Revista mexicana
de bachillerato a distancia

Video por invitación

Pensamiento computacional: un asunto crítico en el bachillerato

Claudia Urrea

Computational thinking: a critical issue in high school

Resumen

En este video Claudia Urrea entreteje aportes de Seymour Papert, Edith Ackermann y Mitch Resnick con sus propias ideas para hablar sobre el pensamiento computacional (PC), y su importancia crítica en el nivel bachillerato. Señala que el PC no es un fin en sí mismo, sino una poderosa forma de pensamiento. Constituye, a la vez, una forma de pensar y una estrategia para desarrollar habilidades de pensamiento, solución de problemas, creatividad e innovación. Esta forma de pensar, se desarrolla a través de diversas actividades, como la composición digital o la creación de videos, pero aún más a partir de la programación. Destaca la relevancia del contacto temprano con la tecnología y, en el caso de los adolescentes, resulta trascendente al pasar "de ser una simple herramienta a una forma de pensar, a un futuro en la vida".

Palabras clave: pensamiento computacional, fluidez tecnológica, programación, Seymour Papert, Edith Ackermann, Mitch Resnick.

Abstract

In this video, Claudia Urrea intertwines contributions of Seymour Papert, Edith Ackermann and Mitch Resnick with her own ideas to talk about computational thinking (CT) and its critical relevance at the high school level. She points out that CT is not an end in itself, but rather the means towards developing a powerful way of thinking . It is, at once, a way of thinking and a strategy to develop skills to think, solve, create or innovate. It develops by means of diverse activities, like digital composition or the creation of videos, but even more through coding. She underscores the importance of early contact with technology and how it is critical in the case of teenagers for it goes from being "a simple tool to a way of thinking, to a future in life".

Keywords: computational thinking, technological fluidity, coding, Seymour Papert, Edith Ackermann, Mitch Resnick.

Video de la doctora Claudia Urrea: https://youtu.be/9HoxqIpM6bQ

Transcripción del video[1]

Mi nombre es Claudia Urrea. Soy colombiana y estoy en el Instituto Tecnológico de Massachusetts. Quiero compartir con ustedes algunas ideas sobre el pensamiento computacional. Voy a hacer referencia a dos personas -o varias personas- con las que he trabajado y creo que vale la pena revisitar algunas ideas que tienen que ver con el pensamiento computacional. Nosotros desde hace mucho tiempo en el MIT; digo "nosotros" porque considero a mi familia de Seymour Papert mi grupo, y me refiero a "nosotros" cuando hablo de ellos.

Seymour Papert [y Mitch Resnick] hablaban no de pensamiento computacional, pero sí de fluidez tecnológica y yo hago esta referencia desde 1993, cuando ellos escriben un artículo que fue una aplicación a un grant de la Fundación Nacional de la Ciencia de Estados Unidos, donde habla de la fluidez tecnológica como se habla del lenguaje. No tiene que ver con aprender las tecnologías, pero sí con usar las tecnologías de una forma fluida. Ellos hablaban de la comunicación: cómo utilizar el lenguaje para comunicarse; utilizar las tecnologías para hacer diferentes cosas, no como un fin, sino como un medio. Y ser fluido quiere decir entenderlas; poder usarlas para hacer algo concreto. Entonces parte de nuestra influencia en el pensamiento computacional tiene que ver con esta idea de fluidez tecnológica que seguimos todavía referenciando.

El término "pensamiento computacional" viene de una ex alumna del MIT que se llama Jeannette Wing.[2] Ella habla del pensamiento computacional como un proceso de pensamiento que tiene que ver con formulación de problemas, formulación de soluciones: crear, inventar. Es una forma de pensar que tiene mucho paralelo con esta idea de que la tecnología no es un fin; no es saber programar; es saber pensar. De ahí hay mucha controversia cuando se han planteado estándares y formas de medir el pensamiento computacional, lo cual a su vez ha influenciado el cómo llegar a estas habilidades de pensamiento computacional que se consiguen a través de la forma concreta de la computación.

En el término "pensamiento computacional" yo veo dos cosas: una es esta definición de forma de pensar; pero es también una estrategia de cómo alcanzar y cómo desarrollar estas habilidades de pensamiento, de solución, de creatividad, de innovación, y hay varias formas de lograrlo.

[1] Con algunos ajustes de estilo, para pasar de lengua oral a escrita, y añadidos a la versión en video, hechas por la doctora Urrea.

[2] Computational Thinking. http://www.cs.cmu.edu/~./15110-s13/Wing06-ct.pdf

Antes de hablar de la computación, quiero hablarles de dos formas en particular. Esto me trajo memorias de una conversación con una colega, que ya murió y que es parte de nuestra familia de MIT, Edith Ackermann,[3] sobre cómo llegar a esta forma de pensamiento porque programar no es la única forma de hacerlo. Les doy un par de ejemplos que me encantan: uno es la composición digital. Para mí, comunicarse a través de un escrito es casi como programar. Yo quiero comunicar una idea y lo hago de diferentes formas; pienso en la estrategia de comunicación. El compartirlo, el tener feedback sobre eso me está garantizando que puedo escribir algo, compartirlo; tener una retroalimentación de algunas personas y corregirlo. Tengo esa oportunidad de compartir, recibir retroalimentación y mejorar mi escrito para llegar a comunicar lo que yo quiero. Sin embargo, hay diferentes estilos y las personas pueden interpretarlo de diferentes formas. Es una manera, también, de llegar a formas de pensamiento crítico.

Otra forma que recuerdo que hablamos con Ackermann era el crear un video, que también es una forma de comunicación, de pensar. Si quiero comunicar una idea, edito, pongo música, pongo diferentes estrategias que hacen a mi video ser más exitoso. De alguna manera, el éxito de esa herramienta, como el video, puede ser interpretada de diferentes modos.

