redie Revista Electrónica de Investigación Educativa
ISSN: 1607-4041
https://redie.uabc.mx

Vol. 22, 2020/e27

Lenguajes de programación y desarrollo de competencias clave. Revisión sistemática

Programming Languages and Development of Key Competences. Systematic Review

Francisca Tejera-Martínez (*) http://orcid.org/0000-0003-4002-9636
David Aguilera (*) https://orcid.org/0000-0002-7996-9097
José Miguel Vílchez-González (*) https://orcid.org/0000-0002-9766-1061

(*) Universidad de Granada
(Recibido: 12 de febrero de 2019; Aceptado para su publicación: 3 de abril de 2019)

Cómo citar: Tejera-Martínez, F., Aguilera, D. y Vílchez-González, J. M. (2020). Lenguajes de programación y desarrollo de competencias clave. Revisión sistemática. *Revista Electrónica de Investigación Educativa, 22*, e27, 1-16.
https://doi.org/10.24320/redie.2020.22.e27.2869

Resumen

El presente estudio trata de dar respuesta a una serie de interrogantes relacionados con los lenguajes de programación en el marco de la educación formal; en particular, en qué etapas educativas se usan los lenguajes de programación y cuáles son los más utilizados. A su vez, se pretende conocer la repercusión de estos lenguajes en el aprendizaje del alumnado, para lo cual se ha estudiado su influencia en el desarrollo de las competencias clave. Se llevó a cabo una revisión sistemática, según los fundamentos de la declaración PRISMA, de los trabajos indexados en las bases de datos Web of Science y Scopus entre 2007 y 2018. Los resultados muestran un incipiente interés de la comunidad científica por los lenguajes de programación en entornos educativos formales. Asimismo, se ha constatado la eficacia del uso de los lenguajes de programación en el desarrollo competencial del alumnado.

Palabras clave: Lenguajes de programación, Competencias, Educación tecnológica, Revisión bibliográfica.

Abstract

This study endeavors to answer a number of questions regarding programming languages in formal education – in particular, in which educational stages programming languages are used and which programming languages are most widely used. Similarly, the study also seeks to explore the impact of these languages on student learning, and to that end, examines how they influence the development of key competences. A systematic review was conducted of studies indexed in the Web of Science and Scopus databases from 2007 to 2018, in accordance with the PRISMA statement. The results show an emerging interest among the scientific community in programming languages in formal learning environments, while confirming the effectiveness of using programming languages for student competence development.

Keywords: Programming languages, Competencies, Technology education, Literature review.

I. Introducción

En los últimos años las instituciones educativas, tanto nacionales como internacionales, han hecho hincapié en el uso de las Tecnologías de la Información y la Comunicación (TIC) como vía para el desarrollo de las competencias necesarias para que el alumnado se desenvuelva en la sociedad actual. No obstante, aunque los docentes suelen reconocer la necesidad de introducir las TIC en el aula, en ocasiones les suele costar hacerlo por tener menor dominio de las TIC que los estudiantes (Almirón y Porro, 2014).

Desde hace tiempo estamos inmersos en una era tecnológica que requiere de nuevos esquemas de análisis para comprender nuevas situaciones (Tezanos, 2001), lo que conlleva una transformación notable tanto en la forma de aprender como en la de enseñar y, por tanto, también en los roles que ejercen los docentes y los estudiantes (Valverde et al., 2015). Por ello es necesario desarrollar nuevas capacidades que permitan desenvolverse en el día a día de una sociedad digitalizada en pleno siglo XXI (Rincón y Ávila, 2016). En este sentido, Llorens (2015) habla de una nueva alfabetización para las sociedades presentes, a la que llama alfabetización digital y en la que se ha de formar a los futuros ciudadanos.

Desde esta concepción, aprender a resolver problemas con el uso de las tecnologías y la programación debe ser uno de los retos principales de la educación actual (Feierherd et al., 2001). En palabras de Espino y González (2015, p. 4): "Debido a que estamos atravesando por esta novedosa etapa, las competencias relacionadas con la programación se están considerando destrezas básicas e instrumentales en la Sociedad del Conocimiento".

Todo esto ocasiona un cambio en el proceso de enseñanza-aprendizaje tradicional, que requiere adaptarse para responder a las competencias demandadas. Este trabajo pretende analizar cómo se está implantando el lenguaje de programación en la educación formal y qué beneficios aporta a las competencias clave del alumnado, a través de las siguientes preguntas:

- ¿Cómo ha evolucionado la producción científica (2007-2018) que aborda el uso del lenguaje de programación en materia educativa?
- ¿En qué etapas educativas está presente el lenguaje de programación?
- ¿Qué lenguajes de programación son los más utilizados en las aulas?
- ¿Cómo incide el lenguaje de programación en el desarrollo de las competencias clave del alumnado?
- ¿Cómo ha ido cambiando el uso del lenguaje de programación en educación durante el período estudiado?

En la actualidad, la mayoría de los jóvenes utilizan la tecnología de forma constante, ya sea para comunicarse con sus amigos, ver videos o jugar, pero muy pocos son capaces de crear sus propios videojuegos (López-Escribano y Sánchez-Montoya, 2012) o aplicaciones para dispositivos móviles. Vivimos en una sociedad dominada por una cultura digital, y en este sentido es como si los jóvenes de ahora "pudieran leer, pero no escribir" (Resnick et al., 2009), pues son simplemente consumidores de tecnología. Hay estudios que concluyen que, pese a que se suele asumir que los estudiantes actuales son nativos digitales (Kirschner y De Bruyckere, 2017), realmente no presentan los rasgos de un nativo digital, en particular los de producir, difundir y consumir cultura a través de Internet (Castellanos et al., 2017).

Los sistemas educativos están incorporando en sus currículos oficiales contenidos y conocimientos relacionados con la tecnología (Cabrera, 2015; Valverde et al., 2015). Desde esta perspectiva podemos observar el surgimiento de una nueva metodología basada en el pensamiento computacional, la programación y la robótica.

Igualmente, Llorens (2015) junto con Rincón y Ávila (2016) defienden la necesidad de un currículo que integre la tecnología con la programación y que sea accesible a todos los estudiantes, apoyando del mismo modo la importancia de la transversalidad de ambas como fin para lograr un desarrollo integral del alumnado. A su vez, otro motivo que impulsa la inclusión de la programación en las aulas es la web 2.0, los dispositivos móviles y los videojuegos, que han puesto de manifiesto la necesidad de formar al alumnado

en el ámbito tecnológico como camino para el desarrollo de las competencias necesarias para vivir en una sociedad dominada por una cultura digital (Valverde et al., 2015).

Asimismo, algunos estudios sobre la repercusión e impacto internacional de los lenguajes de programación en el ámbito educativo han hecho referencia a la prueba internacional PISA. Un ejemplo fue el caso de Estonia, que en 2012 obtuvo excelentes resultados en PISA acaparando titulares al introducir la programación informática y la robótica en los colegios desde la enseñanza básica (Cabrera 2015). De igual forma, en 2007 el Informe Horizon clasificó los mundos virtuales, los videojuegos y la programación como áreas emergentes importantes (Martínez 2017). Años después, en el informe de NMC Horizon Report: 2014 K-12 Edition (Johnson et al., 2014) se sostenía que los juegos virtuales, la gamificación, la programación y la robótica tendrían una implantación importante en el ámbito educativo en torno al 2016 y 2018.

