Hamut'ay
Revista cuatrimestral de divulgación cientitífica
UNIVERSIDAD ALAS PERUANAS
http://revistas.uap.edu.pe/ojs/index.php/HAMUT
ISSN: 2313-7878
Vol. 7 (3). Setiembre-Diciembre. Hamut'ay 2020. Lima-Perú

Lenguajes y Entornos de Programación para Fortalecer El Desarrollo de Competencias Concernientes al Pensamiento Computacional

Languages And Programming Environments To Strengthen The Development Of Competences Concerning Computer Thinking

Maira Isbeth Sarmiento Bolívar [1]
Universidad Pedagógica Nacional, Colombia
https://orcid.org/0000-0001-9066-2726

Recibido:21-08-2020
Aceptado: 22-12-2020

Cita Recomendada
Sarmiento, M. (2020). Lenguajes y entornos de programación para fortalecer el desarrollo de competencias concernientes al pensamiento computacional. Hamut´ay, 7 (3), 86-97
http://dx.doi.org/10.21503/hamu.v7i3.2200

Resumen

El artículo presenta la descripción de una serie de lenguajes y entornos de programación que han sido utilizados en actividades de aula con el fin de promover el pensamiento computacional. Para encontrar estas herramientas de software se hizo una búsqueda sistemática en la que se recopiló información relacionada con iniciativas alrededor de Iberoamérica que tuvieran como meta el desarrollo del pensamiento computacional a través de lenguajes y entornos de programación, de estas experiencias se tomaron las herramientas con mayor aceptación y uso, las cuales caracterizan y señalan aspectos como, la particularidad del entorno utilizado, características de uso, tipo de acceso (libre y gratuito), requisitos del sistema, lugares donde es posible conseguir material de apoyo para docentes y estudiantes, sitio oficial. El objetivo del estudio es brindar información a docentes interesados sobre herramientas de software, lenguajes y entornos de programación que puedan ser utilizados en propuestas para trabajar en aula con el fin de desarrollar las habilidades concernientes al pensamiento computacional, la información compartida permite dar a conocer características generales sobre catorce herramientas de software, que por sus particularidades resultan útiles para aprendices, novatos en el tema de programación, por la manera en que se puede escribir el código a través de bloques de arrastrar y soltar o mediante una sintaxis sencillas; de igual manera resultan asequibles a todos los interesados en adquirir estas herramientas, pues en su mayoría son libres y gratuitas.

Palabras Claves: lenguajes de programación, entornos de programación, pensamiento computacional, competencias relacionadas con el pensamiento computacional.

1. Especialista en Tecnologías de la Información Aplicadas a la Educación de la Universidad Pedagógica Nacional de Colombia, magister en Tecnología Informática Aplicada en Educación y Especialista en Tecnología Informática Aplicada en educación de la Universidad Nacional de La Plata (UNLP), Argentina. Docente, Área de tecnología e informática, Cundinamarca, Institución Educativa Departamental Pío XII, Pacho, Colombia. E-mail: may1740@gmail.com

Abstract

The following article presents the description of a series of programming languages and environments that have been used in classroom activities in order to promote computational thinking. To find these software tools, a systematic search was carried out, in which information related to initiatives around Ibero-America that had as their goal the development of computational thinking through programming languages and environments were collected. From these experiences, the tools with the greatest acceptance and use, were taken and characterized; in this document are indicated aspects such as: the particularity of the environment used, characteristics of use, type of access (available and free), system requirements, places where it is possible to get support material for teachers and student, official site. The objective of this article is to provide information to interested teachers in software tools, languages and programming environments that can be used in a proposal to work in the classroom in order to develop skills related to computational thinking, the shared information allows to publicize general characteristics of fourteen software tools, which due to their peculiarities are useful for novice programming learners, due to the way in which the code can be written through drag and drop blocks or through simple syntax; in the same way, they are accessible to all those interested in acquiring these tools since they are mostly available and free.

Keywords: programming languages, programming environments, computational thinking, and competencies related to computational thinking.

Introducción

Desde hace aproximadamente 20 años las políticas TIC (Tecnologías de Información y comunicación) se hacen presente en los países Iberoamericanos, con el fin de proveer masivamente a las instituciones educativas de dispositivos tecnológicos que puedan ser utilizados en las aulas, para ello por medio de programas tales como, Computadores Para Educar (CPE) Colombia, Conectar Igualdad en Argentina, Enlaces en Chile y Plan Ceibal en Uruguay, hacen entrega de equipos como, PC, laptop, tablet, tableros digitales y kits de robótica entre otros. Al contar con estos equipos en las instituciones se hace necesario buscar la manera de integrarlos al proceso educativo, de manera que se contrarreste la inclusión acelerada que han tenido en los espacios académicos, a la vez se puedan hacer uso adecuado de los mismos a través de estrategias pensadas desde lo pedagógico para facilitar el desarrollo de habilidades con las que deben contar los estudiantes de hoy (Lugo, 2015). Por esto y como complemento a los programas públicos ya mencionados se generan iniciativas como, programando con RITA y Enfoque basado en gamificación para el aprendizaje de un lenguaje de programación (Argentina), El uso de Scratch en el desarrollo de la programación lógica como un aporte interdisciplinario (Brasil) , Desarrollo del Pensamiento computacional con Scratch (Chile), Programa Nacional de Informática Educativa – PRONIE (Costa Rica), Desafío STEM (España), entre otras; estas iniciativas utilizan metodologías propias que pretenden además de incluir los dispositivos tecnológicos en las aulas, fomentar las competencias inherentes al pensamiento computacional haciendo uso de lenguajes y entornos de programación, interés suscitado por la importancia que tiene abordar este tipo de pensamiento que forma parte de las llamadas competencias para el siglo XXI, que según Sanabria y Romero (2018) son las habilidades que debe poseer un ciudadano para tener un papel activo en la sociedad.