Ahora llegamos a la parte de programar como una herramienta, un mecanismo para llegar a esa forma de pensamiento computacional, que para mí es la más efectiva, porque yo programo, creo algo y el feedback me lo da el programa que estoy construyendo. Voy a mencionar a Scratch[4] porque es una de las herramientas que hemos creado en el MIT, que es bastante popular. Hay otras como App Inventor,[5] está StarLogo Nova.[6]

En la programación, divido mi programa en diferentes partes; construyo y corro mi programa y ya sé si me está funcionando o no; ese feedback es inmediato. Esa idea de iteraciones es inmediata y me ayuda a desarrollar esa forma de pensamiento. Para mí, nuevamente, el pensamiento computacional comunica dos ideas: la forma de pensar y el mecanismo con el cual me apoyo para desarrollar esa forma de pensamiento. El verdadero impacto no es solo el saber programar y adquirir esa fluidez tecnológica de la que ya les hablaba, sino el desarrollo de habilidades que son importantes para la vida. Estas se

[3] Edith Ackermann. https://web.media.mit.edu/~edith/

[4] Scratch is a free programming language and online community where you can create your own interactive stories, games, and animations. https://scratch.mit.edu/

[5] MIT App Inventor is an intuitive, visual programming environment that allows everyone, even children, to build fully functional apps for smartphones and tablets. https://appinventor.mit.edu/

[6] StarLogo Nova is an agent-based game and simulation programming environment that combines an easy-to-use blocks-based programming language. https://www.slnova.org/

manifiestan en todo lo que hacemos, en cómo resolvemos problemas, en cómo organizamos información, en cómo tomamos decisiones en la vida, para mencionar algunas.

¿Por qué eso es importante? Casualmente acabo de salir de un seminario en el MIT sobre el trabajo del futuro que tiene que ver con Estados Unidos, pero creo que es súper relevante para otros países. Una de las cosas que me encantaron fue esta idea de que el "futuro del trabajo es de nosotros y que es responsabilidad de nosotros inventarlo".

Cuando uno se pone a pensar cómo se va a inventar, se tiene que trabajar con los jóvenes para lograrlo. Entonces, la pregunta es ¿cómo voy a crear esas habilidades en mis jóvenes para lograr reinventar ese futuro del trabajo y los trabajos del futuro, para crear prosperidad? Una de las cosas de las que hablaban en este seminario sobre el reporte7 era acerca del tema de la automatización, el tema de la tecnología. Y definitivamente es indiscutible la presencia de la tecnología, el impacto que tiene en nuestros trabajos y en nuestras vidas. No podemos simplemente quedarnos esperando el futuro. Una de las recomendaciones era que necesitábamos invertir en nuestros jóvenes. Ellos hablaban de *high school* y también de los primeros años de *college*, de universidad, para hacerlo. Esa es una población crítica porque se están tomando decisiones sobre su futuro. Puede ser un futuro hacia el trabajo, un futuro hacia una carrera, para crear habilidades de tecnología que les permitan no necesariamente ser ingenieros, pero sí inventar los trabajos del futuro.

Si a mí me preguntan cuál es la edad crítica, yo digo que mientras más temprano la tecnología se vuelva una herramienta para todos, para aprender, para crear, para construir… mientras más temprano se presente a los niños, mejor. Pero cuando se llega a una edad como *high school* es crítico porque ya se vuelve algo concreto, ya pasa de ser una simple herramienta a una forma de pensar, a un futuro en la vida.

Hay mucho por hacer en el tema del pensamiento computacional. Puede ser controversial cuando queremos medirlo y volverlo un fin. Insisto en el tema de formas de pensar y por eso creo que algunos de los miembros de mi grupo, y de mi familia del MIT, han decidido definirlo de diferentes formas. Podemos enseñar sobre programación - la sintaxis del lenguaje, las estructuras de control y datos, variables, etc., pero eso no garantiza que podamos crear algo, que podamos solucionar problemas si el énfasis no está en lo que hacemos a medida que programamos. Mi recomendación es aprender a programar a medida que creamos algo. Es como aprender a montar en bicicleta porque de hecho estamos usando la bicicleta, no porque aprendimos sobre sus partes o memorizamos conceptos como balance, velocidad, fricción. Usando

[7] Se refiere a *The work of the future: Building better jobs in an age of intelligent machines*, de Autor, Mindell y Reynolds, publicado en 2020 y disponible en https://bit.ly/37NKfG8

esta analogía, quiero hacer énfasis en el uso de la programación con un propósito que va más allá de programar, y que contribuye al desarrollo de habilidades de pensamiento.

Mitch Resnick habla de pensamiento creativo porque lo más importante es usar la tecnología para crear cosas. Él está convencido, y su grupo está convencido, de la invención y la creatividad como habilidades críticas en el futuro que tenemos que desarrollar. El énfasis es lograr ese tipo de habilidades, y la tecnología es una herramienta para desarrollar y crear diferentes cosas.

Alguien como Hal Abelson, que creó App Inventor, habla de computación en acción porque quiere que no solamente se aprenda a programar, sino que quiere que eso se traduzca en que los jóvenes hagan soluciones concretas usando una herramienta como App Inventor, que es para crear aplicaciones en el teléfono.

Aquí volvemos a Papert, y con esto quiero terminar: mientras más concreta sea la acción es mejor. Crear cosas concretas; inventar cosas concretas me va a llevar a desarrollar esa fluidez tecnológica pero no como un fin, sino como un mecanismo. Muchas gracias.

Dra. Claudia Urrea

Massachusetts Institute of Technology, Abdul Latif Jameel World Education Lab

calla@mit.edu

ORCID: 0000-0003-0286-6601