The aforementioned to determine the initial status of new students to the information technology career at the Universidad Tecnológica de Puebla - Mexico and promote academic performance in the Programming Methodology course in higher education. Through the selection of five items corresponding to five computational thinking skills (abstraction, generalization, decomposition, algorithmic design and evaluation) a relationship is established with the thematic contents of the course; therefore, it has been possible to determine what the strengths and weaknesses of students in order to support their academic performance. Two results stand out from the work done. First, the favorable impact on students for the acquisition of skills through surveys requested during each intervention. Second, from the third intervention, obtain a prediction of enrollment retention for at least the first four-month period. Based on the experimental design carried out in Mexico, the design proposal for the Technological University of Chile is also discussed.

Keywords: Computational thinking, higher education, teaching-learning programming.

INTRODUCCIÓN

Problemática en el aprendizaje de la programación de computadoras

En similitud a otras universidades, varios son los factores que enfrentan los estudiantes para mantener una vida académica universitaria. Echegaray et al. (2017) listan los siguientes: "desmotivación de los estudiantes, confusiones con respecto a la elección del grado, falta de información acerca de la vida universitaria o los planes y contenidos de los grados, confusión con el propio diseño de la estructura universitaria, deficiencias en la formación académica previa, sentimientos de inseguridad acerca de las propias capacidades, etc.". Particularmente, el aprendizaje-enseñanza de la programación de computadoras no es una actividad sencilla para los estudiantes-docentes; Selby (2015) aborda el problema del aprendizaje, destacando que algunas de las razones que ha identificado son: "inadecuada comprensión de cómo trabaja un modelo computacional, una incapacidad para dominar la lectura, rastreo y escritura de código, así como comprender conceptos de alto nivel tales como el diseño". Milne y Rowe (2002) presentaron los resultados de un análisis estadístico a una encuesta realizada en línea a 66 participantes, entre estudiantes y profesores de programación de computadoras alrededor del Reino Unido, indicando que la incapacidad para comprender qué le está ocurriendo al programa en la memoria de la computadora dificulta el entendimiento de conceptos avanzados tales como apuntadores o los relacionados a la gestión de la memoria, en consecuencia, sugieren que a través de la aclaración de tal proceso computacional, los estudiantes sean capaces de crear un claro modelo mental de la ejecución de instrucciones. Sakhnini y Hazzan (2008) presentaron conclusiones en beneficio del proceso enseñanza-aprendizaje de los tipos de datos abstractos. Primero, los profesores deben ser conscientes que los estudiantes tienden a tratar con problemas desconocidos, confiando en los términos e ideas que le son familiares desde su vida diaria o desde sus estudios previos, por lo tanto, las diferencias entre los nuevos problemas y los familiares deben ser explícitamente discutidas con los estudiantes, y deben ser expuestos a un amplio número de problemas de diferentes tipos y niveles con el fin de mejorar sus habilidades para tratar con problemas no familiares. Segundo, con el fin de guiar a los estudiantes en la concepción de los tipos de datos abstractos como objetos, recomiendan separar la definición y uso desde su implementación, es decir, iniciar enseñando el concepto y posteriormente su implementación. Y tercero, los estudiantes deben ser expuestos a muchos métodos, estrategias y herramientas de resolución de problemas, tales como, el método divide y vencerás o las recomendaciones de Pólya (1957) relacionadas al proceso de resolución de problemas, es decir, primero, comprender el problema, segundo, encontrar la conexión entre los datos y lo desconocido, se pueden considerar problemas auxiliares, si una conexión inmediata no puede ser encontrada, se debe eventualmente obtener un plan de la solución; tercero, llevar a cabo el plan, y cuarto, examinar la solución obtenida.

La frustración que puede causar en los estudiantes de programación de computadoras durante los primeros meses de estudio el no poder crear soluciones básicas de software a problemas de la industria o mercado laboral, puede ser el factor principal para que tome la decisión de abandonar los estudios. La demanda de programadores es aún requerida en el sistema

económico de cualquier país y su diversidad de industrias: automotriz, dispositivos embebidos, academia, tecnologías de negocios, publicitaria y de pagos (Mazaika, 2017). La Oficina de Estadísticas Laborales de Estados Unidos (Bureau_of_Labor_Statistics, 2019) pronostica un incremento del 24% del 2016 al 2026; por lo que la pertinencia de competencias adquiridas y su funcionalidad en el ejercicio profesional refleja el éxito de un modelo educativo universitario en el perfil de las tecnologías de la información. Además, hay que considerar el proceso de automatización de empleos que hará necesaria la actividad profesional de analistas de datos, expertos en inteligencia artificial y aprendizaje de máquinas (García-Vega, 2019).

El pensamiento computacional (Selby, 2015; Wing, 2006) es un proceso cognitivo que permite la generación de soluciones a problemas a través del uso de habilidades específicas, tales como abstracción, descomposición, generalización, evaluación y diseño algorítmico. Actualmente existe un gran interés por enseñar el pensamiento computacional a una edad temprana (García-Peñalvo & Mendes, 2018; Chiazzese, Fulantelli, Pipitone, & Taibi, 2018; Chiazzese, Fulantelli, Pipitone, & Taibi, 2017; García-Peñalvo, Reimann, Tuul, Rees, & Jormanainen, 2016), debido a que la resolución de problemas es una competencia útil para las personas independientemente de su perfil profesional; la aplicación del pensamiento computacional en niños sirve también para detectar talento afín a las áreas de ciencia, tecnología, ingeniería y matemáticas (Science, Technology, Engineering, and Matemathics - STEM), además, se considera un elemento clave para el éxito de estudiantes en las ciencias de la computación. Por todo lo anterior, es útil conocer el nivel cognitivo del estudiante en pensamiento computacional a través de una evaluación que permita obtener no solo qué habilidades posee en la resolución de problemas, sino además contribuir en la creación del ambiente de aprendizaje creado por el docente para favorecer el proceso enseñanza – aprendizaje en la asignatura inicial de programación de computadoras.

Objetivo y pregunta de la investigación

Una vez descrita la problemática, el objetivo del presente trabajo es presentar el conocimiento generado con los resultados que se han obtenido al realizar una evaluación de habilidades específicas del pensamiento computacional para determinar el estado inicial de los estudiantes de nuevo ingreso de la carrera de tecnologías de la información en la Universidad Tecnológica de Puebla – México y favorecer el desempeño académico en el curso Metodología de la programación, además de predecir el porcentaje de retención. La pregunta que motivó la investigación realizada fue ¿Qué resultados se pueden obtener de la evaluación de habilidades de pensamiento computacional que favorecen el aprendizaje de las competencias del curso inicial metodología de la programación en el contexto de la investigación?

El artículo está organizado de la siguiente forma con la intención de favorecer la divulgación de los resultados. La sección definición del pensamiento computacional incluye la revisión de algunas publicaciones que han propuesto la definición del concepto. La sección titulada estado del arte, contiene una colección de publicaciones específicas que han trabajado el uso del pensamiento computacional para el aprendizaje de programación de computadoras en la educación superior. En consecuencia, la sección diseño experimental presenta la propuesta

de evaluación del pensamiento computacional y su uso para determinar un diagnóstico a los estudiantes que iniciarán el curso metodología de la programación. La sección resultados, contiene el detalle de los datos obtenidos de la evaluación del pensamiento computacional y su interpretación para el entorno de enseñanza-aprendizaje, así como los resultados de competencias adquiridas al finalizar el curso en cuestión. La sección discusión, comenta las observaciones de la información obtenida en comparación con lo reportado por otros autores para reafirmar el valor agregado del diseño experimental y los resultados obtenidos. Finalmente, la sección de conclusiones expone principalmente la relevancia del trabajo realizado en dos vertientes, el impacto en beneficio de los estudiantes, es decir, el cumplimiento del objetivo y respuesta a la pregunta de investigación, y el trabajo a futuro para las siguientes generaciones.

DEFINICIÓN DE PENSAMIENTO COMPUTACIONAL

El término pensamiento computacional fue hecho popular por Jeannette M. Wing, en su definición "involucra la solución de problemas, diseño de sistemas, y comprensión de la conducta humana, basándose en los conceptos fundamentales de las ciencias de la computación" (Wing, 2006). Wing revisó el concepto y proporcionó una nueva definición: "el pensamiento computacional es el proceso de pensamiento involucrado en la formulación de problemas y sus soluciones para que las soluciones estén representadas en una forma que pueda ser efectivamente llevada a cabo por un agente de procesamiento de información" (Wing, 2011). García-Peñalvo lo define como "la aplicación de un alto nivel de abstracción y un enfoque algorítmico para resolver cualquier tipo de problema" (García-Peñalvo, 2016). Google for Education (2016) lo establece como un conjunto de habilidades y técnicas para "el proceso de solución de un problema, tales como, ordenamiento lógico, análisis de datos y creación de soluciones usando una serie de pasos ordenados que en consecuencia, es esencial para el desarrollo de aplicaciones computacionales". En el documento elaborado por García-Peñalvo et al. (2016) se presentan definiciones que ejemplifican la falta de consenso para establecer una definición formal del término pensamiento computacional, pero a pesar de eso, destacan las siguientes dos:

"El pensamiento computacional es el proceso de reconocer aspectos de computación en el mundo que nos rodea, y aplicar herramientas y técnicas desde la ciencia de la computación para comprender y razonar acerca de sistemas y procesos artificiales y naturales" (Royal Society, 2012).

"El pensamiento computacional es un proceso de resolución de problemas que incluye (pero no esta limitado a) las siguientes características: formulación de problemas de una forma que nos permita usar una computadora y otras herramientas para ayudar a resolverlos; organización lógica y análisis de datos; representación de datos a través de abstracciones tales como modelos y simulaciones; soluciones automáticas a través de pensamiento algorítmico (una serie de pasos ordenados); identificar, analizar e implementar posibles soluciones con la meta de lograr la combinación más eficiente y efectiva de pasos y recursos; generalizar y transferir este proceso de solución a una amplia variedad de problemas" (Barr & Stephenson, 2011) .

Xia (2016) comenta la definición propuesta por Zhou Yizhen del 2006: es el uso de los conceptos fundamentales de ciencias de la computación para la resolución de problemas, comprensión de la conducta humana además de actividades de pensamiento en el diseño de sistemas.

Zapata-Ros (2015) elaboró un estudio con la intención de contextualizar la definición del pensamiento computacional y la relación de habilidades asociadas, desde un análisis y una elaboración interdisciplinar en las teorías del aprendizaje. Como resultado, establece los siguientes componentes:

- Análisis ascendente. En ocasiones, la creación de un algoritmo implica un proceso de diseño de submétodos funcionales que permiten determinar un método general de resolución.
- Análisis descendente. La resolución de un problema complejo se puede llevar a cabo a partir de resolver primero los problemas más concretos para pasar después a resolver los más abstractos.
- Heurística. Conjunto de procedimientos prácticos o informales que se obtienen de la observación, el análisis y el registro para resolver problemas.
- Pensamiento divergente. Pensamiento ajeno a un esquema rígido de pensar y de formular ideas en el aprendizaje, con el objetivo de obtener ideas creativas e innovadoras. Produce soluciones no convencionales e implica fluidez y capacidad para generar una gran cantidad de visiones e ideas sobre el problema que se trabaja, para cambiar de unas a otras, y para establecer asociaciones inusuales.
- Creatividad. Amplia variedad de comportamientos, hábitos e ideas que permiten converger el pensamiento convergente (estructurar los conocimientos de una forma lógica y para aplicar sus leyes) y divergente para producir una novedad en un campo reconocible por una sociedad.
- Resolución de problemas. El pensamiento computacional es una variante del dominio metodológico que se conoce como resolución de problemas.
- Pensamiento abstracto. Es la capacidad de operar con modelos ideales abstractos de la realidad, abstrayendo las propiedades de los objetos que son relevantes para un estudio.
- Recursividad. Metodología que permite resolver un problema con la característica de poder resolverse a partir de una remisión a otro problema de las mismas características o naturaleza, pero más pequeño que sí puede ser resuelto.
- Iteración. Componente importante del pensamiento computacional que permite crear procedimientos repetitivos, con una extensa proyección en otras representaciones cognitivas y en procedimientos complejos que son la base de importantes actividades y tareas en la resolución de problemas.
- Métodos por aproximaciones sucesivas. Ensayo – error. Método de resolución de problemas en donde se confrontan las ideas formadas de la realidad, a través de los sentidos, la experimentación y la representación de las ideas obtenidas de las experiencias, para aceptar o rechazar el conocimiento que la realidad ofrece e inducirlo.