Según la información encontrada en cada una de las iniciativas se diseñan actividades de aula con base en los contenidos a abordar, el lugar de implementación, las características del grupo objetivo y las herramientas de software/hardware elegidas; estas experiencias según exponen los documentos encontrados alcanzaron resultados positivos por la correcta selección de herramientas de software/hardware utilizadas y la planeación adecuada de las actividades para llevar acabo el trabajo en aula. Por estos resultados destacados y al tener en cuenta lo comentado por Stage (como se citó en Eduteka, 2008) don-

de agrega que por medio de la programación los estudiantes encuentran diversas maneras de dar solución a problemas, evaluar opciones y anticipar errores, que son características relacionadas al pensamiento computacional; se hace importante conocer las particularidades y potencialidades de los lenguajes y entornos de programación utilizados en las propuestas, de modo que se pueda proponer una lista de estas herramientas en donde se describan los aspectos más importantes junto con los sitios donde se pueda encontrar mayor cantidad de datos al respecto, con el fin de brindar información a docentes y personas interesadas en generar propuestas que fortalezcan el desarrollo del pensamiento computacional y que pueda ser utilizada en diversos entornos educativos.

Método

Para seleccionar la literatura estudiada, de la cual se toma información para la generación de este artículo, se siguen pasos de la metodología para la revisión sistemática Kitchenham (2004). En la cual se tienen en cuenta los siguientes aspectos:

a. Se utilizan palabras claves y combinación de estas en español, en inglés y en portugués, algunas de ellas fueron: herramientas de software para fortalecer el pensamiento computacional, lenguajes de programación, entornos de programación, desarrollo del pensamiento computacional, programación para novatos.

b. Se realiza la revisión de información por medio de fuentes documentales como:

• Librerías digitales: IEEE Explore Digital Library, ACM Digital Library.

• Artículos publicados por diversas comunidades científicas: JENUI, RIBIE.

• Artículos publicados en actas de congresos: Congreso internacional EDUTEC, Congreso Internacional de Innovación Educativa - CIIE, TISE, CACIC, entre otros.

• Repositorios institucionales: Bdigital - Repositorio U. Nacional de Colombia, SEDICI, DASH, Repositorio U. Francisco José de Caldas RIUD, Reposital UNAM.

• Portales académicos: Dialnet, RELPE, Portales Educativos del MEN Colombia – Eduteka, Colombia aprende.

• Motor de búsqueda Google Academics.

c. Se eligen publicaciones realizadas entre 2007 y 2018.

d. Se definen criterios para incluir y excluir documentos.

Tabla 1. Criterios de inclusión y exclusión

Criterios para incluir	Criterios para excluir
• Publicaciones donde se muestran herramientas de software o entornos y lenguajes de programación que fomentan el pensamiento computacional. • Publicaciones que exponen los diversos conceptos sobre pensamiento computacional. • Publicaciones donde se describen las herramientas de software o lenguajes y entornos de programación encontrados en los documentos revisados. • Publicaciones acerca de propuestas educativas para el fortalecimiento del pensamiento computacional. • Publicaciones realizadas entre los años 2007 y 2018.	• Documentos informales o con el texto incompleto. • Publicaciones en otros idiomas que no incluyera el español, portugués o inglés.

Fuente: Elaboración propia (2020)

Para continuar el proceso se realiza una lista con las publicaciones que se han evaluado por medio de los criterios para incluir o excluir documentos, de allí se eligen referencias que guían la búsqueda de documentos relacionados. Luego se aplican nuevamente criterios para incluir o excluir los documentos que no cumplen los fines de la búsqueda, después se hace una selección con la totalidad de documentos encontrados. Al contar con esta información se toman las herramientas de software, lenguajes y entornos utilizados en las propuestas pedagógicas encontradas, las cuales sirven de insumo para describir y analizar la información que se ha presentado en este articulo.

Pensamiento computacional

Según Wing (2011) el pensamiento computacional está relacionado con la actividad mental para formular problemas que admita soluciones computacionales y esta solución puede ser dada por la conjunción de seres humanos y máquinas (p.1). Acerca de las competencias, habilidades, conocimientos y actitudes pertenecientes al pensamiento computacional existen diversas acepciones, para este caso se toman las más representativas (Barr y Stephenson, 2011; Wing,

2011; ISTE y CSTA, 2011; Kemp, 2014; Linn, Aho, Blake, Constable, Kafai, Kolodner y Bradley, 2010; Sarmiento, Gorga y Sanz, 2016).

Tabla 2. Competencias relacionadas con el pensamiento computacional

Habilidades, conocimientos y actitudes pertenecientes al pensamiento computacional
• Presentar problemas que tengan solución al hacer uso de dispositivos para el procesamiento de información. • Reconocer la información principal abstrayendo datos relevantes. • A través del diseño de algoritmos lograr dar soluciones de manera ordenada. • Segmentar procesos, datos o problemas en partes reducidas que sean manipulables. • Antes de hacer las pruebas, proponer modelos de las posibles soluciones. • Probar los modelos propuestos para verificar si es adecuada la solución. • Utilizar un proceso de solución generalizado para dar solución a problemas en diversas situaciones. • Insistir y permanecer al trabajar con problemas complejos • Tener habilidad para llegar a acuerdos al trabajar en equipo.

Fuente: Elaboración propia (2020)

Por otro lado, se tiene en cuenta que, según autoridades en el tema sobre el desarrollo del pensamiento computacional (Kafai, Resnick, Maloney, Monroy-Hernández, Rusk, Eastmond y Silverman, 2009; Guerrero, Nacional y Luis, 2014), señalan los entornos y lenguajes de programación como un medio para aprender a discernir problemas, proponer soluciones prácticas, expresar las soluciones de manera correcta, entre otras. Motivo por el cual esta serie de herramientas de software se emplean en las iniciativas encontradas y se eligen para ser caracterizadas en este artículo.

Para definir las herramientas del pensamiento computacional se han revisado las experiencias educativas de Iberoamerica cuyo objetivo es fomentar el pensamiento computacional a través de lenguajes y entornos para enseñar programación o temas relacionados (Sarmiento, Gorga y Sanz, 2018). Acerca de las herramientas de software, lenguajes y entornos de programación encontradas se revisan algunas como por ejemplo Robot Inventor to Teach Algorithms - RITA, desarrollada por el grupo de investigación LINTI en la Universidad Nacional de La Plata en Argentina que al hacer uso de esta herramienta impulsa un proyecto de articulación universidad – escuela; el mismo grupo en otra experiencia, utiliza el lenguaje Python para trabajar en el segundo año el Seminario de lenguajes en la facultad de informática, por medio de este lenguaje los estudiantes además de aprender conceptos de programación, desarrollan aplicaciones útiles para proyectos de investigación, extensión y otras cátedras; de la misma forma el grupo de Innovación Educativa Universidad Nacional de La Pampa, Argentina, trabaja mediante el enfoque de gamificación nociones de programación entre ellas: algoritmos, estructuras y variables, haciendo uso de los entornos de programación Ligthbot y Scratch. En Chile, también hacen aportes sobre el tema, al generar varias propuestas, en la que se usa Scratch y los robots de Lego para abordar fundamentos de programación y desarrollar competencias inherentes al pensamiento computacional. De esta manera se eligen una diversidad de herramientas de software, lenguajes y entornos de programación que se caracterizan a continuación.