1.1 Pensamiento computacional

En la última década se han puesto de manifiesto las necesidades de formación en relación con el pensamiento computacional y la programación (Katz, 2016). Pero para entender qué es y en qué consiste la programación debemos atender al término pensamiento computacional, popularizado por Wing (2006) como una forma de pensar y un "conjunto de habilidades útiles para todo el mundo, no sólo para los científicos de la computación", y por Basogain et al. (2015, p. 3): "El pensamiento computacional es una metodología basada en la implementación de conceptos básicos de las ciencias de la computación para resolver problemas cotidianos, diseñar sistemas domésticos y realizar tareas rutinarias".

Los lenguajes de programación (Pascal, Basic, Logo, Scratch, etc.) proporcionan al alumnado una herramienta para aprender de manera lúdica, a la vez que mejoran su pensamiento y su desarrollo cognoscitivo (Liguori, 2000). Según Basogain et al. (2015) existen estudios que avalan las ventajas de los lenguajes de programación mostrando y concluyendo que el alumnado puede adquirir habilidades específicas en el razonamiento lógico a la vez que desarrolla capacidades que derivan en nuevas formas de resolver problemas y que permiten enfrentarse a algunos que de otra forma no sería posible.

Pese a ello, hoy en día, y tal y como se refleja en varios estudios (Basogain et al., 2015; Pérez-Palencia, 2017), la programación se considera una actividad técnica destinada principalmente a un grupo de la población del ámbito de la ingeniería informática y computacional. Esto se debe, principalmente, al alto nivel de abstracción que requiere y la complejidad necesaria para su desarrollo (Zúñiga et al., 2016). Sin embargo, desde hace tiempo se trata de superar estas limitaciones a través de una programación que cuente con un lenguaje visual, así como facilitar cualquier tipo de proyectos y actividades, compartir y difundir los mismos y favorecer el uso de recursos multimedia (Resnick et al., 2009). Además, se crean objetos que materializan esos códigos abstractos haciendo visible lo planificado y fomentando la motivación del alumnado al ver sus progresos.

Del mismo modo, a la vez que se favorece una metodología activa y de colaboración, se beneficia a los estudiantes que, inmersos en el paradigma constructivista, aprenden creando nuevas estructuras mentales que ayudan a entender tanto los conceptos ya adquiridos como los nuevos (Basogain et al., 2015). En este sentido también se hace referencia a generar un conflicto cognitivo y aprender de los errores, conceptos que se tratan de manera directa a través de la programación (Llorens, 2015). Los dos términos, conflicto cognitivo y aprendizaje por errores, surgen en la programación cuando algo no sale como se había planeado. Todo ello inmerso en un aprendizaje autónomo y autodidacta, sin olvidar la colaboración entre iguales (Maloney et al., 2010).

1.2 Competencias clave

A través del proyecto de Definición y Selección de Competencias Clave (OCDE, 2003), la OCDE ha identificado un conjunto de competencias que involucran el desarrollo de habilidades y capacidades que sirven de base para la adaptación a un mundo caracterizado por la complejidad, el cambio y el desarrollo continuo. Katz (2016) señala la necesidad de fomentar el desarrollo de estas competencias junto con el pensamiento computacional y la programación. En la tabla I se observan dichas competencias, enlistadas en categorías,

junto a las capacidades y habilidades que desarrollan y que son la base de la que parte este estudio.

Tabla I. Competencias clave de la OCDE extraídas del Proyecto DeSeco

Competencia	Habilidad/Capacidad
Categoría 1. Usar las herramientas de forma interactiva	
1. Uso interactivo del lenguaje, símbolos y textos	Destrezas lingüísticas orales y escritas, de computación y matemáticas.
2. Uso interactivo del conocimiento y la información	Reconocer y determinar lo que no se sabe; identificar, ubicar y acceder a fuentes apropiadas de información; evaluar la calidad, propiedad y el valor de dicha información, así como sus fuentes y organizar el conocimiento y la información.
3. Uso interactivo de la tecnología	Relacionar las posibilidades que yacen en las herramientas tecnológicas con sus propias circunstancias y metas e incorporar la tecnología en prácticas comunes.
Categoría 2. Interactuar en grupos heterogéneos	
4. Relacionarse bien con otros	Fomentar la inteligencia emocional, respetar y apreciar los valores, las creencias, culturas e historias de otros, desarrollar la empatía, la autorreflexión, el manejo efectivo de las emociones, el conocimiento sobre uno mismo y los demás, etc.
5. Cooperar y trabajar en equipo	Habilidad de presentar ideas y escuchar las de otros, entendimiento en dinámicas de debate y seguimiento de una agenda, construcción de alianzas, negociar, tomar decisiones, etc.
6. Manejar y resolver conflictos	Analizar los elementos e intereses en juego, los orígenes del conflicto y el razonamiento de todas las partes, reconociendo que hay diferentes posiciones posibles; identificar áreas de acuerdo y desacuerdo; recontextualizar el problema y priorizar las necesidades.
Categoría 3. Actuar de manera autónoma	
7. Actuar dentro del contexto del gran panorama	Comprensión de patrones, conocimiento del sistema en el que existen (estructuras, cultura, prácticas, reglas, roles y normas sociales), identificación de consecuencias de sus acciones y reflexión de las mismas, tanto de forma individual como colectiva.
8. Formar y conducir planes de vida y proyectos personales	Definición de proyecto y metas, identificación, balance y evaluación de recursos, priorización de metas, aprender de acciones pasadas proyectando resultados futuros y monitorear el progreso haciendo los ajustes necesarios.
9. Defender y asegurar derechos, intereses, límites y necesidades	Comprender intereses propios y colectivos, conocer las reglas y principios para basar un caso, construir argumentos para que los derechos y necesidades propios sean reconocidos, sugerir arreglos o soluciones alternativas.

II. Método

Se realizó una revisión sistemática sobre el uso del lenguaje de programación en las aulas y sus posibles aportaciones, en relación con el desarrollo de las competencias clave. Para ello, en primer lugar, se establecieron los términos de búsqueda y los booleanos a utilizar: por un lado, "programming language" AND education, y, por otro, "programming language" AND "key competences". Se decidió el uso de estos términos para acotar un primer estudio de revisión sistemática, ampliable en un futuro al uso de otros relacionados con la temática.

Asimismo, se definió el período de estudio entre los años 2007 y 2018 (ambos inclusive), las bases de datos en las que se buscó la información (Web of Science y Scopus) y las categorías de las mismas (Education & Educational Research y Social Sciences, respectivamente).