- Métodos colaborativos. De una aproximación inicial de entender el trabajo colaborativo como el que se produce en una situación en la que dos o mas personas aprenden algo juntos, se nutre el concepto para llevarlo a superar el aprendizaje para trabajar juntos hacia encontrar una cultura en común, unas referencias y unas experiencias que hagan que esa forma de trabajar fluya.
- Patrones. Desde el punto de vista del análisis de la programación, los patrones tienen la facultad útil de evitar el trabajo tedioso que supone repetir procedimientos que en esencia se repiten, pero aplicados a contextos y situaciones distintas; lo que exige la capacidad de distinguir lo que tienen de común situaciones distintas.
- Sinéctica. Estudio enfocado en organizar la integración de diferentes individuos que componen un grupo para la resolución de problemas. Es un punto de confluencia de las teorías que tratan de explicar y estudian la creatividad, de las técnicas de trabajo en grupo como medio para exteriorizar flujos e impulsos que de otra forma no serían observables y por tanto analizados, mejorados y compartidos.
- Metacognición. Condición necesaria en combinar la situación y los recursos cognitivos propios de las personas para que se lleve a cabo un aprendizaje; se trata de un plan de actuación que implica habilidades y destrezas (que la persona ha de poseer previamente) y de una serie de técnicas que se aplican en función de las tareas a desarrollar para lograr el aprendizaje, sobre las que la persona decide y sobre las que tiene una intención de utilizar de forma consciente. En relación al pensamiento computacional adquiere una importancia en la tarea de cómo afrontar un problema y cómo resolverlo.

La definición que sirve de apoyo al trabajo de investigación realizado es la propuesta por Selby (2015) que incluye las habilidades de abstracción, descomposición, diseño algorítmico, generalización y evaluación. El estudio detona un elemento muy importante debido a que explora la relación entre el pensamiento computacional, enseñar programación y la taxonomía de Bloom. Inicialmente desarrolla una definición de las cinco habilidades.

- Generalización: la habilidad para expresar la solución de un problema en términos genéricos, la cual pueda ser aplicada a diferentes problemas que comparten algunas de las mismas características como el problema original.
- Descomposición: fraccionar a piezas más pequeñas, fáciles de resolver, partes de un problema.
- Abstracción: habilidad para decidir qué detalles de un problema son importantes y qué detalles se pueden omitir.
- Diseño Algorítmico: habilidad para crear un conjunto de instrucciones que indiquen paso a paso la solución de un problema.
- Evaluación: habilidad para reconocer y determinar los alcances de realizar procesos, en términos de eficiencia y uso de recursos.

ESTADO DEL ARTE (respecto al pensamiento computacional y el aprendizaje de programación de computadoras en educación superior)

El pensamiento computacional representa una propuesta adecuada para fomentar el aprendizaje de habilidades que beneficien a los estudiantes que ingresan en las áreas STEM (Swaid, 2015; Lye & Koh, 2014; García-Peñalvo & Cruz-Benito, 2016; García-Peñalvo, 2016; García-Peñalvo, Reimann, Tuul, Rees, & Jormanainen, 2016). Representa el "foco de la innovación educativa por el conjunto de habilidades de solución de problemas que debe ser adquirido por las nuevas generaciones de estudiantes" y su evaluación es un tema de investigación que ha permitido la creación de pruebas en educación superior (Román, Pérez, & Jiménez, 2015), vinculación con el aprendizaje de la programación y la taxonomía de Bloom (Selby, 2015). Las habilidades pueden ser aprendidas por medio de un modelo de juego y a su vez introducir conceptos de programación que eviten la "frustración y la demanda de actividad de los estudiantes" (Kazimoglu, Kiernan, Bacon, & MacKinnon, 2012) sin olvidar que el pensamiento computacional no es sinónimo de programación.

Específicamente, para el caso de usar pensamiento computacional con el estudio inicial de programación de computadoras en la educación superior, existen las siguientes propuestas concretas.

Larkins y Harvey (2010) consideran no solo a estudiantes del área de ciencias de la computación sino aquellos que vienen de una variedad de ciencias e ingenierías para solucionar problemas en sus áreas, inclusive a nivel doctoral. Lingling et al. (2015) comentan que se tiene que ir más allá de "entrenar a los estudiantes para solucionar problemas usando un lenguaje de programación en práctica", se tiene que buscar su motivación pues "es un factor importante para mejorar el desempeño del estudiante" (Ramirez-Lopez & Muñoz, 2015).Walker (2015) defiende el hecho de que usar un lenguaje de programación ofrece precisión de las propuestas de los estudiantes para mejorar sus habilidades de pensamiento computacional para analizar una correcta solución y comparar soluciones alternativas, pues existe el problema de la adquisición de las competencias básicas para aprender la programación de computadoras y el desarrollo de software en general; el motivo principal "desde el punto de vista del estudiante es que no se tienen las capacidades mentales adecuadas para la solución de problemas" (Olivares, Jiménez, Ortiz, & Rodríguez, 2015). Particularmente, existen dos trabajos realizados por Czerkawski et al. (2015) y Weese (2016). El primero se propone revisar el estado actual del campo en educación superior y discutir si las habilidades del pensamiento computacional son relevantes fuera de los campos STEM. El segundo confirma el interés en el pensamiento computacional para el desarrollo curricular universitario de programación visual. "Es imperativo desarrollar las habilidades del pensamiento computacional entre los docentes y estudiantes de las áreas STEM para sostener la revolución científica" (Swaid, 2015). Así, el trabajo de la aplicación del pensamiento computacional en la educación superior sigue siendo fuente de investigación para diferentes aristas educativas, no exclusivas de las ciencias de la computación (Shiflet & Shifleta, 2012; Chilana, y otros, 2015).

Romero et al. (2017) inicialmente establecieron el impacto que tiene la creatividad para la creación de soluciones de codificación, lo que denominaron programación creativa,

permitiendo enmarcar el pensamiento computacional en dicho tipo de programación. Así, determinaron una relación de componentes del pensamiento computacional con las fases de la solución de problemas colaborativos que indica el programa de estudios para la evaluación internacional del estudiante. Posteriormente, propusieron una metodología para evaluar el pensamiento computacional en las actividades de la programación creativa. Los resultados sugirieron la necesidad de una evaluación humana de la programación creativa al tiempo que señalan los límites de la herramienta analítica automatizada usada, debido a que no refleja la diversidad creativa de los proyectos entregados por los estudiantes en el entorno de Scratch y anula la complejidad algorítmica. Yinnan y Chaosheng (2012) también comentan en su trabajo el uso del pensamiento computacional para incrementar la capacidad creativa de los estudiantes. En su propuesta organizan el contenido de enseñanza y elaboran experimentos para entrenar la capacidad de pensamiento durante un curso de programación. Zhang et al. (2011) también trabajan con la adecuación de currículo en el curso de programación Visual Basic. El impacto que tiene el entrenamiento del pensamiento computacional de los estudiantes recae en la intención de activar el análisis requerido en la solución de problemas que a su vez permita innovar en la programación de computadoras. Chen (2017) también determina una relación de habilidades del pensamiento computacional y el conocimiento de un lenguaje de programación, sin especificar la plataforma, para mejorar el desempeño de los estudiantes. Hace uso de la metodología de aprendizaje basada en problemas para reforzar el material teórico. Zhi-Mei y Xiang (2016) también hacen uso de la misma metodología, pero en primera estancia para mejorar el entrenamiento del pensamiento computacional a través de una secuencia de actividades y, en consecuencia, que los estudiantes mejoren su capacidad durante el proceso de resolución de problemas y finalmente impacte en el aprendizaje del lenguaje de programación C. Por otro lado, Ni (2017) enfoca su trabajo de investigación para el aprendizaje del lenguaje de programación C++. Promueve el pensamiento computacional de los estudiantes y usa la metodología de aprendizaje basada en casos reales. Xia (2016) hace uso de otras filosofías de enseñanza avanzadas, como así las denomina, para que el entrenamiento del pensamiento computacional favorezca las habilidades del estudiante y su proceso de aprendizaje de programación. Al igual que los anteriores trabajos de investigación, busca complementar la ejercitación del pensamiento computacional. La enseñanza por casos, el manejo de tareas, micro lecciones y MOOC, representan a tales filosofías.

Gao (2014) basa su investigación en dos habilidades del pensamiento computacional: Abstracción y automatización. En sus resultados, indica que a través del pensamiento computacional es posible resolver la contradicción entre enseñanza y aprendizaje en la resolución de problemas durante un curso de programación en C. Los estudiantes deben modelar adecuadamente para mejorar la efectividad del aprendizaje, analizar el modelo, verificar en un escenario práctico, y complementar con investigación, así el estudiante tiene el rol principal en el proceso de aprendizaje y el docente es un líder.

Huang et al. (2009) sin especificar que habilidad del pensamiento computacional trabajan con los estudiantes, proponen cultivar el pensamiento computacional para beneficiar la solución de problemas y el diseño algorítmico en el contexto de la programación orientada a objetos.

Ying y Pingping (2017) en su investigación comentan clases específicas de pensamiento computacional para entrenar a los estudiantes, con la intención de que descubran, analicen, resuelvan y generalicen problemas. En el proceso de enseñanza que proponen, los estudiantes deberán de producir diferentes mecanismos algorítmicos para resolver el mismo problema durante las clases presenciales. Posteriormente, se pasa a la etapa de codificación y poder obtener resultados de programación. Con la anterior estrategia, buscan resolver la problemática planteada: "altas calificaciones y bajas habilidades".

Compañ-Rosique et al. (2015) enfocan su trabajo de investigación desde otra perspectiva. El desarrollo de habilidades de pensamiento computacional para la resolución de problemas, así como el aprendizaje de la programación de computadoras sigue siendo el objetivo, pero la estrategia que proponen para conseguirlo es diferente. Establecen tres etapas para conseguir el aprendizaje: oír, ver y hacer, pero reelaboran su significado al momento de realizar la intervención docente, es decir, apelan a escuchar, observar y practicar.

Michaelson (2015) elabora una pedagogía para el aprendizaje de la programación a través del uso del pensamiento computacional. Las habilidades de descomposición, abstracción, reconocimiento de patrones y diseño algorítmico, las considera como un conjunto de etapas que se empalman e interactúan para generar la resolución de problemas, calificándola como una actividad creativa. Así, argumenta que a través de la información se debe estructurar la computación, es decir, iniciar con una concreta instancia del escenario del problema para que al usar el pensamiento computacional, el estudiante elabore buenas preguntas que le permitan desenredar estructuras de información conocidas y en consecuencia guiar el diseño de la computación.

Finalmente, resulta de interés que en general el pensamiento computacional apoye a los estudiantes no importando su perfil profesional, pero para el caso de género, Hamlin et al. (2010) reportan en su estudio, acerca de resolver problemas de ingeniería usando herramientas de computación y programación a través de un ejercicio de hoja de cálculo, que las "mujeres subestimaron sus habilidades mientras que los hombres sobreestimaron sus propias habilidades", aunque en general, no hay diferencia entre hombres y mujeres para las habilidades del pensamiento computacional, la elección de la profesión es la que hace parecer que el género masculino predomina en las áreas de la computación (Espino & González, 2015).

DISEÑO EXPERIMENTAL

El objetivo de la asignatura Metodología de la programación (MP), de cada unidad temática, así como la descripción del Saber y Saber Hacer de cada tema está redactado en la hoja de asignatura con base a una categoría cognitiva de la taxonomía de Bloom. Tomando como referencia la relación elaborada por Selby (2015) entre los niveles cognitivos de la taxonomía de Bloom con habilidades del pensamiento computacional: en el nivel de Aplicación la habilidad de Generalización, en el nivel de Análisis las habilidades de Abstracción y Descomposición, en el nivel de Síntesis la habilidad de Diseño algorítmico y en el nivel de Evaluación con la habilidad del mismo nombre; se definió una relación entre los conocimientos de la asignatura MP y el pensamiento computacional como se describe a continuación:

- La unidad temática 1 titulada "Conceptos básicos", tiene seis temas y en la redacción del pilar Saber están los verbos Identificar, Reconocer y Describir que corresponden al nivel dos de la taxonomía de Bloom, en el pilar Saber Hacer el verbo Determinar, también del nivel dos de Bloom y otro del cuarto, Esquematizar, por lo que se encuentran en el nivel de Comprender y Analizar, en consecuencia, se eligió que correspondan a las habilidades de Abstracción y Descomposición.
- La unidad temática 2 nombrada "Expresiones", usa en los tres temas el verbo Identificar para el pilar Saber, que corresponde al nivel dos de Bloom, mientras que en el pilar Saber Hacer se usa en tres ocasiones verbos del nivel tres Aplicar por lo que se eligió corresponda a la habilidad de Generalización.
- La unidad temática 3: "Algoritmos y diagramas de flujo" tiene cuatro temas, en el pilar Saber aparecen verbos del nivel cognitivo dos Comprender y en el pilar Saber Hacer un verbo del nivel cinco Evaluar, además de indicar la programación de ejercicios usando el paradigma de programación estructurada por lo que se eligió que correspondan a las habilidades de Evaluación y Diseño algorítmico.

La Tabla 1 contiene la relación establecida de las unidades temáticas del curso MP y las habilidades del pensamiento computacional: Descomposición, generalización, abstracción, diseño algorítmico y evaluación, así como el nombre del reactivo que la evalúa.
Aprovechando la relación mencionada y la forma en que están determinados los contenidos de la materia se eligieron los reactivos para evaluar el pensamiento computacional de los estudiantes de nuevo ingreso y obtener un diagnóstico con base a las habilidades detectadas.

