Un análisis desde la programación estructurada del lenguaje Scratch como entorno lúdico educativo

Antonieta Kuz[1], María Cecilia Ariste[2]
[1]Universidad Metropolitana para la Educación y el Trabajo
[2]Fundación Sadosky, iniciativa Program.AR

Resumen: Aprender a programar implica resolver problemas y automatizar tareas con la ayuda de una computadora. Hoy en día Scratch es considerada una de las herramientas más usadas para enseñar a programar, que puede integrase a la robótica y desarrollo de prototipos que busca promover el pensamiento computacional y habilidades que conllevan la resolución de problemas. Este estudio se basa en los fundamentos teóricos de Scratch desde un enfoque estructurado, teniendo en cuenta sus prestaciones y su adaptación a diferentes asignaturas, su relación con el pensamiento computacional y con los propósitos de aprendizaje, áreas disciplinares.

Palabras clave: Scratch, programación estructurada, sentencias de programación, programación visual

Abstract: Learning to program involves solving problems and automating tasks with the help of a computer. Today Scratch is considered one of the most used tools to teach programming, which can be integrated into robotics and prototype development that seeks to promote computational thinking and the ability to solve problems. This study is based on the theoretical foundations of Scratch from a structured approach, taking into account its benefits and its adaptation to different subjects, its relationship with computational thinking and with the purposes of learning, disciplinary areas.

Key words: Scratch, structured programming, sentences of programming, visual programming

1. Introducción

En la sociedad actual nuevos desafíos surgen a partir de la irrupción de la tecnología donde la información juega un rol protagónico. Las Tecnologías de la Información y la Comunicación (TIC) forman parte de la educación y su uso han ganado terreno de manera progresiva tanto al nivel macro como micro, es decir dentro y fuera de las aulas. Si pensamos en los diversos usos de las TIC, existen una gran variedad de recursos que se han creado para facilitar y mediar en los procesos de enseñanza-aprendizaje. Muchos países están en proceso de introducir o han introducido la programación en el currículum de las enseñanzas obligatorias, lo cual implica trabajar sobre diferentes ángulos en la currícula el paradigma de programación estructurado, el manejo de sentencias, el pensamiento computacional, construir la autonomía y desarrollar la creatividad de los alumnos.

Enseñar a programar en la escuela en los diferentes niveles, tanto primario como secundario, es uno de los nuevos desafíos que enfrentan los profesores en las aulas y es actualmente, dado que programar no significa simplemente codificar en un lenguaje de programación sino que también implica incluir al pensamiento computacional para resolver problemas. Entendiendo que la programación no es una habilidad cerrada que solo sirve para el propósito de crear código para una computadora por un programador o informático, sino que sirve para desarrollar habilidades que permitan al alumno proyectar nuevas posibilidades. Los procesos de enseñanza y aprendizaje apoyados por el uso de Scratch promueven y fortalecen prácticas educativas y posibilitan la interactividad de forma exploratoria y creativa a través de proyectos colaborativos en diferentes contextos.

El objetivo de la presente investigación atañe a la descripción de Scratch desde un análisis de las prestaciones que brinda y cómo puede aportar a la programación y a una nueva dimensión de

pensamiento, ayuda a analizar las acciones y los procedimientos realizados, aportando claridad a la generación de ideas y a la resolución de problemas. El resto del artículo se estructura como sigue: en la sección 2 introducimos el estado de arte de Scratch. En la sección 3, detallamos las bases de la programación estructurada e introducimos las estructuras de control. En la subsección 3.1 definimos las estructuras de control con Scratch. En la subsección 3.2 definimos los componentes de la interfaz gráfica a través del escenario de historias. En la sección 4 definimos el diseño y la didáctica de la enseñanza de la programación. En la sección 5 definimos la discusión. Finalmente, en la sección 6 se presentan las conclusiones y el trabajo futuro.