Para la realización de este estudio se estableció una preselección de los documentos encontrados con los criterios establecidos a partir de la lectura del título, resumen y palabras clave. A continuación, se analizaron los artículos de esta preselección a través de una lectura profunda guiada por las interrogantes de investigación. La búsqueda se inició en la primera semana de mayo de 2018. Asimismo, esta revisión sistemática se fundamentó en la Declaración PRISMA (Preferred Reporting Items for Systematic Reviews and

Meta-Analyses) (Sotos-Prieto et al., 2014).

En la selección de artículos se buscó que cumplieran los criterios siguientes:

1) Artículos publicados en las bases de datos de WoS y Scopus, ubicados en categorías de educación.
2) Artículos publicados entre 2007 y 2018, ambos inclusive.
3) Artículos que hagan referencia, bien en su título, resumen o palabras clave, al lenguaje de programación dentro de la educación formal.
4) Estudios que expongan implicaciones educativas referentes al lenguaje de programación y el desarrollo de las competencias clave del alumnado.

El proceso de selección de artículos se realizó con base en los criterios expuestos y eliminando aquellos que coincidían en ambas bases de datos y eran objeto de estudio a partir de la primera selección.

A continuación se procedió a la extracción y análisis de datos a partir de la lectura en profundidad de los artículos seleccionados y con base en los ítems etapa educativa y lenguaje de programación utilizado. Asimismo, y en relación con la incidencia en las competencias clave del alumnado, se tomaron como referencia las extraídas del Proyecto DeSeCo (OCDE, 2003) y, a partir de la información relacionada con éstas, tanto explícita como implícita, se establecieron conexiones.

III. Resultados y discusión

Tras la selección de documentos y la extracción de datos se obtuvo una muestra final de 61 artículos. En la figura 1 se muestra el proceso seguido a lo largo del trabajo de revisión. Los artículos seleccionados aparecen numerados en el Anexo y se citarán en el texto según el número asignado.

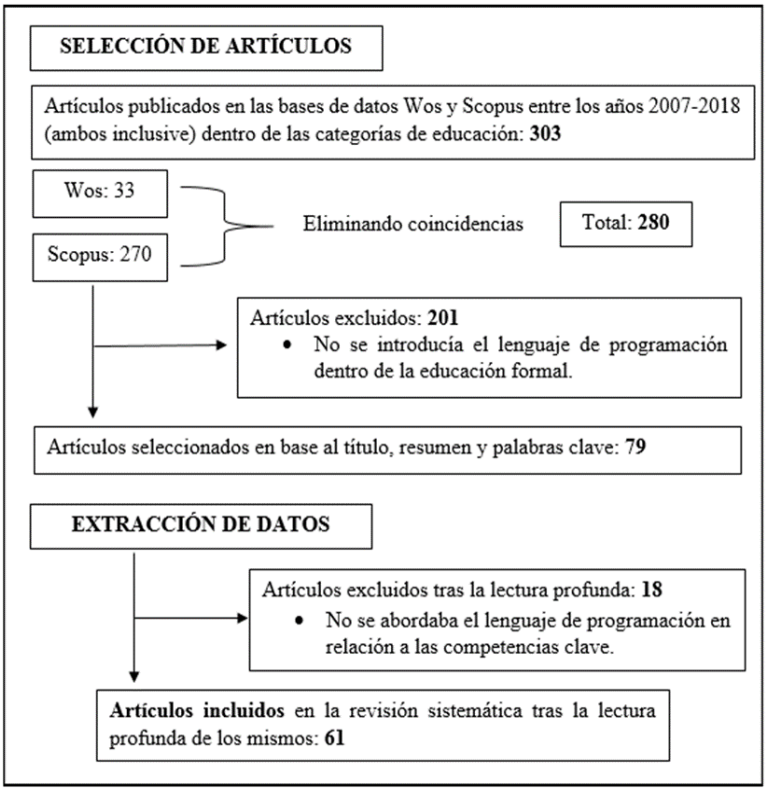

Figura 1. Proceso para la obtención de la muestra de estudio

Una vez realizado este proceso, se procedió a dar respuesta a los interrogantes de investigación.

3.1 Evolución de la producción científica

En la figura 2 se observa cómo se distribuye la producción científica a lo largo de los años estudiados.

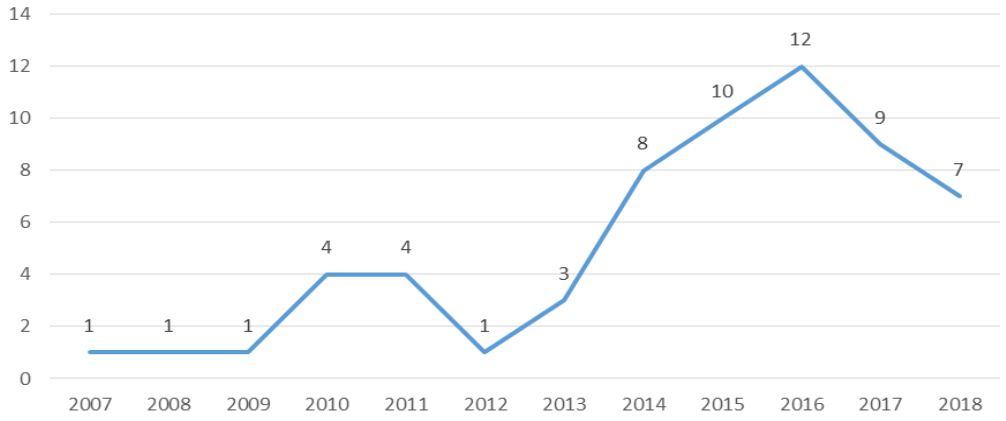

Figura 2. Evolución de la producción científica (2007-2018)

La mayor cantidad de artículos se concentran del 2014 hasta la actualidad, teniendo en cuenta que en 2018 sólo se incluyeron en el estudio documentos indexados hasta la primera semana de mayo. A pesar de esto, podemos observar una tendencia creciente en la preocupación científica relacionada con el uso del lenguaje de programación en la educación formal. Asimismo, este aumento en los últimos cinco años es palpable al representar estos artículos un 75% del total de seleccionados. Ello podría responder a que en los últimos años los sistemas educativos han apostado por la integración de contenidos relacionados con la computación, la robótica y los lenguajes de programación en los currículos oficiales (Cabrera 2015; Valverde et al., 2015). De igual forma, otro de los aspectos impulsores de los resultados obtenidos se debe al Informe Horizon (Johnson et al., 2014), en el que se predecía un auge importante de la programación en torno a los años 2016 y 2018, hecho que queda reflejado en la figura 2.

3.2 Etapas educativas en las que se usa lenguaje de programación

En la tabla II se muestra la distribución de los artículos analizados según la etapa educativa a la que hacen referencia. Se observa que la etapa educativa predominante es la Universitaria (ej.: 3, 6, 7, 16, 24, 27, 28, 30, 34, 42 y 59), seguida de la Secundaria (ej.: 1, 2, 12, 15, 21, 22, 26, 45, 53, 59 y 60). Los artículos publicados antes de 2010 se destinaban casi exclusivamente a la educación superior, y es a partir de 2014 cuando aparecen estudios realizados en otras etapas, como la educación Primaria o la Infantil (ej.: 8, 13, 19, 41, 43, 46, 51 y 52).