Tabla 1

Relación de habilidades del pensamiento computacional y conocimientos del curso MP.

Fuente: Elaboración propia

Unidad Temática	Verbos Usados	Nivel taxonomía Bloom	Habilidad PC – ejercicio
1 Conceptos básicos, **6 temas**	Saber: Identificar, Reconocer y Describir	2 Comprender	Abstracción Canguro
	Saber Hacer: Esquematizar y Determinar	4 Analizar	Descomposición Móviles
2 Expresiones, **3 temas**	Saber: Identificar	2 Comprender	Generalización Espías
	Saber Hacer: Localizar, Resolver y Convertir	3 Aplicar	
3 Algoritmos y diagramas de flujo, **4 temas**	Saber: Reconocer, Identificar y Describir	2 Comprender	Evaluación Salto de charcos
	Saber Hacer: Comparar y Resolver	5 Evaluar	Diseño algorítmico Castores en movimiento

Los reactivos se eligieron de dos fuentes reconocidas a nivel Internacional: el concurso sobre fluidez informática y computacional en edades escolares UK Bebras Computational Thinking Challeng (2015), y la Olimpiada de computación de búsqueda de talento (Talent Search, 2015) que pretende orientar a los estudiantes sudafricanos con más aptitudes en pensamiento computacional hacia carreras técnicas; debido a que su redacción original es en inglés, se realizó una traducción al español. El motivo por el cual solo se seleccionó un ejercicio para cada una de las habilidades del pensamiento computacional, es el tiempo utilizado para responder la evaluación, un promedio de 40 minutos, no es intención que el estudiante invierta mucho tiempo en responder ejercicios; la validación de cada pregunta está garantizada al ser instrumentos de fuentes internacionales usadas en eventos de competencia, aunque cada ejercicio seleccionado no se limita a evaluar exclusivamente la habilidad para la cual fue seleccionada. Para la evaluación del pensamiento computacional a través de los reactivos seleccionados, se creó un formulario en las aplicaciones de Google desde la cuenta institucional de la UTP, los estudiantes la contestaron dentro de la universidad para asegurar que se realizara.

El ejercicio de Bebras denominado Móviles evalúa en gran medida la habilidad de Descomposición. La redacción del ejercicio es la siguiente:

Un móvil es una pieza de arte que cuelga del techo, generalmente en los dormitorios. Un móvil consiste de palos y figuras. Cada palo tiene unos cuantos puntos donde figuras u otros palos pueden ser atados. Además, cada palo tiene un punto para colgar, donde se cuelga a un palo hacia abajo (o hacia el techo). La Figura 1 es un ejemplo de móvil que puede ser descrito usando números y paréntesis, Pregunta: ¿Cuál de los siguientes móviles (Figura 2) puede ser construido usando las siguientes instrucciones? (-3 (-1 4) (2 (-1 1) (1 1))) (2 (-1 6) (2 3)).

Figura 1. Descripción del móvil con una expresión: (-3 (-1 1) (1 1)) (2 3). Fuente: Bebras (2015)

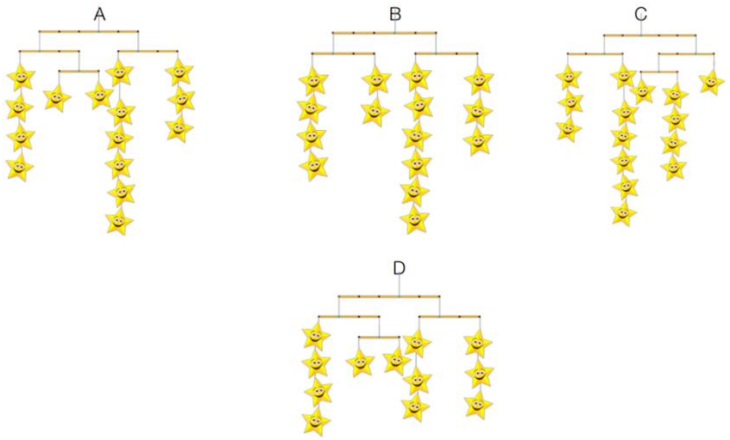

Figura 2. Opciones de respuestas del reactivo móvil. Fuente: Bebras (2015)

El estudiante debe seleccionar la imagen correcta a partir de establecer que la relación de la imagen con la expresión de los números y paréntesis conduce a determinar que el móvil es una composición de pequeños móviles. La respuesta correcta es la opción A.
El ejercicio nombrado Canguro, también de Bebras, destaca por su evaluación de la Abstracción pues el estudiante decide, analiza y determina qué información le va asistiendo para obtener la respuesta correcta del problema planteado, ver Figura 3, la respuesta es Plato I.

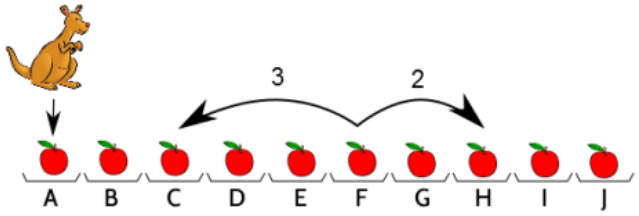

Figura 3. La tarea de recoger las manzanas. Fuente: Bebras (2015)

La redacción del ejercicio anterior en la evaluación es la siguiente:

Hay 10 platos en una fila. Hay una manzana en cada plato. Al canguro Tomás le encanta saltar. Primero, el salta desde el plato más a la izquierda con la letra A. En cada salto después del inicial, salta dos platos hacia adelante, o tres platos hacia atrás. (Un ejemplo de los dos posibles saltos desde un plato es mostrado con flechas en la imagen.) Tomás sólo salta hacia platos con una manzana. Si el salta hacia un plato, recoge la manzana. Pregunta: Si Tomás recoge todas las 10 manzanas, ¿cuál manzana recoge al final? A, B, C, D, E, F, G, H, I, o J.

El tercer ejercicio tomado de Bebras con nombre Espías, se enfoca en evaluar la habilidad de Generalización, pues una vez que verifica la solución indicada para el problema inicial, aplica

a una situación similar la forma de resolver el problema que contiene características parecidas. La respuesta correcta es 4. La redacción del ejercicio es la siguiente:

Cada viernes, seis espías intercambian toda la información que han reunido durante la semana. Un espía nunca puede ser visto con más de otro espía al mismo tiempo. Así, tienen que tener varios encuentros de reunión en pares y compartir la información que poseen. El grupo de 6 espías sólo necesitan tres encuentros para distribuir todos sus secretos. Antes del encuentro cada espía mantiene una sola pieza de información (espía 1 conoce 'a', espía 2 conoce 'b', etc.). En el primer encuentro espía 1 y 2 se encuentran y comparten información entonces ahora ambos conocen 'ab'. La Figura 4 muestra cuales espías se encuentran en cada reunión a través de una línea. También muestra cuáles piezas de información tienen todos. Después de tres encuentros toda la información ha sido distribuida. Pregunta: Después de un incidente internacional un espía ha dejado de atender los encuentros. ¿Cuál es el número mínimo de reuniones necesitadas por los cinco espías restantes para intercambiar toda la información? (ver Figura 5).

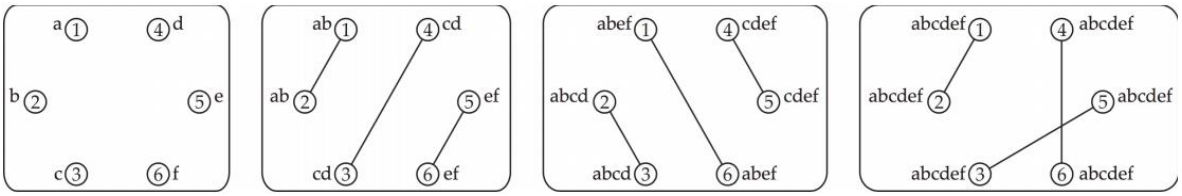

Figura 4. Seis espías intercambiando información. Fuente: Bebras (2015)

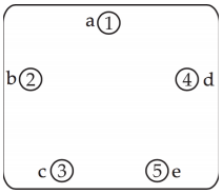

Figura 5. Cinco espías intercambiando información. Fuente: Bebras (2015)

Dos reactivos se tomaron de la Olimpiada de Computación. El primero nombrado Castores en movimiento se eligió para evaluar la habilidad del Diseño algorítmico debido a que el estudiante que lo resuelva muestra su capacidad de entender la organización de instrucciones para resolver un problema, en este caso las indicaciones para que los castores crucen los hoyos. La respuesta es 2 1 6 5 3 4 7. La redacción del ejercicio en la evaluación es la siguiente:

Una colonia de castores está viajando a través de un bosque oscuro. El camino es estrecho, así que tienen que viajar en una fila sin pasar uno del otro. Algunas veces hay un hoyo en el camino. Un hoyo es cruzado de la siguiente manera:

- Primero saltan tantos castores sean necesarios para llenar el hoyo.
- La colonia entera pasará entonces a través del hoyo.
- Los castores que saltaron treparán para salir del hoyo, y unirse al final de la línea.

La Figura 6 muestra cómo cinco castores pasan un pequeño hoyo que se llena con tres castores.

Figura 6. Ejemplo de cinco castores al cruzar un hoyo. Fuente: Talent Search (2015)

Una colonia de 7 castores pasa a través del bosque. Cruzan 3 hoyos. El primer hoyo se ajusta a 4 castores, el segundo se ajusta a 2, y el último hoyo se ajusta a 3 castores, ver Figura 7, ¿En qué orden se encontrarán los castores después de que hayan pasado el tercer hoyo?

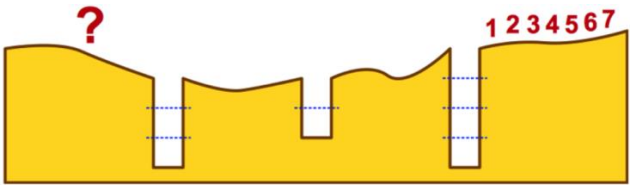

Figura 7. Siete castores buscando cruzar hoyos. Fuente: Talent Search (2015)

El segundo reactivo, de la Olimpiada de Computación, con nombre Salto de charcos, se escogió para determinar la habilidad de Evaluación, es decir, realizar paso a paso las instrucciones indicadas para completar el ejercicio y obtener la respuesta solicitada. La solución es Berta, Dora, Carlos, Ana y Luisa. La redacción del reactivo es la siguiente:

Ana (edad 7), Berta (edad 8), Carlos (edad 9), Dora (edad 10) y Luisa (edad 11) están jugando un juego donde saltan de un charco a otro. Ellos han ubicado flechas entre los charcos, y todos inician del lado izquierdo como se indica en la Figura 8. Cuando un niño salta dentro de un charco él o ella espera la llegada de un segundo niño. El niño mayor en el charco entonces saltará de acuerdo a la flecha gruesa, el más joven sigue la flecha delgada. ¿Cuál es el orden (de arriba hacia abajo) en el cual los niños terminarán a la derecha?

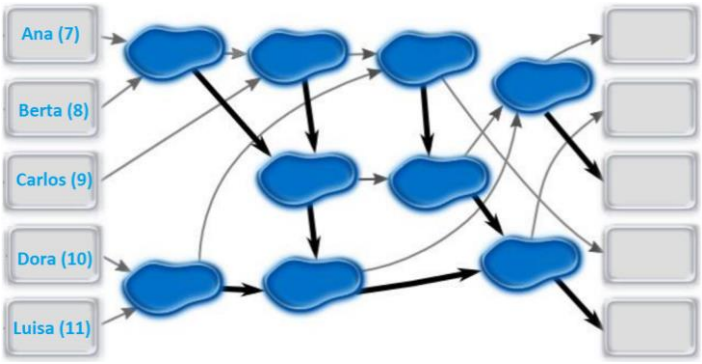

Figura 8. Salto de Charcos. Fuente: Talent Search (2015)

Con base en los resultados obtenidos de la evaluación del pensamiento computacional, el docente pudo ofrecer escenarios de aprendizaje a los estudiantes con la intensión de subsanar las carencias detectadas o favorecer al aprendizaje autónomo para aquellos que mostraron habilidades en la resolución de problemas, es decir, se ofreció la opción de clase presencial o modalidad *b-learning*. Al final del curso, se solicitó realizar de forma voluntaria la encuesta indicada en la Tabla 2 para conocer el impacto del diseño experimental en la adquisición de las competencias respectivas del curso MP.

RESULTADOS

Acerca de los resultados de la evaluación del pensamiento computacional

La primera intervención de la evaluación del pensamiento computacional se dio para el cuatrimestre septiembre – diciembre de 2016. Los resultados en cada uno de los reactivos, de un total de 65 estudiantes donde el investigador dio el curso MP se encuentran en la Tabla 3. Hay que observar que en todos los reactivos los estudiantes eligieron todas las opciones, no existe un contraste entre respuesta correcta y una incorrecta, lo que refleja la diversidad de habilidades en pensamiento computacional entre los 65 evaluados. Tomando en cuenta la relación establecida entre las unidades temáticas y las habilidades del pensamiento computacional, los estudiantes requieren de trabajo académico desde las primeras dos unidades temáticas, debido a que las habilidades de descomposición y generalización no representan una fortaleza para la mayoría de ellos, solo un ligero 52,3% la habilidad de abstracción. A pesar de lo anterior, un 73,8% de los 65 estudiantes indican un perfil adecuado para el diseño algorítmico y la evaluación de los mismos (67,7%).

