Pérez, L. (1997). La evaluación dentro del proceso enseñanza- aprendizaje. Citado por: Cruz Núñez, F. y Quiñones Urquijo, A. *Importancia de la evaluación y autoevaluación en el rendimiento académico. Zona Próxima.* Revista del Instituto de Estudios en Educación Universidad del Norte. (16), pp. 96-104.

Rodríguez Illera (2001) *Aprendizaje colaborativo en entornos virtuales.* Anuario de Psicología, 32 (2), pp. 63-75.

Sáenz Adán, C. (2015). *Apoyo del aprendizaje significativo en matemáticas a través de la gamificación* (tesis de maestría). Universidad de La Rioja, Logroño, La Rioja. España.

Abstract: This experience is part of a research project related to the analysis of the discourse of science in mathematics and the most efficient ways to communicate it to students. Students of the subject Calculus I, from the first year of the different engineering careers of the Faculty of Engineering and Water Sciences of the Universidad Nacional del Litoral, are circumscribed. Currently, the impact that ICTs have on teaching and learning processes is undeniable. In a teaching context characterized by high technological availability, an aulic experience is described referring to the implementation of a didactic proposal linked to gamification, in a non-linear teaching mode, which recovers values of collaborative learning, through a practice motivating, entertaining, challenging and enriching that aims to stimulate the reach of knowledge rich in significance by the student, thus favoring an improvement in their academic performance.

Keywords: Didactic proposal - learning - gamification - motivation

Resumo: Essa experiência faz parte de um projeto de pesquisa voltado para a análise do discurso científico em matemática e das formas mais eficientes de comunicá-la aos alunos.
Limita-se aos alunos da disciplina Cálculo I, desde o primeiro ano das diferentes carreiras de engenharia da Faculdade de Engenharia e Ciências da Água da Universidade Nacional do Litoral. O impacto das TIC nos processos de ensino e aprendizagem é atualmente inegável.
Em um contexto de ensino caracterizado pela alta disponibilidade tecnológica, uma experiência de sala de aula relacionada à implementação de uma proposta didática ligada à gamificação é descrita, em um modo de ensino não linear, que recupera valores aprendizagem colaborativa, através de uma prática motivadora, divertida, desafiadora e enriquecedora que visa estimular o alcance do conhecimento rico em importância por parte do aluno, favorecendo assim uma melhoria em seu desempenho acadêmico.

Palavras chave: Proposta Didática - aprendizagem - gamification - motivação

(*) **Natalia Bas.** Técnica en informática. Bachiller universitario en Letras. Tesista de la Licenciatura en Letras de la Facultad de Humanidades y Ciencias (UNL). Coordinadora del Área de Educación a Distancia de la Facultad de Ingeniería y Ciencias Hídricas (UNL). Docente a cargo del Taller de Tecnologías Digitales de la Escuela Primaria de la (UNL).

(**) **Mario Garelik.** Licenciado en Matemática Aplicada (Facultad de Ingeniería Química, UNL). Magister en Didácticas Específicas (Facultad de Humanidades y Ciencias, UNL). Profesor y Miembro del Consejo Directivo de la Facultad de Ingeniería y Ciencias Hídricas, UNL.

Computación en Ingeniería: la experiencia de pensar para crear juntos

Fecha de recepción: agosto 2019
Fecha de aceptación: octubre 2019
Versión final: diciembre 2019

Elizabeth Jiménez Rey (*)

Resumen: La concepción pedagógica de enseñanza y aprendizaje de Computación en Ingeniería se focaliza en la resolución de problemas con la computadora (algoritmos y programas). Las fases algorítmicas, Análisis del problema y Diseño de la solución, constituyen el desafío creativo de aprendizaje; Codificación del algoritmo y Evaluación del programa, son fases de programación. Los alumnos experimentan el pensar: en aula presencial, con todos los participantes y el docente para construir el conocimiento nuevo, y en aula virtual, en grupos de tres integrantes para ingeniar los algoritmos que solucionan los problemas propuestos acompañados por el docente quien interviene para modelar el pensamiento.

Palabras clave: Algoritmos – programas - pensamiento computacional – creatividad – colaboración

[Resúmenes en inglés y portugués en la página 107]

Contextualización de la Asignatura

El objetivo general de la enseñanza de Computación en dos cursos de la Facultad de Ingeniería de la Universidad de Buenos Aires FIUBA es solucionar problemas con la computadora en coincidencia con uno de los objetivos centrales de las Ciencias de la Computación.

