Education in the Knowledge Society

journal homepage http://revistas.usal.es/index.php/eks/

Ediciones Universidad Salamanca

A Proposal of Programming Didactics in Primary Education following a Gamified Approach with Educational Videogames

Propuesta de didáctica de la Programación en Educación Primaria basada en la gamificación usando videojuegos educativos

Iago Cruz-García[a], Juan Antonio Martín-García[b], Diana Pérez-Marín[c], Celeste Pizarro[d]

[a]*Departamento de Ciencias de la Computación, Arquitectura de Computadores, Lenguajes y Sistemas Informáticos y Estadística e Investigación Operativa, Escuela Técnica Superior de Ingeniería Informática, Universidad Rey Juan Carlos, Madrid, España.*
https://orcid.org/0000-0002-7855-6856 iagocruzgarcia@gmail.com

[b]*Departamento de Ciencias de la Computación, Arquitectura de Computadores, Lenguajes y Sistemas Informáticos y Estadística e Investigación Operativa, Escuela Técnica Superior de Ingeniería Informática, Universidad Rey Juan Carlos, Madrid, España.*
https://orcid.org/0000-0002-7723-9349 juanam9k@gmail.com

[c]*Departamento de Ciencias de la Computación, Arquitectura de Computadores, Lenguajes y Sistemas Informáticos y Estadística e Investigación Operativa, Escuela Técnica Superior de Ingeniería Informática, Universidad Rey Juan Carlos, Madrid, España.*
http://orcid.org/0000-0003-3390-0251 diana.perez@urjc.es

[d]*Departamento de Matemática Aplicada, Ciencia e Ingeniería de los Materiales y Tecnología Electrónica, Escuela Superior de Ciencias Experimentales y Tecnología, Universidad Rey Juan Carlos, Madrid, España.*
http://orcid.org/0000-0003-2447-8239 celeste.pizarro@urjc.es

ARTICLE INFO

Keywords:
Teaching programming
Videogame
Gamification
Primary Education

ABSTRACT

Teaching programming in Primary Education is a worldwide research area of interest. This paper's contribution is a proposal of Programming Didactics in Primary Education following a gamified approach with educational videogames. In the 2019/2020 academic year, 100 students 10-12 years old were asked to follow the Didactics achieving a significant increase in the learning of basic programming concepts as well as they provided positive comments regarding their satisfaction and motivation, validating the proposal.

Palabras clave:
Enseñanza de la programación
Videojuego
Gamificación
Educación Primaria

RESUMEN

La enseñanza de la programación en Educación Primaria es un tema de interés investigador a nivel mundial. En este artículo, la contribución se centra en una propuesta de didáctica de la programación en Educación Primaria basada en la gamificación con un videojuego educativo. En el curso 2019/2020 se realizó un experimento con un grupo de 100 estudiantes de 10 a 12 años que siguieron esta propuesta didáctica, obteniéndose mejoras significativas en el aprendizaje de los conceptos básicos de programación al tiempo que se recibieron comentarios positivos sobre su satisfacción y motivación, validándose así la propuesta realizada.

1. Introducción

La enseñanza de la programación en edades tempranas ha recibido en las últimas décadas un gran interés a nivel mundial (Computer Science Teachers Association, 2012; Kafai et al., 2014; Heintz et al., 2016; García-Peñalvo et al., 2016). Algunos autores afirman que los niños podrán mejorar su comprensión del mundo digital en el que viven, e incluso desarrollar su pensamiento computacional para poder solucionar problemas de su vida con recursos informáticos (Wing, 2006; García-Peñalvo y Mendes, 2018).

Recibido, 24/3/2021. Revisado, 8/5/2021. Aceptado, 18/05/2021. Publicado 28/10/2021
e-ISSN: 2444-8729

En este artículo, el foco está en hacer una propuesta de didáctica de enseñanza de la programación en Educación Primaria mediante un enfoque gamificado usando como recurso videojuegos educativos. La gamificación supone introducir elementos de juego en entornos que en principio no eran de juego para conseguir aumentar la motivación y satisfacción (Deterding et al., 2011a; Deterding et al., 2011b; Torres-Toukoumidis et al., 2018). Gamificar la enseñanza de la programación en otros niveles ya ha resultado en beneficios probados como el uso de juegos serios para la enseñanza de la programación en Educación Secundaria (Montes-León et al., 2020).

En particular, la propuesta se centra en el uso de un videojuego educativo combinado con lenguajes de programación con bloques (i.e. las instrucciones se pueden encajar como un puzle formando bloques de código) y empleando metáforas para simplificar los conceptos de programación relacionándolos con objetos concretos y tangibles de la vida diaria. El uso de videojuegos para educación ha demostrado desarrollar con éxito muchas capacidades de alto nivel en los estudiantes como resolver problemas, trabajar en equipo, desarrollar la visión espacial al tiempo que sirve para reducir el estrés y desarrolla habilidades motoras (Groening y Binnewies, 2019; González-González y Blanco-Izquierdo, 2012). Además, el uso de metáforas ha probado sus beneficios para enseñar programación en Educación Primaria (Pérez-Marín et al., 2018), así como el uso de lenguajes de bloques como Scratch (Resnick et al., 2009) o Blockly (Krishnamoorthy y Kapila, 2016; Baratè et al., 2017).

La hipótesis 1 (H1) es que la propuesta de enseñanza de la programación en Educación Primaria gamificada mediante el uso de videojuegos educativos, lenguajes de bloques y metáforas aumentará las ganancias de aprendizaje de los estudiantes. La hipótesis 2 (H2) es que se recibirá una retroalimentación positiva por parte de los estudiantes respecto a sus niveles de satisfacción y motivación.

Para comprobar estas hipótesis, durante el curso 2019/2020, se ha realizado un experimento con 100 estudiantes de 10 a 12 años cursando 4º, 5º y 6º de Primaria desde octubre 2019 hasta febrero 2020, antes del confinamiento. Éste impidió continuar con el registro de datos. Los resultados soportan ambas hipótesis, validando la propuesta.

El artículo está estructurado en cinco secciones: la Sección 2 revisa el estado del arte; la Sección 3 describe la propuesta, la Sección 4 presenta el experimento con los resultados obtenidos, y la Sección 5 termina el artículo con las principales conclusiones y líneas de trabajo futuro.

2. Estado de la cuestión

2.1. Enseñanza de la programación

Heintz et al. (2016) presentaron una revisión de cómo Australia, Reino Unido, Estonia, Finlandia, Nueva Zelanda, Noruega, Suecia, Corea del Sur, Polonia y Estados Unidos enseñan programación en Informática en su Educación Primaria y Secundaria. El estudio revela que normalmente la asignatura de programación suele ser obligatoria en Educación Primaria y optativa en Educación Secundaria, y muestra el alto interés que tiene la enseñanza de la programación en niveles pre-universitarios a nivel mundial. Muchos otros estudios apoyan la necesidad de explorar la enseñanza de la programación en niveles pre-universitarios y su relevancia (Computer Science Teachers Association, 2012; Kafai et al., 2014; García-Peñalvo et al., 2016; Velázquez Iturbide et al., 2018; Caballero-González y García-Valcárcel 2020).