Tabla 2

Instrumento encuesta para obtener información al final del cuatrimestre. Fuente: Elaboración propia

Pregunta	Opción de respuesta
Selecciona los conceptos con los que estás familiarizado	Opción Múltiple

Tipos de datos	
Operadores aritméticos	
Creación de identificadores para variables	
Operadores lógicos	
Operadores relacionales	
Jerarquía de operadores	
Resolver expresiones aritméticas, lógicas, y relacionales	
Uso de una variable contador y acumulador	
Estructura de selección	
Estructura de repetición	
Definición y creación de un algoritmo	
¿La modalidad de aprendizaje fue adecuada para adquirir las competencias del curso?	Si No
¿La evaluación de tus habilidades al inicio del cuatrimestre fue una actividad acertada para determinar el mejor entorno de aprendizaje?	Si No
¿Qué recomendación tienes para las futuras generaciones acerca de la forma de aprender el contenido del curso?	Abierta

La segunda intervención del diseño experimental se realizó para el cuatrimestre septiembre – diciembre de 2017. La evaluación fue realizada por un total de 110 estudiantes de nuevo ingreso. Para cada uno de los reactivos la Tabla 3 contiene los porcentajes de la respuesta correcta. En esta generación resulta nuevamente favorable que un porcentaje alto de estudiantes tengan la habilidad de diseño algorítmico, aunque ligeramente menor en comparación al año pasado, a pesar de esto, representa una fortaleza para el estudio del curso MP, así como el porcentaje de estudiantes que tienen la habilidad de evaluación. La habilidad de generalización tiene un porcentaje bajo, por lo que el trabajo para la segunda unidad del curso deberá ser reforzada considerando la relación establecida de conocimientos con la evaluación del pensamiento computacional. Finalmente, debido a que las habilidades que corresponden con el contenido de la primera unidad temática son menores al 50%, los conceptos básicos deberán ser parte de un aprendizaje significativo para al final del curso tener buenos resultados.

La tercera intervención de la evaluación del pensamiento computacional se realizó en el cuatrimestre septiembre – diciembre de 2018. La información obtenida fue de 255 estudiantes de nuevo ingreso, en la Tabla 3 se pueden consultar los porcentajes de las respuestas correctas por cada reactivo, desafortunadamente el investigador no estuvo asignado a un curso de MP, pero como se comenta más adelante la información de la evaluación sirvió para realizar un pronóstico de retención académica.

La cuarta intervención de la evaluación del pensamiento computacional fue en el cuatrimestre septiembre – diciembre de 2019, el concentrado de resultados está en la Tabla 3. La información obtenida fue de 263 estudiantes de nuevo ingreso.

Tabla 3

Concentrado de porcentajes de respuestas correctas de cada reactivo para la evaluación del pensamiento computacional del 2016 al 2019. Fuente: Elaboración propia

Reactivo / año	2016	2017	2018	2019
Descomposición	33,8	38,2	29	33,5
Abstracción	52,3	45,5	43,5	47,9
Generalización	36,9	29,1	31	39,5
Diseño algorítmico	73,8	68,2	68,2	67,3
Evaluación	67,7	58,2	60,4	54,8

Finalmente, por primera vez el investigador fue asignado a un curso de Lenguaje de programación para la carrera de Mecatrónica, especialidad de Automatización. El contenido temático de las primeras dos unidades corresponde a los conceptos vistos en el curso de MP, por lo cual se consideró pertinente realizar también la evaluación del pensamiento computacional. Hay que considerar que son estudiantes de cuarto cuatrimestre y se esperaba tener un resultado diferente a los estudiantes de nuevo ingreso. La información obtenida fue de 17 estudiantes. Debido a los resultados favorables que obtuvieron los estudiantes, se espera un curso donde la adquisición de las respectivas competencias no sea una problemática para el estudiante-docente.

Acerca de las encuestas realizadas

Al final del cuatrimestre septiembre – diciembre de 2016, la respectiva encuesta voluntaria fue realizada, 15 de los 65 estudiantes respondieron; los valores a las preguntas están en la Tabla 4. Como se puede observar, casi todos los conceptos tienen una familiaridad mayor al 50% excepto la estructura de selección que representa a la más baja (26,7%). La modalidad de aprendizaje tiene una alta aceptación (86,7%), así como la evaluación del pensamiento computacional (73,3%). De la pregunta abierta que obtiene las recomendaciones para futuras generaciones, un 20% no aporta algún comentario, pero aquellos que lo hacen recomiendan estudiar más de lo visto en clase (46,6%), así como optar por la modalidad presencial donde la enseñanza sea la misma para todos, al parecer la autonomía de estudio no es una capacidad deseada por los estudiantes solo un 6,7% lo recomiendan.

Tabla 4

Encuesta al final del cuatrimestre septiembre-diciembre de 2016. Fuente: Elaboración propia

Pregunta	Opción de respuesta
Selecciona los conceptos con los que estás familiarizado	

Tipos de datos	73,3%
Operadores aritméticos	60,0%
Creación de identificadores para variables	66,7%
Operadores lógicos	73,3%
Operadores relacionales	66,7%
Jerarquía de operadores	73,3%
Resolver expresiones aritméticas, lógicas, y relacionales	66,7%
Uso de una variable contador y acumulador	73,3%
Estructura de selección	26,7%
Estructura de repetición	73,3%
Definición y creación de un algoritmo	66,7%
¿La modalidad de aprendizaje fue adecuada para adquirir las competencias del curso?	Si – 86,7% No – 13,3%
¿La evaluación de tus habilidades al inicio del cuatrimestre fue una actividad acertada para determinar el mejor entorno de aprendizaje?	Si – 73,3% No – 26,7%
¿Qué recomendación tienes para las futuras generaciones acerca de la forma de aprender el contenido del curso?	Estudiar – 46,6% Presencial – 20% Ninguna – 20% Lo mismo para todos – 6,7% Me gustó y motivó a investigar y aprender más por mi cuenta – 6,7%

En el cuatrimestre septiembre-diciembre de 2017 solo en un grupo donde el investigador fue docente se obtuvo información de la encuesta final. Un total de 10 estudiantes respondieron con los valores indicados en la Tabla 5. Como se puede observar, casi todos los conceptos tienen una familiaridad mayor al 80% excepto la estructura de selección que representa por segunda ocasión a la más baja (60%), pero doblemente mejor en comparación al segundo año. La modalidad de aprendizaje tiene una total aceptación (100%), así como la evaluación del pensamiento computacional. De la pregunta abierta que recupera las recomendaciones para futuras generaciones, el 100% indica motivación por la promoción del aprendizaje autónomo en el curso.

Tabla 5

Encuesta al final del cuatrimestre septiembre-diciembre de 2017. Fuente: Elaboración propia

Pregunta	Opción de respuesta
Selecciona los conceptos con los que estás familiarizado	
Tipos de datos	90,0%
Operadores aritméticos	90,0%
Creación de identificadores para variables	100%
Operadores lógicos	90,0%
Operadores relacionales	80,0%
Jerarquía de operadores	90,0%
Resolver expresiones aritméticas, lógicas, y relacionales	80,0%
Uso de una variable contador y acumulador	90,0%

Estructura de selección	60,0%
Estructura de repetición	100%
Definición y creación de un algoritmo	80,0%
¿La modalidad de aprendizaje fue adecuada para adquirir las competencias del curso?	Si – 100% No – 0%
¿La evaluación de tus habilidades al inicio del cuatrimestre fue una actividad acertada para determinar el mejor entorno de aprendizaje?	Si – 100% No – 0%
¿Qué recomendación tienes para las futuras generaciones acerca de la forma de aprender el contenido del curso?	Aceptación total, motivó a investigar y aprender más de forma autónoma.

Acerca de la predicción de retención de matrícula

En el cuatrimestre septiembre – diciembre de 2018, el investigador no fue asignado a ningún curso de MP. En consecuencia, el uso de la información fue predecir cuál sería el porcentaje de matrícula al final del curso considerando que con la evaluación del pensamiento computacional es posible detectar la carencia de ciertas habilidades que impactan en el aprendizaje de los contenidos temáticos. Tomando como referencia los reactivos correctos de la evaluación del pensamiento computacional, se generaron 5 grupos que se describen a continuación.

Grupo 1
Estudiantes que no obtuvieron alguna respuesta correcta de los cinco reactivos, por lo cual se consideró altamente cuestionable su perfil pues al parecer carecían de habilidades para la resolución de problemas.

Grupo 2
Estudiantes que obtuvieron la respuesta correcta de los cinco reactivos, por lo cual se consideró un perfil adecuado para el estudio de la carrera al evidenciar habilidades en la resolución de problemas.

Grupo 3
Estudiantes que obtuvieron solo un reactivo correcto, por lo cual se consideró que podían ser bajas potenciales de matrícula en los primeros meses o al final del cuatrimestre.

Grupo 4
Estudiantes que obtuvieron solo uno de los cinco reactivos incorrecto, por lo cual se consideraron buenos candidatos para estudiar la carrera.

Grupo 5
Estudiantes que obtuvieron dos o tres reactivos correctos, por lo que correspondieron a los jóvenes que requerían trabajo académico en dos sentidos, para incrementar sus habilidades del pensamiento computacional y para mantenerse matriculados al final del cuatrimestre al menos aprobando el curso Metodología de la programación.

Tomando como referencia la descripción de cada grupo determinado, se predicen tres categorías para la matrícula al final de cuatrimestre septiembre – diciembre de 2018 respecto al curso MP.

1. Los que representan una baja escolar muy probable. Aquellos del grupo 1 y 3.
2. Los que se mantendrán para el siguiente cuatrimestre. Estudiantes del grupo 2 y 4.
3. Los que su permanencia es variable debido al trabajo académico que requieren para mantener su motivación o baja frustración ante las problemáticas, dificultades o complejidad de aprendizaje. Aquellos del grupo 5.

Con los resultados de la evaluación del pensamiento computacional se obtuvo la siguiente información: nueve estudiantes de los 255 formaron parte del grupo 1, doce se agruparon al segundo, 48 fueron parte del grupo 3, 37 integraron el grupo 4 y el grupo 5 tuvo la mayor cantidad de estudiantes con un total de 149 de los 255. La Tabla 6 contiene los valores de la información previamente comentada. Además, en la Figura 9 se muestran los porcentajes del total de estudiantes para cada uno de los grupos y sus determinadas descripciones, en la Figura 10 se ilustra la cantidad de los estudiantes que se esperaba tener matriculados, es decir, la predicción al final del curso para el cuatrimestre septiembre- diciembre de 2018.

Tabla 6

Distribución de estudiantes por grupo, descripción y predicción de matrícula en 2018.

Fuente: Elaboración propia

Grupo	Reactivos correctos	Estudiantes (255)	Descripción	Predicción de matrícula
1	0	9	Perfil altamente cuestionable	Deserción – 57 – 22,35%
3	1	48	Bajas potenciales	
2	5	12	Perfil adecuado	Segura – 49 – 19,22%
4	4	37	Buenos candidatos	
5	2 o 3	149	Trabajo académico	Variable – 149 – 58,43%

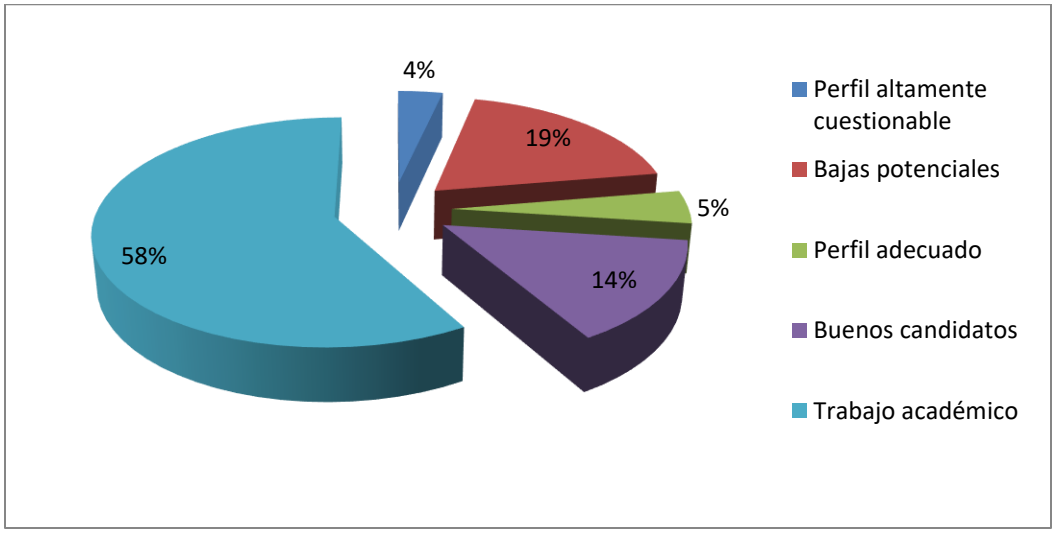

Figura 9 Distribución de estudiantes por grupo a partir de evaluación de pensamiento computacional en 2018. Fuente: Elaboración propia