Lenguajes y entornos de programación para el desarrollo del pensamiento computacional

Expertos en el tema señalan que, los lenguajes y entornos de programación permiten el desarrollo de diversas habilidades y potencian el aprendizaje en otras disciplinas mientras trabajan los conceptos de programación. En este caso se define al lenguaje de programación como la manera de comunicación entre una persona con una máquina, por medio de uso de símbolos y reglas para definir una estructura, y el entorno como un programa compuesto por una serie de herramientas de programación (Quiroz, Muñoz y Noël, 2012).

Para un aprendizaje inicial de programación enfocada al desarrollo del pensamiento computacional existen herramientas de software que tienen una buena aceptación en comunidades educativas interesadas por los resultados favorables descritos en las experiencias revisadas, las cuales hacen uso de estos lenguajes, al respecto se detallará un conjunto de criterios descriptivos como son: C1 nombre, C2 descripción, C3 requisitos software/hardware y C4 sitio web, que enmarcan el lenguaje y entornos de programación para el desarrollo del pensamiento computacional, entre ellos estan: Alice, App Inventor, Astro-Código, Blocky Games, Kodu Game Lab, Light-Bot, Minecraft Hora del Código, Pilas Bloques, Pilas Engine, PSeInt, Python, R.I.T.A Robot Inventor to Teach Algorithms, RoboMind, Scratch,

Nombre: *Alice*
Descripción: Alice es un entorno de programación libre con código abierto, desarrollado en Java por la Carnegie Mellon University en 2004, en el se crean historias, juegos y videos que se pueden compartir en el sitio web. Es un entorno intuitivo y fácil de usar por medio del cual los estudiantes pueden aprender conceptos básicos de programación orientada a objetos, para construir algoritmos se utilizan bloques drag and drop que se visualizan de forma gráfica, esto motiva a los estudiantes porque pueden ejecutar la animación creada para verla paso a paso y de manera inmediata. El entorno cuenta con una interfaz atractiva por ser tridimensional, colorida y de fácil comprensión, este aspecto incentivo a los estudiantes a realizar las actividades que se les proponen. Alice como herramienta de programación promueven aspectos relacionados con el pensamiento computacional como el modelado, la abstracción, el pensamiento algorítmico y la modularización (Ramírez et al., 2011; Werner, Denner, Bliesner y Rex, 2009).
Requisitos software/hardware: Existen versiones para Windows Vista, XP, 2000, Windows 7 y 8, así como para Linux, Huayra, Ubuntu y RedHat, procesador Pentium II, similar o superior, 1Gb de memoria RAM. Para la versión Alice 3.1 se recomienda tener 2GB de memoria RAM y tener instalada Java JDK.
Sitio web: https://www.alice.org/

Nombre: *App Inventor*
Descripción: App Inventor como aplicación web fue desarrollada en Java por Google en 2011, es un Software libre con licencia Apache 2.0, su desarrollo está a cargo del Instituto Tecnológico de Massachusetts (MIT), este entorno permite crear aplicaciones que se pueden usar en tabletas y móviles con sistema operativo Android, su interfaz es visual e intuitivo, permite programar por medio de bloques drag and drop que resulta ser muy parecido a Scratch, lo cual permite escribir programas sin necesidad de usar códigos. El programa que se realice se puede ejecutar vía web o en la computadora, es posible probar la aplicación desarrollada por medio del editor de bloques y el emulador, para luego compilarla y utilizarla en cualquier dispositivo con sistema operativo Android. Las creaciones y proyectos se pueden guardar en la web y en la computadora, esto facilita el acceso desde lugares remotos donde se tenga conexión a internet. App Inventor permite usar elementos que son comunes en los lenguajes de programación como los bucles, las variables y las condiciones entre otras, esta herramienta puede servir para desarrollar el pensamiento lógico y la habilidad para solucionar problemas metódicamente (Universidad de Salamanca, 2015; Rederjo, 2013).
Requisitos software/hardware: existen versiones para Windows, Mac, GNU / Linux. El programa funciona en navegadores Firefox, Safari, Chrome, Explorer. Las aplicaciones creadas con App Inventor funcionan únicamente para sistemas operativos Android.
Sitio web: http://appinventor.mit.edu/explore/

Nombre: *AstroCódigo*
Descripción: es un desarrollo realizado en el 2017 para trabajo de grado en la Universidad Nacional de La Plata, Argentina, AstroCódigo es un juego serio que por medio de la gamificación explica conceptos de programación para novatos, entre ellos, algoritmos, secuenciación, repetición, iteración y estructuras de control. El juego tiene una temática de ciencia ficción donde un astronauta que es operado por el estudiante cumple misiones en el espacio, a través del uso de bloques drag and drop se solucionan las situaciones propuestas por el juego (Bione y Miceli, 2017). Al hacer uso de una herramienta web se pueden crear escenarios personalizados de modo que los docentes pueden hacer los escenarios que deseen, según las metas que tengan planeadas con los estudiantes. Se puede descargar gratuitamente desde el sitio oficial.
Requisitos software/hardware: se debe contar con conexión a internet, funciona en sistemas operativos Windows, Linux, MacOS.
Sitio web: http://www.astrocodigo.com/

Nombre: Blocky Games
Descripción: Blockly Games es un lenguaje gráfi-

co implementado en JavaScript que contiene una serie de juegos educativos para enseñar a programar, está basado en la biblioteca Blocky. Su diseño facilita el aprendizaje a las personas novatas en el tema de programación de computadoras, mediante el juego los estudiantes se alistas para usar otro tipo de lenguajes convencionales, se animan a aprender a programar porque en los desafíos propuestos se puede avanzar a su propio ritmo y aprender de los errores, utiliza bloques drag and drop para representar conceptos de código, expresiones lógicas, bucles y variables, de este modo permite programar sin preocuparse de la sintaxis. Es un programa de código abierto, libre y gratuito, se puede jugar en línea y además los archivos ejecutables se descargan del sitio oficial y pueden ser utilizados en la computadora aún sin tener internet.
Requisitos software/hardware: el programa online es compatible con todos los navegadores, no necesita gran capacidad de componentes en el dispositivo donde se ejecuta, el archivo ejecutable funciona en el sistema operativo Windows.
Sitio de web: https://blockly-games.appspot.com/

