TESINA DE LICENCIATURA

TÍTULO: BlockAr: Aplicación multiplataforma para aprender a programar

AUTORES: Sebastián Ismael Pierini

DIRECTORAS: Claudia Queiruga, Claudia Banchoff Tzancoff

CODIRECTOR/A:

ASESOR/A PROFESIONAL:

CARRERA: Licenciatura en sistemas

Resumen

La presente tesina plantea el desarrollo de la aplicación multiplataforma BlockAr, contextualizada en la Escuela de Educación Secundaria Nº9, cuyo propósito es incorporarla a la materia de NTIC. BlockAr alienta el desarrollo de habilidades vinculadas al pensamiento computacional y al aprendizaje de la programación, con un enfoque que recupera conceptos de la gamificación. En este desarrollo se consideró la infraestructura disponible en la escuela y el uso de los celulares de los estudiantes, se analizaron las tecnologías para crear aplicaciones multiplataforma, así como también como las herramientas con propósito educativo. Se puso especial atención en la elección de tecnologías abiertas y libres. Con el fin de dar continuidad al proyecto, se lo alojó en un repositorio libre.

Palabras Clave

Pensamiento Computacional, Gamificación, Juegos Serios, Educación, Enseñanza de la Programación, Software Libre, Código abierto, NTIC.

Conclusiones

Los temas investigados en esta tesina fueron necesarios para comprender maneras de motivar a los estudiantes, a través de estrategias de gamificación, que los introducen en los conceptos de la programación situados en la asignatura NTIC. Las clases en el aula utilizando BlockAr y las secuencias didácticas elaboradas, dieron cuenta de un resultado alentador, evidenciado en las observaciones áulicas y en las encuestas realizadas a los estudiantes.

Trabajos Realizados

Se investigó sobre gamificación, pensamiento computacional, alfabetización digital, juegos serios.
Se hizo un estudio de campo en la Escuela de Educación Secundaria N°9 de Lisandro Olmos para evaluar la conectividad y los recursos tecnológicos disponibles.
Se analizaron tecnologías y herramientas libres para el desarrollo de aplicaciones multiplataformas.
Se implementó el juego multiplataforma llamado BlockAr y se diseñó una secuencia didáctica que acompañó el uso en las clases.
Se puso a prueba con estudiantes de 4to. año en la materia NTIC en la cual se relevó sus opiniones mediante encuestas.

Trabajos Futuros

Diseñar escenarios de aplicación que articulen los contenidos de programación con propuestas curriculares de otras áreas de conocimiento, que se trabajan en la escuela.
Agregar nuevos desafíos para que los estudiantes y el docente puedan trabajar más contenidos.
Crear un constructor de escenarios para que pueda agregar nuevos desafíos y objetos nuevos.
Darle la posibilidad al estudiante de crear sus propios personajes.
Adecuar BlockAr para interactuar con objetos físicos construidos con Arduino.

Fecha de la presentación: **Agosto 2022**

Tesina de grado de Lic. en Sistemas

BlockAr

Aplicación multiplataforma para aprender a programar

Alumno:
Sebastian Ismael Pierini

Directoras:
Lic. Claudia Queiruga
Lic. Claudia Banchoff Tzancoff

Universidad Nacional de la Plata
Facultad de Informática
La Plata
2022

Capítulo 1: Introducción

En este capítulo se presenta la motivación, los objetivos, la metodología de trabajo, los resultados esperados y la estructura de la tesina.

1.1. Motivación

Esta tesina se inscribe en las ideas de lenguajes digitales, pensamiento computacional y ludificación como saberes a incluir en la enseñanza de la escuela secundaria.

Llorens-Largo F. (2015) señala que la enseñanza y la formación en lenguajes digitales en la escuela constituyen *“las competencias necesarias para desenvolverse con éxito en el mundo digital, con la programación como forma de resolver problemas y el pensamiento computacional como paradigma de trabajo”,* relacionando de esta manera las competencias digitales con la programación y el pensamiento computacional; a la vez que advierte que esta nueva alfabetización, la *alfabetización digital*, permitirá preparar a las y los jóvenes para enfrentarse a los desafíos de la sociedad en las que les tocará vivir, brindándoles las herramientas cognitivas necesarias para desenvolverse en el mundo digital.

Wing (2006) explica que el pensamiento computacional *“implica la resolución de problemas, el diseño de sistemas y la comprensión del comportamiento humano, basándose en los conceptos fundamentales de la informática”.* También menciona que *“incluye una variedad de herramientas mentales que reflejan la amplitud del campo de la informática”.* Desde su punto de vista *“el pensamiento computacional representa una actitud y un conjunto de habilidades de aplicación universal que todos, no sólo los informáticos, estarían ansiosos por aprender y usar.”*

Además Wing (2006) destaca las siguientes características del pensamiento computacional:

- Un informático, debe pensar más allá del simple hecho de programar una computadora.
- Los conceptos computacionales nos brindaran herramientas para resolver problemas, administrar nuestra vida, comunicarnos e interactuar con otras personas.
- Es una gran visión para guiar a los educadores, investigadores y profesionales en Informática a la vez que permite dar a conocer un campo de

conocimiento nuevo y estrechamente vinculado a los procesos productivos, siendo sumamente relevante llegar a la audiencia preuniversitaria, incluidos maestros, padres y estudiantes.

En cuanto de la relevancia de contar con competencias en lenguajes digitales y en pensamiento computacional, Wing (2006) considera importante la incorporación de un curso sobre "Maneras de pensar como un científico de la computación" destinado a estudiantes de la universidad, con el fin de exponer a los estudiantes preuniversitarios a métodos y modelos computacionales, independientemente del campo de estudio disciplinar.