La asignatura Computación:

- Es básica por Resolución Ministerial del año 2001, es decir, sus contenidos abarcan conocimientos comunes a las ingenierías en general;
- Es obligatoria para los estudiantes de las ingenierías no informáticas: Civil, Electricista, Industrial, Química,

Mecánica, Naval, de Alimentos, en Agrimensura, en Petróleo;
- Es cuatrimestral, es decir, se desarrolla en 16 clases, 1 por semana de 4 horas de duración;
- Tiene una única asignatura correlativa posterior, Análisis Numérico I, y ninguna correlativa anterior;
- La cursan los estudiantes que recién ingresan a la FIUBA, en general, luego de haber aprobado todas las asignaturas del Ciclo Básico Común CBC.

El enfoque de la enseñanza y el aprendizaje es procedimental. Se enfatiza no solamente el saber hacer sino también el saber por qué se hace. Por eso, el desarrollo de los contenidos teóricos y prácticos se aborda desde tres núcleos centrales -algoritmo, programa y computadora- en forma interrelacionada, iterativa e incremental. El eje estructurante del curso se organiza en espiral y en cada una de las dieciséis clases el descubrimiento de algoritmos y el desarrollo de programas se complejiza y el conocimiento de la computadora se profundiza.

Modelos para Enseñar a Crear
La enseñanza de Computación se fundamenta en el Modelo prescriptivo de Solución de Problemas del matemático George Polya (Nickerson, Perkins y Smith, 1987) un proceso evolutivo que distingue cuatro fases:

- Comprender el problema.
- Idear un plan implica formular una estrategia general (proceso inductivo).
- Llevar a cabo el plan implica probar la estrategia general (proceso deductivo).
- Evaluar la solución implica verificar los resultados.

Con una visión sistémica, inspirados en el Modelo de Polya, se diseñó un Modelo Evolutivo de Enseñanza, Aprendizaje y Evaluación (Jiménez Rey, 2011) que denominamos CRAC, siglas de Comprensión, Reflexión, Acción y Comunicación, procesos componentes de una trama educativa compleja para generar en el estudiante un pensamiento vinculante, que no aísle y separe los procesos sino que los distinga y los una.
Para solucionar problemas con la computadora los estudiantes utilizan el Modelo de Solución de Problemas de Polya aplicado al ámbito de la construcción de programas que consta de cuatro fases:

- Análisis. Los estudiantes deben comprender en qué consiste el problema a resolver con la construcción del programa.
- Diseño. Los estudiantes deben diseñar una estrategia para definir recursos y descubrir el algoritmo.
- Codificación. Los estudiantes deben implementar la estrategia para construir el programa.
- Evaluación. Los estudiantes deben ejecutar el programa construido para comprobar que la solución es correcta.

El Análisis del problema y el Diseño de la solución, son las fases algorítmicas y constituyen el desafío creativo de aprendizaje. La Codificación y la Evaluación son las fases de programación.

Cada una de las fases del Modelo de Solución de Problemas con la Computadora se expande en una Guía para la Creación de Programas (Jiménez Rey y Perichinsky, 2006) en forma de Mapa Conceptual, construida para ayudar a los estudiantes a descubrir un algoritmo que solucione el problema propuesto y a codificarlo como programa, y en la cual se detalla en qué consiste cada fase del proceso de construcción de un programa.
Los estudiantes utilizan como dispositivo didáctico para la escritura del programa una plantilla que constituye un Modelo de Programa Tipo en Python. Las fases componentes del Modelo de Solución de Problemas de Polya aplicado a la creación de programas están embebidas en el texto del programa construido a través de las secciones declarativa (análisis y diseño), algorítmica (diseño y codificación) y evaluativa (evaluación).