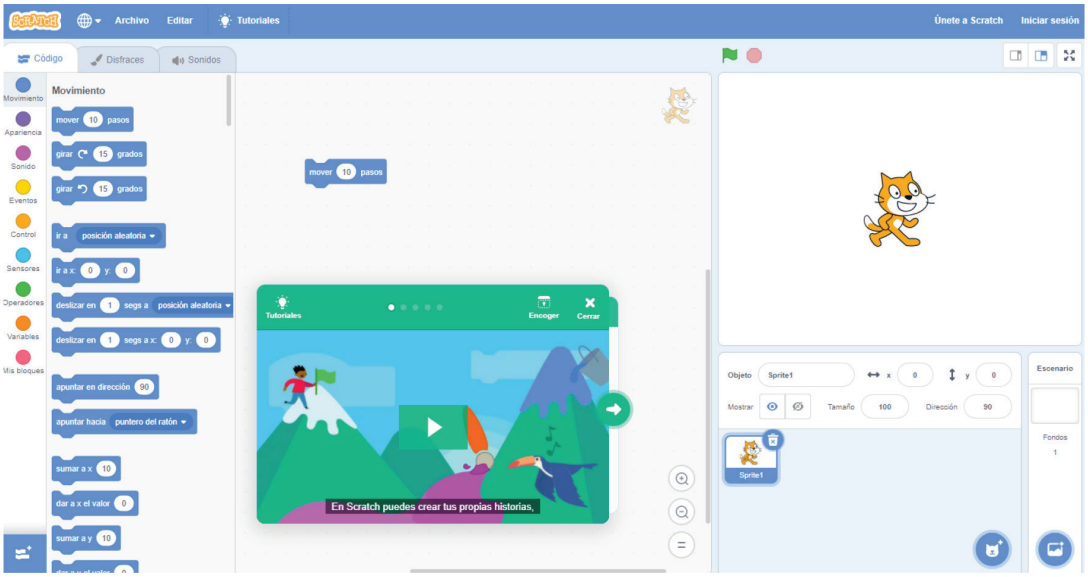
Actualmente, en España, la asignatura donde se incluye la enseñanza de la Programación es de Libre Configuración Autonómica. Esto significa que las distintas Comunidades Autónomas pueden decidir si es obligatoria en cada caso. Cada centro decide cómo organizar su docencia y el enfoque para la enseñanza de programación en Educación Primaria (Conde-Melguizo et al., 2020).

Tan solo se proporcionan para algunas Comunidades Autónomas algunas propuestas genéricas como, por ejemplo, para el caso de la Comunidad de Madrid, dentro de la ley educativa, en el Decreto 89/2014 (Comunidad de Madrid, 2014) donde se nombra la herramienta concreta Scratch dentro de los descriptores de esta asignatura, en el apartado denominado “Fundamentos de programación. Creación de pequeños programas informáticos (Scratch)”.

Sin embargo, muchos centros en España no encuentran el tiempo para integrar esta docencia entre las otras asignaturas de forma transversal, ni de establecerla como una asignatura más igual que Matemáticas o Ciencias Naturales. E incluso cuando el centro encuentra el tiempo para la asignatura, en muchos casos la falta de formación de los profesores les impide impartirla (Hijón-Neira et al., 2017; Pérez-Marín et al., 2018), siguiendo guías tan genéricas como crear pequeños programas informáticos usando Scratch (Resnick et al., 2009).

Scratch permite a los estudiantes programar mediante el arrastre de las instrucciones como si fueran piezas de un puzle que se va creando, y al ejecutarse aplica acciones sobre uno o más objetos. Por ejemplo, en la Figura 1, al ejecutar la instrucción "mover 10 pasos", el gato se moverá 10 pasos.

Figura 1. Ejemplo de pantalla de Scratch.



Este enfoque basado en la enseñanza de la programación usando lenguajes de bloques en un entorno multimedia se está usando a nivel mundial con buenos resultados porque motiva a los niños al estar en un entorno multimedia y evitando errores de sintaxis ya que los niños no llegan a teclear ninguna instrucción en el sistema (Brennan y Resnick, 2012; Ouahbi et al., 2015).

Otros enfoques para la enseñanza de Programación y el desarrollo de su pensamiento computacional incluyen usar robots Lego WeDo o Mindstorms (Sović et al., 2014), interacción en pseudocódigo con compañeros de aprendizaje emocionales (Ocaña et al., 2020; Morales-Urrutia et al., 2021); e incluso enfoques "desenchufados" en los que no se usa tecnología (Brackmann et al., 2016) usándose cuentos, actividades, juegos y ejercicios gratuitos de sitios como Code.org.

2.2. Gamificación

La gamificación supone introducir elementos de juego en entornos que en principio no eran de juego para conseguir aumentar la motivación y satisfacción (Deterding et al., 2011a; Deterding et al., 2011b).

En los estudios realizados, los ambientes gamificados suelen conseguir mejores resultados en educación que cuando no existe esta gamificación, siempre y cuando el diseño se haga correctamente. Esto es, introducir elementos de recompensa, mecanismos de juego, y realizar un diseño global, y no solo aplicar elementos aislados (Groening y Binnewies, 2019).

Para que la gamificación sea efectiva en la educación es necesario implementar mecanismos como objetivos concretos que se puedan conseguir por el jugador (Øygardslia y Aarsand, 2018) y retos con niveles incrementales de dificultad (Dondlinger, 2007).

La gamificación aplicada a la educación es un tema de gran interés investigador porque se comprueba que los estudiantes no solo mejoran su nivel de motivación sino también sus calificaciones (Denny, 2013; Domínguez et al., 2013). Además, esto sucede, en todos los niveles, desde Educación Primaria (Lee et al., 2004), Educación Secundaria (Kebritchi et al., 2010), Educación Superior (Coller y Shernoff, 2009) y en diversas áreas de conocimiento como Biología (Mcclean et al., 2001), métodos numéricos (Coller y Shernoff, 2009), electromagnetismo (Squire, 2002) y también programación (Moreno, 2012; Astudillo et al., 2015; Talingdan y Llanda, 2019; Shahid et al., 2019).

3. Propuesta de didáctica de la programación en Educación Primaria

3.1. Visión global

Esta propuesta se centra en dotar a alumnos de tercer ciclo de primaria (entre 10 y 12 años) de los conocimientos más básicos, los fundamentos de programación: entrada y salida (la capacidad de un sistema para emitir o recibir información), estructuras condicionales (conocer las posibles ramificaciones de un problema y sus soluciones) y estructuras de repetición o bucles (realizar tareas repetitivas).

Habrá que tener en cuenta una serie de características que tiene la Educación Primaria en el tercer ciclo: las sesiones constan de 50 minutos, las motivaciones de un alumno de 10-12 años, su reducida capacidad de atención y utilizar un lenguaje familiar y concreto, pues la abstracción todavía es un concepto que están desarrollando (Piaget, 1971).

Por lo tanto, si el paradigma usual en programación suele conllevar (Gómez-López, 2002):

1. Explicación del concepto y usos.
2. Concepción teórica.
3. Concepción abstracta.
4. Concepción práctica.
5. Trabajar sobre el concepto para asentar información.

Se propone simplificar esta estructura, sin concepción abstracta, con una teoría más básica, sin profundizar: explicar el concepto brevemente con ejemplos cotidianos (metáforas) y trabajar sobre ese concepto múltiples ocasiones para asentar lo aprendido.