Tabla II. Etapas educativas con presencia de programación

Ítem	Variables	N	Porcentaje
Etapa educativa	Educación Infantil	4	6.3
	Educación Primaria	7	11.0
	Educación Secundaria	21	32.9
	Universidad	30	45.3
	Posgrado	1	1.6

Fuente: Elaboración propia.

A pesar del auge creciente de la programación en todas las etapas educativas se puede observar cómo muchos estudios siguen considerándola una actividad técnica destinada al ámbito universitario y, consecuentemente, con un nivel de especialización elevado (Basogain et al., 2015; López-Escribano y Sánchez-Montoya, 2012; Pérez-Palencia, 2017). Sin embargo, y a medida que han transcurrido los años, se viene dando la implementación del lenguaje de programación en todas las etapas del sistema educativo. Esto ocasiona también un cambio en el uso y la enseñanza del mismo, de forma que autores como Zúñiga et al. (2016) han defendido la necesidad de establecer grados de aprendizaje, es decir, comenzar desde habilidades cognitivas de nivel inferior y avanzar hacia otras de nivel más alto. Este hecho queda respaldado por algunos estudios que asocian el aprendizaje de estos lenguajes con "la zona de desarrollo próximo", tan citada en la corriente constructivista (ej.: 1, 9, 14, 15, 16, 28, 29, 31, 38, 43 y 46) y la Taxonomía de Bloom (Bloom, 1956) (ej.: 25, 27, 28 y 33).

3.3 Lenguajes de programación utilizados en Educación

La figura 3 muestra que el lenguaje de programación más extendido en el ámbito educativo es Scratch, representando un 22.4% (N = 15) del total. Esto se debe principalmente a que este lenguaje puede ser utilizado en cualquier etapa educativa (ej.: Infantil, 41; Primaria, 19, 36 y 46; Secundaria, 1, 12, 15, 20 y 40; y Universitaria, 16, 35 y 56) y adaptado fácilmente a las características específicas de cada situación docente. Desde esta concepción, múltiples estudios han concluido que los lenguajes visuales fomentan el aprendizaje de forma efectiva, mejorando dicho proceso y aumentando la motivación respecto al mismo, además de otras ventajas (Basogain et al., 2015; Resnick et al., 2009) (ej.: 1, 7, 8, 17, 31, 40, 46, 54, 56 y 57). Asimismo, lenguajes como Java (ej.: 1, 5, 23, 24, 42 y 53), C (ej.: 10, 11, 18, 38, 48 y 61) y C++ (ej.: 33) tienen una gran representación debido a que son lenguajes utilizados en el ámbito universitario con alumnado especializado en el entorno de la informática y la ingeniería.

Dentro del apartado "Otros" se encuentran lenguajes de programación como Alice (ej.: 16), Greenfoot (ej.: 16), Visual Basic (ej.: 24), Lisp y Scheme (ej.: 21), Fortran (ej.: 45), Basic (ej.: 45), Algol y Cobol (ej.: 44), Processing (ej.: 33) o Pascal (ej.: 61), que se aglutinaron en dicha categoría debido a su baja representación. Por otra parte, el apartado "No especifica" hace referencia a aquellos artículos (N = 15) que utilizan los lenguajes de programación como vía para el desarrollo de software o aplicaciones web que requieren el uso y aprendizaje de estos, pero no determinan el tipo de lenguaje utilizado.

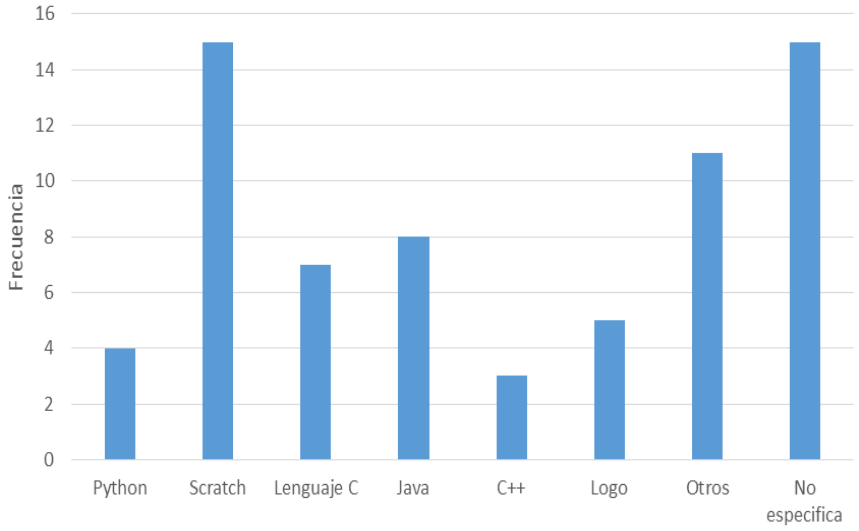

Figura 3. Lenguajes de programación más utilizados en Educación

3.4 Influencia del lenguaje de programación en las competencias clave

La tabla III presenta las competencias clave utilizadas para la categorización de los beneficios de los lenguajes de programación según los trabajos seleccionados. Además, muestra las frecuencias absoluta y relativa de los trabajos que relacionan el lenguaje de programación con el desarrollo de cada competencia.

Tabla III. Contribución de los lenguajes de programación a las competencias clave de la OCDE