2. Estado de Arte de Scratch

Para que una computadora realice las acciones que deseamos necesitamos poder establecer una comunicación, y contar con un lenguaje que ellas comprendan. Mediante un lenguaje de programación se pueden realizar y planificar instrucciones que nos lleve a poder obtener de las máquinas, resultados esperados. La mayoría de los lenguajes de programación son puramente textuales, es decir, utilizan secuencias de texto que incluyen palabras, números y signos de puntuación, de manera similar a los lenguajes naturales escritos. La sintaxis es la parte visible de un lenguaje de programación y define el conjunto de reglas que se deben seguir al escribir el código fuente de los programas para considerarse correctos. Aprender a utilizar la sintaxis presenta ciertas dificultades hasta manejar correctamente un lenguaje de programación, ya que hay lenguajes con una sintaxis bastante compleja, se necesitan de muchos símbolos, muchas líneas para hacer una instrucción. Analizando estas dificultades y retomando las ideas construccionistas (Kafai y Resnick, 1996) plasmadas en el lenguaje Logo, el grupo de investigación Lifelong Kindergarten del laboratorio de medios del MIT (Massachusetts Institute of Technology) desarrolló la plataforma Scratch. Scratch es un lenguaje de programación basado en bloques, con una gramática y una sintaxis tales como el lenguaje C, Java o Python entre otros.

Los bloques tienen una gramática visual y sus reglas de combinación tienen el mismo rol que la sintaxis en los lenguajes basados en texto. Los bloques representan instrucciones con textos en modo imperativo en varios idiomas e incluyen estructuras de control, manejo de eventos, uso de variables, mensajes entre objetos, movimiento y sonido. También cuanta con una biblioteca de escenarios y objetos a la que se les pueden agregar material importado desde otras plataformas. Para que el propio alumno pueda darle su impronta y originalidad, permite editar las imágenes, tomar fotografías, grabar y editar videos y sonidos. De esta manera, pueden construirse proyectos interactivos originales y propios del alumno, tales como video juegos, cuentos, canciones, coreografías, y simulaciones, entre otras. De esta manera se logra que aquello que se construye tome cierta identidad y pertenencia para con el alumno o grupo de alumnos, lo que despierta el interés para poder profundizar más aún en las posibilidades de la herramienta.

3. Las bases de la programación estructurada: estructuras de control

Como vimos en la sección previa los lenguajes de programación tienen una estructura compleja que se compone de varias partes, la sintaxis, la semántica, elementos del lenguaje, el nivel de abstracción, estructuras de control para ordenar la ejecución de los programas, tipos de datos (números, letras, etc.), funciones o procedimientos (unidades) que contienen un conjunto de instrucciones, entre otras. Para realizar el planteo y solución a un problema es necesario utilizar las estructuras de control de programación. Para Eckerdal et al. (2005) programar implica entender el lenguaje de programación, y utilizarlo para escribir códigos. Concordando con esto, para comprender las bases de la programación con Scratch es importante analizar las estructuras de control. Las mismas controlan el flujo de ejecución de los programas. Influyen en la legibilidad y en la facilidad de escritura. Aumentan el control que el programador tiene sobre un programa, y por lo tanto aumenta la confiabilidad.

Un algoritmo se define como un *"conjunto ordenado y finito de operaciones que permite hallar la solución de un problema"* (Real Academia Española). Ahora bien, cualquier algoritmo puede diseñarse e implementar utilizando únicamente tres tipos de estructuras de control (Böhm y Jacopini, 1996) demostraron el teorema de la programación estructurada que especifica que todo algoritmo puede construirse utilizando estructuras de control