Nombre: Kodu Game Lab
Descripción: es un lenguaje desarrollado por laboratorios Future Social Experiences (FUSE) de Microsoft en el año 2009, se creó específicamente para crear juegos por medio de su interfaz visual, puede ser usado por niños y adultos novatos en el tema de la programación, quienes logran diseñar programas sin necesidad de escribir código, por medio de una interfaz simple y basada en iconos que contiene diversos escenarios y mundos, se usan bloques drag and drop que se encastran en páginas y estas a su vez forman líneas de programación que son llamadas reglas donde se contienen las condiciones y acciones. Este lenguaje promueve el ensayo y error para solucionar problemas, así como también la creatividad, el desarrollo de narraciones, también promueve en los estudiantes el reconocimiento de patrones en el código, las simulaciones mentales al anticipar el proceso que se realizará al ejecutar el programa creado, al predecir el comportamiento desde el código (Microsoft Student Partners, 2010).
El programa se encuentra en el sitio oficial y su descarga es gratuita, se puede ejecutar en una computadora o en un Xbox, existen versiones en inglés y en español, cuenta con material de apoyo donde se describen las funciones del programa.
Requisitos software/hardware: Sistema operativo Windows Xp, Vista, 7,8 y 10. disponible como juego Xbox 360 Live Indie.
Sitio web: https://www.kodugamelab.com/

Nombre: LightBot
Descripción: Es un programa desarrollado por Danny Yaroslavski de Lightbot Inc. lanzado en el 2008, cuyo objetivo es enseñar conceptos de programación enfrentando a los estudiantes a retos donde construyen algoritmos inconscientemente para superar el nivel en que se encuentran. La interfaz es amigable, cuenta con un escenario de casillas con colores y algunas indicaciones en texto, tiene un personaje que es un robot, el cual debe ser programado por medio de instrucciones al usar botones que representan acciones cuya finalidad es encontrar la casilla de salida, en el recuadro de acción se utilizan botones que cuentan con movimientos de distintas direcciones como avanzar, saltar, girar 90 grados en diversos sentidos y encender la bombilla. Al terminar la programación se pulsa el botón de ejecutar para probar el procedimiento, si este es correcto y el robot termina el recorrido, la casilla de salida se enciende y se continúa con otro nivel. Ligthbot hace parte de la iniciativa Hora del Código, donde los usuarios cuentan con varios niveles de practicas (Yaroslavski, 2014; Saturio, García, y Hernández, 2015).
Requisitos software/hardware: existe una versión paga para Mac, Android y Windows cuenta con versión de algunos niveles gratuitos que se puede jugar en línea o descargar y la versión lightbot hour of code disponible en el sitio.
Sitio web: https://lightbot.com/hocflash.html
https://lightbot.com/hour-of-code.html

Nombre: Minecraft Hora del Código
Descripción: Minecraft es un videojuego para

construir mundos, el nombre Minecraf está compuesto por dos actividades, excavar y construir, en ella se propone un desafío donde se debe sobrevivir a medios hostiles, fue creado por Markus Persson para Mojang AB, la primera versión estuvo disponible en el 2009 y lanzado en el 2011 (Villa, 2013). En el año 2014 la empresa Mojang fue adquirida por Microsoft y el juego fue integrado al movimiento La Hora del Código en el año 2015 gracias al interés de Microsoft por utilizar las tecnologías en el aula y como medio para promover la enseñanza de la programación en todo el mundo, para poder integrar el juego Code.org junto a Mojang, crearon la versión gratuita Minecraft Hora del Código, además de esta creación Microsoft continúa aportando en la construcción de herramientas educativas como es el caso del lanzamiento de Minecraf Education Edition, lanzada en 2016 con costo de 5 dólares anuales para cada usuario.
Requisitos software/hardware: la versión comercial de Minecraft está diseñada para consolas, ordenador y dispositivos móviles, para acceder a la versión Hora del Código se puede jugar en línea o descargar la versión que funciona en Windows 7 , 8 o 10.
Sitio web: https://code.org/minecraft
https://www.minecraft.net/es-es

Nombre: Pilas Bloques
Descripción: es un entorno de programación desarrollado por el proyecto Program.AR. Con el interés de enseñar conceptos básicos de programación basado en Pilas Engine, propone pequeños desafíos que se resuelven al programar usando bloques drag and drop, cuenta con varios niveles de dificultad en los cuales se debe utilizar estrategias para la solución de problemas que se resuelven por medio del diseño de algoritmos, la modularización y el uso de diversos bloques entre ellos comandos, procedimientos y repeticiones. Pilas Bloques es un programa creado específicamente para el uso educativo dentro del aula, fue diseñado para acompañar secuencias didácticas enfocadas a enseñar programación desde la escuela, cuenta con una interfaz sencilla, es una herramienta gratuita y libre, en idioma español, en el sitio oficial se encuentra material educativo como apoyo de docentes y estudiantes, es posible realizar actividades en línea y también se cuenta con una versión descargable (Artiles, 2019).
Requisitos software/hardware: La aplicación al ser instalada, ocupa normalmente entre 100 MB y 200 MB de disco duro según el sistema operativo, requiere entre 100 MB y 200 MB de memoria RAM también según el sistema operativo, funciona correctamente en navegadores Mozilla Firefox y Google Chrome, requiere Windows 7 SP1 o superior de 32 ó 64 bits. Puede correrse en cualquier versión de Mac OS versión 10.9 o superior, para Linux se puede instalar en Huayra 2, Ubuntu 12.04, Debian 8 y posteriores.
Sitio web: http://pilasbloques.program.ar/.