Desafíos de Aprendizaje
La mayor dificultad para los estudiantes cuando deben solucionar un problema con la computadora reside en ingeniar un algoritmo que solucione el problema. En Computación, el algoritmo es el procedimiento que permite resolver un problema y el programa es la expresión del algoritmo en un lenguaje de programación para que la computadora pueda ejecutarlo.
El mayor desafío cognitivo al que se enfrentan los estudiantes de Computación al abordar la solución de problemas con la computadora es el descubrimiento de un algoritmo efectivo (eficaz y eficiente) que solucione un problema propuesto. Las principales causas de las dificultades detectadas en la programación de computadoras son (Bruno, 2005): 1. Los alumnos no están habituados al uso de la lógica para resolver problemas. 2. La programación de algoritmos representa un caso de resolución de problemas que requiere representación mental del mundo real (abstracción), adaptación para tener una solución computable (sintaxis y semántica del lenguaje de programación) y criterio para elegir una alternativa eficiente de implementación (toma de decisión). Y a estas se agregan, entre otras: a. Motivación secundaria (aprobar) dominante pues los alumnos son estudiantes de carreras de ingenierías no informáticas. b. Expectativas de baja exigencia de cursado y de dedicación. c. Deficiente organización para el estudio y planificación de cursado de otras asignaturas.
Para idear un plan, formular una estrategia general o diseñar la solución, los estudiantes deben descomponer el problema propuesto en una secuencia de subproblemas cada vez más simples hasta el último nivel de descomposición en el que los subproblemas se puedan expresar en un lenguaje de programación para codificar un programa a ejecutar por la computadora y así solucionar el problema con la computadora. Es decir, deben diseñar y representar el algoritmo aplicando el principio de dividir y conquistar y el método de refinamientos sucesivos.
El desarrollo de las competencias de orden superior, análisis, síntesis y creación impactan en la formación del estudiante cuando se centra la mirada del alumno en la búsqueda de la solución de un problema con la computadora (competencia general) y se la descentra de la reducción de la búsqueda a la implementación de una

secuencia de instrucciones para desarrollar un programa (competencia específica) sin una comprensión auténtica del problema que se está resolviendo (Morin, 2001). Es decir, cuando se conjugan y se distinguen pero no se aíslan, la Algoritmia (descubrimiento de un algoritmo en la plataforma Google Drive, un lugar para algoritmiar) y la Programación (representación de un algoritmo en forma de programa en el Entorno Integrado de Desarrollo y Aprendizaje IDLE Python, un lugar para programar).

Experimentar el pensar para crear con otros

Se exploran nuevas formas de enseñar para favorecer en los estudiantes de ingeniería la solución de problemas mediante algoritmos y programas. Para enseñar a solucionar problemas con la computadora se plantea la dicotomía de solamente enseñar a programar con dificultad progresiva o también favorecer el pensamiento algorítmico y computacional (Zapata Ros, 2015).

Jeannette Wing (2006) utilizó el término "pensamiento computacional" para articular una visión en la que todos, no solo los especialistas en Informática, pueden beneficiarse de pensar como un científico informático. Según Wing (2010), de manera informal, el pensamiento computacional describe la actividad mental al formular un problema para admitir una solución computacional. La solución puede ser realizada por un humano o una máquina, o más en general, por combinaciones de humanos y máquinas; se superpone con el pensamiento lógico y el pensamiento sistémico; incluye el pensamiento algorítmico y el pensamiento paralelo, que a su vez involucran otros tipos de procesos de pensamiento.

El Aula de Computación está constituida por espacios de clases presenciales, Aula real, Lab E, y espacios de clases virtuales, Aula virtual, Google Drive, donde los estudiantes aprenden a través de la experiencia de pensar para crear con otros (Swartz, Costa, Beyer, Reagan y Kallick, 2008). En Aula real, todos los participantes piensan de manera sincrónica en el gran grupo constituido por sus pares y el profesor para construir el conocimiento nuevo a partir del conocimiento previo. Y en Aula virtual, los estudiantes piensan de manera asincrónica en los pequeños grupos de tres integrantes distribuidos en talleres propios para ingeniar los algoritmos que solucionan los problemas propuestos acompañados por el profesor quien interviene para modelar el pensamiento.

El profesor utiliza rutinas de pensamiento (Ritchhart, Church y Morrison, 2014) en Aula real y Aula virtual durante el proceso de descubrimiento de algoritmos. Una rutina de pensamiento promueve movimientos específicos de pensamiento e implica un proceso que al ponerse en práctica permite que el pensamiento de los estudiantes se haga visible a medida que expresan sus ideas, debaten y reflexionan en torno a ellas.

