DIGITALEDUCATIONREVIEW

Programas de pensamiento computacional en educación primaria: una revisión sistemática

Rafael Sánchez-Camacho[1,*], Mariona Grané[2]
[1] Universitat de Barcelona, Spain, rsanchca49@alumnes.ub.edu, https://orcid.org/0000-0003-2007-4245
[2] Universitat de Barcelona, Spain, mgrane@ub.edu, https://orcid.org/0000-0002-1435-0664

RESUMEN

El pensamiento computacional (PC) describe un tipo de pensamiento analítico relacionado con la informática y la programación. La Sociedad Internacional para la Tecnología en la Educación (ISTE) y la Asociación de Profesores de Ciencias de la Computación (CSTA) han desarrollado una definición para promover la inclusión de PC en la educación en los EE. UU. El objetivo de esta revisión sistemática es describir los programas educativos que enseñan PC en educación primaria (rango de edad 6-14 años) publicados en revistas científicas indexadas, siguiendo las pautas PRISMA. La estrategia de búsqueda se llevó a cabo en cinco bases de datos: Dialnet, Psycinfo, Scopus, PyscNET y ERIC. Un total de 40 artículos cumplieron los criterios de inclusión. Los entornos de programación más utilizados fueron Scratch (55%), actividades unplugged (15%) y Code.org (10%). Los softwares encontrados trabajaban varias de las competencias de PC establecidas como básicas según la definición de la establecidas por CSTA & ISTE. Esta revisión sistemática recoge y sintetiza por primera vez los programas educativos que han demostrado ser útiles para trabajar el PC y los relaciona con los estándares de PC establecidos por CSTA, contribuyendo al conocimiento de las herramientas disponibles para utilizar en el ámbito educativo.

PALABRAS CLAVE: Pensamiento computacional, educación primaria, programas educativos, aprendizaje de programación.

1 INTRODUCCIÓN

El pensamiento computacional (computational thinking) (PC) es un concepto que ganado popularidad en el ámbito educativo en los últimos años. En consecuencia, la investigación alrededor del término se ha incrementado a nivel mundial, sin embargo, sigue sin existir un consenso amplio respecto a la definición ni a las habilidades y competencias que comporta el trabajar el PC en el Sistema Educativo (Acevedo, 2023).

El origen del término Pensamiento Computacional fue marcado por Jeannette M. Wing (2006), que lo definió como un tipo de pensamiento analítico que está directamente relacionado con la informática y con la programación. Según esta autora, el PC consiste en adoptar un enfoque para resolver problemas, diseñar sistemas y comprender el comportamiento humano basándose en conceptos fundamentales de la computación (Wing 2006). Por ello, el PC comparte con el pensamiento científico las formas de abordar la comprensión de la computación, la inteligencia, la mente o el comportamiento humano (Wings, 2008).

Posteriormente, autores como Brennan y Resnick (2012) dividen el concepto de PC en tres dimensiones que se relacionan entre sí: (a) conceptos computacionales -los conceptos con los que los diseñadores se involucran mientras programan, como iteración, paralelismo, etc.-, (b) prácticas computacionales -procedimientos en el desarrollo de un programa a medida que se involucran con los conceptos, como depurar proyectos o mezclar el trabajo de otros-, y perspectivas computacionales -las perspectivas que los diseñadores forman sobre el mundo que los rodea y sobre sí mismos-. Esta conceptualización del PC es utilizada como base fundamental de herramientas de programación como Scratch, muy utilizado en las aulas de todo el mundo. Sin embargo, otros autores hacen un mayore hincapié en que el PC es un tipo de pensamiento de resolución de problemas que, aunque deriva de la informática puede aplicarse en cualquier dominio y en diferentes ámbitos de aprendizaje (Aho, 2012; Yadav et al., 2017).

A causa de la diferencia de definiciones y teorías sobre el PC, algunas instituciones han intentado aunar el concepto para el desarrollo y la implementación de programas educativos en pensamiento computacional. La International Society for Technology in Education (ISTE) y la Computer Science Teachers Association (CSTA) (2011) decidieron elaborar una definición operacional y una serie de competencias computacionales a fin de promover la inclusión del PC en la educación obligatoria de los EE.UU. (Adell et al., 2019). Finalmente, la CSTA operativiza en 5 componentes de computación el currículo del PC: sistema de computación, redes e internet, datos y análisis, algoritmos y programación e impactos de la computación (CSTA, 2017). Estos componentes se distribuyen a lo largo de 120 estándares de educación que se trabajan a lo largo del llamado K-12 (ver más detalles en Tabla 1).

	Descripción general	Nºde estándares	Número de estándares según etapa en la que se trabaja				
			K2	K3-5	K6-8	K9-10	K11-12
Sistema de computación (*computing system*)	Uno o más dispositivos informáticos, incluido su hardware y software, integrados con el fin de realizar tareas compartidas.	14	3	3	3	3	2
Redes e internet (*Networks and the internet*)	Grupo de dispositivos informáticos conectados a una red global y que utilizan protocolos compartidos para comunicarse o intercambiar información o recursos.	13	1	2	3	5	2
Datos y análisis (*data and analysis*)	Proceso de inspección, limpieza, transformación y modelado de datos con el objetivo de descubrir información útil, encontrar conclusiones o respaldar la toma de decisiones.	15	3	2	3	4	3
Algoritmos y programación (*algorithms and programing*)	Proceso paso a paso de analizar problemas, diseñar, escribir, probar y mantener programas para resolver un problema o una tarea.	56	8	10	10	11	17
Impactos de la computación (*impacts of computing*)	Las formas (positivas, neutrales y negativas) en que la computación afecta muchos aspectos del mundo a nivel local, nacional y global.	22	3	4	4	7	4
TOTAL		120	18	21	23	30	28

Tabla 1. Estándares de PC que se trabajan en las distintas etapas educativas según CSTA (2017)

La Unión Europea junto con el Instituto de Tecnología Educativa del Consejo Nacional de Investigación de Italia (ITD-CNR) y la European Schoolnet (EUN) también definieron el PC para intentar llevarlo a la práctica educativa. El servicio de ciencia y conocimiento de la Comisión Europea (EU SCIENCE HUB) considera que el PC es una abreviatura de "pensar como científico de la computación", o lo que es lo mismo, es la capacidad de utilizar los conceptos de la informática para formular y resolver problemas (The Computational Thinking Study, 2019).

1.1 Pensamiento Computacional en la educación

El PC se ha promovido como una habilidad o competencia tan fundamental como la aritmética y la alfabetización (Wing, 2006), ya que se considera una destreza necesaria para afrontar los retos del siglo XXI. Esto implica la necesidad de que todo el mundo desarrolle competencias de PC (no solo los profesionales de las TIC) y que estas se incluyan en el curriculum educativo.

En la actualidad se están llevando a cabo diversos esfuerzos para establecer modelos educativos con el propósito de fomentar y cultivar el PC en el contexto escolar. Dichos modelos ayudarían a proporcionar un marco estructurado con enfoques pedagógicos concretos que incluyan actividades y estrategias que permitan a educadores, estudiantes y diseñadores de currículos integrar el PC de manera efectiva en las aulas. En el año 2011, Barr y Stephenson propusieron integrar los conceptos del PC no sólo en la informática, sino también en asignaturas como matemáticas, ciencias naturales, ciencias sociales y lenguaje, abarcando desde las etapas de Educación Infantil y Primaria. Este modelo incluía además un ejemplo de actividad para incorporar el PC en cada una de las asignaturas. Sin embargo, los propios autores reconocen que para la correcta implementación en los planes de estudios sería necesario la formación docente, consensuar definiciones claras para cada componente y desarrollar un mayor número de actividades por asignatura (Barr y Stephenson, 2011).

A pesar del interés generalizado y de los esfuerzos por integrar el PC en todos los niveles de la educación (tanto en el profesorado como en el alumnado), especialmente en la educación obligatoria, aún no existe una idea clara sobre cómo incorporar el PC en el sistema educativo debido a la variedad de intervenciones y programas educativos que se encuentran en la literatura (Tangney et al., 2010; Baytak et al., 2011; Burke, 2012). El objetivo de esta revisión sistemática fue sintetizar y analizar los programas educativos utilizados para desarrollar el PC en el ámbito de la Educación Primaria (rango de edad 6-14 años) publicados en revistas científicas indexadas y analizar las competencias y habilidades de PC trabajadas con dichos programas. Esta revisión recoge los diferentes softwares utilizados, los programas educativos empleados específicamente para trabajar el PC y los relaciona con los estándares de PC establecidos por CSTA y ISTE, contribuyendo al conocimiento de las herramientas disponibles para utilizar en el ámbito educativo.