Esto es la concepción más clásica, tanto en educación como en la enseñanza en la programación, donde el profesor recita la teoría y los alumnos absorben la información siguiendo la tradicional lección magistral (De Miguel, 2006). La propuesta es utilizar metodologías activas, aquellas donde el propio alumno es partícipe en su aprendizaje, descubriendo o siendo guiado. Esto permite el uso de nuevos estándares como alumno investigador (Adell et al., 2015) y gamificando el aula de Primaria (Contreras, 2018; Gil-Quintana y Prieto Jurado, 2020).

En todo caso, la gamificación en el aula de Primaria no es nueva. En las últimas décadas, como se ha visto en el estado del arte, se ha estado estudiando, trabajando e investigando la mejor manera de acercar un comportamiento lúdico a los entornos educativos. No hay que confundirlo con jugar en el aula sin objetivo, sino como un apoyo en la formación, pues el objetivo es seguir las pautas de diseño de juegos, concretamente las mecánicas, para conseguir transmitir los conocimientos a los estudiantes. Esto es, establecer objetivos a corto, medio y largo plazo, fomentar la cooperación o competición (dependiendo del objetivo), establecer desafíos de dificultad incremental creando una sensación de progreso gratificante, recompensar los hitos y la capacidad de observar una evolución constante en el aprendizaje.

Con el objetivo de enseñar programación siguiendo un enfoque gamificado, se presenta esta propuesta:

1. Presentar brevemente los conceptos teóricos, acercándose al lenguaje de los estudiantes. Se aconseja utilizar paralelismos y metáforas de la vida cotidiana para facilitar su comprensión.
2. Realizar una serie de ejercicios guiados, de dificultad incremental, donde los alumnos tienen que intentar predecir los siguientes pasos.
3. Proponer ejercicios sencillos para que los estudiantes practiquen. A medida que los estudiantes vayan terminando los ejercicios iniciales más sencillos, se aconseja presentarles otros ejercicios más complejos o premiarles con tiempo libre.

Las sesiones transcurren bajo un enfoque constructivista (Piaget, 1971), según el cual, el profesor guía al alumno al principio para que este pueda continuar el aprendizaje con ejercicios posteriormente. Además, se plantea potenciar este aspecto con gamificación con desafíos y recompensas.

Es importante también destacar que, para poder impartir estos contenidos, los profesores deben recibir formación en la enseñanza de la programación. El Instituto Nacional de Tecnologías Educativas y de Formación de profesorado (INTEF) ofrece cursos para que los profesores reciban estos contenidos y puedan impartirlos en sus clases. También, recientemente en los Grados de Educación Infantil y Primaria se está incorporando en las asignaturas de Informática y Competencia Digital Docente (o TIC en la Educación) temas y prácticas para que los profesores adquieran las nociones básicas de programación y su didáctica que pueden seguir reforzando con los cursos del INTEF durante su carrera profesional.

3.2. Contenidos

Los contenidos que se propone que se deben cubrir son los propios de un curso de introducción a la programación (Mathieu, 2014):

- Conceptos básicos de programación: variable, memoria, instrucción, secuencia y programa.
- Entrada/Salida.
- Estructuras condicionales.
- Estructuras de repetición (bucles).

La Tabla 1 muestra un ejemplo práctico de sesiones con actividades para conocer los conceptos básicos previos a Entrada/Salida que son: variable, memoria, instrucción, secuencia y programa a realizar durante al menos dos semanas para que los conceptos puedan ir siendo asimilados por parte de los estudiantes.

Tabla 1. Propuesta de sesiones para enseñar los conceptos previos a Entrada/Salida.

Concepto	**Sesión**	**Actividad**
Variable, memoria, instrucción, secuencia y programa	1	Se presentan los conceptos con paralelismos y metáforas del día a día. Por ejemplo, la instrucción sería una acción cotidiana (Lavarse los dientes) y una secuencia sería el conjunto de acciones realizadas (Levantarse de la cama, hacer la cama, desayunar…)
	1	Se hacen ejercicios sobre papel para consolidar los conocimientos. Se usa la preparación de un pastel, pues con la cocina es muy sencillo realizar paralelismos (ingredientes como variables, por ejemplo).
	2	Se repasan los conceptos con más ejemplos cotidianos.
	2	Se muestra un entorno de programación con lenguajes de bloques para familiarizarse con el entorno, enseñando donde se encuentran las opciones que se van a utilizar. Se deja un tiempo para explorar.
	3	Se repasan los conceptos una vez más, esta vez con los nombres más técnicos, pero siguiendo con metáforas cotidianas. Se presenta Entrada/Salida de manera superficial.
	3	Se sigue trabajando con ejercicios, esta vez con mímica, para mostrar que las acciones se pueden reducir a variables e instrucciones incluso en el día a día.

La asignatura en la que se podría integrar estos contenidos en Educación Primaria en España es de Libre Configuración Autonómica. Esto implica que cada Comunidad Autónoma puede decidir si es obligatoria o no. Por ejemplo, en la Comunidad de Madrid la asignatura sería "Tecnología y Recursos Digitales para la mejora del aprendizaje". En el caso de centros donde esta asignatura no se oferte al ser no ser obligatoria, se podrían impartir estos contenidos de forma transversal al resto de asignaturas o como actividad extraescolar.

3.3. Recursos

Para el desarrollo de las lecciones, se propone combinar los siguientes recursos con el fin de enfocar la enseñanza de programación desde distintos enfoques y, animar a los estudiantes a seguir programando, aprendiendo de forma lúdica, variada y manteniendo su motivación durante todas las sesiones.

Recurso 1. Metáforas. Durante el desarrollo de clases de 50 minutos de duración, se plantea a los alumnos un problema a solucionar mediante un script con instrucciones sencillas y concretas, se pueden ver ejemplos de scripts en Pérez-Marín et al., (2018). Se plantean estos problemas mediante el uso de metáforas, por ejemplo, fijando como objetivo preparar un plato de comida. Así, se requiere establecer una lista de ingredientes con sus respectivas cantidades (metáfora aplicada para la declaración de variables y asignación de su valor) y determinar una secuencia de instrucciones ordenada para llevar a cabo la elaboración (dando así a entender la importancia del orden de las instrucciones con el fin de obtener un funcionamiento correcto y esperado en los programas). El profesor debe intentar ser ameno en su explicación usando multitud de ejemplos e incluso llevando al aula los utensilios de cocina para que los estudiantes pueden manipularlos con ejercicios de *role-playing* en el que teatralizan las metáforas para integrarlas mejor.

Recurso 2. Lenguajes de programación de bloques. Se recomienda el uso de un entorno de programación multimedia con lenguaje simplificado como Scratch o Blockly. Desde el punto de vista gamificado, se propone a

los niños que los programas a crear sean juegos. Además, se les recompensará cada vez que creen un juego más complejo según los conceptos realizados, con insignias y medallas que quedarán visibles en la clase.