secuenciales, condicionales (selección) y repetitivas. Por lo tanto, tres tipos de estructuras lógicas se pueden combinar para armar programas que brinden una solución a diversos planteos que se realicen: secuencia, selección o condicionales e iteración o ciclos, como se observa en las Figuras 1, 2 y 3 a través de esquemas representativos como el pseudocódigo y los diagramas de flujo para expresar dichas estructuras de programación, los cuales son una representación esquemática de las secuencias de un programa y permiten representar en forma gráfica los algoritmos (Joyanes, 2008). Por un lado, tenemos la sentencia condicional que se utiliza cuando en un algoritmo se quiere establecer que una acción o secuencia de acciones solo se ejecuten si se cumple una determinada condición. Las sentencias condicionales pueden ser de varios tipos: sentencia condicional simple, sentencia condicional doble o alternativa, o sentencia condicional múltiple. Estructura iterativa o bucle se utiliza cuando la ejecución de una acción o secuencia de acciones debe repetirse varias veces. Una condición determinará cuándo deben continuar las iteraciones y determina la continuidad del bucle puede definirse antes o después de definir la secuencia de acciones.

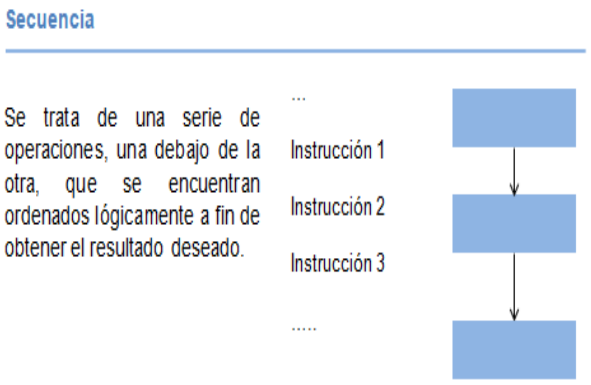

Figura 1. Estructura de Control: Secuencia

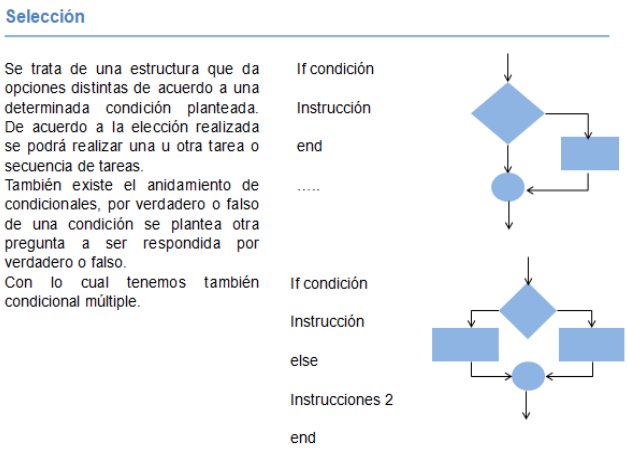

Figura 2. Estructura de Control: selección

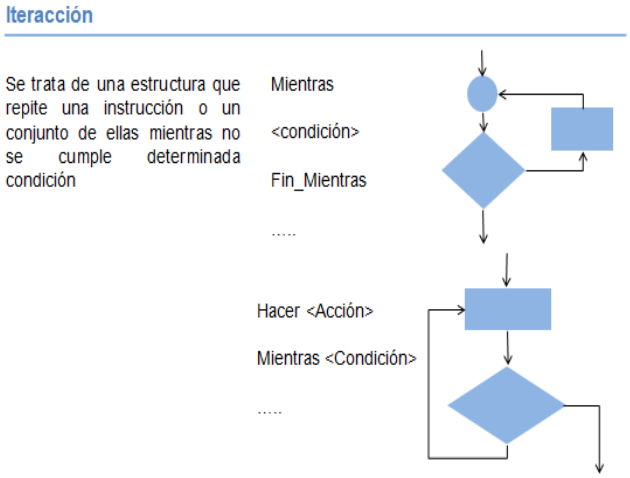

Figura 3. Estructura de control: iteración

3.1 Estructuras de control con Scratch

El papel de Scratch como herramienta de aprendizaje de la programación es muy importante ya que permite crear programas y compartirlos (Nárvaez Perez, 2017). Al ser Scratch un software muy dinámico se puede trabajar con las mismas estructuras de control mencionadas en la sección previa. A continuación detallamos en la Figura 4 los bloques para expresar las estructuras de control.