Competencia	F (%)*	Habilidad/Capacidad
Categoría 1. Usar las herramientas de forma interactiva		
1. Uso interactivo del lenguaje, símbolos y textos	38 (62.3%)	Destrezas lingüísticas orales y escritas, de computación y matemáticas (1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 14, 16, 17, 19, 20, 21, 24, 28, 29, 30, 31, 32, 33, 36, 41, 42, 43, 44, 46, 47, 49, 51, 52, 54, 57, 58, 60).
2. Uso interactivo del conocimiento y la información	22 (36.1%)	Reconocer y determinar lo que no se sabe; identificar, ubicar y acceder a fuentes apropiadas de información; evaluar la calidad, propiedad y el valor de dicha información, así como sus fuentes y organizar el conocimiento y la información (5, 6, 10, 18, 20, 23, 25, 27, 34, 35, 39, 41, 42, 43, 45, 49, 53, 55, 57, 59, 60, 61).
3. Uso interactivo de la tecnología	33 (54.1%)	Relacionar las posibilidades que yacen en las herramientas tecnológicas con sus propias circunstancias y metas e incorporar la tecnología en prácticas comunes (3, 4, 6, 7, 8, 9, 10, 11, 12, 14, 17, 18, 19, 20, 22, 23, 25, 34, 35, 36, 38, 40, 46, 47, 48, 50, 52, 53, 55, 56, 57, 58, 59).
Categoría 2. Interactuar en grupos heterogéneos		
4. Relacionarse bien con otros	16 (26.2%)	Fomentar la inteligencia emocional, respetar y apreciar los valores, las creencias, culturas e historias de otros, desarrollar la autorreflexión, la empatía, el manejo efectivo de las emociones, el conocimiento sobre uno mismo y los demás, etc. (1, 5, 13, 15, 18, 20, 26, 27, 31, 37, 40, 41, 43, 46, 50, 51).
5. Cooperar y trabajar en equipo	15 (24.6%)	Habilidad de presentar ideas y escuchar las de otros, entendimiento en dinámicas de debate y seguimiento de una agenda, construcción de alianzas, negociar, tomar decisiones, etc. (13, 15, 16, 19, 22, 32, 37, 38, 40, 46, 49, 50, 52, 57, 58).
6. Manejar y resolver conflictos	14 (22.9%)	Analizar los elementos e intereses en juego, los orígenes del conflicto y el razonamiento de todas las partes, reconociendo que hay diferentes posiciones posibles; identificar áreas de acuerdo y desacuerdo; recontextualizar el problema y priorizar las necesidades (3, 9, 15, 18, 23, 35, 36, 38, 42, 46, 47, 50, 56, 58).
Categoría 3. Actuar de manera autónoma		
7. Actuar dentro del contexto del gran panorama	26 (42.6%)	Comprensión de patrones, conocimiento del sistema en el que existen (estructuras, cultura, prácticas, reglas, roles y normas sociales), identificación de consecuencias de sus acciones y reflexión de las mismas, tanto de forma individual como colectiva (1, 6, 9, 13, 18, 22, 24, 25, 26, 27, 28, 31, 32, 36, 39, 40, 41, 46, 48, 50, 51, 52, 53, 54, 56, 59).
8. Formar y conducir planes de vida y proyectos personales	25 (40.9%)	Definición de proyecto y metas, identificación, balance y evaluación de recursos, priorización de metas, aprender de acciones pasadas proyectando resultados futuros y monitorear el progreso haciendo los ajustes necesarios (1, 2, 7, 8, 9, 10, 11, 12, 17, 18, 19, 20, 25, 29, 33, 35, 38, 44, 46, 49, 53, 54, 55, 56, 58).
9. Defender y asegurar derechos, intereses, límites y necesidades	16 (26.2%)	Comprender intereses propios y colectivos, conocer las reglas y principios para basar un caso, construir argumentos para que los derechos y necesidades propios sean reconocidos, sugerir arreglos o soluciones alternativas (2, 3, 6, 14, 15, 17, 23, 27, 35, 38, 42, 45, 47, 48, 57, 61).

*Frecuencia absoluta (F) y frecuencia relativa (%).

Los resultados obtenidos (tabla III) parecen apuntar a que la implementación del lenguaje de programación en las aulas favorece en mayor medida las competencias relacionadas con la categoría 1 "Usar las herramientas de forma interactiva". Así, más del 50% de los trabajos presentan beneficios para las competencias 1 y 3. Esto se debe principalmente a que los lenguajes de programación son lenguajes en sí mismos y a que a través de estos se busca conseguir un uso adecuado e interactivo de la tecnología. Este hecho podría responder a que el lenguaje de programación se ha destinado principalmente a un grupo élite de la población, conocedor del ámbito tecnológico y especializado en el mismo (Basogain et al., 2015; López-Escribano y Sánchez-Montoya, 2012; Llorens, 2015; Pérez-Palencia, 2017; Rincón y Ávila, 2016). Por tanto, parece lógico que los estudios centren su atención en la adquisición de aquellas competencias relacionadas con habilidades cognitivas y procedimentales (competencias 1 y 3).

La categoría 3 "Actuar de manera autónoma" es la siguiente beneficiada. El lenguaje de programación no sólo implica conocimiento sobre el uso de las herramientas de forma interactiva, sino que también, a partir de distintas tareas y ejercicios, contribuye al conocimiento sobre sí mismo, a valorarse y a confiar en las propias posibilidades. En relación con esta categoría, la mayor parte de los estudios muestran una gran implicación en el desarrollo de la autoconfianza del alumnado en sus propias capacidades y habilidades para la resolución de problemas (competencias 7 y 8, con frecuencia relativa en torno al 42%) (Feierherd et al., 2001; Llorens, 2015).

Por último, la categoría 2 "Interactuar en grupos heterogéneos" es la que menos beneficio parece recibir del uso del lenguaje de programación, ninguna de sus tres competencias alcanza una frecuencia relativa del 25%. Además, se observa un interés creciente hacia ella en los trabajos publicados en los últimos cinco años (26 de los 33 artículos mencionados en la categoría). Pese a que hasta el momento el lenguaje de programación y su implantación en el ámbito educativo ha sido observado como algo complejo, difícil y sólo destinado a un grupo especializado de la población, se comprueba que con él se puede fomentar el trabajo en equipo (competencias 4 y 5) y la resolución de conflictos (competencia 6). Por tanto, los lenguajes de programación no sólo se limitan a tareas específicas del ámbito tecnológico, sino que también desarrollan propuestas para resolver problemas de la vida cotidiana a través del pensamiento creativo y crítico, pudiendo así trasladar los conocimientos adquiridos a otra serie de tareas más propias del día a día (Feierherd et al., 2001; Liguori, 2000; Llorens, 2015; Valverde y Fernández, 2015).

3.5 Evolución del uso del lenguaje de programación

La mayoría de los artículos anteriores a 2010 (ej.: 16, 25, 28 y 55) tratan el lenguaje de programación en el ámbito universitario y destinado exclusivamente al uso de la informática y lenguajes basados en texto escrito. En el período comprendido entre 2011 y 2013 (ej.: 20, 26, 43 y 45) comienza a observarse un incremento en los estudios realizados en otras etapas educativas como la Primaria y la Secundaria. Por último, en los últimos cinco años (ej.: 1, 7, 9, 15, 17, 19, 23, 33, 40, 41, 42, 44, 46, 47, 48, 50, 51, 52, 54, 56, 57 y 59) los estudios abarcan todas las etapas educativas y se orientan, centrados en el desarrollo de competencias, a una enseñanza de conceptos de programación orientada a objetos y, en consecuencia, más visual que textual. Tanto es así que puede percibirse una transición respecto al empleo de los distintos lenguajes, pasando de una programación textual a una visual en aras de paliar el elevado nivel de abstracción que requieren los lenguajes textuales (Resnick et al., 2009).

También es interesante comentar qué recursos educativos han ido cobrando relevancia por distintos motivos a lo largo de los años, entre los que encontramos la Plataforma de evaluación EduPCR (58), los bloques LEGO (ej.: 25, 29 y 51), el entorno de programación de LabVIEW (ej.: 55), Squeak Smalltalk (ej.: 45), Lost in Space (ej.: 47), Sprego (ej.: 3) (método de la gestión de hoja de cálculo), Robotics First-an Entorno móvil Android (ej.: 2), Space Race (ej.: 50), Prolog en mundos virtuales (ej.: 57), RoboBuilder (ej.: 59), mashups de video musical (ej.: 15), kit de robótica KIWI (ej.: 52), Raptor (ej.: 48) o Tableros Arduino (ej.: 33), entre otros. Estos recursos nos ayudan a comprender diversas metodologías y formas de implantación de los lenguajes de programación en las aulas y su contribución directa al desarrollo de habilidades y capacidades relacionadas con la ciencia y la tecnología, esenciales para el día a día del alumnado actual.