Recurso 3. Videojuegos educativos. Se propone también el uso de al menos un videojuego educativo de programación para satisfacer las necesidades de los estudiantes que quieran aprender a programar de una forma más independiente y lúdica. La idea es que tanto los estudiantes usen el videojuego para aprender a programar, como que el profesor muestre el videojuego e incluso otros ejemplos de videojuegos (profesionales como desarrollados por estudiantes), para motivar a los estudiantes en el aprendizaje de la programación. Además, se puede explicar que dichos videojuegos han sido creados a partir de las instrucciones que ellos mismos han estado estudiando.

4. Experiencia práctica

4.1. Objetivos

Los objetivos de la propuesta son: i) introducir conceptos de programación en edades tempranas para reforzar y mejorar la comprensión de las tecnologías actuales. Se convierte en una necesidad que los estudiantes entiendan los conceptos básicos del desarrollo, pues aunque su futuro profesional pueda no estar destinado a programar, las habilidades que se atribuyen a esta práctica como pensamiento abstracto, lógica y resolución de problemas, serán de gran utilidad en cualquier entorno (Wing, 2006); y ii) aplicar entornos gamificados para el aprendizaje, la ludificación de los entornos educativos es una manera de mantener a los jóvenes motivados a lo largo de su carrera estudiantil. Además, puede fomentar la cooperación y por ello la tolerancia, transmitiendo valores positivos a los estudiantes (Groening y Binnewies, 2019; González-González y Blanco-Izquierdo, 2012).

Las preguntas de investigación con sus hipótesis asociadas son:

P1. ¿La propuesta de didáctica de la programación siguiendo un enfoque gamificado aumentará las notas de los estudiantes de forma significativa?

H1. La propuesta de enseñanza de la programación en Educación Primaria mediante el uso de metáforas, lenguajes de bloques y videojuegos aumentará las ganancias de aprendizaje de los estudiantes.

P2. ¿Los estudiantes tendrán una visión positiva de este proceso de aprendizaje?

H2. Se recibirá una retroalimentación positiva por parte de los estudiantes respecto a sus niveles de satisfacción y motivación al seguir esta propuesta gamificada.

4.2. Participantes y contexto

Los participantes fueron 100 estudiantes entre 10 y 12 años cursando 4º, 5º y 6º de Educación Primaria en centros educativos públicos de la Comunidad de Madrid en España. El 46% de los estudiantes fueron niños y el 54% fueron niñas.

Este centro no tiene la asignatura "Tecnología y Recursos Digitales para la mejora del aprendizaje", pero el director sí tiene interés en que los estudiantes del último ciclo de Primaria puedan aprender a programar por lo que habló con los profesores de estos cursos para que cedieran parte del tiempo de sus asignaturas para conseguir tener una hora de programación semanal desde el inicio de curso en septiembre 2019 hasta febrero 2020. No se pudo continuar la experiencia por causa de la COVID-19 (García-Peñalvo et al., 2021), que impidió también la recogida de cuestionarios finales que estaba prevista para el mes de junio si se hubiese podido completar el curso académico.

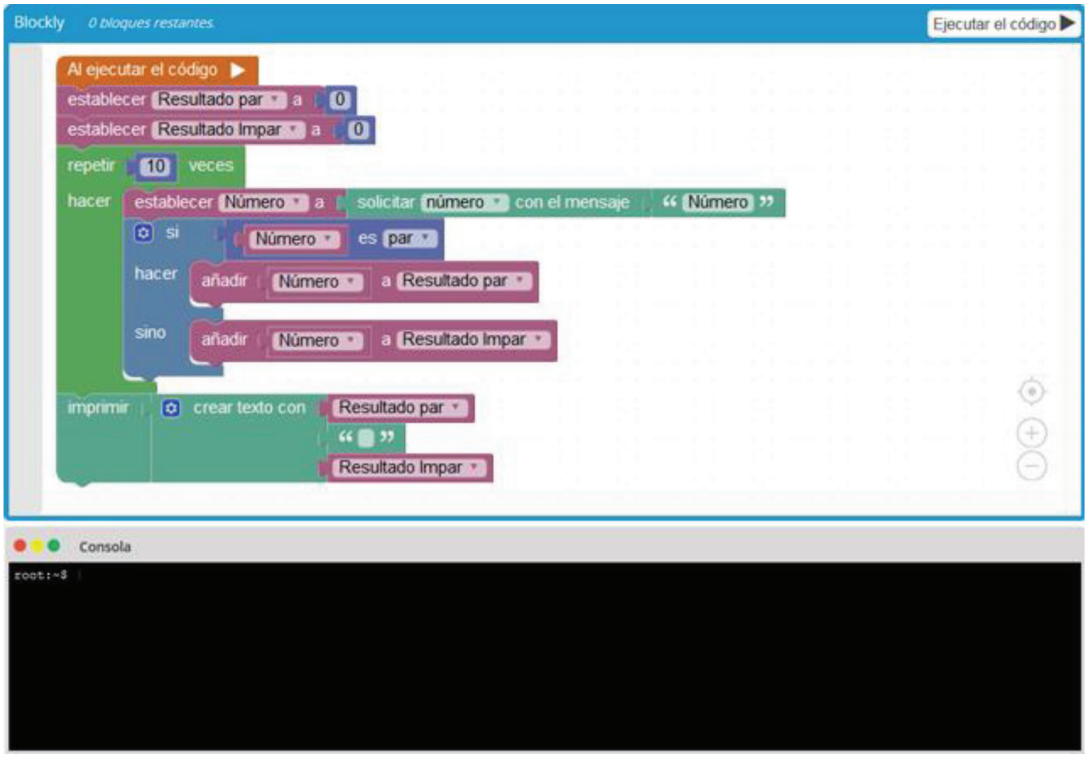
Todos los estudiantes participaron de forma voluntaria y anónima. No se recogió ningún dato personal que revelase su identidad y se siguieron todas las normas éticas de respeto a los niños y a su formación. No se premió en ningún momento la participación en el estudio con subida de notas por parte del colegio, ni se permitió la grabación de observaciones directas de los estudiantes usando los sistemas, ni la reproducción de fotografías ni videos.

4.3. Recursos

Además de los scripts para la enseñanza de la programación basada en metáforas descritos en Pérez-Marín et al. (2018) y que no se reproducen aquí por falta de espacio, como entorno de programación con lenguaje de

bloques se escogió Blockly porque como Scratch es un lenguaje de programación visual de código abierto en el que las instrucciones toman forma de bloque y se pueden formar programas ensamblando unos bloques con otros a modo de puzle, pero además Blockly cuenta con una consola en la que se muestra el código en texto. La Figura 2 muestra un ejemplo de pantalla de Blockly.

Figura 2. Ejemplo de pantalla de Blockly.



El uso de Blockly permite aplicar técnicas de gamificación, mediante objetivos concretos y retos con dificultad incremental. Al ser Blockly un lienzo abierto sin ningún tipo de guía para su uso, se requiere que el profesor proponga estos objetivos al estudiante y vaya explicando los pasos necesarios a lo largo de la lección.

Como videojuego se eligió DevelopLearning (ver Figura 3) que se desarrolló específicamente para implementar la propuesta siguiendo un Diseño Centrado en el Usuario (Lorés, 2002) para tener en cuenta la opinión de los estudiantes. Los estudiantes solicitaron tener como sistema de ayuda del juego un perro, al que se ha llamado Binary, que va guiando al jugador (representado por un avatar humanoide) en los retos a superar mediante la programación en pseudocódigo. Esta mascota muestra textos a los jugadores, de manera que no se encuentran en un entorno en blanco como ocurre en Scratch o Blockly que, pese a contar con tutoriales, no presentan un objetivo o indicaciones precisas sobre qué hacer.

