

# Programación en la Escuela

¿Para qué?

**E**n el número de agosto 2012 de SG, Ignacio Cabral Perdomo presentó un interesante artículo titulado «Enseñando a niños a programar: ¿Imposible o una oportunidad?». La respuesta me parece clarísima: Claro que se puede. Esto viene siendo demostrado con gran éxito, desde los 1960s, empleando el lenguaje BASIC diseñado por Kemeny y Kurtz, y muy particularmente con el lenguaje Logo, conocido principalmente gracias al trabajo de uno de sus autores, Seymour Papert. En esta columna haré un planteamiento diferente.

El énfasis que presentan las conclusiones de Ignacio apunta al mercado del desarrollo de cómputo: “Es muy clara la necesidad de más profesionistas en el área de la Computación y las Tecnologías de Información, en especial en los departamentos de desarrollo de software de diferentes corporaciones pero, desgraciadamente, el interés de los alumnos por carreras de este tipo está reduciéndose de una forma alarmante. Una posible solución es el inculcar desde temprana edad el pensamiento lógico y algorítmico en los niños siguiendo el itinerario de aprendizaje que propongo”.

Si bien el artículo refiere que la enseñanza de programación a partir de nivel primaria “ayuda a los chicos a formar ese pensamiento lógico y algorítmico que tanto necesitan”, no profundiza en este aspecto, que considero fundamental. ¿Por qué los chicos pueden necesitar un pensamiento lógico y algorítmico?

## Los hijos de Logo

Soy parte de una minoría afortunada (pido disculpas anticipadas si el presente artículo se ve como un viaje a mi anecdotario personal), aprendí computación cuando el acceso al equipo de cómputo era extremadamente poco común. Mi primera experiencia fue en la minicomputadora Foonly que había en el IIMAS (UNAM) en 1983, a los 7 años, escribiendo LaTeX con el editor Emacs. Cabe mencionar que el presente artículo, casi 30 años más tarde, lo estoy escribiendo con las mismas herramientas. Tuve acceso a la Foonly gracias a que mi padre trabajaba como investigador en dicho Instituto, y a que tuvo la paciencia de enseñar a su ávido niño ese lenguaje cargado de símbolos y comandos.

Pero creo que mi experiencia con la Foonly se habría mantenido como meramente incidental de no ser porque, uno o dos años más tarde, me inscribieron en IDESE, una de las primeras escuelas de verano dedicadas al cómputo en México. IDESE era una apuesta pedagógica muy interesante que consistía en que durante tres semanas alternábamos dos horas frente a la computadora con dos horas más con juegos de mesa. Si bien no recuerdo los detalles de la interacción, esta alternancia ilustra claramente cómo veían nuestros instructores su tarea: Llevar a los niños a emplear sus habilidades cognitivas de una manera más completa.

IDESE derivó de la versión de Logo desarrollado por el MIT para la Apple, traduciendo todos sus comandos y mensajes al español. Sólo otra vez, también en los 1980, vi un esfuerzo similar: El hecho por la BBC al traducir el lenguaje BASIC de su BBC Micro para crear el EBASIC. Esto permitía enseñar a los niños a programar la computadora sin preocuparse al mismo tiempo de aprender otro idioma. El caso del EBASIC me resulta notorio porque, con un comando, se podía ver el código escrito en EBASIC en BASIC “normal”. Para 1985, me tocó formar parte del taller de computación que se impartía en mi escuela a partir de 4° de primaria. A partir de 1986, estuve inscrito para varios cursos de los Centros Galileo. Tuve la suerte de haber pasado por escuelas muy motivadoras, con lo cual a esas tempranas alturas ya estaba descubierta mi vocación.

El gran acierto que hizo tan importante a Logo como lenguaje educativo fue eliminar las capas de abstracción que debía tener en mente un niño. Si bien el lenguaje permite un desarrollo complejo y formal de programación funcional, el niño podía ver la concreción de sus programas graficándolos a través de una “tortuga”. Permitir que el niño viera directa e inmediatamente sus resultados, hace 45 años, resultó un cambio fundamental y un gran motivador.

Cuando Logo fue planteado, no existía el plan de formar a los niños en programación por la gran demanda que dichas habilidades tendrían en la sociedad. La enseñanza de programación era vista como una forma de enseñar pensamiento abstracto y algorítmico.