Figura 4. Estructuras de control utilizando Scratch

3.2 Interfaz gráfica: el escenario de historias

Scratch cuenta con una interfaz gráfica que tiene diseño HCI (Human Computer Interaction) el cual permite crear fácilmente animaciones y elementos interactivos. A través de su diseño cualquier persona puede comprender y manejar la lógica del software, crear objetos interactivos, añadir música y sonidos, crear animaciones y manipular imágenes en 2D. La interfaz gráfica como se ve en la Figura 5 consta de un escenario fijo que puede ir cambiando de fondo y una serie de objetos movibles. Cada objeto recibe el nombre de sprite y puede tener uno o más disfraces, es decir, diferentes aspectos que puede tomar un mismo objeto (por ejemplo, hacer que un personaje camine). En la parte inferior de la escena hay un área que muestra las miniaturas de todos los sprites del proyecto. El panel central de la interfaz gráfica cuenta con tres pestañas: programas, disfraces y sonidos. Seleccionando cada una de esas pestañas cambiará el panel de la derecha, donde el usuario podrá ver y modificar los disfraces (imágenes), los sonidos, o el comportamiento (el programa) del sprite elegido (Resnick et al. 2009)

El uso de bloques de colores permite y facilita que se hagan tangibles muchos conceptos expresados en las sentencias, con lo cual facilita el aprendizaje de conceptos abstractos y complejos que no son simples de asimilar por la falta de referencias concretas. Además, la interfaz de Scratch se puede dividir en las siguientes secciones que se detallan en la Tabla 1. En la misma es posible realizar la edición de contenido multimedia ya sea de imágenes o sonidos tales como objetos o disfraces o música que quiera incorporarse a los proyectos o cualquier otro efecto sonoro (Maloney et al. 2010).