2 MÉTODO

Se realizó una revisión sistemática integradora que permitió la síntesis de estudios mixtos siguiendo los ítems de informes preferidos para revisiones sistemáticas y pautas de metaanálisis PRISMA (Eaton et al., 2017). Se llevó a cabo la estrategia de búsqueda en las bases de datos de Dialnet, Psycinfo, Scopus, PyscNET y ERIC by ProQuest hasta mayo de 2022 sin restricción del año de publicación.

La estrategia de búsqueda se realizó en mayo de 2022 e incluyó las siguientes palabras clave: (computational thinking) AND child* NOT (teenager OR adolescent OR young adult). La estrategia de búsqueda se adaptó al lenguaje específico de cada base de datos. Los artículos identificados a través de la estrategia de búsqueda se exportaron a un archivo formato RIS.

2.1 Criterios de elegibilidad y selección de estudios

Se incluyeron todos los artículos originales publicados en inglés o español que recogen programas de enseñanza del PC en niños de 6 a 14 años. Para seleccionar los artículos se tuvieron en cuenta los siguientes criterios de inclusión: a) programas educativos para trabajar el PC, b) población de edades comprendidas entre los 6 y 14 años (educación primaria), c) lenguaje: inglés y español, y d) alumnos sin discapacidad o desorden psicológico o tratamiento médico.

Se recuperó un total de 518 artículos en búsqueda inicial en las bases de datos. El proceso de selección del estudio fue realizado de forma independiente de acuerdo con las pautas de PRISMA (Eaton et al., 2017) siguiendo dos fases: (1) revisión del título y abstract utilizando el programa Rayyan y (2) revisión por texto completo. Tras eliminar los duplicados, se revisaron un total de 446 artículos.

Tras la revisión por título y abstract, 350 artículos fueron eliminados por no cumplir criterios de inclusión. En total se revisaron por texto completo 96 artículos, de los cuales se excluyeron 56, dejando un resultado final de 40 artículos para el análisis (Figura 1).

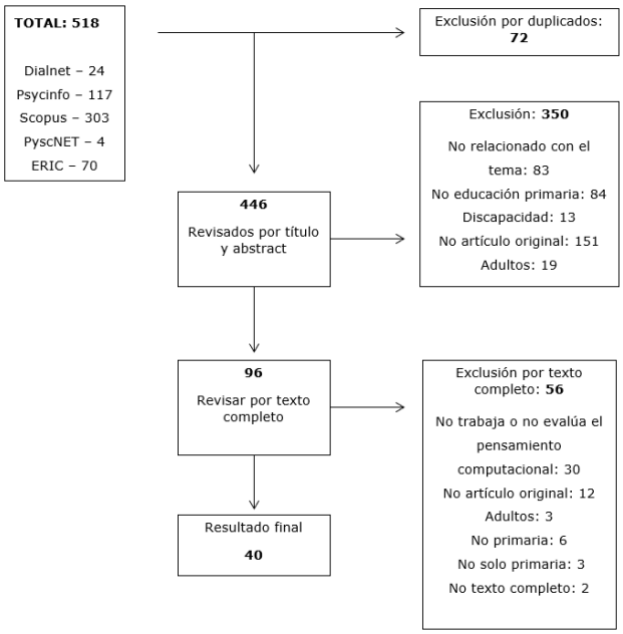

Figura 1. Diagrama de flujo de la selección de artículos incluidos en la revisión.

2.2 Extracción de datos y análisis

El autor principal extrajo y sintetizó los datos de todos los estudios incluidos. El primer paso fue desarrollar una matriz para extraer las características principales de cada uno de los artículos: autor; año de publicación; país donde se realizó el estudio; muestra que participó en el estudio (tamaño, edad y cursos escolares); método del estudio, software o material utilizado y cómo se trabajó el PC, que competencias del PC se trabajaron, modo de desarrollo del programa (actividades, sesiones, etc.) e instrumentos para evaluar PC. Se extrajeron los datos relacionados con los diferentes métodos de evaluación y luego se clasificó y sintetizó.

3 RESULTADOS

Los 40 artículos analizados en esta revisión sistemática fueron desarrollados en su mayoría en China (20%), España (17,5%), Estados Unidos (12,5%), Corea del Sur, Tailandia y Turquía (7,5%) y Reino Unido (5%). Las publicaciones obtenidas iban desde los años 2015 a 2022, pudiéndose observar un incremento considerable de publicaciones en los últimos años (ver figura 2).

Figura 2. Gráfica de representación del número de publicaciones en los últimos diez años.

Los sujetos que participaron en los distintos programas tenían edades comprendidas entre los 7 y 14 años. El 42,5% de los artículos incluían en el programa a niños menores de 10 años. La duración de los programas encontrados para trabajar PC resultó muy variable, desde el que solo trabajó 1 sesión de 70 minutos de duración hasta el que dedicó 60 horas lectivas al programa.

Los nombres y enfoques de los programas y softwares utilizados se describen en la Tabla 2. Se encontraron un total de 21 programas o entornos de programación junto con 6 estudios que incluían actividades unplugged como metodología de aprendizaje de competencias de PC. Entre los programas o software utilizados, nos encontramos con softwares de programación visual basadas en bloques y mosaicos, codificación por texto y software y aplicaciones diseñadas para la resolución de problemas o el pensamiento algorítmico.

Referencias Autor artículo (Año) País	**Entornos de programación**	**Enfoque**	**Duración y edad de los participantes**	**Evaluación PC**
Aksit y Wiebe (2022) Estados Unidos	Scratch	Introducción a Scratch, bucles, variables, eventos y actividades prácticas	5 sesiones 12-14 años	*Computational Thinking test* (CTt): 28 ítems de medida de pensamiento computacional (Román-González, et al. 2017)
Benton et al. (2018) Reino Unido	Scratch	Programa *ScratchMaths*, módulos sobre conceptos matemáticos	3 módulos 10-11 años	Cuestionario ad-hoc sobre algoritmos.
Jun et al. (2017) Corea del Sur	Scratch	*Program and Creative Computing Guidebook* del MIT	15 horas 10-12 años	Cuestionario ad-hoc sobre programación
Li et al. (2021) China	Scratch	*Computational Thinking through Design-Based Learning* (en 6 fases.)	2 horas/semana durante 5 semanas 10 años	Escala de habilidad de PC de Korkmaz et al. (2017)
Ma et al. (2021) China	Scratch	Programa *IGGIA*	14 sesiones a lo largo de 14 semanas 10-12 años	Adaptación de Korkmaz y Bai (2019) de *Computational Thinking test* (CTt) (Román-González, et al. 2017)
Moreno-León et al. (2016) España y Argentina	Scratch	Creación de proyectos variados durante cuatro semanas.	No especificado +7 años.	*Dr. Scratch* (Moreno-León et al. 2015)
Melander (2019) Suecia	Scratch	Rediseño de un juego ya existente	Clases durante dos semanas (sin especificar duración) 10 años	Estudio cualitativo de comportamiento de los sujetos en la resolución de