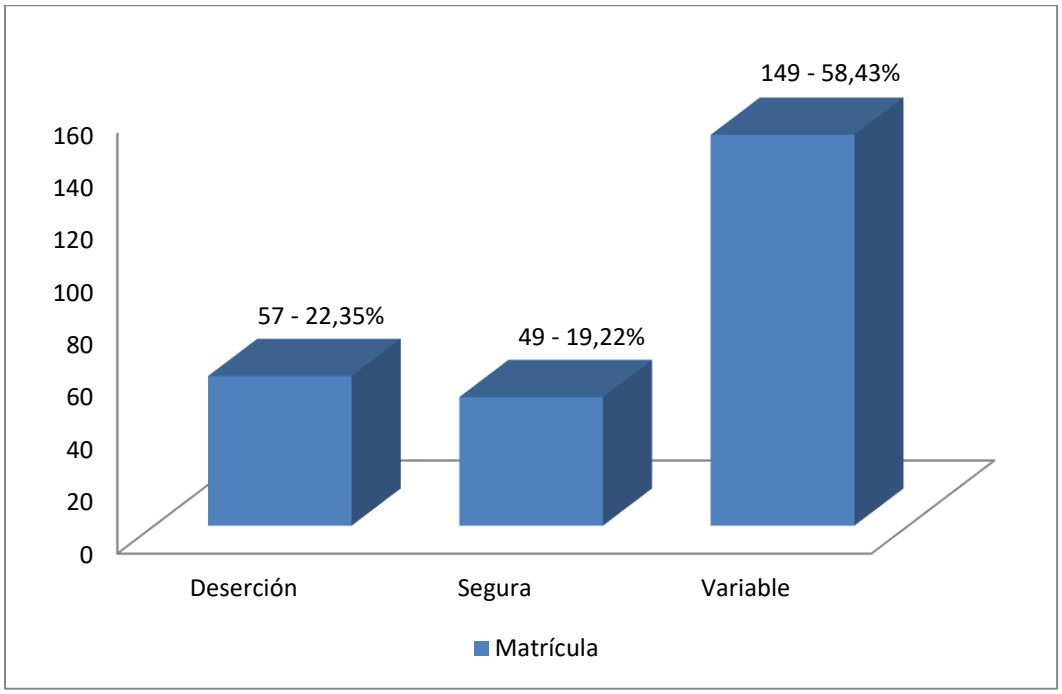

Figura 10. Predicción de matrícula al final de cuatrimestre septiembre – diciembre de 2018. Fuente: Elaboración propia

Los valores reales de matrícula fueron 325 estudiantes en primer cuatrimestre y 202 para el segundo. Lo que representa un 62,15% de retención o 37,85% de deserción. El pronóstico determinado representa una posibilidad de mejora al poder tener solo la pérdida de 57

estudiantes en lugar de 123. Es importante destacar que dentro del pronóstico existe una alta recomendación de trabajo académico para poder lograrlo, en el caso de estudio representó a una población de 149 estudiantes que requerían un seguimiento individualizado.
En el cuatrimestre septiembre – diciembre de 2019, el ejercicio de predicción de matrícula para esta generación de estudiantes es la siguiente considerando los mismos grupos y categorías indicadas previamente para el año 2018 con los datos contenidos en la Tabla 7.
La Figura 11 contiene los porcentajes para cada uno de los grupos. El 5% de los estudiantes evaluados no tienen un perfil adecuado para el estudio de las T.I. debido a que no demostraron alguna habilidad para la solución de problemas. Un 16% de los estudiantes representan una población alta que puede desertar en los primeros meses del cuatrimestre septiembre – diciembre de 2019 debido a que solo demostraron una habilidad para la solución de problemas, muchos de ellos ni las básicas como abstracción y descomposición que impactan en la primera unidad del curso MP por lo que debe considerarse como una pérdida con alta probabilidad de realización junto con el porcentaje anterior. Un 3% de los estudiantes evaluados demostraron un perfil alto para el estudio de T.I. representando a una población con talento para el desarrollo de software. Un 16% de estudiantes forman al grupo de trabajo estable que debido a las habilidades que demostraron tener requieren de una actividad de seguimiento para mantener su motivación académica. Finalmente, un 61% de estudiantes requieren de mucho trabajo académico de los profesores para regularizar sus habilidades, mantener una motivación y acreditar los cursos del primer y segundo cuatrimestre. Las acciones para cada estudiante pueden ser determinadas con base a los reactivos correctos que obtuvo, por lo que se requiere de trabajo personalizado para evitar baja escolar. Por lo anterior, se predice una deserción del 22% lo que representa a 57 estudiantes; una permanencia segura de 46 estudiantes (17%) y una permanencia variable de 61% (160 estudiantes).

Tabla 7

Distribución de estudiantes por grupo, descripción y predicción de matrícula en 2019.

Fuente: Elaboración propia

Grupo	Reactivos correctos	Estudiantes (263)	Descripción	Predicción de matrícula
1	0	14	Perfil altamente cuestionable	Deserción – 57 – 22%
3	1	43	Bajas potenciales	
2	5	8	Perfil adecuado	Segura – 46 – 17%
4	4	38	Buenos candidatos	
5	2 o 3	160	Trabajo académico	Variable – 160 – 61%

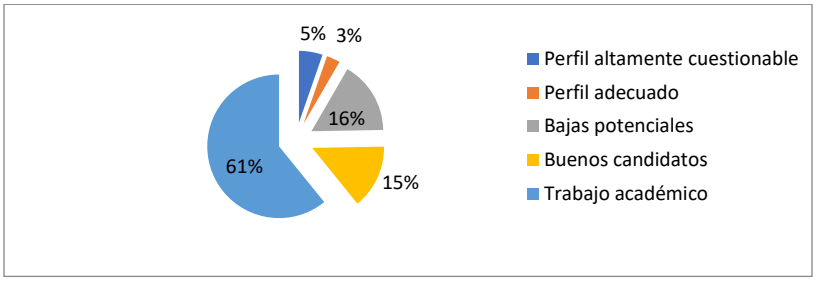

Figura 11. Distribución de estudiantes por grupo a partir de evaluación de pensamiento computacional en 2019. Fuente: Elaboración propia

Acerca del diseño experimental para INACAP-Chile

El trabajo de investigación realizado en una estancia del 1 de marzo al 30 de junio de 2019 en INACAP-Chile, tuvo el objetivo de usar la evaluación del pensamiento computacional a los estudiantes de nuevo ingreso de las carreras Ingeniería en informática y Analista programador de la Universidad Tecnológica de Chile y del Centro de Formación Técnica respectivamente, para favorecer la motivación y autonomía de estudio a través del reconocimiento de habilidades y el uso del diseño instruccional del curso presencial.

Tomando como referencia la asignatura presencial Fundamentos de programación, primer curso en ambas mallas curriculares para el desarrollo de software, donde en la descripción se indica una "asignatura práctica orientada a desarrollar el pensamiento lógico del estudiante mediante el análisis y resolución de problemas", se puede considerar una evaluación del pensamiento computacional para determinar las habilidades que posee un estudiante en la resolución de problemas, y obtener un diagnóstico inicial que represente una primer guía académica para el docente ante la dificultad que puede tener el estudiante cuando se desarrolle el contenido del curso; con el objetivo de favorecer la motivación y autonomía de estudio, además de reducir la deserción escolar a través de acciones preventivas en lugar de correctivas durante el desarrollo de la asignatura.

En trabajo colaborativo con las direcciones de Investigación y Desarrollo (I+D), INACAP online y del Área de Informática y Telecomunicaciones, se logró una propuesta de investigación robusta y factible en su ejecución. Lo anterior debido a la fuerte vinculación de la intervención realizada en México con el modelo educativo del INACAP y el diseño instruccional de la asignatura.

El curso presencial Fundamentos de Programación tiene cinco unidades de aprendizaje y 90 horas asignadas para el semestre de 18 semanas. Debido al modelo educativo de INACAP el diseño instruccional integra horas de trabajo en línea (72 presenciales y 18 en línea para el curso), el detalle de la distribución de horas, semanas, ponderación y número de criterios de evaluación está indicado en la Tabla 8. En cada unidad se describe el aprendizaje esperado (ver Tabla 9) y una lista de los criterios de evaluación (ver Tabla 10 a Tabla 14), para cada asignatura se realizan tres acciones evaluativas: diagnóstica, formativa y sumativa; las dos

primeras corresponden a actividades sugeridas que contribuirán a mejorar los resultados de aprendizaje y permitirá conocer el avance del estudiante en distintos momentos de la asignatura, la evaluación sumativa es de carácter obligatorio y se realiza según lo estipule el curso. Cada unidad de aprendizaje aporta un porcentaje de la evaluación sumativa, la suma de las cinco en el curso genera un 80%, el 20% faltante lo específica el docente a su criterio por medio de evaluar presentaciones, proyectos o trabajos extras.

Tabla 8

Planeación del curso Fundamentos de programación en INACAP. Fuente: Elaboración propia

Unidad de aprendizaje	**Horas**			**Semanas**	**Criterios de Evaluación/ Ponderación**
	Presenciales	En línea	Totales		
1. Fundamentos de procesamiento de datos	12	3	15	3	5 / 10%
2. Estructuras de control en DFD	16	4	20	4	6 / 20%
3. Estructuras de control en pseudocódigo	20	5	25	5	6 / 20%
4. Estructura de arreglo	12	3	15	3	5 / 15%
5. Subrutinas	12	3	15	3	5 / 15%

Tabla 9

Aprendizajes esperados por unidad de aprendizaje. Fuente: Elaboración propia

Unidad de aprendizaje	**Aprendizaje esperado**
1	Resuelve problemas de procesamiento de datos, aplicando principios de almacenamiento y tablas de verdad.
2	Representa gráficamente la solución de un problema mediante diagramas de flujo de datos aplicando estructuras de control.
3	Desarrolla Algoritmos en Pseudocódigo, aplicando estructuras de control en la solución de un problema planteado.
4	Desarrolla Algoritmos en Pseudocódigo, utilizando Arreglos unidimensionales y bidimensionales en la solución de un problema planteado.
5	Desarrolla algoritmos básicos en Pseudocódigo, mediante subrutinas en la solución de un problema planteado.

Tabla 10

Criterios de evaluación y actividades, unidad de aprendizaje 1. Fuente: Elaboración propia

Criterio de evaluación	Actividad – modalidad
1.1.1. Identificando datos de entrada, proceso y salida a partir de un problema de procesamiento de datos. **1.1.2. Aplicando las etapas de la metodología de Polya en el análisis de la solución del problema planteado.** **1.1.3. Considerando operaciones de entrada, procesos y salida de datos en memoria.** **1.1.4. Aplicando operadores lógicos en la solución de problemas de procesamiento de datos.**	- Evaluación diagnóstica – presencial - Actividad formativa 1 – en línea - Guía de ejercicios 1 / evaluación formativa – presencial - Actividad formativa 2 – en línea - Guía de ejercicios 2 / evaluación formativa – presencial - Evaluación sumativa – presencial

Tabla 11

Criterios de evaluación y actividades, unidad de aprendizaje 2. Fuente: Elaboración propia

Criterio de evaluación	Actividad – modalidad
2.1.1. Aplicando estructuras de decisión en la solución del problema. **2.1.2. Incorporando operadores lógicos en la solución del problema.** **2.1.3. Utilizando estructuras de repetición en la solución del problema.** **2.1.4. Considerando la validación de datos en la solución del problema.** **2.1.5. Realizando la traza de la solución propuesta.**	- Evaluación diagnóstica – presencial - Guía de ejercicios 3 / evaluación formativa – presencial - Foro – en línea - Guía de ejercicios 4 / evaluación formativa – presencial - Evaluación sumativa – presencial

Tabla 12

Criterios de evaluación y actividades, unidad de aprendizaje 3. Fuente: Elaboración propia

Criterio de evaluación	Actividad – modalidad
3.1.1. Aplicando estructuras de decisión en la solución del problema. **3.1.2. Utilizando operadores lógicos en la construcción de algoritmos.** **3.1.3. Incorporando estructuras de repetición en algoritmos en Pseudocódigo.** **3.1.4. Considerando la validación de datos en la solución del problema.** **3.1.5. Realizando la traza de la solución propuesta.**	- Evaluación diagnóstica – presencial - Actividad formativa – en línea - Guía de ejercicios 5 / evaluación formativa – presencial - Foro – en línea - Evaluación sumativa – presencial

Tabla 13

Criterios de evaluación y actividades, unidad de aprendizaje 4. Fuente: Elaboración propia

Criterio de evaluación	Actividad – modalidad
4.1.1. Ingresando datos en arreglos unidimensionales y bidimensionales.	- Evaluación diagnóstica – presencial - Actividad formativa – en línea

4.1.2. Realizando un recorrido sobre arreglos unidimensionales o bidimensionales. **4.1.3. Realizando búsquedas en arreglos unidimensionales o bidimensionales.** **4.1.4. Realizando la traza de la solución propuesta.**	- Guía de ejercicios 6 / evaluación formativa – presencial - Foro – en línea - Evaluación sumativa – presencial

Tabla 14

Criterios de evaluación y actividades, unidad de aprendizaje 5. Fuente: Elaboración propia

Criterio de evaluación	Actividad – modalidad
5.1.1. Incorporando paso de parámetros a la subrutina. **5.1.2. Incorporando el retorno de datos al programa principal.** **5.1.3. Realizando llamadas a las subrutinas creadas.** **5.1.4. Realizando la traza de la solución propuesta.**	- Evaluación diagnóstica – presencial - Guía de ejercicios 7 / evaluación formativa – presencial - Foro – en línea - Evaluación sumativa – presencial

Inicialmente, en similitud a la intervención realizada en México, se establece una relación de conocimientos de la asignatura Fundamentos de programación con las habilidades del pensamiento computacional como se indica en la Tabla 15. En cada relación se indica la justificación y los criterios de evaluación con los que se determina un impacto, además del nombre del reactivo seleccionado, los cuales también corresponden a los usados previamente (Rojas-López & García-Peñalvo, 2018).