Para sintetizar y organizar ideas se utiliza principalmente, entre otras, la rutina Conectar-Ampliar-Desafiar porque ofrece una estructura en la que el pensamiento de los estudiantes inspirado por una nueva situación problemática se puede hacer visible. Se interpela a los estudiantes y a través de preguntas, se provoca la escucha activa en lugar del oír pasivo y se los prepara para que tomen conciencia de la nueva experiencia de aprendizaje estableciendo conexiones (Conectar) entre el conocimiento nuevo y el conocimiento previo, identificando nuevas ideas para pensar en nuevas direcciones (Ampliar) y buscando cómo las nuevas ideas desafían a pensar en nuevas maneras de solucionar problemas (Desafiar). Esta rutina permite crear oportunidades para que los estudiantes puedan escudriñar con una mirada pensante lo que necesitan comprender.

Para explorar las ideas más profundamente se utiliza principalmente, entre otras, la rutina Afirmar-Apoyar-Cuestionar porque ofrece una estructura abarcadora para examinar las ideas y generar nueva comprensión. Los estudiantes buscan patrones, detectan generalizaciones e identifican aseveraciones (Afirmar), el profesor les ayuda a que tomen conciencia de sus afirmaciones y se focalicen en las evidencias (Apoyar) y formula preguntas para que prueben y validen sus interpretaciones (Cuestionar).

Tanto en Aula real como en Aula Virtual los estudiantes trabajan en tablas sistematizadas como dispositivos didácticos para favorecer el pensamiento computacional necesario para el descubrimiento de un algoritmo solución de un problema propuesto. Las descripciones de cada columna son precisas para facilitar a los estudiantes la comprensión de las acciones a realizar. Las tablas son dinámicas porque las orientaciones del profesor en cada columna cambian con las características del problema a resolver.

En la Tabla Análisis del Problema los estudiantes se focalizan en las especificaciones del enunciado del problema como conocimiento previo y trabajan con valores de los datos para:

- Seleccionar los casos representativos (columna1) del problema, es decir, numerar la cantidad de casos distintos que requerirán del procesador la ejecución de procedimientos diferentes para obtener los resultados;
- Establecer el estado inicial (columna2), es decir, seleccionar los valores de los datos de entrada en correspondencia con cada caso representativo que ayuden a pensar cómo proceder para solucionar el problema;
- Establecer el estado final (columna3), es decir, expresar los valores del resultado de salida esperado en correspondencia con los valores representativos elegidos;
- Determinar el estado intermedio de transformación (columna 4), es decir, relacionar el estado final del problema con el estado inicial para hallar una secuencia de operaciones que definen una vía de solución para la obtención del resultado en cuestión y representar cada uno de los estados intermedios de transformación desde el estado inicial hasta el estado final.

En la Tabla Diseño de la Solución los estudiantes amplían el conocimiento del problema y avanzan a la primera columna de esta Tabla considerando la última columna de la Tabla anterior para:

- Determinar el estado genérico de adaptación (columna 1), es decir, representar con identificadores de recursos en forma genérica los valores de cada uno de los estados intermedios de transformación, observar y pensar computacionalmente para escudriñar el procedimiento algorítmico que generalice la solución del problema cualesquiera sean los datos;

- Realizar la descomposición del problemas en subproblemas (columna 2), es decir, organizar una secuencia de descripción de los subproblemas a resolver desde la obtención de los datos hasta la comunicación de resultados y analizar la condición o condiciones que gobiernan operaciones selectivas y/o repetitivas y que determinan el conjunto de órdenes necesarias al procesador para lograr la generalización;
- Detectar la naturaleza de los subproblemas (columna 3), es decir, secuencial, selectiva simple, selectiva múltiple o repetitiva;
- Deleccionar las primitivas de programación (columna 4) apropiadas en lenguaje Python para implementar cada uno de los subproblemas.

Pensar la experiencia para mejorar el aprendizaje

La sistematización y formalización de las fases algorítmicas en las tablas Análisis del Problema y Diseño de la Solución tanto en Aula real como en Aula virtual permite al profesor a través de la conversación detectar el pensamiento y las ideas importantes de los estudiantes en el contexto de aprendizaje y destacar los conceptos y prácticas necesarios en el proceso de descubrimiento de un algoritmo. En la elaboración de las tablas están embebidas las principales rutinas que estructuran, movilizan y visibilizan el pensamiento computacional que asegura la calidad de diseño de un algoritmo para solucionar el problema propuesto con la computadora.