				problemas en la interacción entre pares.
Noh y Lee (2020) Corea del Sur	Scratch	Nueva actividad semanal siguiendo un esquema	11 semanas (sin especificar horas) 12 años	*Bebras task* (2015) parcialmente convergente con el Test de Pensamiento Computacional de Román-González et al. (2017)
Rodríguez-Martínez et al. (2020) España	Scratch	Conceptos matemáticos de Currículum español y resolver problemas relacionados con estos conceptos	3 sesiones de 1 hora 12 años	*Computational Thinking test* (CTt) (Román-González, et al. 2017)
von Wangenheim et al. (2015) Brasil	Scratch	Programa *UNIfICA*: Enseñar conceptos de programación.	6 lecciones de 2 horas 8-14 años	*Dr. Scratch* (Moreno-León et al. 2015)
Xuefeng Wei et al. (2021) China	Scratch	Curso "Pensamiento computacional con Scratch"	1 clase semanal de 40 minutos durante 16 semanas 10 años	*Dr. Scratch* (Moreno-León et al. 2015)
Allsop (2019) Reino Unido	Scratch	Creación de un videojuego de temática libre	28 horas (14 horas de Scratch y 14 horas de Alice) 10-11 años	Estudio cualitativo mediante la observación. Rúbrica de evaluación (ad-hoc) de conceptos computacionales.
	Alice	Creación de un videojuego de temática libre		
Anuar et al. (2020) Malasia	Unplugged coding game	1: Descomposición de problemas y pensamiento algorítmico	7 lecciones (sin especificar horas) Edad no especificada	Cuestionario con cinco preguntas: cuatro ítems se enfocaron en el conocimiento sobre Scratch y Programación, y otro ítem en el pensamiento algorítmico
	Scratch	Resolución de problemas, depuración		
Hsiao et al., (2022) Tailandia	Scratch	Proyecto *Crab Robot*	18 semanas. 2 sesiones semanales de 40 minutos 11-12 años	*Computational thinking ability test* (CTA test) (Bebras, 2018)
Jiang y Wong, (2022) China	Unplugged	5 lecciones en los que se tenía que adaptar en *Scratch*	5 semanas (sin especificar hora) 9-12 años	Instrumento Ad-hoc *CT Assesment*
	Scratch			
		los juegos unplugged presentados		
Kert et al. (2020) Turquía	LEGO® Mindstorms EV3 robotics kits	Unidad *Programming*	10 semanas (sin especificar horas) 11-12 años	*CT self-efficacy perception scale* (CTS) *Academic achievement test* (AAT) *Word association test* (WAT)
	Scratch			
Ntourou et al. (2021) Grecia	Scratch	Construir un modelo con *Arduino* y programarlo utilizando *Scratch*	4 semanas (sin especificar horas) 10-11 años	Adaptación del cuestionario *Bebras* (2018) de Psycharis y Kotzampasaki (2019)
Özmutlu et al. (2021) Turquía	MakeBlock MBot (programado con Scratch)	3 sesiones divididas en varias tareas	2 días (3 sesiones sin especificar número de horas) 10-11 años	*Computational Thinking Self-efficacy scale* (CTS)
	LEGO® WeDo 2.0			
Özcan et al. (2021) Turquía	Algo Dijital	10 sesiones con actividades varias, de introducción a la informática a creación de un proyecto libre	20 horas repartido entre 10 semanas 10 años	Escala de pensamiento computacional de Tran (2018)
	Code.org			
	Scratch			
Pérez-Marín et al. (2020) España	Scratch	*MECOPROG* (Pérez-Marín et al., 2018a)	3 bloques de trabajo (2 horas de duración) 9-12 años	*Computational Thinking test* (CTt) de (Román-González, et al. 2017) *PCNT* (Pérez-Marín et al., 2018b) *CONT*: cuestionario de medida de conocimiento de conceptos de programación
	CompThink App (2018)	Ocasionalmente, entre la aplicación de *MECOPROG*		
Qu y Fok (2021) China	Scratch	12 lecciones, desde introducción hasta crear un robot	3 sesiones semanales de 70 min. durante 4 semanas 7-9 años	Rúbrica de evaluación que se evaluó (Ad-hoc)
Wong et al. (2018) China	Scratch	Desarrollo de 1 video juego en 3 niveles distintos	14 lecciones de programación (8 horas aprox.) 11-12 años	Adaptación del cuestionario de resolución de problemas de Chow et al. (2009) Entrevista
	Kodu Game Lab	Proyecto de nivel 3	14 lecciones de programación (8 horas aprox.) 9-11 años	

Chou (2020) Tailandia	Scratch JR	8 experimentos en clase y actividades de programación en casa	8 semanas (sin especificar horas) 7-8 años	*CT competence test* (Strawhacker et al., 2018)
Kyza et al. (2021) Chipre	Scratch JR	Diferentes actividades	4 sesiones de 90 minutos 6-12 años	Evaluación de los proyectos
Román-González et al. (2018) España	Code.org	*Intro to CS Course*	12 semanas (sin especificar horas) 8-14 años	*Computational Thinking test3* (CTt) (Román-González et al., 2015)
Asad et al. (2016) Israel	Plastelina interactive Logic Game	15 juegos lógicos	12 sesiones de 90 minutos 9-11 años	Cuestionario de Baser, 2013; Morris y Trushell, 2014, adaptado Estudio cualitativo basado en entrevistas, observación del comportamiento y comentarios Presentación de tareas y proyecto final
	Code.org	Itinerario de 20 lecciones de codificación		
	Turtle Academy	Itinerario de 24 lecciones de codificación con el lenguaje de programación LOGO		
del Olmo-Muñoz et al. (2020) España	Unplugged	3 actividades	5 sesiones de 45 minutos (3 sesiones *unplugged* y 2 sesiones *plugged*) 6-7 años	Instrumento had-oc con 10 ítems de menor a mayor dificultad
	Code.org	5 actividades		
Caballero-González y García-Valcárcel (2020) España	Bee-Bot	6 sesiones en las que se hicieron actividades de programación y robótica	30 horas 6-7 años	Rúbrica *TangibleK* (Bers, Flannery, Kazakoff, & Sullivan, 2014)
Cervera et al. (2020) España	Bee-Bot	Completar una tarea de ruta y resolución de tareas de ruta y uso de la caja de codificación	2 sesiones de 45 minutos 10 años	*Computational Thinking test3* (CTt). (Román-González et al., 2015) adaptado
Chen y Chi (2020) Tailandia	Coding Ocean (unplugged)	Juego de estrategia de cartas de programación	70 minutos (10 minutos de explicación 50 de juego 10 minutos de conclusión) 10-12 años	Estudio cualitativo por observación y entrevistas
Hooshyar et al. (2021) Estonia	Autothinking	3 niveles dónde se trabajan las habilidades y conceptos del PC propuesto por el autor	Sesiones de 60-75 minutos 11-12 años	Adaptación de los autores del *Computational Thinking test3* (CTt). Román-González, et al. (2015)
		dónde se tiene que resolver diferentes tareas dentro de un laberinto dependiendo del nivel de dificultad		
Kawada et al. (2019) China	IchigoJam	Taller de construcción y programación de un árbol de navidad con 5 actividades con tarjetas de comandos para vincular las acciones concretas y los símbolos abstractos (comandos)	5 actividades de 30 min. de duración 5-12 años	Estudio cualitativo que analiza mediante la observación los algoritmos producidos por los estudiantes
Leonard et al. (2016) Estados Unidos	LEGO® Mindstorms EV3 robotics kits	Utilizar el robot *LEGO® EV3* para desarrollar habilidades visoespaciales y de manipulación.	60 horas 11-14 años	Rúbrica de evaluación de pensamiento computacional (ad-hoc)
	AgentSheets	Construcción de un videojuego 2D de laberinto y Frogger.		
	AgentCubes	Construcción de un videojuego 3D de laberinto y *Frogger*.		
Min y Kim (2020) Corea del Sur	LEGO® WeDo 2.0	Dos actividades para crear y programar un robot en el que los alumnos debían: 1: Analizar el problema y encontrar ideas 2: Formular y jugar con algoritmos 3: Jugar y ejecutar algoritmos	6 sesiones de 80 minutos 12 años	Conceptos computacionales: *UK Bebras Computational Challenges* (2015) Practicas computacionales: adaptación de la rúbrica del *Software* Directrices de Educación (Ministry of Education, 2015)