Tabla 15

Relación de habilidades del pensamiento computacional con unidades de aprendizaje. Fuente: Elaboración propia

Unidad de aprendizaje	Habilidad Pensamiento Computacional / reactivo	Justificación
1. Fundamentos de procesamiento de datos	Abstracción / Canguro	La abstracción ayuda a determinar los datos que contribuyen a desarrollar la resolución de alguna problemática como lo indica el aprendizaje esperado de la unidad. Impactará en los criterios de evaluación: 1.1.1, 1.1.3, 1.1.4
2. Estructuras de control en DFD	Evaluación / Salto de Charcos	La evaluación permite reconocer y determinar el alcance de ejecución de los diagramas de flujo de datos en la resolución de un problema que es parte del aprendizaje esperado de la unidad. Impactará en el criterio de evaluación: 2.1.5
3. Estructuras de control en pseudocódigo	Diseño algorítmico / Castores	El diseño algorítmico permite el desarrollo de algoritmos en pseudocódigo (o lenguaje de

		programación) que es objetivo del aprendizaje esperado de la unidad. Impactará en los criterios de evaluación: 3.1.1, 3.1.2, 3.1.3, 3.1.4
4. Estructura de arreglos	Generalización / Espías	El manejo de arreglos representa el primer incremento en la complejidad del diseño algorítmico, debido a que el estudiante ha visualizado la usabilidad de la estructura para resolver problemas donde previamente había creado una solución, pero ahora ha incorporado la comprensión de características similares, así puede generalizar una solución de forma óptima. Impactará en los criterios de evaluación: 4.1.2, 4.1.3
5. Subrutinas	Descomposición / Móviles	La creación de subrutinas o funciones representa la habilidad para fragmentar (descomponer) un problema en bloques funcionales y de menor tamaño. Impactará en los criterios de evaluación: 5.1.1, 5.1.2, 5.1.3

La interpretación diagnóstica de los resultados que se propone con la evaluación del pensamiento computacional, está determinada por 8 escenarios que el estudiante puede enfrentar en la modalidad presencial del curso. El número de escenarios corresponde a los casos que representan desde la detección de posibles talentos (cinco reactivos correctos) hasta carencia de habilidades para la resolución de problemas (cinco reactivos incorrectos). Se consideran escenarios donde se establecen posibles carencias desde el inicio, a mitad o final del curso respecto a las unidades de aprendizaje con los reactivos correspondientes. Por lo anterior, los escenarios propuestos no son todas las combinaciones de respuestas correctas de los cinco reactivos, solo de aquellos según la relación con la unidad de aprendizaje. Las unidades 1 y 2 tienen un nivel básico y de gran importancia para el curso por el contenido formativo y de conceptos. La unidad 3 representa un nivel medio de aprendizaje al repasar el concepto de estructuras de control en pseudocódigo. Finalmente, las unidades 3 y 4 se asocian con un nivel alto de aprendizaje dentro del curso. La Tabla 16 muestra la determinación de los escenarios a partir de lo comentado previamente.

El primer escenario corresponde a los estudiantes que exhiben habilidades en la resolución de problemas, por lo que se considera que no tendrán problemáticas con los contenidos de aprendizaje del curso y pueden tener un alto grado de autonomía. El segundo escenario determina el caso contrario del primero. A partir del tercero hasta el octavo se toman en cuenta los reactivos correctos según la unidad de aprendizaje con la que fue relacionada.

Tabla 16

Determinación de los escenarios de aprendizaje en INACAP. Fuente: Elaboración propia

Nivel	**Habilidad**	**Unidad**	**Reactivo**	**Escenarios**							
			1 – correcto, 0 – incorrecto	1	2	3	4	5	6	7	8

Básico	Abstracción	1	Canguro	1	0	0	1	0	1	0	1
	Evaluación	2	Salto de charcos	1	0	0	1	0	1	0	1
Medio	Diseño algorítmico	3	Castores	1	0	1	0	1	0	0	1
Alto	Generalización	4	Espías	1	0	1	0	0	1	1	0
	Descomposición	5	Móviles	1	0	1	0	0	1	1	0

Escenario 1. 5 respuestas correctas
El estudiante no tendrá ninguna problemática para acreditar siguiendo el diseño instruccional, realizando las actividades presenciales y las establecidas para la asignatura en línea a través de la plataforma, es decir, uso de las actividades formativas, guía de aprendizaje y ejercicios, así como las evaluaciones formativas y sumativas de cada unidad de aprendizaje. Un escenario alternativo para el estudiante, si le fuera posible, podría ser avanzar el desarrollo de la asignatura a su propio ritmo y solo recibir la guía del docente cuando la solicite. Finalmente, también puede ser un buen candidato para tomar el curso en línea.

Escenario 2. Incorrecto en todos los reactivos
Existe la posibilidad de que el estudiante no tenga habilidades para el estudio de la asignatura y en la unidad de aprendizaje tres sea difícil la comprensión de las estructuras de control y diseño algorítmico al momento de crear algoritmos en pseudocódigo, por lo que se recomienda en reunión presencial docente-directivo-estudiante valorar el perfil vocacional de este último. Es muy seguro que un curso presencial sea la mejor opción para el estudiante.

Escenario 3. Incorrectos los ejercicios de Canguro y Salto de charcos.
El estudiante requiere observación y trabajo presencial del docente para que desde las unidades iniciales adquiera aprendizaje significativo. El estudiante podría acreditar el curso presencial sin inconveniente teniendo en cuenta reforzar en su momento (en la semana 4 del curso), de forma presencial o a través de la plataforma la segunda unidad de aprendizaje, Estructuras de control en DFD, debido a que sería su primer contacto con las estructuras de control (secuenciales, decisión y repetición) para la definición de algoritmos, codificando ejercicios extras a la guía correspondiente que resuelven un problema usando tales estructuras. En caso contrario, posiblemente tenga dificultad desde la unidad de aprendizaje 3 y se dificulte su acreditación.

Escenario 4. Correctos los ejercicios de Canguro y Salto de charcos
El estudiante exhibe habilidades básicas para la actividad creativa del diseño de algoritmos. Posiblemente no tendrá ninguna problemática para acreditar los criterios de evaluación en las primeras tres semanas siguiendo el diseño instruccional, es decir, realizando las actividades presenciales y en línea establecidas para la asignatura en la plataforma. A partir de la cuarta semana, trabajar una atención especial con una práctica y retroalimentación de ejercicios extras en las unidades de aprendizaje dos y tres de forma presencial, así garantizar también la prevención de alguna problemática en las últimas dos unidades de aprendizaje.

Escenario 5. Correcto el ejercicio de Castores en movimiento
El estudiante requiere observación y trabajo presencial del docente para que desde las unidades iniciales adquiera aprendizaje significativo. Habrá que reforzar la evaluación de expresiones aritméticas, lógicas y relacionales considerando la jerarquía de operadores por medio de codificar ejercicios que usen cálculo de operaciones y condicionales en la unidad de aprendizaje dos (semana

4 del curso). Lo anterior para prevenir problemáticas al momento de abordar la unidad de aprendizaje 4 y enfrente sin conflicto el primer incremento de complejidad en el curso.

Escenario 6. Incorrecto el ejercicio de Castores en movimiento
Si le fuera posible al estudiante, podría avanzar el desarrollo de la asignatura a su propio ritmo y solo recibir la guía del docente cuando la solicite a través de la plataforma misma en las primeras dos unidades de aprendizaje, en caso contrario, seguir la planeación indicada en el curso presencial. El estudiante posiblemente tenga problema con la unidad de aprendizaje 3, Estructuras de control con pseudocódigo, por ello sería conveniente reforzar el diseño algorítmico con ejercicios extras a la guía correspondiente a través de la plataforma o de forma presencial cuando se llegue a esta unidad de aprendizaje (en la semana 8 del curso), así garantizar también la prevención de alguna problemática en las últimas dos unidades de aprendizaje.

Escenario 7. Correctos los ejercicios de Espías y Móviles
El estudiante requiere observación y trabajo presencial del docente para que desde las unidades iniciales adquiera aprendizaje significativo, además de atención para que el material del curso y las actividades se realicen puntualmente con base al tiempo determinado en el diseño instruccional, también puede requerir sesiones de asesoría con el profesor, así que deberán ser alentadas para que la confianza del estudiante se incremente respecto a la capacidad de crear programas, y planeadas para que no representen un trabajo extra del docente fuera del horario presencial. En caso contrario, posiblemente tenga dificultad desde la unidad de aprendizaje 3 y se dificulte su acreditación.

Escenario 8. Incorrectos los ejercicios de Espías y Móviles
El estudiante tiene habilidades básicas, pero necesarias para el aprendizaje particularmente de programación de computadoras, por lo que el diseño instruccional creado es favorable, pero con una atención de asesoría presencial regular en las prácticas de laboratorio que diseñe el docente.

Las circunstancias ideales para la intervención del experimento, es aquel donde todos los estudiantes que inician el curso realicen la evaluación del pensamiento computacional en línea, como una actividad durante la primera semana. Posteriormente, determinar los escenarios iniciales que posiblemente enfrentarán los docentes-estudiantes durante el curso a partir de los reactivos correctos que obtuvo cada uno de los jóvenes. Establecer los grupos de control y experimental. El o los grupos experimentales han de realizar las acciones o condiciones preventivas que consideran las recomendaciones de cada escenario para su sección respectiva del profesor. Durante el curso registrar y medir la correspondencia del escenario con los resultados de los criterios de evaluación de los estudiantes en su respectivo momento, es decir, verificar que en los grupos de control se ha de observar un porcentaje de deserción y acreditación en correspondencia con los datos históricos del área Informática y Telecomunicaciones de los últimos 3 a 5 años; para los grupos experimentales, se espera favorecer la motivación y autonomía de estudio para acreditar el curso y en consecuencia aumentar el porcentaje de retención escolar.

Es recomendable realizar dos encuestas para la obtención de datos, las preguntas se indican en la Tabla 17 y Tabla 18, la primera para ser contestada en la semana 9 del curso y la segunda al final del semestre por los estudiantes del o los grupos experimentales. Si los directivos lo consideran pertinente, la propuesta puede ser realizada para el semestre inicial de 2020.

Tabla 17

Encuesta a mitad del semestre para INACAP. Fuente: Elaboración propia

Pregunta	Opción de respuesta
La modalidad de trabajo te parece adecuada con tu expectativa de aprendizaje	Si No
¿Conoces los objetivos a los que tienes que llegar o tienes claro el conocimiento y lo qué debes saber hacer al final del curso?	Si No
¿Te sientes perdido usando la plataforma, no sabes qué hacer y para qué?	Si No
De las actividades contenidas en la plataforma ¿Cuál te aportó más?	Selección múltiple - Evaluación diagnóstica - Actividad formativa - Guía de ejercicios / evaluación formativa - Foro - Evaluación sumativa
¿Qué acción sugieres para mejorar el aprendizaje o estás de acuerdo con tu entorno de aprendizaje?	Abierta

Tabla 18

Encuesta al final del semestre para INACAP. Fuente: Elaboración propia

Pregunta	Opción de respuesta
Selecciona los criterios de aprendizaje con los que estás familiarizado - Selección múltiple	
Identificando datos de entrada, proceso y salida a partir de un problema de procesamiento de datos.	
Aplicando operadores lógicos en la solución de problemas de procesamiento de datos.	
Aplicando estructuras de decisión en la solución del problema.	
Utilizando estructuras de repetición en la solución del problema.	
Considerando la validación de datos en la solución del problema.	
Realizando la traza de la solución propuesta.	
Utilizando operadores lógicos en la construcción de algoritmos.	
Incorporando estructuras de repetición en algoritmos en Pseudocódigo.	
Ingresando datos en arreglos unidimensionales y bidimensionales.	
Realizando un recorrido sobre arreglos unidimensionales o bidimensionales.	
Realizando búsquedas en arreglos unidimensionales o bidimensionales.	
Incorporando paso de parámetros a la subrutina.	