Gunnar Wolf es administrador de sistemas para el Instituto de Investigaciones Económicas de la UNAM y desarrollador del proyecto Debian GNU/Linux.  
<http://gwolf.org>

## »»“LA TAREA ES LLEVAR A LOS NIÑOS A EMPLEAR SUS HABILIDADES COGNITIVAS DE UNA MANERA MÁS COMPLETA.”

¿Y para qué enseñar pensamiento abstracto y algorítmico si no para formar profesionales que comprendan más fácil los paradigmas de cómputo? Citando a un buen amigo: “de lo que se trata no es de aprender más que a programar, se trata de aprender lo que significa programa”. Dicho de otro modo, ¿Para qué se enseñan matemáticas, filosofía, historia o biología? Para formar personas más completas, no sólo en su cultura, sino que en la manera de estructurar el pensamiento. Habilidades que indudablemente impactan en su crecimiento como adultos.

### La OLPC

Ninguna herramienta dura para siempre, sin embargo, y ni siquiera el gran Logo se salva de la obsolescencia que las nuevas tecnologías van causando. Hoy en día, sería iluso pensar que mover una “tortuga” por la pantalla pudiera impresionar a un niño. Afortunadamente, no han sido pocos los estudios en este campo que se han realizado — El artículo de Ignacio presentó cuatro entornos de programación orientados a la enseñanza en diferentes edades: Scratch, Alice, Greenfoot y BlueJ.

Agrego a los anteriores uno de los trabajos más comentados de los últimos años, que tiene un impacto muy medible: El proyecto OLPC (One Laptop Per Child)[1], iniciado —al igual que Logo— en el MIT y con el decidido apoyo de Seymour Papert, entre otras muchas personalidades.

La OLPC no es cualquier computadora: Fue planteada como el vehículo sobre del cual correría Sugar[2]. Yo no tengo experiencia de primera mano con el entorno, por lo cual prefiero dirigir a quienes estén interesados en una descripción más completa al artículo que publicó Werner Westermann dentro del libro «Construcción Colaborativa del Conocimiento»[3].

En resumen, Sugar es un entorno dirigido a facilitar un aprendizaje constructor, en que cada alumno debe ir explorando y construyendo su camino por medio de la experiencia personal, lo cual lleva a una mayor apropiación del contenido que cuando éste es dictado. A partir de una interfaz sencilla, una orientación más a actividades que a aplicaciones y empleando a fondo la colaboración entre todos los alumnos, la computadora se vuelve un actor, un facilitador de la transmisión del conocimiento. Y una característica fundamental de Sugar es que el alumno no sólo puede utilizar las actividades, sino que puede modificarlas. Las actividades están escritas en Python, un lenguaje de sintaxis limpia y conceptualmente fácil de adoptar.

OLPC fue planteado como un proyecto necesariamente a gran escala: Su objetivo es que una computadora sea entregada a cada niño en edad escolar en los países receptores. El proyecto busca además

resolver problemáticas específicas de los países menos favorecidos; con ciertas modificaciones al planteamiento inicial, actualmente hay despliegues de OLPC en once países de escasos recursos[4].


Y siguiendo con el tono personal que he mantenido en esta ocasión, relato lo que me contó Manuel Kauffman, un desarrollador argentino de Sugar, en una visita que hizo a una escuela en Uruguay: Un niño, de 11 o 12 años, le explicó que prefería editar en texto los iconos de las actividades que iba creando o modificando directamente en un editor de texto, en SVG (un lenguaje basado en XML para representar gráficos vectoriales) porque le quedaban más limpios que utilizando un editor gráfico.

Este ejemplo habla como pocas cosas de apropiación de la herramienta y de la apreciación estética de la representación en código de un objeto. Hay programadores de larga carrera profesional que son incapaces de desarrollar estas habilidades.

### Conclusiones

Enseñar a programar a un niño va mucho más allá de transmitirle una habilidad para su futuro profesional. La enseñanza básica no puede basarse solamente en la transmisión de competencias medibles en el mercado.

Hay, sin embargo, puntos importantes a considerar. Si bien algunos tuvimos la gran suerte de aprender de la forma y en el momento correcto, es una materia con la que muchos se enfrentan con dificultad, el desarrollo de las capacidades de abstracción necesarias para esta materia se produce de forma muy desigual, y la frustración que esta materia puede causar en algunos alumnos puede ser muy grande. Cabe mencionar, claro, que esto mismo se presenta en varias otras materias que forman ya parte del currículo básico.

Otro punto importante a considerar es la formación de los docentes. Para incorporar una materia al currículo básico, es necesario contar con un cuerpo docente suficientemente amplio y capacitado, no sólo con las habilidades técnicas sino pedagógicas. 

>> Por Gunnar Wolf

### Referencias

[1] *One Laptop per Child*. <http://one.laptop.org>

[2] *Sugar*. <http://sugarlabs.org>

[3] Autores varios. *Seminario Construcción Colaborativa del Conocimiento*. <http://seminario.edusol.info/seco3>

[4] *LPC Stories*. <http://one.laptop.org/stories>