Moore et al. (2020) Estados Unidos	Code and Go™ Robot Mouse Activity Set	Actividad "*Puzzles and Robots*" que van desde la introducción al robot hasta el juego libre.	4 semanas (1 hora por sesión) 6-7 años	Estudio cualitativo que evalúa los procesos de iteración de codificación, comparación y condensación de datos de 4 tareas realizadas por los participantes.
Pinkard et al. (2020) Estados Unidos	Unplugged	Proyector "*Digital Youth Divas*"	3 horas 10-14 años	Dos preguntas dónde se evaluó el conocimiento y entendimiento de conceptos computacionales
	Virtual Environment Interactions (VENVI)	2: Prueba de baile escribiendo un pseudocódigo.		
Relkin et al. (2021) Estados Unidos	KIBO robotics platform	Lecciones con desafíos *KIBO* estructurados y conceptos de programación	12 sesiones de 1h durante 6-7 semanas consecutivas 5-7 años	Cuestionario *TechCheck* (Relkin et al., 2020) adaptado
Rios et al. (2020) México	EasyLogic3D	10 ejercicios de algoritmos: 3 ejercicios de estructuras secuenciales, 3 condicionales ejercicios de estructuras, 3 ejercicios de estructuras repetitivas y 1 ejercicio que incluía todos los bloques visuales	3 semanas (1 hora por cada ejercicio) 9-12 años	Cuestionario ad-hoc para evaluar los conceptos básicos en lógica algorítmica.
Soleimani et al. (2019) Estados Unidos	Arduino	*CyberPLAYce,* que trabaja la demostración y juego exploratorio con los componentes, reescribir la historia de Jane y narración de una actividad sobre el calentamiento global.	5 sesiones de 90 minutos (1 sesión la primera semana y 2 sesiones las dos semanas siguientes) 11-12 años	Estudio cualitativo que analiza mediante la observación presencial, la recogida de audios y videos el comportamiento de los sujetos.
Zhan et al. (2022) China	Micro:bit	Actividad "*Maze Treasure Hunt*": Construcción de un robot con *Micro:bit* divido en 6 módulos	1,5 horas cada semana (sin especificar) 6-8 años	*Computational Thinking Ability Test* (Laboratorio de Ciencias del Aprendizaje de la Universidad de Pekín, 2019)
		Unplugged: Codificación *unplugged* para completar un laberinto con el robot construido		

Tabla 2. Programas educativos revisados para el desarrollo del Pensamiento Computacional en niños de 6 a 14 años

Entre los métodos de enseñanza de codificación por bloques, Scratch sería el programa más utilizado (55%), seguido de actividades unplugged (15%) y Code.org (10%). Otros entornos de programación encontrados como LEGO® Mindstorms EV3 robotics kits, Alice, LEGO® WeDo 2.0, Beebot, AgentCubes o AgentSheets, fueron utilizados de manera más puntual o en combinación con Scratch. Los softwares Kodu Game Lab, LEGO® WeDo 2.0 y Scratch JR fueron utilizados para trabajar el PC con el lenguaje visual en mosaicos. Entre los softwares de programación por lenguaje se encuentran IchigoJam, Arduino (BASIC) y Turtle Academy. Finalmente, los softwares utilizados para trabajar la resolución de problemas fueron CompThink, EasyLogic 3D y Plastelina interactive Logic Game. En la Tabla 3 se describen en detalle las características de cada uno de estos entornos de programación.

Entorno de programación	Características
Algo dijital	Plataforma online donde se pueden desarrollar habilidades de pensamiento algorítmico y explorar el mundo digital con juegos. Enseña el alfabeto del mundo digital con codificación basada en bloques.
Alice	Entorno de programación visual que se centra en la codificación por bloques, utilizando la técnica de "arrastrar y soltar", y en la programación orientada a objetos en 3D. La interfaz proporciona un escenario visual claro donde se puede visualizar el resultado del proyecto en desarrollo.
Arduino	IDE basado en el lenguaje Wiring (c++), facilita la introducción a la programación y la robótica. Su hardware permite la conexión de actuadores y sensores para interactuar con el entorno. El entorno de programación incluye funciones como setup y loop, simplificando la escritura de código para realizar diversas funciones con placas Arduino.
Autothinking	Juego en un entorno adaptativo desarrollado para estudiantes de primaria y secundaria que utiliza iconos en lugar de texto, código o comandos de programación, reduciendo la carga cognitiva al excluir las posibilidades de errores sintácticos para enseñar tres conceptos centrales de PC (secuencia, condicional y bucle) y cuatro habilidades (identificación y descomposición de

	problemas, construcción de algoritmos, depuración y simulación).
Beebot y Code and Go™ Robot Mouse	Robots de suelo educativos diseñados para enseñar a los niños programación y habilidades de PC. Están destinados especialmente para utilizarlos en entornos educativos para niños de edad preescolar y primaria. Ambos robots se programan de manera sencilla con comandos direccionales que los niños ingresan manualmente en el dispositivo.
Code.org	Plataforma online de programación visual centrada en la codificación por bloques, similar a Scratch, pero diseñada para enseñar programación a usuarios jóvenes con guías y tutoriales progresivos.
Compthink	App para Android diseñada para niños en edad preescolar y primeros años de primaria, que introduce conceptos computacionales a través de un entorno parecido a un juego. Ofrece tareas visuales de arrastrar y soltar divididas en 7 categorías basadas en el modelo de Brennan y Resnick (2012), que incluyen conceptos como bucles, secuencias, datos, condicionales y operaciones. Las actividades se centran en bucles, algoritmos, patrones, condicionales, pasos, instrucciones y autómatas.
EasyLogic3D	Entorno diseñado para instruir en los principios fundamentales del PC: algoritmos, abstracción, reconocimiento de patrones y descomposición. Sus características distintivas incluyen la aplicación de técnicas de aprendizaje automático para el reconocimiento emocional y la implementación de un motor gráfico en 3D.
IchigoJam	Versión de la placa Raspberry Pi que incluye un software para programar utilizando el lenguaje Beginner's All-purpose Symbolic Instruction Code (BASIC). Se pueden agregar componentes para transformarlo en un pequeño ordenador con el que programar.
KIBO Robotics	Kit de robótica diseñado para niños en edad preescolar y primaria. Utiliza bloques de programación físicos para enseñar conceptos de programación y PC de manera lúdica. Permite a los niños crear secuencias de comandos para el robot KIBO, fomentando el aprendizaje a través del juego y la creatividad.
LEGO® Mindstorms EV3 y LEGO® WeDo 2.0	Kits de construcción de robots que se utilizan para desarrollar habilidades visoespaciales, de manipulación y programación. Utilizan un entorno de programación visual y un sistema de codificación visual por bloques. A través de este sistema se pueden programar diferentes actuadores que incluyen cada kit.
Makeblock Mbot	kit STEAM de robótica educativa, para iniciarse en robótica, programación y electrónica basado en Arduino y Scratch.
Microbit	Microcontrolador que se puede programar con diferentes lenguajes de código (bloques, python, Java) para crear juegos y animaciones o para controlar un hardware. Se le pueden añadir diferentes actuadores para mejorar la experiencia.
Plastelina interactive Logic Game	Son una serie de 15 juegos de lógica basados en web dónde el usuario tiene que ir resolviendo problemas utilizando el pensamiento algorítmico.
Scalable Game Design	Entorno que utiliza un conjunto de proyectos de simulación y creación de videojuegos cada vez más sofisticados para trabajar patrones de PC.
Scratch	Sistema de programación basado en la codificación visual por bloques y la programación orientada a objetos en un entorno en 2D donde se trabajan las dimensiones del PC del modelo de Brennan y Resnick (2012). Tiene una interfaz de usuario intuitiva y sencilla que se ordena por colores en categorías de acciones. En el entorno de trabajo se muestra un escenario con el resultado del proyecto que se está trabajando y un banco de trabajo con las miniaturas o sprites que tenemos en el proyecto.
Scratch JR	Versión simplificada de Scratch especialmente diseñado para introducirlo en prelectores (5-7 años). Es un lenguaje de programación visual por bloques diseñado para introducir habilidades de codificación.
Turtle Academy	Entorno online de programación por texto basado en el lenguaje de programación LOGO. Su finalidad es enseñar los principios de programación en el lenguaje LOGO a través de una serie de lecciones o retos dónde se amplía la dificultad de las acciones y del código.
Actividades de Pensamiento Computacional *Unplugged*	Son una serie de ejercicios que no requieren el uso de dispositivos electrónicos y se centran en desarrollar habilidades de PC sin programar directamente en una computadora. Estas actividades pueden incluir juegos, rompecabezas y ejercicios que fomentan la resolución de problemas, el pensamiento lógico y la abstracción.
Virtual Environment Interactions (VENVI)	Entorno virtual que integra danza y programación y para enseñar PC. Los estudiantes crean espectáculos de danza para personajes virtuales, empleando bloques de código para diseñar movimientos de baile. La plataforma fomenta el uso de conceptos de programación, como variables y bucles, para agregar complejidad a las coreografías.