Figura 3. Ejemplo de pantalla de DevelopLearning.

En los niveles inferiores del juego, el código lo muestra el videojuego, mientras que en los niveles superiores el reto aumenta y son los propios estudiantes quienes deben escribir el código para superar los retos incrementales que se les van presentando en los diferentes mundos por los que se van moviendo.

Cuando se inicia la aplicación, el alumno encontrará un menú principal compuesto de los siguientes apartados:

- Menú de jugar donde podrá acceder al tutorial y a los diferentes mundos y sus niveles.
- Menú de opciones para poder cambiar los colores de la interfaz dentro de los minijuegos.
- Menú de ayuda para consultar conceptos de manera más extensa con ejemplos. Este menú es accesible dentro de los propios minijuegos también.
- Botón de salir para cerrar la aplicación.

La interfaz del videojuego se realizó de manera que pudiera ser utilizada sin el uso de ratón y teclado simultáneo. Es decir, cuando el usuario interacciona con los menús no necesita usar el teclado y cuando se encuentra en un minijuego, no necesita el ratón. Tampoco necesita conexión a Internet. Esto se decidió por las características de los equipos informáticos y la conexión a Internet que tienen algunos colegios, para que se pudiera usar en el máximo posible de centros independientemente de la calidad de la conexión a Internet y los dispositivos que tuvieran.

Existen tres mundos para cada uno de los conceptos principales a enseñar: entrada/salida, condicionales y bucles. Todos los mundos comparten la misma estructura: cuatro minijuegos (descripciones, elegir, reordenar y rellenar), un nivel de repaso y un jefe como prueba final, según solicitaron los estudiantes de los colegios que sirvieron de usuarios para completar el Diseño Centrado en el Usuario (Lorés, 2002) de DevelopLearning. La Figura 4 muestra un ejemplo del menú de selección de los minijuegos para el primer mundo de entrada/salida. Dentro de cada minijuego (ver Figura 5), la interfaz es la misma: la consola con el código arriba a la derecha y, si hay algún tipo de explicación, abajo en el centro el cuadro de texto. Cada vez que se completa un minijuego se colorean estrellas como parte del enfoque gamificado de recompensas.

Figura 4. Ejemplo de pantalla con los minijuegos de entrada/salida en DevelopLearning.

La Tabla 2 muestra todos los juegos categorizados según el nivel al que corresponden.

En el minijuego de *descripciones,* el jugador deberá leer atentamente las descripciones de la consola en la esquina superior derecha e introducir en el pozo la palabra que corresponda. Si acierta, el cubo permanecerá en el pozo. Deberá realizar esta acción cinco veces en total. Si escoge el bloque equivocado, éste y todos los que estén dentro del pozo saldrán despedidos (en una parábola para evitar volver a caer en el agujero) y deberá volver a empezar.

Esto se basa en que, al comenzar un aprendizaje, se debe prestar atención a los conceptos que se están estudiando. Una forma de conseguirlo es describir aquellas palabras propias o con un significado diferente en el contexto actual para entender así las explicaciones y enseñanzas posteriores.

En el minijuego de *elegir*, el jugador debe discernir la salida del programa, cayendo por el agujero, de tres posibles. Si falla, volverá al inicio de ese piso. Si acierta, pasará al siguiente piso, hasta superar el nivel. La intención es mejorar el pensamiento abstracto, para entender escenarios posibles y adelantarse a las soluciones para independizar los conocimientos del contexto, es decir, no memorizar ejercicios.

Figura 5. Ejemplos de los minijuegos: a) descripciones; b) elecciones; c) ordenar; d) rellenar; e) repaso; y, f) jefe.

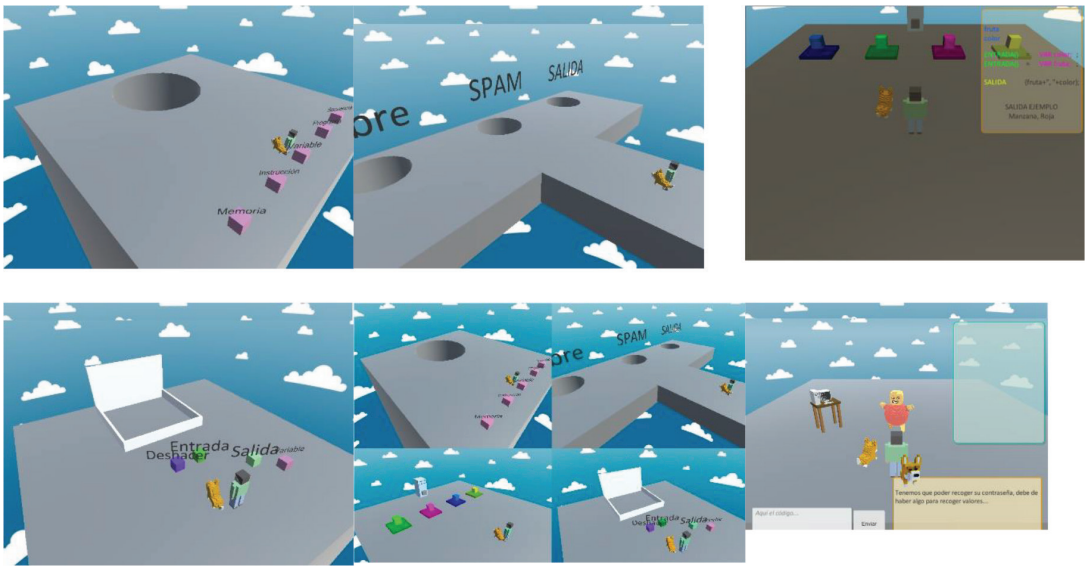


Tabla 2. Mundos y juegos en DevelopLearning.

Mundo 1. Entrada/Salida	Minijuegos: descripciones, elegir, ordenar y rellenar Nivel de repaso y jefe.
Mundo 2. Condicionales	Minijuegos: descripciones, elegir, ordenar y rellenar Nivel de repaso y jefe.
Mundo 3. Bucles	Minijuegos: descripciones, elegir, ordenar y rellenar Nivel de repaso y jefe.

En el minijuego de *reordenar*, el jugador debe colocar los bloques en los pedestales que correspondan con la posición en la consola. El orden de la consola y los pedestales coinciden con la lectura occidental: la consola de arriba a abajo y los pedestales de izquierda a derecha. Asimismo, a cada bloque le corresponde un color idéntico a la porción de consola que se ordena. El usuario debe fijarse en la consola arriba a la derecha y los colores de los bloques de texto que debe ordenar, pues corresponden a los colores de los bloques y las plataformas situadas en la escena. De izquierda a derecha, deberá colocar los bloques en sus posiciones equivalente en la consola, de arriba abajo. Si está seguro del resultado, pulsará 'F' cerca del botón en el fondo del nivel para comprobarlo. Si falla, los bloques se desordenarán. Una habilidad importante para resolver problemas es la capacidad de organización de los elementos que lo componen. Igual que en matemáticas lo primero que se pide en un ejercicio es agrupar los datos conocidos o en análisis morfosintáctico localizar sujeto y predicado primero, en programación se debe estructurar de manera lógica el programa con las variables para las instrucciones posteriores.