Nombre: Pilas Engine
Descripción: inspirado en las teorías de Seymour Papert, es un proyecto desarrollado por Hugo Ruscitti lanzado el 1 de agosto de 2010, es una herramienta de programación sencilla y llamativa para construir videojuegos, se enfoca en estudiantes sin conocimientos previos, novatos o casuales interesados en aprender a programar. Cuenta en su interfaz con actores y ejemplos que se pueden usar para construir un juego rápidamente, estas rutinas pueden ser modificadas y facilitan el desarrollo, se pueden añadir comportamientos para hacer una creación interactiva, a su vez se puede visualizar las instrucciones escritas mientras se programa. Pilas Engine tiene versión en español, se puede descargar del sitio oficial, es libre y gratuito de licencia LGPL, se puede copiar, modificar y distribuir libremente, posee material y videotutoriales educativos. Por ser basado en Phyton las instrucciones que se utilizan para programar son sencillas, el entorno se modifica continuamente gracias al aporte y colaboración de los participantes en la comunidad de programadores, quienes colaboran en el desarrollo de la herramienta; cualquier persona interesada puede hacer parte de la comunidad (Ruscitti, 2016).
Requisitos software/hardware: es compatible con Mac OS X, Windows y Gnu/Linux. Se puede trabajar en línea en los navegadores Mozilla Firefox y Google Chrome, no necesita gran capacidad de

máquina para poderse ejecutar.
Sitio web: http://pilas-engine.com.ar/

Nombre: *PSeInt*
Descripción: PSeInt, Pseudo Intérprete, es un entorno educativo desarrollado por Pablo Novara, lanzado en diciembre de 2003, útil en la enseñanza básica de programación como estructuras de control, variables, expresiones, etc. Y el desarrollo de la lógica de programación, pensado para estudiantes novatos, facilita la escritura de algoritmos por medio de herramientas que ayudan a revisar los errores, su interfaz es simple e intuitiva, como complemento tiene un editor de diagramas de flujo, cuenta con numerosas ayudas y recursos didácticos como los ejemplos donde se exponen la aplicación de estructuras básicas, la versión existente es en español. Es un programa con versiones para Windows, GNU/Linux y macOS con licencia GPL que permite el uso libre y gratuito, se puede descargar de su sitio oficial.
Requisitos software/hardware: compatible con Windows, GNU/Linux y Mac OS X. el tamaño de su instalador es de 7 Mb en Windows.
Sitio web: http://pseint.sourceforge.net/

Nombre: *Python*
Descripción: programa desarrollado por el investigador holandés Guido Van Rossum para el Centrum Wiskunde & Informatica en los Países Bajos en los años 90, donde se pueden escribir con pocas líneas de código programas completos por su sintaxis y semántica sencilla, cuenta con librerías de varios propósitos que posibilitan a los estudiantes explorar funciones variadas que se pueden realizar con el lenguaje como, interfaces gráficas y aplicaciones web (Lovos, Gibelli, y Bertone, 2014). Python es lenguaje orientado a objetos, multiplataforma, interpretado y con tipado dinámico, posee licencia "Python Software Foundation License" de código abierto, se encuentra disponible para descargarse en el sitio oficial. Según Ortiz (2010) en el artículo "Phyton como primer lenguaje de programación" menciona que además de ser orientado a objetos soporta un tipo de programación funcional y procedural, comparado con Java o C cuenta con una sintaxis sencilla y consistente, al utilizar tipado dinámico es posible trabajar programas scripts junto con aplicaciones de gran tamaño. Se modifica continuamente mediante la comunidad con la que cuenta, conformada con personas interesada en su desarrollo, Phyton es utilizado en entornos educativos para llevar programación experimental.
Requisitos software/hardware: cuenta con versiones para Windows, Mac, Linux, Aix, IBMi, Solaris, entre otros. No necesita gran capacidad de máquina para ser ejecutado.
Sitio web: https://www.python.org/

Nombre: *R.I.T.A Robot Inventor to Teach Algorithms*
Descripción: R.I.T.A. fue desarrollado por el grupo de investigación LINTI como tesis de grado en la Universidad Nacional de La Plata año 2012, integra los frameworks OpenBlocks y Robocode, es un juego didáctico con el cual se pueden trabajar conceptos básicos de programación, fomenta el pensamiento computacional, el análisis, la creatividad, junto con el trabajo en equipo, la resolución de problemas y la transferencia de conocimientos en diferentes contextos. El entorno permite programar juegos de estrategia en el cual los robots programados deben sobrevivir a batallas, se programan por medio de bloques estilo Lego drag and drop que se encastran y se pueden visualizar en código Java. R.I.T.A motiva el interés porque se puede visualizar lo que se ha programado en el mismo instante y porque escribir código con bloques resulta agradable para los estudiantes, puesto que no necesitan conocer acerca de sintaxis de programación, el código generado se compilla en Java y se ejecuta en el escenario brindado por Robocode, en el que es posible comprobar las acciones programadas para ser corregidas en caso de ser necesario, así mismo resulta interesante para los estudiantes diseñar estrategias de ataque y defensa para competir con sus compañeros y lograr ganar las batallas (Aybar, Querigua, Banchoff, Kimura, y Brown, 2017; Aybar, Queiruga, Kimura, Brown y Goméz, 2015; Aybar, Queiruga, y Banchoff, 2012).
Requisitos software/hardware: tiene versiones

para Windows y Linux, es entorno libre y gratuito de código abierto, con licencia GNU GPL.
Sitio de descarga: https://github.com/vaybar/RITA

Nombre: *RoboMind*
Descripción: Es un programa de simulación con robots que se mueven, fue desarrollado por Arvid Halma estudiante de la Universidad de Ámsterdam, lanzado en 2005, diseñado para enseñar programación a estudiantes interesados en aprender pero que no cuentan con ningún tipo de conocimiento acerca del tema, es un lenguaje de programación con interfaz sencilla en el que se puede programar rápidamente, por medio del cual se aprenden conceptos básicos mientras se programa un robot para cumplir ciertas tareas, el entorno consigue acercar a los estudiantes a áreas relacionadas con la inteligencia artificial y la robótica, se pueden crear programas propios al seguir principios que rigen a los lenguajes de programación comunes, sin necesidad de entornos de desarrollo ni compiladores externos porque RoboMind cuenta con un entorno completamente integrado que permite revisar su script en el momento que se requiera, cuenta con 24 idiomas entre ellos alemán, turco, árabe, sueco, chino, polaco, español, holandés, francés, griego. Es un programa gratuito al cual no se le puede modificar el código, descargable de su sitio oficial (Ministerio de Educación, 2018).
Requisitos software/hardware: Cuenta con versiones para Windows, Linux y Mac OS, no necesita gran capacidad de máquina para poderse ejecutar.
Sitio web: https://www.robomind.net/en/index.html