Incorporando el retorno de datos al programa principal. Realizando llamadas a las subrutinas creadas.	
¿La modalidad de aprendizaje fue adecuada para adquirir las competencias del curso?	Si No
¿La evaluación de tus habilidades al inicio del cuatrimestre fue una actividad acertada para determinar el mejor entorno inicial de aprendizaje?	Si No
¿Qué recomendación tienes para las futuras generaciones acerca de la forma de aprender el contenido del curso?	Abierta

DISCUSIÓN

Tomando como referencia las 5 ocasiones en que se ha realizado la evaluación del pensamiento computacional, la Figura 12 ilustra los porcentajes obtenidos de las respuestas correctas por cada intervención en cada uno de los reactivos-habilidades. El diseño algorítmico es la habilidad con mejor porcentaje obtenido en cada año, un promedio del 71,98%. Lo anterior es muy conveniente debido a que es una habilidad clave en la programación de computadoras. Por lo anterior, se tiene una primera aportación de conocimiento generado, es decir, evaluar la habilidad de diseño algorítmico permite detectar aptitudes favorables para los estudiantes que determinan estudiar una carrera a fin a las ciencias de la computación, en el caso de estudio, tecnologías de la información. La descomposición y generalización han obtenido el porcentaje más bajo en dos años diferentes, 29% en el 2018 y 2017 respectivamente, representando a las habilidades que en promedio tienen un 38,66% y 35,54%. Este dato indica las habilidades que mayor trabajo académico requieren los estudiantes para el curso inicial de programación. La habilidad de abstracción y evaluación tienen un favorable promedio de 54,32% y 64,7%. A partir de la relación con los contenidos temáticos del curso MP, el inicio del cuatrimestre no debe representar una problemática para los docentes-estudiantes, pero sí se debe realizar un trabajo de enseñanza eficiente para garantizar el aprendizaje significativo de los estudiantes, es decir, trabajar con la adecuada comprensión de los conceptos tipos de datos, variables e identificadores. Posteriormente, la unidad temática dos (Expresiones) es clave para tener buenos resultados debido a que corresponde con la habilidad de generalización y ha sido la más baja al momento de su evaluación. El aprendizaje significativo de los conceptos operadores aritméticos, lógicos y relacionales, así como la evaluación de expresiones con tales operadores permitirá un mejor desempeño para la unidad temática tres y los temas de condicionales y ciclos. Realizarlo de tal forma, permite abonar la motivación de los estudiantes detectada con el buen resultado de la habilidad de diseño algorítmico. Por lo anterior, una segunda aportación de conocimiento generado es la capacidad que tiene el docente de establecer condiciones iniciales del proceso enseñanza aprendizaje para un curso inicial de programación a partir de la relación de reactivos correctos en correspondencia con habilidades del pensamiento computacional con los contenidos temáticos del curso.

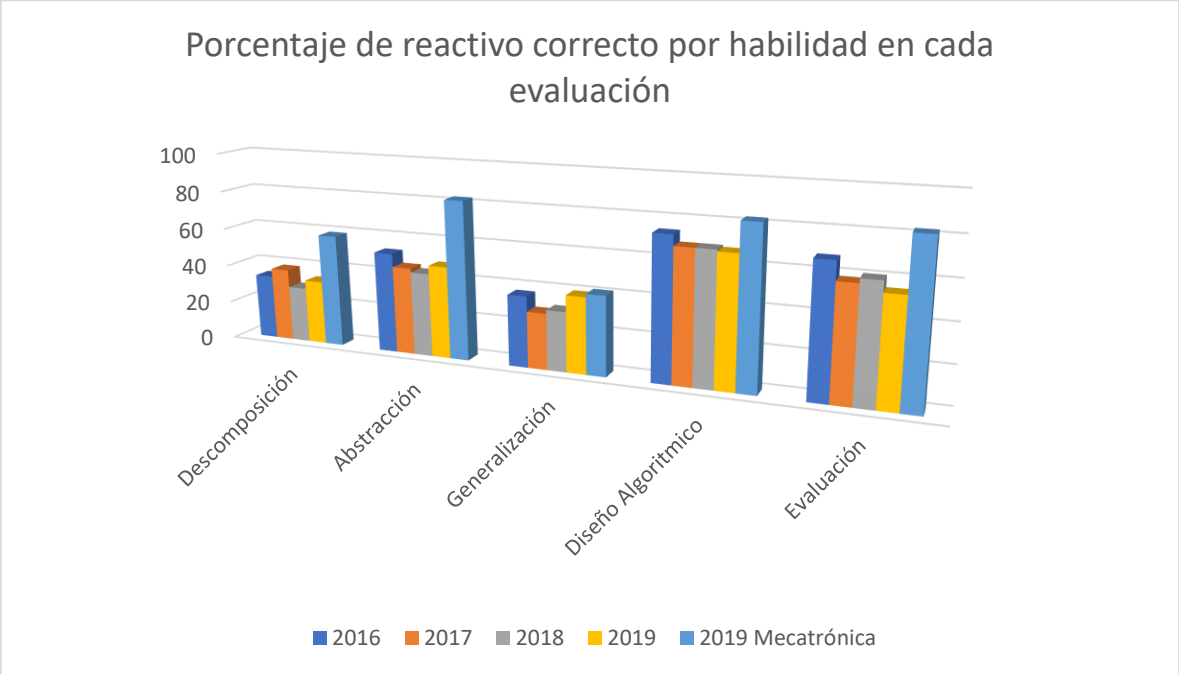

Figura 12. Porcentajes obtenidos de las respuestas correctas por cada intervención en cada uno de los reactivos-habilidades. Fuente: Elaboración propia

Antes de discutir los resultados obtenidos respecto a las competencias que obtuvieron los estudiantes, es conveniente destacar la debilidad del proceso de obtención de datos por medio de la autoevaluación de los estudiantes, es decir, debido a la forma en que perciben los estudiantes que han aprendido algo y lo que realmente saben. En la encuesta realizada en 2016, los estudiantes desconocen estar familiarizados al menos con un concepto, lo que coincide con el hecho de que los porcentajes de las habilidades evaluadas no sean cercanos al 100%. El 73,3% de los estudiantes reconoce estar familiarizado con los conceptos de tipos de datos, operadores lógicos, jerarquía de operadores, uso de variable contador y acumulador, y estructura de repetición; el 66,7% con los conceptos de creación de identificadores, operadores relacionales, resolver expresiones aritméticas, lógicas y relacionales, y la definición y creación de algoritmos; el 60% con el concepto de operadores aritméticos; solo un 26,7% con el concepto de estructura de selección. Los anteriores porcentajes coinciden con la recomendación de los estudiantes para futuras intervenciones donde aconsejan estudiar más, un 46,6% lo indican, la modalidad de aprendizaje no es un problema que ellos perciban, un 86,7% la considera adecuada y solo un 6,7% recomienda que sea la misma forma de aprendizaje para todos.

En la evaluación realizada en el 2017, los resultados de competencias adquiridas son mejores, dos conceptos son 100% familiarizados por los estudiantes, creación de identificadores y estructura de repetición; cinco conceptos tienen el 90% de familiaridad, tipos de datos, operadores aritméticos, operadores lógicos, jerarquía de operadores y uso de variable contador y acumulador; tres conceptos tienen el 80%, operadores relacionales, resolver

expresiones aritméticas, lógicas y relacionales y definición y creación de un algoritmo; nuevamente el concepto estructura de selección tiene el porcentaje más bajo, 60% lo que indica un foco de atención para reforzar su práctica en las sesiones presenciales. La mejoría en los resultados coincide con el hecho de que el 100% de los encuestados considera adecuada la modalidad de aprendizaje para la adquisición de las competencias y no indican una recomendación negativa para las siguientes intervenciones, sus comentarios son de aceptación y motivación por investigar y aprender más de forma autónoma.

La actividad de predicción realizada desde el 2018 debido a la ausencia de curso impartido por el investigador, dio un nuevo uso a la información generada con la evaluación. El porcentaje de retención pronosticado requería de trabajo académico por parte de cada uno de los docentes. En comparación con el obtenido, hacer ver que si se quieren obtener números favorables de matrícula hay que personalizar el aprendizaje de los estudiantes tomando como punto de partida el diagnóstico obtenido que se relaciona con los contenidos temáticos del curso MP. Los anterior es posible debido a que se tiene una pertinente relación de las unidades temáticas del curso y las habilidades del pensamiento computacional, como también lo propuso Romero et al. (2017). Existe motivación de los estudiantes al reconocer sus habilidades del pensamiento computacional, beneficiando su confianza en la capacidad de resolución de problemas, que como en los trabajos de Yinnan y Chaosheng (2012), y Zhang et al. (2011) impacta en el aprendizaje de la programación de computadoras. También, se puede promover aprendizaje autónomo, en semejanza con lo reportado por Gao (2014), Chen (2017) y Ni (2017) que parten de la cultivación del pensamiento computacional.

Finalmente, a pesar de que cuatro generaciones de estudiantes han contestado los reactivos seleccionados en la evaluación, su vigencia de uso se reafirma porque representan una complejidad adecuada al nivel de egreso del nivel pre universitario en la población de estudiantes de Puebla, México aspirantes a estudias en la Universidad Tecnológica de Puebla. La ejecución del diseño experimental creado para INACAP – Chile puede ser realizado para el semestre inicial de 2020, pero requiere de un seguimiento del área de informática y telecomunicaciones y de INACAP online. El interés existe y la disposición del investigador también para obtener datos de la intervención. Los ocho escenarios de aprendizaje para los estudiantes de nuevo ingreso del curso Fundamentos de programación están propuestos para beneficiar el porcentaje de retención del INACAP. Su diseño está basado a partir del estudio de los contenidos de las unidades de aprendizaje y el material disponible en la plataforma, es decir, considerando los criterios de evaluación y el diseño instruccional del curso.

CONCLUSIONES

En esta sección se incluyen los comentarios finales que describen la respuesta a la pregunta de investigación. Desde la primera evaluación se observaron dos resultados conclusivos positivos. En primer lugar, existen estudiantes que sienten motivación por que sean reconocidas sus habilidades y, en segundo lugar, la evaluación permite determinar un diagnóstico acorde con los contenidos de aprendizaje del curso Metodología de la programación para el caso de estudio en México.

Las generaciones de estudiantes evaluados, han reportado un porcentaje promedio de 71,98% para la habilidad de diseño algorítmico, lo que es muy alentador para el docente y una

fortaleza para el estudiante. El porcentaje promedio de 35,54% de la habilidad de generalización puede considerarse como una debilidad, pero hay que tomar en cuenta, que en correspondencia con el contenido del curso, es decir, la unidad de aprendizaje expresiones, refleja que algunos estudiantes aún en nivel pre universitario tienen problemas con la evaluación de expresiones aritméticas, lógicas y relacionales. Por lo anterior, ante tal diagnóstico se debe practicar en cursos propedéuticos la evaluación de tales expresiones. La habilidad de descomposición con un promedio del 38,66% también debe ser una actividad a ser trabajada para favorecer la resolución de problemas complejos. En resumen, es relevante el trabajo realizado porque permite determinar el estado académico de los estudiantes respecto a la resolución de problemas. Así, es posible determinar estrategias educativas que impacte en el aprendizaje de los contenidos temáticos de los cursos iniciales de programación y la adquisición de las respectivas competencias. La mención anterior representa también un conocimiento científico valioso para aquellos docentes que buscan herramientas, estrategias y acciones que mejoren el proceso enseñanza-aprendizaje de la programación de computadoras, docentes que en su contexto concuerdan con el objetivo de investigación.

En INACAP – Chile se realizó una planificación pertinente y con un diseño experimental robusto para su posible ejecución si los directivos lo consideran para el próximo semestre de nuevo ingreso. Por lo anterior, se tiene un resultado muy favorable para la investigación hasta ahora realizada en México, es decir, es posible adaptar y ajustar la evaluación del pensamiento computacional que mejoren los indicadores de retención de Chile con los resultados y diseño experimental de investigación previamente ejecutado en México desde el 2016 al 2018.

Para las futuras intervenciones, tomando como referencia el conocimiento obtenido, se proponen las siguientes acciones para llevar a cabo la evaluación de habilidades del pensamiento computacional:

i. Realizar la evaluación del pensamiento computacional, usando los reactivos previamente elegidos, de preferencia durante los tres días iniciales del cuatrimestre a los grupos donde se quiera obtener el diagnóstico.
ii. Es muy recomendable realizar una encuesta a mitad del cuatrimestre para conocer el resultado de la experiencia que está teniendo el estudiante con su entorno de aprendizaje, a partir del reconocimiento de las habilidades detectadas y acciones de prevención para subsanar la carencia de habilidades que impacte de forma favorable en la adquisición de las competencias.
iii. Realizar la encuesta final para favorecer la actividad de predicción en la matrícula sumada a una encuesta a los profesores que permitiría obtener información acerca de las habilidades adquiridas de los estudiantes, así como mejorar el porcentaje de retención.