Desde una perspectiva social, en los últimos años el lenguaje de programación utilizado como recurso en las aulas parece haberse impregnado de las corrientes educativas constructivistas (López-Escribano y Sánchez-Montoya, 2012; Llorens, 2015; Resnick, 2009) (ej.: 27, 44, 58 y 59). Así, se busca conseguir la atención de los estudiantes, informarles de los objetivos, conectar nuevos aprendizajes con los previos, mostrar el contenido, facilitar orientación y tutorización, evaluar el rendimiento y mejorar sus capacidades y habilidades en torno al aprendizaje y la resolución de problemas en su vida diaria (Feierherd et al., 2001; Llorens, 2015) (ej.: 13, 19, 20, 23, 31, 43, 45, 55 y 57). Por tanto, podríamos afirmar que el fin con el que se utiliza el lenguaje de programación en el ámbito educativo ha trasladado su foco de atención a las habilidades personales y sociales del alumnado. No en vano el constructivismo pretende fomentar las interacciones sociales en entornos de aprendizaje, que a su vez favorezcan una serie de habilidades necesarias para el día a día del alumnado (Espino y González, 2015) (ej.: 13, 16, 27, 31 y 43).

IV. Conclusiones

El sistema educativo actual prioriza el desarrollo de las competencias clave, a las cuales se puede contribuir mediante el uso del lenguaje de programación. A través de este estudio se pretende analizar cómo se está implantando éste en la educación formal y qué beneficios aporta a dichas competencias. Después de esta revisión podemos concluir en que:

Primero. Se observa un aumento del interés por parte de la sociedad científica respecto al uso de dichos lenguajes en educación en los últimos cinco años. Esto contribuye a que su estudio dentro del ámbito de la investigación se torne más accesible para futuros autores.

Segundo. Las etapas educativas en las que los lenguajes de programación están más presentes son, en primer lugar, la Universitaria seguida de la educación Secundaria. La educación Primaria y la educación Infantil quedan menos representadas, aunque se aprecia un creciente interés, especialmente en los últimos años, en incluir en ellas estos lenguajes.

Tercero. El lenguaje de programación más utilizado en entornos educativos es Scratch, lo que se debe a su adaptabilidad a distintas etapas educativas por ser un lenguaje visual y facilitar el aprendizaje significativo y el desarrollo competencial. Asimismo, abundan los artículos en los que no se especifica ningún lenguaje concreto por utilizar la programación para el desarrollo de software o aplicaciones web que requieren un aprendizaje y uso previo de algún lenguaje, sin especificar uno en concreto.

Cuarto. Las competencias que han recibido mayor contribución por parte de estos lenguajes son las relacionadas con las habilidades disciplinares (Categoría 1. “Usar las herramientas de forma interactiva”). A éstas le siguen las relacionadas con habilidades sociales, en primer lugar, las de carácter personal (Categoría 3. “Actuar de manera autónoma”), y por último las de carácter interpersonal (Categoría 2. “Interactuar en grupos heterogéneos).

Quinto. El uso del lenguaje de programación durante el período analizado ha evolucionado en los siguientes sentidos: a) su inclusión en todas las etapas educativas; b) el uso de programación orientada a objetos y la sustitución de entornos de programación textuales por entornos visuales; c) el aumento de recursos educativos que requieren el uso del lenguaje de programación; y d) la inclusión del uso de lenguaje de programación en estrategias didácticas constructivistas.

Para concluir, debemos enfatizar la importancia y necesidad de continuar investigando y profundizando en el presente tema de estudio, con el fin de conseguir una verdadera inclusión tecnológica de los lenguajes de programación en los entornos educativos para fomentar las habilidades y capacidades propias de las competencias necesarias en el siglo XXI. Con perspectivas de futuro, proponemos líneas de investigación relacionadas con la combinación de los lenguajes de programación con otras metodologías y recursos (gamificación, realidad aumentada y robótica), o con el período de iniciación idóneo para el aprendizaje de la programación. Esto implicaría la inserción de nuevos términos de búsqueda, tales como los nombres de los lenguajes de programación (Python, C++, Scratch, etc.), relacionados con educación (*teaching, learning, instruction*, etc.) o con la propia programación (p. ej.: *coding*).

Agradecimientos

Al Grupo de investigación HUM 613 Didáctica de las Ciencias Experimentales y de la sostenibilidad y al Fondo Social Europeo, la Junta de Andalucía y la Universidad de Granada por la financiación del contrato de joven personal investigador Nº 6161.

Referencias

Almirón, M. E. y Porro, S. (2014). Las TIC en la enseñanza: un análisis de casos. *Revista Electrónica de Investigación Educativa*, *16*(2), 152-160. https://redie.uabc.mx/redie/article/view/341

Basogain X., Olabe M. A. y Olabe J. C. (2015). Pensamiento computacional a través de la programación: paradigma de aprendizaje. *Revista de Educación a Distancia, 46.* https://doi.org/10.6018/red/46/6

Bloom S. (1956). *Taxonomy of educational objectives: Book 1 cognitive domain*. Longman.

Cabrera, J. M. (2015). Programación informática y robótica en la enseñanza básica. *Revista Avances en Supervisión Educativa*, *24*, 1-26. https://doi.org/10.23824/ase.v0i24.17

Castellanos, A., Sánchez, C. y Calderero, J. F. (2017). Nuevos modelos tecnopedagógicos. Competencia digital de los alumnos universitarios. *Revista Electrónica de Investigación Educativa*, *19*(1), 1-9. https://doi.org/10.24320/redie.2017.19.1.1148

Espino E. y González C. (2015). Estudio sobre diferencias de género en las competencias y las estrategias educativas para el desarrollo del pensamiento computacional. *Revista de Educación a Distancia, 46*(12). https://doi.org/10.6018/red/46/12

Feierherd, G. E., Depetris, O. y Jerez, M. (2001). *Una evaluación sobre la incorporación temprana de algorítmica y programación en el ingreso a Informática*. En VII Congreso Argentino de Ciencias de la Computación. El Calafate, Santa Cruz, Argentina. http://sedici.unlp.edu.ar/bitstream/handle/10915/23229/IE00134.PDF?sequence=1

Johnson L., Adams-Becker, S., Estrada V., Freeman A. y Hall C. (2014). *NMC Horizon Report: 2014 K-12 Edition*. The New Media Consortium. https://www.nmc.org/publication/nmc-horizon-report-2014-k-12-edition/

Katz, R. (2016). TIC, digitalización y políticas públicas. En M. T. Lugo (Coord.), *Entornos digitales y políticas educativas: dilemas y certezas* (pp.17-58). IIPE-UNESCO.

Kirschner, P. A. y De Bruyckere, P. (2017). The myths of the digital native and the multitasker. *Teaching and Teacher Education*, *67*, 135-142.