Nombre: *Scratch*
Descripción: entorno de programación creado por el laboratorio "Lifelong Kindergarten" MIT de la Universidad de California en el año 2007, basado en "Squeak", permite el desarrollo de habilidades computacionales por medio del aprendizaje de programación sin necesidad de tener conocimientos anticipados sobre este tema, su interfaz es atractiva e intuitiva y por medio de bloques de colores drag and drop se pueden dar instrucciones para crear animaciones, videojuegos y pequeños proyectos, sin necesidad de escribir líneas de código, sino que por medio de los bloques encastrados se logra programar haciendo uso de los escenarios y personajes que brinda el programa o que pueden ser creados según el interés del estudiante. De este modo es posible aprender conceptos básicos de programación como variables, repeticiones, condiciones, entre otros. Se puede acceder al entorno en línea o se realiza la descarga del instalador para PC desde su sitio oficial, también puede ser instalado en su versión para Tablet, es un programa multiplataforma y multilenguaje, libre y gratuito, está disponible en más de 40 idiomas. Scratch cuenta con una comunidad en línea donde se pueden publicar proyectos y creaciones para permitir el trabajo colaborativo, de modo que todos aportan para mejorar y rediseñar los trabajos compartidos, de esta manera se posibilitan crear aplicaciones complejas que una sola persona no habría podido generar. Otro uso para Scratch es el desarrollo de proyectos en robótica con tecnología hardware libre, que mediante la variante del entorno llamada S4A (Scratch para Arduino) creada por Citilab, puede controlar una tarjeta de Arduino haciendo uso de instrucciones en Scratch (Brennan y Resnick, 2012; Sánchez de la Calle y Hernández, 2016; Vázquez-cano y Delgado, 2015).
Requisitos software/hardware: Windows XP y superiores, Mac OS X 10.4 y posteriores, Linux, cuenta con licencia GPLv2 y Scratch Source Code License. La configuración de la pantalla debe ser 800x480 o más, 16 bits de color o más, para su instalación necesita espacio en disco de por lo menos 120 megabytes.
Sitio web: https://scratch.mit.edu/

A continuación, en la Tabla 3 y 4 se muestran las observaciones generales de los lenguajes y entornos de programación revisados, donde se resume la información de los criterios con los cuales se analizaron.

Tabla 3. Descripción de lenguajes un entorno de programación

Nombre(C1)	Descripción(C2)	Requerimientos (C3)	Sitio Web(C4)
Alice	Libre y de código abierto Creación de juegos, historias y video Comunidad en Internet Interfaz lúdica Bloques drag and drop	Windows 7 y 8, Vista,XP y 2000 Linux 2 Gb de memoria RAM, mínimo 1 GB Java JDK	https://www.alice.org/
App Inventor	Enseñanza básica de programación Software Libre bajo la licencia Apache 2.0 Diseño aplicaciones para moviles y tablet Android Bloques drag and drop Enseñanza básica de programación	Mac OS X 10.5, 10.6 Windows XP, Vista,7 y posteriores Ubuntu 8 +, 5 + Debian, posteriores Navegadores, Mozilla Firefox 3.6 o superior Safari 5.0 o superior Google Chrome 4.0 o superior Microsoft Internet Explorer 7 o superior	http://appinventor.mit.edu/explore/
Astro Código	Enseñanza básica de programación Bloques drag and drop Juego de desafíos	Contar con conexión a internet *Windows, Linux o MacOS*	http://www.astrocodigo.com/
Blocky Games	Juegos educativos Bloques drag and drop Enseñanza básica de programación Libre y gratuito de código abierto	Navegadores Chrome, Firefox, Safari, Opera e IE No necesita gran capacidades de componentes para que el programa pueda ser ejecutado	https://blockly-games.appspot.com/
Kodu	Interfaz simple Basada en íconos Enseñanza básica de programación Libre y gratuita	*Windows 10, Windows 7, Windows 8, Windows Vista o Windows XP* *Tarjeta gráfica compatible con DirectX 9.0c y Shader Model 2.0 o superior*	https://www.kodugamelab.com/
Light Bot	Enseñanza básica de programación Juego de casillas con escenario tridimensional Interfaz amigable Utiliza botones que representan las acciones	*Windows, Mac, Android y Iphone*	https://lightbot.com/hocflash.html
Minecraft Hora del Código	Enseñanza básica de programación Versiones especiales gratuitas y libres Juego de construcción	*Windows 7* o *Windows 10.*	https://hourofcode.com/es/learn

Fuente: Elaboración propia (2020).

Tabla 2 Descripción de lenguajes un entornos de programación

Nombre (C1)	Descripción(C2)	Requerimientos (C3)	Sitio Web(C4)
Pilas Bloques	Enseñanza básica de programación Bloques drag and drop Libre y gratuita Juego de desafíos	100 MB y 200 MB de espacio en disco y en memoria RAM Mozilla Firefox y Google Chrome Windows 7 o posterior Mac OS mayor a 10.9 Huayra 2 y posteriores, Ubuntu 12.04 y posteriores, Debian 8 y posteriores	http://pilasbloques.program.ar/
Pilas Engine	Diseño de video juegos Enseñanza básica de programación Instrucciones escritas Libre y gratuita bajo la licencia LGPL Comunidad abierta	*Windows, Gnu/Linux y Mac OS X* No necesita gran capacidades de componentes para que el programa pueda ser ejecutado	http://pilas-engine.com.ar/
PSeInt	PseudoCódigo permite editar e interpretar programas escritos Enseñanza básica de Programación Libre y gratuito GPLv2	Microsoft Windows, GNU/Linux y Mac OS X Poca capacidades de componentes para que el programa pueda ser ejecutado	http://pseint.sourceforge.net/
Python	Programas escritos sintaxis sencilla Licencia de código abierto Lenguaje orientado a objetos	Windows, Mac,Linux, Aix, IBMi, Solaris No necesita gran capacidades de componentes para que el programa pueda ser ejecutado	http://www.pygame.org
R.I.T.A	Juego de programación Arrastre de bloques y conectores Puede visualizarse en Java Gratuita y de código abierto licencia GNU GPL	*Windows y Linux*	https://github.com/vaybar/RITA
Scratch	Programación básica Interfaz atractiva Bloques de arrastrar y soltar Libre y gratuito Comunidad libre Crea aplicaciones robóticas con S4A	Windows XP y posteriores Mac OS X 10.4 o posterior Pantalla (Display): 800x480 o más Por lo menos 120 megabytes	https://scratch.mit.edu/

Fuente: Elaboración propia (2020)

Conclusiones

La cantidad de lenguajes y entornos de programación analizados es pequeña, por lo tanto, no es posible brindar conclusiones categóricas, pero permite dar a conocer características generales de manera que se convierte en una guía útil de docentes o personas interesadas en la temática acerca del desarrollo de las competencias concernientes al pensamiento computacional.