Tabla 3. Descripción de las características de los entornos de programación

En relación a la evaluación de las competencias de pensamiento computacional, el 17,5% optó por un diseño cualitativo basado principalmente en la observación o en las entrevistas en profundidad mientras que el 82,5% de los estudios utilizaron una metodología cuantitativa. La revisión arroja un total de 11 instrumentos de evaluación del PC distintos junto con otros 14 cuestionarios desarrollados ad-hoc por los propios autores del trabajo. Los cuestionarios más utilizados para evaluar habilidades o competencias de PC fueron el Computational thinking Test (CTt) desarrollado por Román-González, et al. (2017) el Computational Thinking Ability (Bebras challenge) desarrollado por Bebras (2018) y el Dr. Scratch desarrollado por Moreno-León y Robles (2015). Cada uno de estos cuestionarios evalúa competencias diferentes del PC (tabla 4), aunque todos comparten la característica de ser herramientas que no evalúan la autopercepción, sino que se

centran en evaluar la habilidad o competencias del individuo en tareas que implican el PC.

Métodos de evaluación del PC	**Competencias que se evalúan**	**% de veces utilizado**
Academic achievement test (AAT)	Abstracción Pensamiento algorítmico Reconocimiento de patrones Habilidades de programación Depuración razonamiento lógico Perspectivas computacionales	2,5
Dr. Scratch	Abstracción y descomposición de problemas Pensamiento lógico Sincronización Paralelismo Notaciones algorítmicas de control de flujo Representación de datos Interactividad del usuario	7,5
Computational thinking Test (CTt)	Creatividad Pensamiento algorítmico Cooperación Pensamiento crítico Resolución de problemas	20
Escala de habilidad de PC de Korkmaz	Creatividad Pensamiento algorítmico Pensamiento cooperativo Pensamiento crítico Resolución de problemas	2,5
CT self-efficacy perception scale (CTS)	Competencia de diseño de algoritmos Competencia de resolución de problemas Competencia de procesamiento de datos Competencia de programación básica Autoconfianza operativa	5
Computational Thinking Ability Test	Modelo abstracto Pensamiento lógico Descomposición del problema Evaluación del problema y resumen	2,5
Computational Thinking Ability (Bebras challenge)	Estructura de secuencias Estructuras de bucles Estructuras de condicionales Variables **Existe cierta variabilidad en función de la edad y la adaptación propia del autor*	10
TechCheck	Desafíos de secuenciación Acertijos de ruta más corta Serie de símbolos faltantes Descomposición de objetos Laberintos de obstáculos Forma de símbolo Acertijos Identificación de conceptos tecnológicos Problemas de simetría	2,5
CT competence test	Depuración de programa Marcar la respuesta Emparejando el programa Ingeniería inversa	2,5
Escala de Pensamiento Computacional	Secuencia de medición Algoritmo Bucle Depuración Uso de condicional	2,5
Word association test (WAT)	Dar sentido a conceptos de pensamiento computacional	2,5
Instrumentos ad-hoc	No existe homogeneidad, cada autor diseñó su propio instrumento que evaluaba distintas competencias y factores del PC	35

Tabla 4. Métodos de evaluación de competencias del PC.

3.1 Comparativa de las competencias PC trabajadas en los distintos programas

Basándonos en las competencias del PC establecidas por CSTA & ISTE (2011) para la enseñanza en Educación Primaria, en la Tabla 5 aparece representado cuales de los programas identificados en esta revisión sistemática pueden ser utilizados para trabajar cada una de las competencias incluidas en dicha definición.

	Formulación de problemas	**Resolución de problema**	**Organización lógica y análisis de datos**	**Representación de datos (abstracción)**	**Pensamiento algorítmico**	**Generalización y transferencia**
Alice	X	X	X	X	X	X
Arduino	X	X	X	-	X	-
Autothinking	X	X	X	X	X	X
CompThink App	X	X	-	X	X	-
Code	X	X	X	X	X	X
EasyLogic3D	X	X	X	X	X	X
Ichigo Jam	X	X	X	-	X	X
KIBO robotics platform	X	X	X	-	X	-
Kodu Game Lab	X	X	X	X	X	X
LEGO® Mindstorm Ev3	X	X	X	X	X	X
LEGO® WeDo 2.0	X	X	X	X	X	X
Plastelina Game	-	X	X	-	X	-
Scalable Game Desing	X	X	X	X	X	X
Scratch	X	X	X	X	X	X
Turtle Academy	X	X	X	-	X	X
Virtual Environment Interactions (VENVI)	-	-	-	X	X	X

Tabla 5. Comparativa de las características del PC (CSTA & ISTE; 2011) trabajadas en los distintos softwares/entornos/aplicaciones.

Dentro de las competencias del PC, la resolución de problemas, el pensamiento algorítmico y la organización lógica de datos se pueden trabajar con todos los softwares identificados en esta revisión sistemática, a excepción de Virtual Environment Interactions (VENVI). Sin embargo, hay otras competencias que pueden ser más difíciles de trabajar, como la representación de datos (abstracción) y la generalización y transferencia. Por ejemplo, la generalización y transferencia pueden estar limitadas a

la hora de utilizar dispositivos como Arduino, Micro:bit o KIBO, ya que utilizan un lenguaje de programación propio (código escrito o lectura de código de barras) que difícilmente es aplicable a otros entornos de programación.

4 DISCUSIÓN

En esta revisión sistemática se encontraron un total de 21 softwares, robots y entornos de programación además de varios juegos y dinámicas unplugged para trabajar el PC en niños de educación primaria. Estos softwares utilizan distintos lenguajes de codificación. El lenguaje de programación por bloques (arrastrar y soltar) es el más utilizado. Esto puede ser debido a lo sencillo de su uso, que permite a los usuarios, sin conocimiento de código, programar objetos con un resultado inmediato. Además, este tipo de entornos permiten profundizar en la programación incluyendo patrones o secuencias cada vez más complicadas, aumentando el nivel de abstracción y el pensamiento algorítmico. Al tratarse de una programación por objetos o agentes, es más sencillo el desarrollo de la generalización o transferencia, ya que se puede copiar patrones de comportamiento a los distintos objetos dentro de una misma programación e incluso a diferentes entornos de programación basados en bloques.

Desde que Papert y su equipo (1980) desarrollaron LOGO con la intención de llevar la codificación a las escuelas e introducir a los niños en el uso de las computadoras como instrumentos para el aprendizaje y para mejorar la creatividad, se han desarrollado diferentes proyectos o iniciativas para llevar el PC a la educación formal. El PC se ha promovido en los últimos años como una habilidad o competencia tan fundamental como la aritmética y la alfabetización (The Computational Thinking Study, 2019). Como se puede ver en esta revisión sistemática, desde el año 2016 ha aumentado el número de publicaciones trabajando el PC en educación primaria. Actualmente, Scratch es el software más utilizado para trabajar el PC. La preferencia en el uso de Scratch puede deberse a su accesibilidad, facilidad en el manejo y atractivo visual. Además, presenta una comunidad con millones de usuarios que comparten sus proyectos libremente y dónde se pueden reinventar estos proyectos, buscando distintas soluciones al problema que se presente.

Al igual que ha aumentado el número de programas para trabajar PC, también ha aumentado el número de instrumentos de evaluación del PC. Herramientas como Computational Thinking Ability (Bebras challenge) y Dr. Scratch son fácilmente aplicables en entornos escolares, por lo que facilitan la implementación de programas de desarrollo del PC y permiten que los docentes puedan evaluar los proyectos de sus alumnos y las competencias adquiridas en PC. Otro test disponible y que en esta revisión resultó ser el más utilizado, es el Computational Thinking test (CTt), con 28 ítems de elección múltiple, que evalúa la formulación y resolución de problemas basándose en conceptos de computación y usando la lógica del lenguaje de programación: secuencias básicas, bucles, iteraciones, condicionales, funciones y variables (Román-González et al., 2015). Cualquiera de estos métodos de evaluación refleja las competencias de PC adquiridas por el alumno y permite observar unos resultados objetivos, a diferencia de las pruebas en las que se pregunta al alumno por su percepción o las evaluaciones cualitativas.

Otro de los resultados que arroja este estudio es la gran variedad de habilidades o competencias relacionadas con el PC que son trabajadas y evaluadas por los diferentes autores. Desde que Wing (2006) definiese el concepto de PC diversos autores han arrojado distintos puntos de vista acerca del concepto, de lo que implica y cómo trabajarlo. La falta de consenso supone una limitación a la hora de desarrollar programas estatales o bajo un marco común que definan cómo trabajar y evaluar el PC en las escuelas, pues la actual heterogeneidad de propuestas y competencias que se trabajan impiden la comparación de resultados y de la efectividad de dichos programas para desarrollar competencias de PC en los niños (Bocconi et al., 2016).