Para mejorar los resultados, principalmente en el aprendizaje de los contenidos del curso inicial de programación de computadoras, se debe trabajar con los escenarios de aprendizaje creados para INCAP-Chile con los estudiantes desde un enfoque personalizado.

Presentación del artículo: 14 de enero de 2020
Fecha de aprobación: 10 de marzo de 2020
Fecha de publicación: 30 de abril de 2020

Rojas-López, A. y García-Peñalvo, J.F. (2020). Evaluación de habilidades del pensamiento computacional para predecir el aprendizaje y retención de estudiantes en la asignatura de programación de computadoras en educación superior. *RED. Revista de Educación a Distancia, 20(*63). DOI: http://dx.doi.org/10.6018/red.409991

Financiación
Esta investigación no ha recibido ninguna subvención específica de los organismos de financiación en los sectores públicos, comerciales o sin fines de lucro.

REFERENCIAS

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48-54. doi:10.1145/1929887.1929905

Bureau_of_Labor_Statistics. (2019). *U.S. Department of Labor*. Obtenido de Occupational Outlook Handbook, Software Developers: https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm#tab-6

Chen, G.-m. (2017). Programming Language Teaching Model Based on Computational Thinking and Problem-based Learning. *Proceedings of the 2017 2nd International Seminar on Education Innovation and Economic Management (SEIEM 2017).* Atlantis Press. doi:10.2991/seiem-17.2018.31

Chiazzese, G., Fulantelli, G., Pipitone, V., & Taibi, D. (2017). Promoting computational thinking and creativeness in primary school children. *TEEM 2017 Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality* (pág. Article No. 6). Cádiz, Spain — October 18 - 20: ACM New York, NY, USA. doi:10.1145/3144826.3145354

Chiazzese, G., Fulantelli, G., Pipitone, V., & Taibi, D. (2018). Involucrando a los niños de educación primaria en el Pensamiento Computacional: diseñando y desarrollando videojuegos. *Education in the Knowledge Society, 19*(2), 63-81. doi:10.14201/eks20181926381

Chilana, P. K., Alcock, C., Dembla, S., Ho, A., Hurst, A., Armstrong, B., & Guo, P. J. (2015). Perceptions of non-CS majors in intro programming: The rise of the conversational programmer. *2015 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)* (págs. 251-259). Atlanta, GA, USA: IEEE. doi:10.1109/VLHCC.2015.7357224

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*(11). doi:10.6018/red/46/11

Czerkawski, B. C., & Lyman, E. W. (2015). Exploring Issues About Computational Thinking in Higher Education. *TechTrends, 59*(2), 57–65. doi:10.1007/s11528-015-0840-3

Echegaray, G., Barroso, N., Laskurain, I., Zuza, K., & Barragués, J. I. (2017). Investigación cualitativa para la mejora de los resultados académicos en primer curso en los grados de Ingeniería de la Escuela de Ingeniería de Gipuzkoa. *IV Congreso Internacional sobre Aprendizaje, Innovación y Competitividad-CINAIC (Zaragoza 4-6 Octubre 2017).* doi:10.26754/CINAIC.2017.000001_093

Espino, E. E., & González, C. S. (2015). Influence of Gender on Computational Thinking. *Interacción '15 Proceedings of the XVI International Conference on Human Computer Interaction* (pág. art. no. 36). Vilanova i la Geltru, Spain — September 07 - 09: ACM New York, NY, USA. doi:10.1145/2829875.2829904

Gao, Q. (2014). The computational thinking- oriented inquiry teaching mode for advanced programming language course. *Bio Technology: A Indian Journal, 10*(12), 6588-6595. Obtenido de https://www.tsijournals.com/articles/the-computational-thinking-oriented-inquiry-teaching-mode-for-advanced-programming-language-course.pdf

García-Peñalvo, F. J. (2016). What Computational Thinking Is. *Journal of Information Technology Research, 9*(3), v-viii.

García-Peñalvo, F. J. (2018). Computational thinking. *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje (IEEE RITA), 13*(1), 17-19. doi:10.1109/RITA.2018.280993

García-Peñalvo, F. J., & Cruz-Benito, J. (2016). Computational thinking in pre-university education. En F. J. García-Peñalvo (Ed.), *TEEM '16 Proceedings of the Fourth International Conference on Technological Ecosystems for Enhancing Multiculturality* (págs. 13-17). Salamanca, Spain: ACM, New York, NY, USA. doi:10.1145/3012430.3012490

García-Peñalvo, F. J., & Mendes, A. J. (2018). Exploring the computational thinking effects in pre-university education. *Computers in Human Behavior, 80*, 407-411. doi:10.1016/j.chb.2017.12.005

García-Peñalvo, F. J., Reimann, D., & Maday, C. (2018). Introducing Coding and Computational Thinking in the Schools: The TACCLE 3 – Coding Project Experience. En M. Khine (Ed.). Computational Thinking in the STEM Disciplines, Springer, Cham. doi:10.1007/978-3-319-93566-9_11

García-Peñalvo, F. J., Reimann, D., Tuul, M., Rees, A., & Jormanainen, I. (2016). *An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers.* Belgium: TACCLE 3 Consortium. doi:10.5281/zenodo.16512

García-Vega, M. Á. (2019). Automatización: así es la batalla entre trabajo y tecnología. *Revista Retina*. Obtenido de https://retina.elpais.com/retina/2019/05/24/tendencias/1558680372_855666.html?id_externo_rsoc=TW_CM

Google for Education. (2016). *Exploring Computational Thinking*. Obtenido de https://www.google.com/edu/resources/programs/exploring-computational-thinking/

Hamlin, B., Riehl, J., Hamlin, A. J., & Monte, A. (2010). Work in progress — What are you thinking? Over confidence in first year students. *2010 IEEE Frontiers in Education Conference (FIE)* (págs. F2H1-F2H2). Washington, DC, USA: IEEE. doi:10.1109/FIE.2010.5673354

Huang, W., Deng, Z., & Rongsheng, D. (2009). Programming Courses Teaching Method for Ability Enhancement of Computational Thinking. *2009 International Association of Computer Science and Information Technology - Spring Conference* (págs. 182-185). Singapore, Singapore: IEEE. doi:10.1109/IACSIT-SC.2009.52

Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning Programming at the Computational Thinking Level via Digital Game-Play. *Procedia Computer Science, 9*, 522-531. doi:10.1016/j.procs.2012.04.056

Larkins, D. B., & Harvey, W. (2010). Introductory computational science using MATLAB and image processing. *Procedia Computer Science, 1*(1), 913-919. doi:10.1016/j.procs.2010.04.100

Lingling, Z., Xiaohong, S., & Tiantian, W. (2015). Bring CS2013 Recommendations into c Programming Course. *Procedia - Social and Behavioral Sciences, 176*, 194-199. doi:10.1016/j.sbspro.2015.01.461

Lye, S. Y., & Koh, J. H. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51-61. doi:10.1016/j.chb.2014.09.012

Mazaika, K. (2017). Will The Demand For Developers Continue To Increase? *Forbes*. Obtenido de https://www.forbes.com/sites/quora/2017/01/20/will-the-demand-for-developers-continue-to-increase/#1272429e33ee

Michaelson, G. (2015). Teaching Programming with Computational and Informational Thinking. *Journal of Pedagogic Development, 5*(1). Obtenido de https://www.beds.ac.uk/jpd/volume-5-issue-1-march-2015/teaching-programming-with-computational-and-informational-thinking

Milne, I., & Rowe, G. (2002). Difficulties in Learning and Teaching Programming—Views of Students and Tutors. *Education and Information Technologies, 7*(1), 55-66. doi:10.1023/A:1015362608943

Ni, Z. (2017). Discussion on Case Teaching Method Based on Computational Thinking in Programming Teaching. *2017 International Conference on Social science, Education and Humanities Research (ICSEHR 2017).* Atlantis Press. doi:10.2991/icsehr-17.2017.9

Olivares, J. C., Jiménez, J. A., Ortiz, O., & Rodríguez, N. (2015). Desarrollo de una aplicación para fortalecer el aprendizaje de los fundamentos de programación. *Revista de ciencia e ingeniería del instituto tecnológico superior de Coatzacoalcos, 2*(2).

Pólya, G. (1957). *How to solve it.* United States of America: Anchor books edition.

Ramirez-Lopez, A., & Muñoz, D. F. (2015). Increasing Practical Lessons and Inclusion of Applied Examples to Motivate University Students during Programming Courses. *Procedia - Social and Behavioral Sciences, 176*, 552-564. doi:10.1016/j.sbspro.2015.01.510

Rojas-López, A., & García-Peñalvo, F. J. (2018). Learning Scenarios for the Subject Methodology of Programming from Evaluating the Computational Thinking of New Students. *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje, 13*(1), 30-36. doi:10.1109/RITA.2018.2809941

Román, M., Pérez, J. C., & Jiménez, C. (2015). Test de Pensamiento Computacional: diseño y psicometría general Computational Thinking Test: design & general psychometry. *III Congreso Internacional sobre Aprendizaje, Innovación y Competitividad (CINAIC 2015), Octubre 14-16.* Madrid, ESPAÑA.

Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education*, 14-42. doi:10.1186/s41239-017-0080-z

Royal Society. (2012). *Shut down or restart: The way forward for computing in UK schools.* Obtenido de https://royalsociety.org/~/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf

Sakhnini, V., & Hazzan, O. (2008). Reducing Abstraction in High School Computer Science Education: The Case of Definition, Implementation, and Use of Abstract Data Types. *Journal on Educational Resources in Computing (JERIC), 8*(2), 1-13. doi:10.1145/1362787.1362789

Selby, C. C. (2015). Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy. *WiPSCE '15 Proceedings of the Workshop in Primary and Secondary Computing Education* (págs. 80-87). London, United Kingdom: ACM New York, NY, USA. doi:10.1145/2818314.2818315

Shiflet, G. W., & Shifleta, A. B. (2012). Introducing Life Science Doctoral Students in Oz to the Wizardry of Computational Modeling: Introducing Computational Thinking with CellDesigner™. *Procedia Computer Science, 9*, 1753-1762. doi:10.1016/j.procs.2012.04.193

Swaid, S. I. (2015). Bringing Computational Thinking to STEM Education. *Procedia Manufacturing, 3*, 3657-3662. doi:10.1016/j.promfg.2015.07.761

Talent Search. (2015). *Elite: Grade 12+, Institute of IT Professionals South Africa.* Obtenido de http://www.olympiad.org.za

UK Bebras Computational Thinking Challenge. (2015). *answers (2015), University of Oxford.* Obtenido de http://www.bebras.org

Walker, H. M. (2015). Computational thinking in a non-majors CS course requires a programming component. *ACM Inroads, 6*(1), 58-61. doi:10.1145/2727126

Weese, J. L. (2016). Mixed Methods for the Assessment and Incorporation of Computational Thinking in K-12 and Higher Education. *ICER '16 Proceedings of the 2016 ACM Conference on International Computing Education Research* (págs. 279-280). Melbourne, VIC, Australia — September 08 - 12: ACM New York, NY, USA. doi:10.1145/2960310.2960347

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35. doi:10.1145/1118178.1118215

Wing, J. M. (2011). *Research Notebook: Computational Thinking--What and Why?* Obtenido de The Link, The magazine of the Carnegie Mellon University School of Computer

Science: https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why

Xia, A. (2016). On the Basis of the Program Design Teaching and Research of Cultivation of Computational Thinking Ability. *Proceedings of the 2016 2nd International Conference on Social Science and Higher Education.* Atlantis Press. doi:10.2991/icsshe-16.2016.83

Ying, L., & Pingping, L. (2017). Research on the teaching of programming language based on Computational Thinking. *Proceedings of the 2017 International Conference on Social science, Education and Humanities Research (ICSEHR 2017).* Atlantis Press. doi:10.2991/icsehr-17.2017.17

Yinnan, Z., & Chaosheng, L. (2012). Training for computational thinking capability on programming language teaching. *2012 7th International Conference on Computer Science & Education (ICCSE)* (págs. 1804-1809). Melbourne, VIC, Australia : IEEE. doi:10.1109/ICCSE.2012.6295420

Zapata-Ros, M. (2015). Pensamiento computacional: Una nueva alfabetización digital. *Revista De Educación a Distancia, 46*. Obtenido de https://revistas.um.es/red/article/view/240321

Zhang, C., Chen, X., & Li, J. (2011). Research of VB programming teaching mode based on the core of computational thinking ability training. *2011 6th International Conference on Computer Science & Education (ICCSE)* (págs. 1260-1263). Singapore, Singapore: IEEE. doi:10.1109/ICCSE.2011.6028861

Zhi-Mei, C., & Xiang, L. (2016). The PBL teaching method research based on computational thinking in C programming. *2nd International Conference on Modern Education and Social Science,* (págs. 405-409). doi:10.12783/dtssehs/mess2016/9624