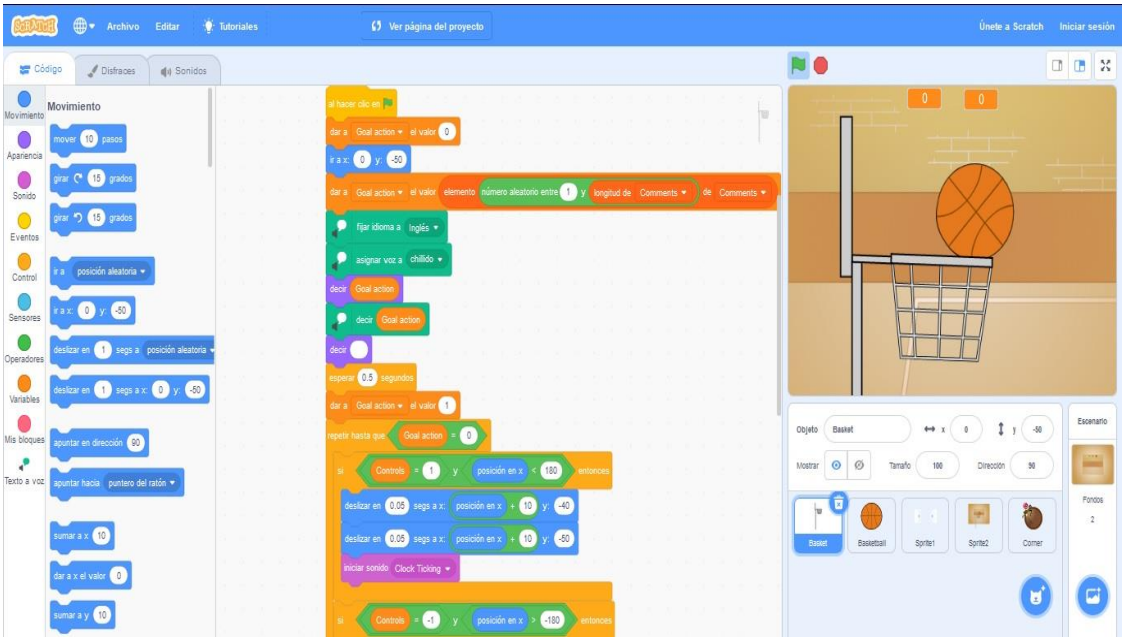

Figura 5. Interfaz gráfica de Scratch

TABLA 1. SECCIONES DE LA INTERFAZ GRÁFICA DE SCRATCH

Interfaz de Scratch	Descripción
Escenario	Es donde se programan los objetos que se mueven e interactúan entre ellos. Además se le pueden añadir uno o más fondos intercambiables.
Gestión de objetos	Incluye la lista de objetos donde muestra las miniaturas de todos los objetos incluidos en el proyecto, y las herramientas que Scratch dispone para ellos.
Paleta de bloques	Se encuentran distribuidas todas las instrucciones en apartados según su función, al hacer click en una agrupación, se nos mostrarán las instrucciones que contiene.
Área de programa	Es donde se sitúan todas las instrucciones que queremos que realice cada uno de los objetos.
Multimedia	**Disfraces:** Scratch dispone de una galería de objetos en la que algunos de ellos tienen más de un disfraz y además es posible cargar disfraces desde el disco duro, y a su vez, mediante el editor de imagen que tiene el programa incorporado, y también es posible modificar imágenes que ya se encuentren en la galería de Scratch y crear nuevos disfraces. **Sonidos:** Los proyectos en Scratch admiten sonidos y música. Para acceder a las herramientas de sonido basta con seleccionar, desde el área de gestión de objetos, el objeto en el cual queremos incorporar audio para después hacer clic en la pestaña Sonido que se encuentra encima del área de programación y que nos mostrará la interfaz para sonidos.

4. Diseño y didáctica de la enseñanza de la programación: caso “Elije tu propia aventura”

Aprender a programar involucra la resolución de problemas con el empleo de un lenguaje de programación, como Scratch, el usuario construye programas agrupando bloques gráficos. Estos bloques pueden utilizarse como si fueran piezas y representan

las estructuras de programación, y las acciones que se pueden realizar dentro del programa (mover un objeto, reproducir un sonido, etc.). Las instrucciones y estructuras de control detalladas, que se pueden asociar a cada objeto, se agrupan en bloques según su funcionalidad. Los bloques se encajan entre sí unos con otros para crear los programas o scripts y se ejecutan las instrucciones en orden de arriba abajo.

Adicionalmente programar implica analizar y aproximarse a la comprensión sobre una situación a resolver, mediante el reconocimiento de las estructuras algorítmicas necesarias para garantizar una lógica coherente implicando un proceso de algoritmización computacional. La algoritmización involucra varios componentes: la práctica para comprender el diseño y la didáctica de los ejercicios, formulación del problema lo cual conlleva comprensión lingüística y lectora, delimitar los resultados que se desea obtener, identificar la información que se dispone organizando los datos, tener presente cuales son las restricciones del problema y definir los procesos que llevan desde los datos disponibles hasta el resultado deseado a través de la secuencia de pasos.

A través del siguiente ejemplo podemos ver un modelo de enunciado (Fernández Casal, 2016): "Elige tu propia Aventura" es una propuesta educativa para desarrollar en Scratch que se integra con Prácticas del Lenguaje y hace honor a esta clase de cuentos. La idea es que los niños elijan un cuento o historia que hayan o estén trabajando en el área de lengua, ellos van a modificar el cuento para transformarlo en un formato del tipo elige tu propia aventura. El cuento debe analizarse utilizando mapas conceptuales u otros recursos con el objetivo de conocer exactamente los momentos de la historia, los personajes principales y secundarios. Luego hay que comenzar a llevar el relato a escenas gráficas con personajes, dramatizando situaciones. Una vez ubicados en esta instancia, la idea es diseñar las bifurcaciones, es decir, los distintos caminos por los que puede transcurrir la historia. La elección de un camino u otro será tomada en tiempo de ejecución por otro niño o adulto a quién le mostremos la historia que estamos armando. De esta manera los niños están diseñando e implementando interacciones humano-computadora. Como puede verse en la Figura 6, para realizar las bifurcaciones se trabajan las estructuras de control condicionales, el uso de variables y de eventos.