En el minijuego de *rellenar,* el jugador debe introducir los bloques en el orden establecido. Si acierta, se iluminarán en verde, si no, en rojo y deberá repetirlo. Esto intenta emular los ejercicios usuales a la hora de aprender el lenguaje de completar huecos vacíos con una palabra de un conjunto para que la oración tenga sentido. Este minijuego busca complementarse con los anteriores, pues en este punto el usuario debería entender la estructura y las funciones de las instrucciones.

En el minijuego de *repasar*, el jugador recorrerá los cuatro niveles anteriores de manera reducida y en orden. Es decir, deberá completar los minijuegos de *descripción, elegir, reordenar* y *rellenar* para superar estos niveles. En este nivel aparecerán todos los minijuegos anteriores, en versiones reducidas, para asentar lo aprendido y prepararse para la fase de final con el jefe. Se realiza para asemejarse a la preparación previa a un examen de un estudiante, para recordar lo aprendido y poder afrontar la prueba con mayor éxito.

En el minijuego del *jefe*, el jugador tendrá que atender a las indicaciones del perro ayudante Binary y del jefe para escribir código manualmente para poder avanzar. Si el jugador se equivoca se le ofrecerá una pista básica, una guía de qué es lo que se le pide. Si vuelve a fallar, se le otorgará una segunda pista más explícita. Con otro fallo, el jugador deberá repetir el nivel. En estos niveles, el jugador debe escribir, en el área designada para ello, las instrucciones para completar el programa necesario, cuando se le indique, hasta finalizar el nivel. A lo largo

de esta fase, si el jugador falla se le ofrecerá una pista para saber que debería escribir. Si vuelve a fallar, se le dará otra más explícita. En el tercer error, deberá repetir el nivel.

Este minijuego se inspiró en los exámenes y las pruebas de conocimientos finales para poner en una ligera presión al alumno con el fin de abordar la situación de una manera más lúdica. Intentando cumplir así la necesidad de tener objetivos concretos y retos para conseguir buenos resultados con el uso de videojuegos.

4.4. Instrumentos y procedimiento

El primer instrumento utilizado fue un test de conocimientos en el que se pedía a los estudiantes que realizaran las siguientes tareas:

- Implementación de una instrucción de salida.
- Implementación de una instrucción de entrada y una de salida.
- Implementación de instrucciones de salida determinadas por bloques condicionales.
- Implementación de un bucle finito, con instrucciones de salida.

Se realizaron tres tests, todos ellos cubriendo las tareas anteriormente indicadas:

- El primer test se realizó en octubre de 2019 a modo de pre-test, para comprobar el conocimiento previo de los estudiantes en la materia.
- El segundo test se realizó en diciembre de 2019, tras cuatro sesiones, para determinar el progreso hasta la fecha. Las cuatro sesiones se habían centrado en dar a conocer los conceptos básicos de programación: variable, memoria, instrucción, secuencia y programa. Además, se trabajó en ejercicios que introducían el flujo de entradas y salidas de un programa mediante ejemplos basados en lenguaje natural.
- El tercer test se realizó en febrero de 2020, con el objetivo de medir la evolución de las notas de los estudiantes, y también se pidió a los profesores su impresión por observación directa del nivel de satisfacción y motivación que tenían los estudiantes con la forma en que habían aprendido a programar, así como la recogida de algunos comentarios de los estudiantes.

4.5. Resultados

A partir de las notas obtenidas en los tests realizados se observó que los estudiantes partían de un nulo conocimiento inicial. Posteriormente, fueron incapaces de contestar correctamente en diciembre tan solo con cuatro sesiones, y el aprendizaje significativo se registró en el último test que es cuando los estudiantes fueron capaces de realizar los programas solicitados. La Tabla 3 muestra los valores de la media, mediana y desviación típica de las puntuaciones obtenidas por los estudiantes en los tres tests.

La Figura 6 muestra los diagramas de cajas referente a la puntuación en los diferentes tests realizados en octubre 2019, diciembre 2019 y febrero 2020 ilustrando los datos reflejados en la Tabla 3. Como puede observarse, las puntuaciones obtenidas por todos los estudiantes en octubre fue cero. Tanto en diciembre como febrero hay algunos valores atípicos, referidos a algunos alumnos que obtuvieron un valor en los tests atípicamente alto. En estos dos meses, como puede verse reflejado en el valor de la mediana, el 50% de los alumnos tenía una nota igual a cero en el caso de diciembre o menor a 0,25 en el caso de febrero. Sin embargo, la dispersión de las notas en torno a la media es bastante similar. Este valor medio aumenta en las notas de febrero.

Para comparar las notas en estos tres meses se usa un test no paramétrico debido a la falta de normalidad y homocedasticidad de los distintos grupos de datos resultantes de cada fecha evaluatoria. Debido a las características particulares de este estudio, el test de Friedman para varias muestras apareadas es la opción escogida. La Tabla 4 recoge la diferencia significativa entre ambos meses (p=0,000).

Tabla 3. Media, mediana y desviación típica de las puntuaciones obtenidas por los estudiantes en los tests.

n=100	$\bar{x}$	Med	sd
Octubre	0,00	0,00	0,00
Diciembre	0,29	0,00	0,41
Febrero	0,32	0,25	0,42

Figura 6. Boxplots para las puntuaciones obtenidas.

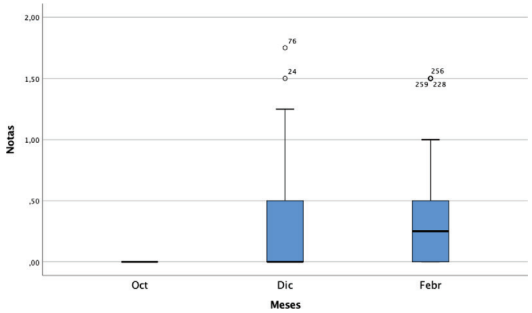


Tabla 4. Test F de Friedman para muestras apareadas. Valores de rango promedio y significación estadística.

	Rango Promedio	Chi-cuadrado	gl	p-valor
Octubre	1,51	60,793	2	0,000
Diciembre	2,24			
Febrero	2,25			

Desde el punto de vista cualitativo, a lo largo de las lecciones impartidas durante el curso, por observación directa, se pudo observar un gran interés por parte de los participantes. Las clases eran dinámicas y participativas. Además, siguiendo el Diseño Centrado en el Usuario del videojuego educativo, con la retroalimentación proporcionada por los alumnos se pudo ir mejorando el diseño llegando a satisfacer con éxito los requisitos del diseño planteado. El juego en su versión final fue recibido positivamente y con gran aceptación. Algunos comentarios realizados por los estudiantes, según fueron recogidos por los profesores, fueron los siguientes:

- Respecto al minijuego de rellenar: *"es divertido porque lanzo los bloques".*
- Respecto al minijuego de las descripciones: *"es divertido, aunque falle porque las piezas salen volando".*
- En general: *"nos gusta más aprender jugando que solo escuchando las clases".*

Es destacable como la mayoría de los comentarios incluían la palabra "divertido", "gustar" o "encantar" denotando cómo los estudiantes apreciaban la gamificación en la enseñanza y la preferían respecto a otras alternativas didácticas. Incluso enfatizando como lo hacían varios estudiantes en sus comentarios no tener miedo a equivocarse, y sentir que podían intentar la tarea varias veces hasta que la conseguían.