Liguori, L. (2000). Las nuevas tecnologías de la información y la comunicación en el marco de los viejos problemas y desafíos educativos. En E. Litwin y M. Libedinsky (Coords.) *Tecnología educativa. Política, historias, propuestas* (pp. 123-151). Paidós.

Llorens, F. (2015). Dicen por ahí… que la nueva alfabetización pasa por la programación. *ReVision*, *8*(2), 11-14. http://www.aenui.net/ojs/index.php?journal=revision&page=article&op=viewArticle&path%5B%5D=181&path%5B%5D=325

López-Escribano, C. y Sánchez-Montoya, R. (2012). Scratch y necesidades educativas especiales: programación para todos. *Revista de Educación a Distancia*, *34*, 1-14. http://www.um.es/ead/red/34

Maloney J., Resnick M., Rusk N., Silverman B. y Eastmond E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education* , *10*(4), 15. https://doi.org/10.1145/1868358.1868363

Martínez G. (2017). Diseño de una guía didáctica basada en la integración de mundos virtuales al entorno educativo de la Universidad de Cundinamarca. *Formación Universitaria*, *10*(1), 3-14. https://doi.org/10.4067/S0718-50062017000100002

OCDE. (2003). The definition and selection of key competencies. Executive summary. [La definición y selección de competencias clave. Resumen ejecutivo]. http://www.oecd.org/dataoecd/47/61/35070367.pdf

Pérez-Palencia, M. (2017). El pensamiento computacional para potenciar el desarrollo de habilidades relacionadas con la resolución creativa de problemas. *3C TIC*, *6*(1), 38-63. https://doi.org/10.17993/3ctic.2017.55.38-63

Resnick, M., Maloney, J., Monroy-Hernández A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Siver, J., Silverman, B. y Kafay, Y. (2009). *Scratch: programming for all. Communications of the ACM*, *52*(1), 60-67. https://doi.org/10.1145/1592761.1592779

Rincón A. I. y Ávila W. D. (2016). Una aproximación desde la lógica de la educación al pensamiento computacional. *Sophia, Colección de Filosofía de la Educación*, *21*, 161-176.

Sotos-Prieto, M., Prieto, J., Manera, M., Baladia, E., Martínez-Rodríguez, R. y Basulto, J. (Trads.). (2014). Ítems de referencia para publicar revisiones sistemáticas y metaanálisis: la declaración PRISMA. *Revista Española de Nutrición Humana y Dietética*, *18*(3), 172-181.

Tezanos J. F. (2001). *Hacia un nuevo paradigma social. La emergencia de las sociedades tecnológicas avanzadas. La sociedad dividida. Estructuras de clases y desigualdades en las sociedades tecnológicas*. Biblioteca Nueva.

Valverde, J., Fernández, M. R. y Garrido, M. C. (2015). El pensamiento computacional y las nuevas ecologías del aprendizaje. *Revista de Educación a Distancia*, *46*(3). https://doi.org/10.6018/red/46/3

Wing, J. M. (2006). Computational thinking. It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use. *Communications of the ACM*, *49*(3), 33-35. https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf

Zúñiga, R., Hurtado, J. y Paderewski, P. (2016). Discovering the mechanisms of abstraction in the performance of work teams in children to solve computational problems. *Sistemas y Telemática*, *14*(36), 69-87. https://doi.org/10.18046/syt.v14i36.2216

ANEXO

Artículos incluidos en la Revisión Sistemática

(1) Armoni, M., Meerbaum-Salant, O. y Ben-Ari, M. (2015). From scratch to "real" programming. *ACM Transactions on Computing Education, 14*(4).

(2) Banduka, M. L. (2016). Robotics first-a mobile environment for robotics education. *International Journal of Engineering Education, 32*(2).

(3) Biró, P., Csernoch, M., Abari, K. y Máth, J. (2015). Testing algorithmic and application skills. *Turkish Online Journal of Educational Technology*, 530-536.

(4) Blake, B. (2010) BLAKE: a lenguage designed for Programming I. *Education and Information Technologies, 15*(4), 277-291.

(5) Cabada, R., Estrada, M., Hernández, F., Bustillos, R. y Reyes-García, C. (2018). An affective and Web 3.0-based learning environment for a programming language. *Telematics and Informatics, 35*(3), 611-628.

(6) Cedazo R., Garcia C., Al-Hadithi B. (2015). A friendly online C compiler to improve programming skills based on student self-assessment. *Computer Applications in Engineering Education, 23*(6), 887-896.

(7) Chang C., Yang, Y. y Tsai, Y. (2017). Exploring the engagement effects of visual programming language for data structure courses. *Education for Information, 33*(3), 187-200.

(8) Chen, G., Shen, J., Barth-Cohen, L., Huang, X. y Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. *Computers and Education, 109*, 162-175.

(9) de Aquino, A. y Ferreira, D. (2016). Learning programming patterns using games. *International Journal of Information and Communication Technology Education, 12*(2), 23-34.

(10) Dolgopolovas, V., Jevsikova, T. y Dagiene, V. (2018). From Android games to coding in C – An approach to motivate novice engineering students to learn programming: a case study. *Computer Applications in Engineering Education, 26*(1), 75-90.

(11) Dos Santos, M., Gomes, I., Trindade, R., Da Silva, A. y Lima A. (2017). Web environment for programming and control of a mobile robot in a remote laboratory. *IEEE Transactions on Learning Technologies,10*(4), 526-531.

(12) Feijóo, P. y de la Rosa, F. (2016). RoBlock – Web App for programming learning. *International Journal of Emerging Technologies in Learning, 11*(12), 45-53.

(13) Fessakis, G., Gouli, E. y Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: a case study. *Computers and Education, 6*3, 87-97.

(14) Feurzeig, W., Papert, S. y Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487-501.

(15) Fields, D., Vasudevan, V. y Kafai, Y. (2015). The programmers' collective: fostering participatory culture by making music videos in a high school Scratch coding workshop. *Interactive Learning Environments, 23*(5), 613-633.

(16) Fincher, S. y Utting I. (2010). Machines for thinking. *ACM Transactions on Computing Education, 10*(4).

(17) Howland, K. y Good J. (2015). Learning to communicate computationally with Flip: a bi-modal programming language for game creation. *Computers and Education, 80*, 224-240.

(18) Ibáñez, M., Di-Serio, A. y Delgado, C. (2014) Gamification for engaging computer science students in learning activities: a case study. *IEEE Transactions on Learning Technologies, 7*(3). 291-301.

(19) Kalelioğlu, F. y Gülbahar, Y. (2014). The effects of teaching programming via scratch on problem solving skills: a discussion from learners' perspective. *Informatics in Education, 13*(1), 33-50.

(20) Kim, S., Chung, K. y Yu, H. (2013). Enhancing digital fluency through a training program for creative problem solving using computer programming. *Journal of Creative Behavior, 47*(3), 171-199.

(21) Konidari, E. y Louridas, P. (2010). When students are not programmers. *ACM Inroads, 1*(1), 55-60.

(22) Koorsse, M., Cilliers, C. y Calitz, A. (2015). Programming assistance tools to support the learning of IT programming in South African secondary schools. *Computers and Education, 82*, 162-178.