La implicación de organizaciones como CSTA y ISTE y su definición consensuada del PC han ayudado a establecer, al menos parcialmente, un marco común de trabajo. En EEUU y México llevan años aplicando programas para trabajar el PC como son el ejemplo de Scalable Game Design (Repenning, 2010). En el proceso de consenso de definición del PC, también cabe destacar la labor que está realizando la UE a través del estudio CompuThink (The Computational Thinking Study, 2019), que pretende contribuir al debate sobre codificación y habilidades y competencias transversales a nivel europeo. A pesar de estas iniciativas y de la inversión pública y privada que se está realizando en la UE, existen una serie de problemas y desafíos a la hora de integrar el PC en los planes de estudio de los escolares europeos. Uno de los mayores desafíos que se destaca en la literatura es la necesidad de implicar al personal docente y fomentar la formación entre este grupo para facilitar la implementación del PC en el currículum (Zapata-Ros, 2015).

El desarrollo de las competencias del PC no debería limitarse solo a las personas que se dedican a las ciencias o ingenierías, ya que son de gran utilidad para adquirir competencias tan básicas como la resolución de problemas y el pensamiento lógico. En este sentido, la visión de Wing de que el PC es para todos muy atractiva, ya que todos pueden beneficiarse de pensar como un científico de la computación (Wing 2011, 2016). El uso de herramientas sencillas y visualmente atractivas facilitan el acercamiento de todo el mundo al PC y son el primer paso en la transformación digital, cambiando la percepción y la interacción con la tecnología.

La educación a día de hoy ya incorpora entre sus objetivos crear una sociedad digitalmente competente, sin embargo, en muchos casos carecen de un currículo y una metodología aplicable. Esta revisión pretende arrojar información acerca de los software, robots y entornos de programación que tienen a su disposición los equipos docentes, así como los posibles instrumentos de evaluación del PC y su relación con las competencias de PC establecidas por CSTA y ISTE.

Las conclusiones de este estudio están limitadas a los resultados encontrados en cuatro bases de datos, por lo que puede haber otro tipo de literatura que haya quedado excluida (editoriales, revisiones o tesis). Además, al focalizar la búsqueda en el término pensamiento computacional, otros conceptos que puedan estar relacionados tampoco se han abordado.

5 CONCLUSIONES

En general, ha crecido el interés por la implementación del PC en las aulas a pesar de las limitaciones existentes. Sin embargo, aún quedan muchos pasos por recorrer hasta la creación de un marco común y su definitiva implementación en el currículo. Para poder implementar satisfactoriamente programas que trabajen el PC en las escuelas desde edades tempranas, es fundamental

consensuar una definición, así como establecer cuáles son las habilidades o competencias que se tienen que desarrollar y su correspondiente instrumento de evaluación. Además, no se puede aplicar eficazmente un programa sin la participación de la comunidad educativa, especialmente docentes a los que habría que proporcionar una formación adecuada.

REFERENCIAS

Acevedo, H. M. V., Suarez, L. J. L., & Medina, L. D. F. (2024). Pensamiento Computacional: una competencia del siglo XXI: Revisión sistemática en Scopus. Revista Latinoamericana Ogmios, 4(9), 1-16.

Adell Segura, J., Llopis Nebot, M. Á., Esteve Mon, F. M., & Valdeolivas Novella, M. G. (2019). El debate sobre el pensamiento computacional en educación. RIED. Revista Iberoamericana de Educación a Distancia. https://doi.org/10.5944/ried.22.1.22303

Aho, A. V. (2012). Computation and computational thinking. The computer journal, 55(7), 832-835. https://doi-org.sire.ub.edu/10.1093/comjnl/bxs074

*Aksit, O., & Wiebe, E. N. (2020). Exploring force and motion concepts in middle grades using computational modeling: A classroom intervention study. Journal of Science Education and Technology, 29(1), 65-82. https://doi.org/10.1007/s10956-019-09800-z

*Allsop, Y. (2019). Assessing computational thinking process using a multiple evaluation approach. International journal of child-computer interaction, 19, 30-55. https://doi.org/10.1016/j.ijcci.2018.10.004

*Anuar, N. H., Mohamad, F. S., & Minoi, J. L. (2020). Contextualising computational thinking: A case study in remote rural sarawak borneo. *International Journal of Learning, Teaching and Educational Research, 19*(8), 98-116. https://doi.org/10.26803/ijlter.19.8.6

*Asad, K., Tibi, M., & Raiyn, J. (2016). Primary School Pupils' Attitudes toward Learning Programming through Visual Interactive Environments. *World* journal of education, 6(5), 20-26. https://doi.org/10.5430/wje.v6n5p20

Barr, V., & 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. doi:10.1145/1929887.1929905

Baser, M. (2013). Attitude, Gender and Achievement in Computer Programming. Middle-East Journal of Scientific Research, 14(2), 248-255. 10.5829/idosi.mejsr.2013.14.2.2007

Baytak, A., & Land, S. M. (2011). An investigation of the artifacts and process of constructing computers games about environmental science in a fifth-grade classroom. Educational Technology Research and Development, 59(6), 765-782. https://doi-org.sire.ub.edu/10.1007/s11423-010-9184-z

Bebras Computational Challenges Recommended by UK Bebras Challenges (2015). Retrieved from http://bebras.uk

Bebras. (2018). International Challenge on Informatics and Computational Thinking. Retrieved from http://www.bebras.org/

*Benton, L., Kalas, I., Saunders, P., Hoyles, C., & Noss, R. (2018). Beyond jam sandwiches and cups of tea: An exploration of primary pupils' algorithm-evaluation strategies. Journal of Computer Assisted Learning, 34(5), 590-601. https://doi-org.sire.ub.edu/10.1111/jcal.12266

Bers, M. U., & Resnick, M. (2015). The official ScratchJr book: Help your kids learn to code. No Starch Press.

Bers, M. U. (2017). Coding as a playground: Programming and computational thinking in the early childhood classroom. Routledge.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Exploring the field of computational thinking as a 21st century skill. Proceedings of the EDULEARN16, 16, 4725-4733. Doi: 10.21125/edulearn.2016.2136

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada (Vol. 1, p. 25). http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf

Burke, Q. (2012). The markings of a new pencil: Introducing programming-as-writing in the middle school classroom. Journal of Media Literacy Education, 4(2), 121-135. http://files.eric.ed.gov/fulltext/EJ985683.pdf

*Caballero-González, Y. A., & García-Valcárcel, A. (2020). Learning with robotics in primary education? A means of stimulating computational thinking. Education in the Knowledge Society, 21(10), 1-15. https://doi.org/10.14201/eks.21443

Cearreta-Urbieta, I. (2015). Scratch como recurso didáctico para el desarrollo del Pensamiento Computacional de los alumnos de Secundaria y Bachillerato en la asignatura de Informática y como recurso transversal en el resto de asignaturas (Master's thesis). Universidad Internacional de la Rioja (UNIR) https://reunir.unir.net/handle/123456789/3150

*Cervera, N., Diago, P. D., Orcos, L., & Yáñez, D. F. (2020). The acquisition of computational thinking through mentoring: an exploratory study. Education Sciences, 10(8), 202. 10.3390/educsci10080202

*Chen, K. Z., & Chi, H. H. (2020). Novice young board-game players' experience about computational thinking. Interactive Learning Environments, 1-13. https://doi.org/10.1080/10494820.2020.1722712

*Chou, P. N. (2020). Using ScratchJr to foster young children's computational thinking competence: A case study in a third-grade computer class. Journal of Educational Computing Research, 58(3), 570-595. 10.1177/0735633119872908

Computer Science Teachers Association (2017). CSTA. https://www.csteachers.org/page/standards

Eaton, S. E. (2017). Book review: Coe, R., Waring, M., Hedges, LV, & Arthur, J. (Eds.). (2017). Research methods and methodologies in education. Canadian Journal of Educational Administration and Policy, (184). http://cjc-rcc.ucalgary.ca/index.php/cjeap/article/view/42501/30934