Con la información recopilada se realiza un análisis donde se compara las características de los lenguajes y entornos de programación revisados, de esta manera se definen los siguientes resultados:

Entre los lenguajes descritos (Alice, App Inventor, AstroCódigo, Blocky Game, Kodu, LightBot, Minecraft hora del código, Pilas Bloques, R.I.T.A, RoboMind, Scratch se encuentran aquellos que son ideales para introducir al tema de programación, pueden ser utilizados con estudiantes novatos que no cuenten con conocimientos previos gracias a su facilidad de uso, puesto que se programa por medio de bloques drag and drop y su interfaz es intuitiva.

Aunque los lenguajes de programación orientada a objetos son utilizados por personas con algún tipo de experiencia en este tema, los entornos descritos en este estudio permiten de manera sencilla entender y aprender la sintaxis de los principales lenguajes de script.

Los lenguajes y entornos de programación descritos cuentan con interfaces gráficas llamativas y amigables, que motivan a los estudiantes en su aprendizaje puesto que muestran los resultados de la codificación al momento de ser escritos si el

estudiante lo ejecuta.

La mayoría de lenguajes y entornos descritos son libres y gratuitos, esto implica la posibilidad de ser utilizado por toda clase de estudiantes, sobre todo los que no cuentan con recursos económicos para comprar programas, brindándoles la oportunidad de acceder a estas herramientas para su aprendizaje.

Para concluir, este artículo describe diversas herramientas de software que fueron utilizadas en iniciativas encontradas alrededor de Iberoamérica cuyo objetivo estaba centrado en el desarrollo de las competencias concernientes al pensamiento computacional. La información brindada sirve para observar detalladamente las características de los lenguajes de programación, de tal manera que se pueda seleccionar la herramienta más conveniente según el grupo objetivo al que se espera llegar y la meta pedagógica por lograr. Una recomendación a tener en cuenta es que, aunque las herramientas se pueden utilizar sin restricción alguna, por ser libres y gratuitas, su uso debe ser planificado por medio de metodologías que guie el proceso para el diseño de actividades de aula donde se tengan en cuenta los objetivos proyectados y los contenidos temáticos a abordar. Para los docentes interesados en este tema, la información proporcionada en el artículo brinda una idea de las potencialidades de cada una de las herramientas, características que se puede verificar y complementar visitando los sitios web oficiales, cuyo link se encuentra en cada una de las descripciones.

Referencias Bibliográficas

Artiles, C. (2019). Pilas Bloques aprender a programar jugando. Observatorio de tecnología educativa. Recuperado de https://intef.es/wp-content/uploads/2019/06/Pilas-Bloques-1.pdf

Astudillo, G. J., Bast, S. G. y Willging, P. A. (2016). Enfoque basado en gamificación para el aprendizaje de un lenguaje de programación. Virtualidad, Educación Y Ciencia, (12), 125-142. Recuperado de https://revistas.unc.edu.ar/index.php/vesc/article/view/14739

Aybar, V., Querigua, C., Banchoff, C., Kimura, I. M. y Brown, M. (2017). Programming competitions in high school classrooms: RITA en RED. En A. Villa (Coordinación ejecutiva), CLEI 2017. Simposio llevado a cabo en XLIII Latin American Computer Conference, Córdoba, Argentina.

Aybar, V. del C., Queiruga, C., Kimura, I. M., Brown, M. y Goméz, S. (2015). Enseñando a programar con RITA en escuelas secundarias. En A. De Giusti (Coordinador), CACIC 2015. Simposio llevado a cabo en XXI Congreso Argentino de Ciencias de la Computación, Junín, Argentina.

Aybar, V., Querigua, C., y Banchoff, C. (2012). Aplicaciones complementarias a ROBOCODE que faciliten el aprendizaje de programación en escuelas secundarias (tesis de pregrado). Facultad de Informática Universidad Nacional de La Plata, Argentina. Recuperado de http://sedici.unlp.edu.ar/bitstream/handle/10915/47050/Documento_completo__.pdf?sequence=1

Barr, V. y 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. https://doi.org/10.1145/1929887.1929905

Bione, J. y Miceli, P. (2017). AstroCódigo. Un juego serio para la introducción de jóvenes en los conceptos básicos de la programación (tesis de pregrado). Universidad Nacional de La Plata UNLP, La Plata, Argentina.

Brennan, K. y Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. En A. Ball (Presidencia), Annual American Educational Research Association Meeting. Simposio llevado a cabo en el AERA 2012, Vancouver, Canada.

Capot, R. B. y Espinoza, R. M. (2015). Desarrollo del Pensamiento Computacional con Scratch. En J. Sánchez (Presidente), Nuevas ideas en informática educativa. Simposio llevado a cabo en el XX Congreso internacional de Informática Educativa, Santiago, Chile.

Csta.Iste. (2011). Computational Thinking in K-12 Education leadership toolkit. 43. Recuperado de http://www.iste.org/learn/computational-thinking

Eduteka. (2008). En pro de los computadores. Cali, Colombia: Freepik. Recuperado de http://eduteka.icesi.edu.co/articulos/ProComputadores

Guerrero, R. A., Nacional, L.U. y Luis, D. S. (2014). El Desarrollo del Pensamiento Computacional para la Resolución de Problemas en la Enseñanza Inicial de la Programación. WICC 2014. Simposio llevado a cabo en el XVI Workshop de Investigadores en Ciencias de la Computación, Ushuaia, Argentina.

Harari, V. y Tzancoff, C. (2014). Desarrollando juegos educativos para incrementar la participación de los alumnos en una materia de programación. En D. Pulfer (Presidencia) , Simposio llevado a cabo en el Congreso Iberoamericano de Ciencia, Tecnología, Innovación y Educación, Buenos Aires, Argentina.