No solamente se pueden desarrollar enunciados o problemas, pueden incluirse el desarrollo de las actividades lúdicas mediante la construcción de videojuegos educativo reforzando conocimientos de programación, dentro de un marco de referencia constructivista del aprendizaje. Como observamos en el ejemplo el programa que desarrollaron los alumnos es una secuencia de instrucciones cuya ejecución producen una serie de acciones que cambian o transforman el estado inicial del entorno, pasando por diversos estados intermedios para llegar a un estado final, el cual representa es la solución del problema.

5. Discusión

El principal objetivo de Scratch es hacer que la programación sea el medio por el cual se lleva a cabo el proceso de aprendizaje, un aprendizaje que debe caracterizarse por tener la mayor autonomía posible permitiendo un mayor desarrollo de la capacidad creativa. Es por tal motivo que hacer un análisis de la sintaxis que ofrece Scratch nos permite entender cuál es la experiencia que el usuario tiene al interactuar con el software, y cuáles pueden ser las infinitas posibilidades de desafíos que pueden proponer los docentes para incentivar la exploración y el aprendizaje.

Como hemos analizado, Scratch permite que los alumnos utilicen las estructuras de control

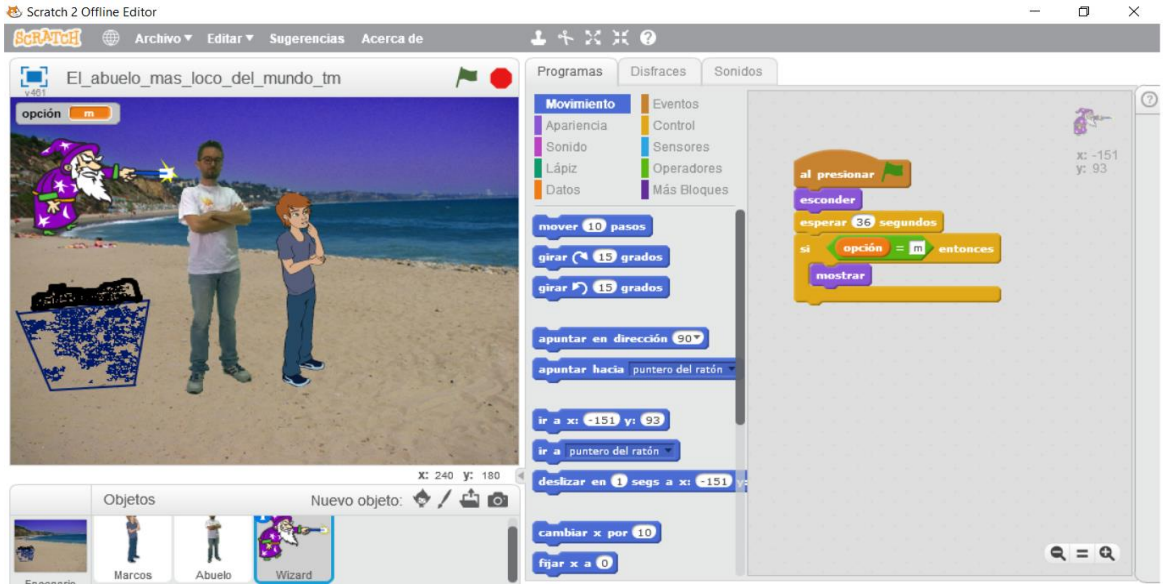

Figura 6. Ejemplo de historia interactiva desarrollada por alumnos de primaria

fundamentales para la escritura de cualquier algoritmo además de poder aplicar otros conceptos como variables y eventos. Además, Scratch cuenta con posibilidades de edición de imagen para armar video, agregar sonido y generar proyectos interactivos, lo que lo hace atractivo para los alumnos y amplía las posibilidades de trabajar con recursos digitales de cualquiera de las asignaturas de la trayectoria escolar (Monjelat y San Martín 2016).