*del Olmo-Muñoz, J., Cózar-Gutiérrez, R., & González-Calero, J. A. (2020). Computational thinking through unplugged activities in early years of Primary Education. Computers & Education, 150, 103832. https://doi.org/10.1016/j.compedu.2020.103832

*Hooshyar, D., Malva, L., Yang, Y., Pedaste, M., Wang, M., & Lim, H. (2021). An adaptive educational computer game: Effects on students' knowledge and learning attitude in computational thinking. Computers in Human Behavior, 114, 106575. https://doi.org/10.1016/j.chb.2020.106575

*Hsiao, H. S., Lin, Y. W., Lin, K. Y., Lin, C. Y., Chen, J. H., & Chen, J. C. (2022). Using robot-based practices to develop an activity that incorporated the 6E model to improve elementary school students' learning performances. *Interactive Learning Environments, 30*(1), 85-99. https://doi.org/10.1080/10494820.2019.1636090

International Society for Technology in Education (ISTE) y Computer Science Teachers Association (CSTA). (2011). *Operational Definition of Computational Thinking for K–12 Education*. http://www.iste.org/docs/ct-documents/computationalthinking-operationaldefinition-flyer.pdf?sfvrsn=2

International Society for Technology in Education (ISTE). (2021). Computational Thinking Competencies. https://www.iste.org/standards/computational-thinking

* Jiang, S., & Wong, G. K. (2022). Exploring age and gender differences of computational thinkers in primary school: A developmental perspective. *Journal of Computer Assisted Learning, 38*(1), 60-75. 10.1111/jcal.12591

*Jun, S., Han, S., & Kim, S. (2017). Effect of design-based learning on improving computational thinking. *Behaviour & Information Technology, 36*(1), 43-53. https://doi-org.sire.ub.edu/10.1080/0144929X.2016.1188415

*Kawada, K., Okamoto, K., Tamai, T., & Ohnishi, Y. (2019). A Study on Developmentally Appropriate Programming Education Learning Materials for Lower-Elementary School Students. *Journal of Robotics and Mechatronics, 31*(3), 441-451. https://doi-org.sire.ub.edu/10.20965/jrm.2019.p0441

*Kert, S. B., ErkoÇ, M. F., & Yeni, S. (2020). The effect of robotics on six graders' academic achievement, computational thinking skills and conceptual knowledge levels. *Thinking Skills and Creativity, 38*, 100714. 10.1016/j.tsc.2020.100714

Korkmaz, O., Cakir, R., & Ozden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior, 72*, 558–569. https://doi.org/10.1016/j.chb.2017.01.005

Korkmaz, O., & Bai, X. (2019). Adapting computational thinking scale (CTS) for Chinese high school students and their thinking scale skills level. *Participatory Educational Research, 6*(1), 10–12. http://dx.doi.org/10.17275/per.19.2.6.1

*Kyza, E. A., Georgiou, Y., Agesilaou, A., & Souropetsis, M. (2022). A Cross-Sectional Study Investigating Primary School Children's Coding Practices and Computational Thinking Using ScratchJr. *Journal of Educational Computing Research, 60*(1), 220–257. https://doi.org/10.1177/07356331211027387

Learning Science Laboratory of Peking University (2019). *Computational Thinking grade measurement*. https://ceping.gamification.org.cn/

* Leonard, J., Buss, A., Gamboa, R., Mitchell, M., Fashola, O. S., Hubert, T., & Almughyirah, S. (2016). Using robotics and game design to enhance children's self-efficacy, STEM attitudes, and computational thinking skills. *Journal of Science Education and Technology*, *25*(6), 860-876. https://doi.org/10.1007/s10956-016-9628-2

Journal of Science Education and Technology, 25(6), 860-876. https://doi-org.sire.ub.edu/10.1007/s10956-016-9628-2

*Li, X., Xie, K., Vongkulluksn, V., Stein, D., & Zhang, Y. (2021). Developing and Testing a Design-Based Learning Approach to Enhance Elementary

Students' Self-Perceived Computational Thinking. *Journal of Research on Technology in Education*, 1-24. https://doi.org/10.1080/15391523.2021.1962453

*Ma, H., Zhao, M., Wang, H., Wan, X., Cavanaugh, T. W., & Liu, J. (2021). Promoting pupils' computational thinking skills and self-efficacy: a problem-solving instructional approach. *Educational Technology Research and Development, 69*(3), 1599-1616. https://doi.org/10.1007/s11423-021-10016-5

*Melander Bowden, H. (2019). Problem-solving in collaborative game design practices: epistemic stance, affect, and engagement. Learning, Media and Technology, 44(2), 124-143. https://doi.org/10.1080/17439884.2018.1563106

* Min, S. H., & Kim, M. K. (2020). Developing children's computational thinking through physical computing lessons. International Electronic Journal of Elementary Education, 13(2), 183-198. 10.26822/iejee.2021.183

Ministry of Education (2015). Activation plan for SW education in K-12. Korea.

*Moore, T. J., Brophy, S. P., Tank, K. M., Lopez, R. D., Johnston, A. C., Hynes, M. M., & Gajdzik, E. (2020). Multiple representations in computational thinking tasks: a clinical study of second-grade students. Journal of Science Education and Technology, 29(1), 19-34. https://doi.org/10.1007/s10956-020-09812-0

Moreno-León, J., Robles, G., & Román-González, M. (2015). Dr. Scratch: Automatic analysis of scratch projects to assess and foster computational thinking. RED. Revista de Educación a Distancia, (46), 1-23. https://revistas.um.es/red/article/view/240251

*Moreno-León, J., Robles, G., & Román-González, M. (2016). Code to learn: Where does it belong in the K-12 curriculum. Journal of Information Technology Education: Research, 15, 283-303. https://doi.org/10.28945/3521

Morris, D., & Trushell, J. (2014). Computer programming, ICT and gender in the classroom: a male-dominated domain or a female preserve? Research in teacher education, 4(1), 4-9. https://doi.org/10.15123/uel.859y6

*Noh, J., & Lee, J. (2020). Effects of robotics programming on the computational thinking and creativity of elementary school students. Educational technology research and development, 68(1), 463-484. https://doi.org/10.1007/s11423-019-09708-w

*Ntourou, V., Kalogiannakis, M., & Psycharis, S. (2021). A study of the impact of Arduino and Visual Programming In self-efficacy, motivation, computational thinking and 5th grade students' perceptions on electricity. Eurasia Journal of Mathematics, Science and Technology Education, 17(5), em1960. https://doi.org/10.29333/ejmste/10842

* Özcan, M. Ş., Çetinkaya, E., Göksun, T., & Kisbu-Sakarya, Y. (2021). Does learning to code influence cognitive skills of elementary school children? Findings from a randomized experiment. British Journal of Educational Psychology, 91(4), 1434-1455. DOI:10.1111/bjep.12429

*Özmutlu, M., Atay, D., & Erdoğan, B. (2021). Collaboration and engagement-based coding training to enhance children's computational thinking self-efficacy. Thinking Skills and Creativity, 40, 100833. https://doi.org/10.1016/j.tsc.2021.100833

Papert, S. (1980). Computers for children. Mindstorms: Children, computers, and powerful ideas, 3-18.