Inspirados en Mariana Maggio (2018) quien imagina el futuro de la educación .co (punto co, no punto com) como una construcción colectiva y considerando la formas de enseñar y aprender los contenidos de la asignatura en dos cursos nos animamos a denominarla Computación.Co porque su práctica educativa implica Construir, Conectar, Comprender, Comunicar, Conversar, Compartir, Colaborar, Cooperar, Concebir, Codificar, Computar, entre otras acciones que deben desplegar los estudiantes.

Referencias Bibliográficas

Bruno, O. (2005) *Análisis de la percepción de los alumnos y de los docentes para la incorporación de un sistema tutor inteligente como facilitador del aprendizaje de algoritmia*. Revista de Informática Educativa y Medios Audiovisuales, 2 (4), 1-31. ISSN 1667-8338 © LIE-FI-UBA.

Jiménez Rey, E. (2011) Enseñanza y Aprendizaje de Computación en Carreras de Ingeniería: una visión sistémica de los procesos esenciales y sus vinculaciones con las NTICs. *I Congreso Argentino de Tecnología de Información y Comunicaciones.* Las Nuevas Tecnologías en la Sociedad de la Información y del Conocimiento. Buenos Aires: COPITEC.

Jiménez Rey, E. y Perichinsky, G. (2006) El Mapa Conceptual como Representación del Modelo de Polya para la Creación de Programas. *VIII Workshop de Investigadores en Ciencias de la Computación.* Recuperado de:

Maggio, M. (2018) *Reinventar la clase en la universidad.* Buenos Aires: Paidós.

Morin, E. (2001) *Los siete saberes necesarios para la educación del futuro.* Buenos Aires: Nueva Visión.

Nickerson, R. S., Perkins, D. N. y Smith, E. E. (1987) *Enseñar a pensar. Aspectos de la aptitud intelectual.* Barcelona: Paidós.

Ritchhart, R., Church, M. y Morrison, K. (2014) *Hacer visible el pensamiento.* Buenos Aires: Paidós.

Swartz, R.; Costa, A.; Beyer, B.; Reagan, R. y Kallick, B. (2008) *El aprendizaje basado en el pensamiento. Cómo desarrollar en los alumnos las competencias del siglo XXI.* Nueva York: Ediciones SM.

Wing, J. (2006) "Computational Thinking". En: *Communications of the ACM.* Marzo. Vol. 49, n°. 3, 33-35.

Wing, J. (2010) *Computational Thinking: What and Why?* Recuperado de: https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

Zapata Ros, M. (2015) *Pensamiento Computacional: Una nueva alfabetización digital.* RED Revista de Educación a Distancia, 46(4). Recuperado de: https://www.um.es/ead/red/46/

Abstract: The pedagogical conception of teaching and learning of Computer Engineering is focused on solving problems with the computer (algorithms and programs). The algorithmic phases, Analysis of the problem and Design of the solution, constitute the creative learning challenge; Algorithm coding and program evaluation are programming phases. Students experience thinking: in classroom, with all participants and the teacher to build new knowledge, and in virtual classroom, in groups of three members to engineer the algorithms that solve the proposed problems accompanied by the teacher who intervenes to model the thought.

Keywords: Algorithms - programs - computational thinking - creativity – collaboration

Resumo: A concepção pedagógica do ensino e da aprendizagem da ciência da computação em engenharia centra-se na resolução de problemas com o computador (algoritmos e programas). As fases algorítmicas, Análise do problema e Design da solução, constituem o desafio da aprendizagem criativa; Codificação de algoritmos e avaliação de programas são fases de programação. Os alunos experimentam o pensamento: em sala de aula cara a cara, com todos os participantes e o professor para construir novos conhecimentos, e em sala de aula virtual, em grupos de três membros para projetar os algoritmos que resolvem os problemas propostos, acompanhados pelo professor que intervém para modelar o pensamento.

Palavras chave: Algoritmos - programas - pensamento computacional - criatividade - colaboração

(*) **Elizabeth Jiménez Rey.** Docente Investigadora en el Departamento de Computación (Facultad de Ingeniería, UBA). Ingeniera Civil. Especialista en Ingeniería de Sistemas. Diplomada en Inteligencia y Pedagogía Compleja. Maestranda en Tecnología Informática Aplicada en Educación (Facultad de Informática, Universidad Nacional de La Plata).