En general, los estudiantes mantuvieron la motivación durante todo el curso, asistiendo contentos a clase, estando atentos durante toda la sesión, preguntando dudas y siendo partícipes de su propio aprendizaje.

5. Conclusiones

Se valida la propuesta de didáctica de enseñanza de la programación siguiendo un enfoque gamificado combinando el uso de metáforas, lenguajes de bloques y videojuegos al haber obtenido una mejora significativa en las notas de los estudiantes cuando se les pide hacer programas que involucren los conceptos de entrada/salida, condicionales y bucles, a la vez que los estudiantes se han mantenido motivados alcanzando un alto nivel de satisfacción al conseguir ir realizando las tareas que se le iban pidiendo de una forma lúdica y sin temor a equivocarse.

La mayor limitación que ha tenido este estudio es que a partir de marzo 2020 la pandemia por COVID-19 impidió que se pudiera continuar con las sesiones y se paró el registro de datos. En el curso 2020/2021 se ha intentado volver a los colegios, pero ha sido imposible, incluso habiendo clases presenciales, se intenta minimizar cualquier posible riesgo impidiendo el acceso a cualquier persona que no sea del equipo docente o estudiante del centro.

Como trabajo futuro, una vez validada la propuesta, se quiere formar a los profesores para que puedan llevar la propuesta al aula sin necesidad de apoyo externo y, por lo tanto, que pueda ser aplicable incluso durante

la pandemia por COVID-19. El objetivo es aliviar la carencia detectada en formación del profesorado para que puedan seguir los scripts de las metáforas y usar entornos como Scratch o Blockly con lenguajes de bloques combinados con videojuegos con los minijuegos propuestos.

En general, la enseñanza de la programación en niveles preuniversitarios tiene múltiples beneficios que promueven que se siga trabajando en la investigación de propuestas didácticas que integren en las aulas los resultados encontrados por los investigadores en sus estudios científicos.

Referencias

Adell, J., Mengual-Andrés, S. y Roig-Vila, R. (2015). Presentación del Monográfico. Webquest: 20 años utilizando Internet como recurso para el aula. *Edutec. Revista Electrónica de Tecnología Educativa*, *52*, a298. https://doi.org/10.21556/edutec.2015.52.622

Astudillo, G., Bast, S. y Willging, P. (2015). Enfoque basado en gamificación para el aprendizaje de un lenguaje de programación. *Virtual Educación y Ciencia*, *12*(7), 125-142.

Baratè, A., Formica, A., Ludovico, L. A. y Malchiodi, D. (2017). Fostering computational thinking in secondary school through music-an educational experience based on Google Blockly. En *International Conference on Computer Supported Education* (pp. 117-124). https://doi.org/10.5220/0006313001170124

Brackmann, C., Barone, D., Casali, A., Boucinha, R. y Muñoz-Hernandez, S. (2016). Computational thinking: Panorama of the Americas. En *International Symposium on Computers in Education (SIIE)*. https://doi.org/10.1109/SIIE.2016.7751839

Brennan, K. y Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. En *Proceedings of the 2012 annual meeting of the American Educational Research Association* (pp. 1-25).

Caballero-González, Y. A. y García-Valcárcel, A. (2020). ¿Aprender con robótica en Educación Primaria? Un medio de estimular el pensamiento computacional. *Education in the Knowledge Society*, *21*, Article 10. https://doi.org/10.14201/eks.21443

Coller, B. y Shernoff, D. (2009). Video game-based education in mechanical engineering: a look at student engagement. *International Journal of Engineering Education*, *25*(2), 308–317.

Computer Science Teachers Association. (2012). Computer Science K–8: Building a Strong Foundation. https://bit.ly/3kjzCS1

Comunidad de Madrid. (2014). Decreto 89/2014, de 24 de julio, del Consejo de Gobierno, por el que se establece para la Comunidad de Madrid el Currículo de la Educación Primaria. Boletín Oficial de La Comunidad de Madrid, 175, 10–89.

Conde Melguizo, R., Vega-Barbas, M. y García-Vázquez, C. (2020). Analizando el auge de Scratch para la enseñanza de la programación. Revisión del conocimiento científico publicado en España. *Tarbiya, Revista De Investigación e Innovación Educativa*, *48*, 7-32. https://doi.org/10.15366/tarbiya2020.48.001

Contreras, R. S. (2018). *Experiencias de gamificación en aulas*. Bellaterra.

De Miguel, M. (2006). *Metodologías de Enseñanza Centrados en el Desarrollo de Competencias. Orientaciones para Promover el Cambio Metodológico en el Espacio Europeo de Educación Superior*. Ediciones Universidad de Oviedo.

Denny, P. (2013). The effect of virtual achievements on student engagement. En *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 763–772). https://doi.org/10.1145/2470654.2470763

Deterding, S., Dixon, D., Khaled, R. y Nacke, L. (2011a). From game design elements to gamefulness: defining "gamification". En *Proceedings of the 15th International Academic MindTrek Conference Envisioning Future Media Environments* (pp. 9–15). https://doi.org/10.1145/2181037.2181040

Deterding, S., Sicart, M., Nacke, L., O'Hara, K. y Dixon, D. (2011b). Gamification. using game-design elements in non-gaming contexts. En *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems* (pp. 2425–2428). https://doi.org/10.1145/1979742.1979575

Domínguez, A., Saenz-de Navarrete, J., Marcos, L., Fernández-Sanz, L., Pagés, C. y Martínez-Herráiz, J. J. (2013). Gamifying learning experiences: practical implications and outcomes. *Computers & Education*, *63*, 380–392. https://doi.org/10.1016/j.compedu.2012.12.020

Dondlinger, M. J. (2007). Educational video game design: A review of the literature. *Journal of Applied Educational Technology*, *4*(1), 21-31.

García-Peñalvo, F. J., Corell, A., Abella-García, V. y Grande-de-Prado, M. (2021). Recommendations for Mandatory Online Assessment in Higher Education During the COVID-19 Pandemic. In D. Burgos, A. Tlili, & A. Tabacco

(Eds.), *Radical Solutions for Education in a Crisis Context. COVID-19 as an Opportunity for Global Learning* (pp. 85-98). Springer Nature. https://doi.org/10.1007/978-981-15-7869-4_6

García-Peñalvo, F. J. y Mendes, A. J. (2018). Exploring the computational thinking effects in pre-university education. *Computers in Human Behavior*, *80*, 407-411. https://doi.org/10.1016/j.chb.2017.12.005

García-Peñalvo, F. J., Reimann, D., Tuul, M., Rees, A. y Jormanainen, I. (2016). *An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers*. TACCLE3 Consortium. https://doi.org/10.5281/zenodo.165123

Gil-Quintana, J. y Prieto Jurado, E. (2020). La realidad de la gamificación en Educación Primaria. Estudio multicaso de centros educativos españoles. *Perfiles educativos*, *42*(168), 107-123. https://doi.org/10.22201/iisue.24486167e.2020.168.59173

Gómez López, R. (2002). Análisis de los métodos didácticos en la enseñanza. *Publicaciones*, *32*, 261-334.