Pérez-Marín, D., Hijón-Neira, R., & Martín-Lope, M. (2018). A methodology proposal based on metaphors to teach programming to children. IEEE Revista Iberoamericana de tecnologias del aprendizaje, 13(1), 46-53. https://doi-org.sire.ub.edu/10.1109/RITA.2018.2809944

*Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., & Pizarro, C. (2020). Can computational thinking be improved by using a methodology based on metaphors and scratch to teach computer programming to children? Computers in Human Behavior, 105, 105849. https://doi-org.sire.ub.edu/10.1016/j.chb.2018.12.027

*Pinkard, N., Martin, C. K., & Erete, S. (2020). Equitable approaches: opportunities for computational thinking with emphasis on creative production and connections to community. Interactive Learning Environments, 28(3), 347-361. DOI: 10.1080/10494820.2019.1636070

Portelance, D.J., & Bers, M.U. (2015). Code and tell: assessing young children's learning of computational thinking using peer video interviews with ScratchJr. Proceedings of the 14th International Conference on Interaction Design and Children [Conference presentation]. 10.1145/2771839.2771894

Psycharis, S., & Kotzampasaki, E. (2019). The Impact of a STEM Inquiry Game Learning Scenario on Computational Thinking and Computer Self-confidence. Eurasia Journal of Mathematics, Science and Technology Education, 15(4), em1689. https://doi.org/10.29333/ejmste/103071

*Qu, J. R., & Fok, P. K. (2021). Cultivating students' computational thinking through student–robot interactions in robotics education. International Journal of Technology and Design Education, 1-20. https://doi.org/10.1007/s10798-021-09677-3

Relkin, E., de Ruiter, L., & Bers, M. U. (2020). TechCheck: Development and validation of an unplugged assessment of computational thinking in early childhood education. Journal of Science Education and Technology. https://doi.org/10.1007/s10956-020-09831-x

*Relkin, E., de Ruiter, L. E., & Bers, M. U. (2021). Learning to code and the acquisition of computational thinking by young children. Computers & Education, 169, 104222. https://doi.org/10.1016/j.compedu.2021.104222

Repenning, A., Webb, D., & Ioannidou, A. (2010, March). Scalable game design and the development of a checklist for getting computational thinking into public schools. In Proceedings of the 41st ACM technical symposium on Computer science education (pp. 265-269). https://doi.org/10.1145/1734263.1734357

*Ríos Félix, J. M., Zatarain Cabada, R., & Barrón Estrada, M. L. (2020). Teaching computational thinking in Mexico: A case study in a public elementary school. Education and Information Technologies, 25(6), 5087-5101. https://doi.org/10.1007/s10639-020-10213-4

*Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using Scratch: an experiment with sixth-grade students. Interactive Learning Environments, 28(3), 316-327. https://doi.org/10.1080/10494820.2019.1612448

Román-González, M. (2015) Computational Thinking Test: Design Guidelines and Content Validation, in: Proc. 7th Annu. Int. Conf. Educ. New Learn. Technol. (EDULEARN 2015), 2015: pp. 2436–2444. doi:10.13140/RG.2.1.4203.4329.

*Román-González, M., Pérez-González, J. C., Moreno-León, J., & Robles, G. (2018). Can computational talent be detected? Predictive validity of the Computational Thinking Test. International Journal of Child-Computer Interaction, 18, 47-58. https://doi-org.sire.ub.edu/10.1016/j.ijcci.2018.06.004

Scalable Game Design wiki. (2015, September 14). Scalable Game Design wiki. Retrieved 11:29, March 29, 2021 from http://wiki.computationalthinkingfoundation.org/wiki/index.php?title=Scalable_Game_Design_wiki&oldid=12459.

*Soleimani, A., Herro, D., & Green, K. E. (2019). CyberPLAYce—A tangible, interactive learning tool fostering children's computational thinking through storytelling. International Journal of Child-Computer Interaction, 20, 9-23. https://doi-org.sire.ub.edu/10.1016/j.ijcci.2019.01.002

Tangney, B., Oldham, E., Conneely, C., Barrett, S., & Lawlor, J. (2009). Pedagogy and processes for a computer programming outreach workshop—The bridge to college model. IEEE Transactions on Education, 53(1), 53-60. https://doi-org.sire.ub.edu/10.1109/TE.2009.2023210

The Computational Thinking Study. (2019). EU Science Hub - European Commission. https://ec.europa.eu/jrc/en/computational-thinking

Tran, Y. (2018). Computational thinking equity in elementary classrooms: What third-grade students know and can do. Journal of Educational Computing Research, 57, 3–31. https://doi.org/10.1177/0735633117743918

*Von Wangenheim, C. G., Alves, N. C., Rodrigues, P. E., & Hauck, J. C. (2017). Teaching Computing in a Multidisciplinary Way in Social Studies Classes in School-A Case Study. International Journal of Computer Science Education in Schools, 1(2), 3-16. https://doi.org/10.21585/ijcses.v1i2.9

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.

Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366(1881), 3717-3725. https://doi-org.sire.ub.edu/10.1098/rsta.2008.0118

Wing, J. M. (2011). Research Notebook: Computational Thinking - What and Why? The Link. Pittsburg: Carnegie Mellon. https://www.cs.cmu. edu/link/research-notebookcomputational-thinking- what-and-why.

Wing, J. (2016). Computational thinking, 10 years later. Microsoft Research Blog. https:// blogs.msdn.microsoft.com/msr_er/2016/03/23/computational-thinking-10-years-later/

*Wong, G. K. W., & Cheung, H. Y. (2018). Exploring children's perceptions of developing twenty-first century skills through computational thinking and programming. Interactive Learning Environments, 28 (4), 438-450. https://doi-org.sire.ub.edu/10.1080/10494820.2018.1534245

*Xuefeng Wei, Lin Lin, Nanxi Meng, Wei Tan, Siu-Cheung Kong, Kinshuk. (2021). The effectiveness of partial pair programming on elementary school students' Computational Thinking skills and self-efficacy. Computers & Education, Volume 160, 104023,ISSN 0360-1315, https://doi.org/10.1016/j.compedu.2020.104023.

Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In Emerging research, practice, and policy on computational thinking (pp. 205-220). Springer, Cham. https://doi-org.sire.ub.edu/10.1007/978-3-319-52691-1_13

Zapata-Ros, M. (2015). Pensamiento computacional: Una nueva alfabetización digital. Revista de Educación a Distancia (RED), (46). https://revistas.um.es/red/article/vweiiew/240321

*Zhan, Z., He, W., Yi, X., & Ma, S. (2022). Effect of Unplugged Programming Teaching Aids on Children's Computational Thinking and Classroom Interaction: with Respect to Piaget's Four Stages Theory. Journal of

Educational Computing Research, 60(5), 1073–1083. https://doi-org.sire.ub.edu/10.1177/07356331211057143

* Referencia de publicación incluida en la muestra de publicaciones revisadas.

PROGRAMES DE PENSAMENT COMPUTACIONAL EN EDUCACIÓ PRIMÀRIA: UNA REVISIÓ SISTEMÀTICA

El pensament computacional (PC) descriu un tipus de pensament analític relacionat amb la informàtica i la programació. La Societat Internacional per a la Tecnologia a l'Educació (ISTE) i l'Associació de Professors de Ciències de la Computació (CSTA) han desenvolupat una definició per promoure la inclusió de PC a l'educació als EUA L'objectiu d'aquesta revisió sistemàtica és descriure els programes educatius que ensenyen PC en educació primària (rang d'edat 6-14 anys) publicats a revistes científiques indexades, seguint les pautes PRISMA. L'estratègia de cerca es va dur a terme a cinc bases de dades: Dialnet, Psycinfo, Scopus, PyscNET i ERIC. Un total de 40 articles van complir els criteris d"inclusió. Els entorns de programació més utilitzats van ser Scratch (55%), activitats unplugged (15%) i Code.org (10%). Els programaris trobats treballaven diverses de les competències de PC establertes com a bàsiques segons la definició de les establertes per CSTA & ISTE. Aquesta revisió sistemàtica recull i sintetitza per primera vegada els programes educatius que han demostrat ser útils per treballar el PC i els relaciona amb els estàndards de PC establerts per CSTA, contribuint al coneixement de les eines disponibles per utilitzar a l'àmbit educatiu.

PARAULES CLAU: pensament computacional; educació primària; programes educatius; aprenentatge de programació.

COMPUTATIONAL THINKING PROGRAMS IN PRIMARY EDUCATION: A SYSTEMATIC REVIEW

Computational thinking (CT) describes a type of analytical thinking related to computer science and programming. The International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) have developed a definition to promote the inclusion of CT in education in the US. The goal of this systematic review is to describe the educational programs that teach CT in primary education (age range 6-14 years) published in indexed scientific journals, following the PRISMA guidelines. The search strategy was carried out in the following five databases: Dialnet, Psycinfo, Scopus, PyscNET and ERIC. A total of 40 articles met the inclusion criteria. The most used programing environment were Scratch (55%), unplugged activities (15%) and Code.org (10%). The software found worked several of the core PC competencies as defined by CSTA & ISTE. This systematic review collects and synthesizes for the first time the educational programs that have proven to be useful for working on CT in children aged 6 to 14 years and relates them to the CT standards established by CSTA, contributing to the knowledge of the tools available to use in the educational field.

KEYWORDS: Computational thinking, primary education, educational programs, programming learning.

The authors retain copyright and grant the journal the right of first publication. The texts will be published under a Creative Commons Attribution-Non-Commercial-NoDerivatives License.