Kafai, Y., Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., & Silverman, B. (2009). Scratch: Programming for All. Communications of the ACM, 52, 60-67. https://doi.org/10.1145/1592761.1592779

Linn, M. C., Aho;, A. V, Blake;, M. B., Constable;, R., Kafai;, Y. B., Kolodner;, J. L., & Bradley, S. (2010). Report of

a Workshop on The Scope and Nature of Computational Thinking. https://doi.org/10.17226/12840
Lovos, E., Gibelli, T. y Bertone, R. (2014). Programación Estructurada en un Curso Introductorio. Una Experiencia Explorando Python. En A. De Giusti (Presidencia), CACIC 2015. Simposio llevado a cabo en el XXI Congreso Argentino de Ciencias de la Computación, Junín, Argentina.
Lugo, M. T., Toranzos, L., Lopez, N. & Corbetta, S. (2014). Políticas tic en los sistemas educativos de América Latina. Recuperado de http://unesdoc.unesco.org/images/0023/002300/230080s.pdf
Lugo, M. (2015). Diálogo con M. Teresa Lugo. Avances en la integración de las TIC en los sistemas educativos latinoamericanos. Recuperado de http://www.siteal.iipe.unesco.org/debates/521/dialogo-con-m-teresa-lugo-avances-en-la-integracion-de-las-tic-en-los-sistemas-educativo
Microsoft Student Partners. (2010). Kodu Game Lab. Recuperado de https://studentambassadors.microsoft.com/
Ministerio de Educación. (2018). Del control automático a la robótica. Recuperado de https://www.buenosaires.gob.ar/sites/gcaba/files/profnes_educ_tecnologica_robotica_-_actividades.pdf
Muñoz, R., Barcelos, T. S., Villarroel, R., Barría, M., Becerra, C., Noel, R. y Silveira, I. F. (2015). Uso de Scratch y Lego Mindstorms como Apoyo a la Docencia en Fundamentos de Programación. En M. Castaño (presidencia), JENUI 2015.Simposio llevado a cabo en el XXI Jornadas de La Enseñanza Universitaria de Informática, Andorra La Bella, Principado de Andorra.
Queiruga, C., Fava, L., Banchoff, C., Aybar, V. del C., Kimura, I. M. & Brown, M. (2016). RITA: una herramienta didáctica-pedagógica innovadora en la escuela secundaria. Recuperado de https://jets.linti.unlp.edu.ar/uploads/docs/rita__una_herramienta_didactica_pedagogica__innovadora_en_la_escuela_secundaria.pdf
Quiroz, P., Muñoz, R. & Noël, R. (2012). Desarrollo de un lenguaje de programación y entorno de desarrollo que facilite la programación de robots LEGO mindstorms. En J. Sanchez (presidencia), TISE 2012. Simposio llevado a cabo en el XVII Congreso Internacional de Informática Educativa, Santiago de Chile, Chile.
Ramírez, M., Castillo, M., Garza, J., García, L. y Vargas, J. (2011). "Alice": un entorno diferente para aprender programación orientada a objetos. CienciaUAT, 6(2), 64-68. Recuperado de https://www.redalyc.org/pdf/4419/441942926010.pdf
Rederjo, J. (2013). Observatorio Tecnológico. Madrid, España.: Ministerio de Educación, Cultura y Deporte. Recuperado de http://recursostic.educacion.es/observatorio/web/en/software/programacion/1090-uso-de-appinventor-en-la-asignatura-de-tecnologias-de-la-comunicacion-y-la-informacion
Resnick, M. (2007). Cultivando las semillas para una sociedad más creativa. Act. Inv. En Educ., 8(1). https://doi.org/10.15517/aie.v8i1.9306
Ruscitti, H. (2016). pilas-engine - manual. Recuperado de https://docplayer.es/32129896-Pilas-engine-manual.html
Sanabria, J y Romero, M (2018). Competencias del siglo XXI en proyectos co-tecnocreativos. Revista Mexicana de Bachillerato a Distancia, 10(19), p.16-25.
https://doi.org/10.22201/cuaed.20074751e.2018.19
Sanchez, C y Hernandez, A (2016). Carmen, ¿Qué es eso del Scratch y para qué sirve?. Iberoamericadivulga. Recuperado de https://www.oei.es/historico/divulgacioncientifica/?Carmen-que-es-eso-del-Scratch-y-para-que-sirve
Sarmiento, M., Gorga, G. y Sanz, C. (2016). Análisis de experiencias y estrategias educativas con TIC para el desarrollo del pensamiento computacional en estudiantes de secundaria y primeros años de universidad en Iberoamérica (tesis de especialización). Universidad Nacional de La Plata UNLP, La Plata, Argentina.
Sarmiento, M. S., Sanz, C. y Gorga, G. (2018). Diseño de una propuesta metodológica para el desarrollo de competencias relacionadas con el pensamiento computacional (tesis de maestría). Universidad Nacional de La Plata UNLP. La Plata, Argentina.
Saturio, L. M., García, S. y Hernandez, M. (2015). Videojuegos para aprender a programar videojuegos (tesis de pregrado) Recuperado de http://eprints.ucm.es/32853/1/Memoria del Proyecto - Videojuegos para Aprender a Programar Videojuegos.pdf
Universidad de salamanca. (2015). Básicos-APPInventor-Manual-de-Introducción. Recuperado de https://diarium.usal.es/igallego/files/2015/06/Basicos-APPInventor-Manual-de-Introduccion.pdf
Vázquez-cano, A., & Delgado, F. (2015). La creación de videojuegos con scratch en educación secundaria. Communication papers -media literacy & gender studies, 4(6), 63-73. Recuperado de https://dialnet.unirioja.es/servlet/articulo?codigo=5182831
Villa, E. (2013). Minecraft: una interpretación. Revista Luthor, 14(3), 1-11.
Werner, L., Denner, J., Bliesner, M., & Rex, P. (2009). Can Middle-Schoolers use Storytelling Alice to Make Games? Results of a Pilot Study. En J. Whitehead (Presidencia), ICFDG 2009. Simposio llevado a cabo en el 4th International Conference on Foundations of Digital Games, Orlando, USA.
https://doi.org/10.1145/1536513.1536552
Wing, J. M. (2010). Computational Thinking: What and Why? Thelink - The Magaizne of the Carnegie Mellon University School of Computer Science. Recuperado de http://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why
Yaroslavski, D. (2014). ¿How does Lightbot teach programming? Waterloo, Canadá: lightbot.com. Recuperado de https://lightbot.com/Lightbot_HowDoesLightbotTeachProgramming.pdf