González-González, C. y Blanco-Izquierdo, F. (2012). Designing social videogames for educational uses. *Computers & Education*, *58*(1), 250-262. https://doi.org/10.1016/j.compedu.2011.08.014

Groening, C. y Binnewies, C. (2019). Achievement unlocked! - The Impact of digital achievements as a gamification element on motivation and performance. *Computers in Human Behavior*, *97*, 151-166. https://doi.org/10.1016/j.compedu.2011.08.014

Heintz, F., Mannila, L. y Färnqvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K-12 education. En *Proceedings of the 2016 IEEE Frontiers in Education Conference (FIE)*. https://doi.org/10.1109/FIE.2016.7757410

Hijón-Neira, R., Santacruz-Valencia, L., Pérez-Marín, D. y Gómez-Gómez, M. (2017). Un análisis de la situación sobre el estado de la enseñanza de la Programación en Primaria y su didáctica. *Atas do XIX Simpósio Internacional de Informática Educativa e VIII Encontro do CIED–III Encontro Internacional* (pp. 103-108). http://hdl.handle.net/10400.21/11916

Kafai, Y. B., Burke, Q. y Resnick, M. (2014). *Connected code: why children need to learn programming*. MIT Press. https://doi.org/10.7551/mitpress/9992.001.0001

Kebritchi, M., Hirumi, A. y Bai, H. (2010). The effects of modern mathematics computer games on mathematics achievement and class motivation. *Computers & Education*, *55*(2), 427–443. https://doi.org/10.1016/j.compedu.2010.02.007

Krishnamoorthy, S. P. y Kapila, V. (2016). Using a visual programming environment and custom robots to learn c programming and k-12 stem concepts. En P*roceedings of the 6th Annual Conference on Creativity and Fabrication in Education* (pp. 41-48). https://doi.org/10.1145/3003397.3003403

Lee, J., Luchini, K., Michael, B., Norris, C. y Soloway, E. (2004). More than just fun and games: assessing the value of educational video games in the classroom. En *Extended Abstracts on Human Factors in Computing Systems, CHI conference* (pp. 1375–1378). https://doi.org/10.1145/985921.986068

Lorés J. (2002). *La Interacción Persona-Ordenador.* Asociación Interacción Persona Ordenador. https://aipo.es/libro/pdf/00Portad.pdf

Mathieu, M. J. (2014). *Introducción a la programación*. Grupo Editorial Patria.

Mcclean, P., Saini-eidukat, B., Schwert, D., Slator, B. y White, A. (2001). Virtual worlds in large enrollment science classes significantly improve authentic learning. En *Proceedings of the 12th International Conference on College Teaching and Learning*, (pp. 111–118). Center for the Advancement of Teaching and Learning.

Montes-León, H., Hijón-Neira, R., Pérez-Marín, D. y Montes-León, R. (2020). Mejora del Pensamiento Computacional en Estudiantes de Secundaria con Tareas Unplugged. *Education in the Knowledge Society*, *21*, Article 24. https://doi.org/10.14201/eks.23002

Morales-Urrutia, E.K., Ocaña, J M., Pérez-Marín, D. y Pizarro, C. (2021). Can Mindfulness Help Primary Education Students to Learn How to Program with an Emotional Learning Companion? *IEEE Access*, *9*, 6642-6660. https://doi.org/10.1109/ACCESS.2021.3049187

Moreno, J. (2012). Digital competition game to improve programming skills. *Education & Technology Society*, *15*(3), 288–297.

Ocaña, J.M., Morales-Urrutia, E.K., Pérez-Marín, D. y Pizarro, C. (2020). Can a learning companion be used to continue teaching programming to children even during the COVID-19 pandemic? *IEEE Access*, *8*, 157840-157861. https://doi.org/10.1109/ACCESS.2020.3020007

Ouahbi, I., Kaddari, F., Darhmaoui, H., Elachqar, A. y Lahmine, S. (2015). Learning basic programming concepts by creating games with Scratch programming environment. *Procedia-Social and Behavioral Sciences*, *191*, 1479-1482. https://doi.org/10.1016/j.sbspro.2015.04.224

Øygardslia, K. y Aarsand, P. (2018). Move over, I will find Jerusalem: artifacts in game design in classrooms. *Learning, Culture and Social Interaction*, *19*, 61-73. https://doi.org/10.1016/j.lcsi.2018.04.013

Pérez-Marín, D., Hijón-Neira, R. y Martín-Lope, M. (2018). A Methodology Proposal based on Metaphors to teach Programming to children. *IEEE Revista Iberoamericana de Tecnologías del Aprendizaje*, *13*(1), 46-53. https://doi.org/10.1109/RITA.2018.2809944

Piaget, J. (1971). *Psychology and epistemology: towards a theory of knowledge*. Viking.

Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E. y Brennan, K. (2009). Scratch: Programming for all. *Communications of the ACM*, *52*(11), 60-67. https://doi.org/10.1145/1592761.1592779

Shahid, M., Wajid, A., Haq, K. U., Saleem, I. y Shujja, A. H. (2019). A Review of Gamification for Learning Programming Fundamental. En *Proceedings of the 2019 International Conference on Innovative Computing (ICIC).* https://doi.org/10.1109/ICIC48496.2019.8966685

Sović, A., Jagušt, T. y Seršić, D. (2014). How to teach basic university-level programming concepts to first graders? En *Proceedings of the 2014 IEEE Integrated STEM Education Conference*. https://doi.org/10.1109/ISECon.2014.6891050

Squire, K.D. (2002). Video games in Education. *International Journal Intelligent Games Simulation*, *2*(1), 49–62.

Talingdan, J. A. y Llanda, C. R. (2019). Assessment of the effectiveness of learning theories using gamified Android app in teaching C programming. *IOP Conference Series: Materials Science and Engineering*, *482*(1), 012030. https://doi.org/10.1088/1757-899X/482/1/012030

Torres-Toukoumidis, Á., Ramírez-Montoya, M. S. y Romero-Rodríguez, L. M. (2018). Valoración y evaluación de los Aprendizajes Basados en Juegos (GBL) en contextos e-learning. *Education in the Knowledge Society*, *19*(4), 109-128. https://doi.org/10.14201/eks2018194109128

Velázquez-Iturbide, J. Á., Bahamonde, A., Dabic, S., Escalona, M. J., Feito, F., Fernández Cabaleiro, S., Ferrero Martín, B., Garay Vitoria, N., García, J. C., García Borgoñón, L., García Martínez, M., García Molina, J., García Varea, I., Hermenegildo Salinas, M., Larraza Mendiluze, E., Llorens Largo, F., Mateos, J. A., Moratel Muñoz, A., Mozos, D., Pimentel, E., Sahelices, B., Toro, M. y Zapata Ros, M. (2018). *Informe del Grupo de Trabajo SCIE/CODDII sobre la enseñanza preuniversitaria de la informática*. Sociedad Científica Informática de España Conferencia de Decanos y Directores de Ingeniería Informática. https://goo.gl/dmCPgm

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM, 49*(3), 33-35. https://doi.org/10.1145/1118178.1118215