6. Conclusiones

Los alumnos son considerados como nativos digitales, ya que han nacido y están inmersos entorno rodeado de tecnología, y están interiorizados en el manejo de las computadoras y smartphones. Sin embargo, que un alumno esté utilizando la tecnología no significa que sepa cómo programar. La programación es un conocimiento vital para un mundo gobernado por lo digital, la cual no es una habilidad cerrada que solo sirve para el propósito de crear código, sino que también sirve para desarrollar otras habilidades y aprender otras nuevas.

Para llegar a los objetivos de aprendizaje y lograr despertar en el alumno la curiosidad por este tipo de recursos educativos, es fundamental que el equipo docente conozca las posibilidades que brinda la herramienta, de su alcance, de sus variantes y de los desafíos que pueden plantearse y cómo han resultado en otros casos, para evaluar la viabilidad de aplicación en un contexto determinado. Con lo cual el aprendizaje de la programación con Scratch no es diferente a otros procesos de aprendizaje y la flexibilidad permite a los docentes implementar lecciones conceptuales visuales. Es decir, el docente puede utilizar la herramienta para crear animaciones que ayudan a visualizar conceptos difíciles de física, química o matemáticas pero también armar cuentos utilizando escenarios y personajes dándoles vida y proponiendo diferentes interacciones.

Referencias

Böhm, C., Jacopini, G. (1966) Flow diagrams, Turing machines and languages with only two formation rules. In: Communications of the ACM, pp. 366–371

Eckerdal, Anna & Thuné, Michael & Berglund, Anders. (2005)What does it take to learn 'programming thinking'? In Proceedings of the first international workshop on Computing education research (ICER '05). Association for Computing Machinery, New York, NY, USA, 135–142. DOI:https://doi.org/10.1145/1089786.1089799

Fernández Casal, F. (2016) Proyecto para aprender programando: Elige tu propia aventura. España. Recuperado de: http://procomun.educalab.es/es/articulos/elige-tu-propia-aventura-con-scratch.

Joyanes Aguilar, L. (2008). Metodología de la programación, Diagramas de flujo algoritmos y

programación estructurada. Estados Unidos: McGraw-Hill. Edición 4

Kafai, Y., Resnick, M. (1996). *Constructionism in practice: Designing, thinking, and learning in a digital world*. Hillsdale, Erlbaum

Luque, M. (2019). "Cómo influye Scratch en el aprendizaje de las estructuras de programación en estudiantes secundarios de escuela técnica de 1er. Año 1ra. Div. del ciclo superior de informática personal y profesional". Universidad Tecnológica Nacional Facultad Regional Resistencia Licenciatura en Tecnología Educativa, Resistencia

Maloney, J., Resnick, M., Rusk, N., Silverman, B., and Eastmond, E. (2010). The scratch programming language and environment. ACM Transactions on Computing Education, Vol. 10, No. 4, Article 16

Monjelat, N., San Martín P. S. (2016) Programar con Scratch en contextos educativos: ¿Asimilar directrices o co-construir Tecnologías para la Inclusión Social? Praxis educativa, Vol. 20, Nº 1; pp. 61-71

Nárvaez Perez, H. O. (2017) Uso de Scratch como herramienta para el desarrollo del pensamiento computacional en Programación I de la carrera de Informática de la Universidad Central del Ecuador. Universidad de Alicante. Alicante, España
Real Academia Española. (s.f.). Algortimo. En *Diccionario de la lengua española*. Recuperado en 17 de diciembre de 2020, de https://dle.rae.es/?w=algoritmo&origen=REDLE

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K. Millner, A., Rosenbaum, E., Silver, J., Silverman, B and Kafai, Y. (2009) Scratch: programming for all. Commun. Communications of the ACM 52(11):60-67