(23) Koulori, T., Lauria, S. y Macredie R. (2014). Teaching introductory programming: a quantitative evaluation of different approaches. *ACM Transactions on Computing Education, 14*(4).

(24) Kunkle, W. y Allen, R. (2016). The impact of different teaching approaches and languages on student learning of introductory programming concepts. *ACM Transactions on Computing Education, 16*(1).

(25) LaCombe, J. C., Vollstedt, A. y Wang, E. (2008). Teaching structured programming using LEGO programmable bricks. *Computers in Education Journal, 18*(2), 28-37.

(26) Lau, W. y Yuen, A. (2011). Modelling programming performance: beyond the influence of learner characteristics. *Computers and Education, 57*(1), 1202-1213.

(27) López, J., Miyata, Y. y Dominguez, M. (2016). Creative coding and intercultural projects in higher education: a case study in three universities. *Revista Iberoamericana de Educación a Distancia, 19*(2), 145-165.

(28) Machanick, P. (2007). Teaching Java backwards. *Computers and Education, 48*(3), 396-408.

(29) Majherová, J. y Králík, V. (2017). Innovative methods in teaching programming for future informatics teachers. *European Journal of Contemporary Education, 6*(3).

(30) Marowka, A. (2018). On parallel software engineering education using Python. *Education and Information Technologies, 23* (1), 357-372.

(31) Miller, P. (2009). Learning with a missing sense: what can we learn from the interaction of a deaf child with a Turtle? *American Annals of the Deaf, 154* (1), 71-82.

(32) Monsalvez, J. C. (2017). Python as first textual programming language in secondary education. *Education in the Knowledge Society, 18*(2), 147-162.

(33) Montironi, M., Qian, B. y Cheng, H. (2017). Development and application of the ChArduino toolkit for teaching how to program Arduino boards through the C/C++ interpreter Ch. *Computer Applications in Engineering Education, 25*(6), 1053-1065.

(34) Moons, J. y De Backer C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers and Education, 60*(1), 368-384.

(35) Moreno, J., Robles, G. y Román-González, M. (2015). Dr. Scratch: automatic analysis of scratch projects to assess and foster computational thinking. *Revista de Educación a Distancia*, 46.

(36) Moreno, J., Robles, G. y Román-González, M. (2016). Code to learn: where does it belong in the K-12 curriculum? *Journal of Information Technology Education, 15*, 283-303.

(37) Olelewe, C. y Agomuo, E. (2016). Effects of B-learning and F2F learning environments on students' achievement in QBASIC programming. *Computers and Education, 103*, 76-86.

(38) Ortiz, O., Franco, J., Garau, P. y Martin R. (2017). Innovative mobile robot method: improving the learning of programming languages in engineering degrees. *IEEE Transactions on Education, 60* (2), 143-148.

(39) Ott, C., Robins, A. y Shephard, K. (2016). Translating principles of effective feedback for students into the CS1 context. *ACM Transactions on Computing Education,16*(1).

(40) Pellas, N. (2014). The development of a virtual learning platform for teaching concurrent programming languages in Secondary education: the use of open sim and Scratch4OS. *Journal of E-Learning and Knowledge Society, 10*(1), 129-143.

(41) Portelance, D., Strawhacker, A. y Bers, M. (2016). Constructing the ScratchJr programming language in the early childhood classroom. *International Journal of Technology and Design Education, 26*(4), 489-504.

(42) Radivojević, Z., Cvetanović, M. y Jovanović, Z. (2014). Reengineering the SLEEP simulator in a concurrent and distributed programming course. *Computer Applications in Engineering Education, 22*(1), 39-51.

(43) Ratcliff, C. y Anderson, S. (2011). Reviving the turtle: exploring the use of logo with students with mild disabilities. *Computers in the Schools, 28*(3), 241-255.

(44) Rinderknecht, C. (2014). A survey on teaching and learning recursive programming. *Informatics in Education, 13*(1), 87-119.

(45) Rodhouse, K., Cooper, B. y Watkins, S. (2011). Programming for pre-college education using Squeak Smalltalk. *Computers in Education Journal, 21*(2), 101-111.

(46) Sáez-López, J., Román-González, M. y Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: a two year case study using "Scratch" in five schools. *Computers and Education, 97*, 129-141.

(47) Serrano-Laguna, A., Torrente, J., Iglesias, B. y Fernandez-Manjon, B. (2015). Building a scalable game engine to teach computer science languages. *Revista Iberoamericana de Tecnologías del Aprendizaje,10*(4), 253-261.

(48) Silva-Maceda, G., Arjona-Villicana, P. y Castillo-Barrera, F. (2016) More time or better Tools? A large-scale retrospective comparison of pedagogical approaches to teach programming. *IEEE Transactions on Education, 59*(4), 274-281.

(49) Simpkins, N. (2014). I scratch and sense but can I program? An investigation of learning with a block based programming language. *International Journal of Information and Communication Technology Education, 10*(3), 87-116.

(50) Smith, S. y Chan, S. (2017). Collaborative and competitive video games for teaching computing in higher education. *Journal of Science Education and Technology, 26*(4), 438-457.

(51) Strawhacker, A. y Bers, M. (2015). ''I want my robot to look for food'': Comparing Kindergartner's programming comprehension using tangible, graphic, and hybrid user interfaces. *International Journal of Technology and Design Education, 25*(3), 293-319.

(52) Sullivan, A. y Bers, M. (2016). Robotics in the early childhood classroom: learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade. *International Journal of Technology and Design Education, 26*(1), 3-20.

(53) Taub, R., Armoni, M., Bagno, E. y Ben-Ari, M. (2015). The effect of computer science on physics learning in a computational science environment. *Computers and Education, 87*, 10-23.

(54) Teng, C., Chen, J. y Chen, Z. (2018). Impact of augmented reality on programming language learning: efficiency and perception. *Journal of Educational Computing Research, 56*(2), 254-271.

(55) Tiernan, P. (2010). Enhancing the learning experience of undergraduate technology students with LabVIEW software. *Computers and Education, 55*(4), 1579-1588.

(56) Topalli, D. y Cagiltay, N. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers and Education, 120*, 64-74.

(57) Vosinakis, S., Anastassakis, G. y Koutsabasis, P. (2018). Teaching and learning logic programming in virtual worlds using interactive microworld representations. *British Journal of Educational Technology, 49*(1), 30-44.

(58) Wang, Y., Li, H., Feng, Y., Jiang, Y. y Liu, Y. (2012). Assessment of programming language learning based on peer code review model: Implementation and experience report. *Computers & Education, 59*(2), 412-422.

(59) Weintrop, D. y Wilensky, U. (2014). Situating programming abstractions in a constructionist video game. *Informatics in Education, 13*(2), 307-321.

(60) Weintrop, D. y Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computing Education, 18*(1).

(61) Xinogalos, S., Pitner, T., Ivanović, M. y Savić, M. (2018). Students' perspective on the first programming language: C-like or Pascal-like languages? *Education and Information Technologies, 23*(1), 287-302.