Por otro lado, el estudio del juego y sus implicaciones culturales y sociales fue uno de los principales temas tratados por Huizinga (1949), quien nos presenta el concepto de Homo ludens (el hombre que juega) como una dimensión adicional al ser humano. Los juegos están presentes en diferentes ámbitos de la experiencia humana, en la guerra, la escuela y hasta los negocios, esfera donde se ha observado un crecimiento significativo del uso de herramientas de juego en los últimos años. Ahora bien, como dice Jean Piaget (1983): "*El principio de la programación (que Skinner ha intentado en sus propias lecciones de psicología antes de generalizarlo a toda la enseñanza) es, en efecto, el siguiente. Dadas las definiciones, en primer término el alumno debe sacar las consecuencias correctas y para ello elegir entre dos o tres soluciones que la máquina le ofrece. Si elige la buena (presionando un botón) el trabajo continúa, mientras que si se equivoca vuelve a comenzar el ejercicio. Cada nueva información que la máquina proporciona da lugar a elecciones que ponen de manifiesto la comprensión obtenida, con tantas repeticiones como haga falta y con progresos ininterrumpidos en el caso de éxitos constantes, Cualquier rama puede ser programada de esta manera, tanto si se trata de razonamiento puro como de simple memoria.*"

Tasneem Raja (2014) en el post "We Can Code It!", de la revista-blog Mother Jones, señala: *"El enfoque computacional se basa en ver el mundo como una serie de puzzles, a los que se puede romper en trozos más pequeños y resolver poco a poco a través de la lógica y el razonamiento deductivo"*, proponiendo un enfoque para enseñar a programar en ámbitos no universitarios. Tomando este concepto de *puzzles* como piezas para armar programas, varias herramientas didácticas orientadas a la enseñanza de programación han sido desarrolladas, entre las más

adoptadas en ámbitos escolares podemos mencionar Scratch[1], Pilas Bloques[2], permitiendo acercar conceptos de programación como sentencias, iteradores, variables, entre otras.

Miguel Zapata-Ros (2015) hace un análisis de lo mencionado en el párrafo anterior acerca de Tasneem Raja (2014) en el post “We Can Code It!” , de la revista-blog Mother Jones, y dice: *“Esta es una forma intuitiva en la que una autora, que proviene del mundo computacional, aborda una serie de métodos ampliamente conocidos en el mundo de la psicología del aprendizaje. Implícitamente está hablando de análisis descendente y de elaboración: Puzzles —problemas— que se pueden dividir en puzzles—problemas— más pequeños, para ir resolviéndolos. También, en el mismo párrafo, vemos una alusión implícita a la recursividad, falta la cláusula de parada y la vuelta atrás, pero evidentemente después de armar los puzzles pequeños cada uno hay que ensamblarlo en el puzzle general. Y también habrá que decir en qué nivel habrá que parar y dar marcha atrás.”*

Entrando en el terreno local, aún en Argentina no se ha consolidado un enfoque de enseñanza de la Informática como campo de conocimiento en la escuela, sin embargo existen experiencias de enseñanza y formación docente que dan cuenta de una intención de incorporar los *lenguajes digitales* en los diseños curriculares, tal como ocurre en la “Especialización Docente de Nivel Superior en Didáctica de las Ciencias de la Comunicación” destinada a docentes y profesionales de las áreas de Matemática, Física, Química, Tecnología e Informática. Estas especializaciones se pueden encontrar en (Unidad Académica Escuela Normal Superior N° 1 "Mary O. Graham", s.f..) y (UNIPE, s.f.).

1.2. Objetivos

1.2.1. Objetivo general

Desarrollar e implementar una herramienta educativa, denominada BlockAr, que será de acceso libre y abierto, para favorecer la introducción a estudiantes de la Escuela Secundaria N° 9 de Lisandro Olmos**,** en conceptos de programación**,** con un **enfoque situado.** La herramienta será multiplataforma.

[1] Sitio oficial: https://scratch.mit.edu/
[2] Sitio oficial: http://Pilasbloques.program.ar/

1.2.2. Objetivos específicos

- Identificar las características del pensamiento computacional que podrían desarrollarse en BlockAr.
- Definir los contenidos de programación a enseñar y su didáctica, en acuerdo con los docentes de la escuela.
- Caracterizar las herramientas disponibles en el mercado para desarrollo de juegos para celulares, con especial atención en el código fuente abierto.
- Determinar las posibilidades de conectividad a Internet de la Escuela Secundaria N° 9 y de disponibilidad de dispositivos de los estudiantes.
- Realizar una evaluación de BlockAr en la Escuela Secundaria N° 9.

1.3. Metodología de trabajo

- Realizar un estudio del estado del arte sobre las herramientas libres para programar en bloques, al estilo puzzles, y seleccionar la más adecuada para su implementación en la Escuela N° 9.
- Estudiar técnicas de gamificación para incorporar en BlockAr.
- Evaluar tecnologías de desarrollo para la implementación de BlockAr en plataformas móviles y web, con especial atención en estándares abiertos.
- Investigar y evaluar diferentes formas de visualización e interacción en dispositivos móviles con pantallas de resoluciones diferentes.
- Seleccionar contenidos de programación a ser enseñados en el marco de la materia NTIC.
- Diseñar y desarrollar la aplicación BlockAr a partir de las necesidades de la escuela.
- Subir el proyecto a GitHub[3] y compartirlo con la comunidad para que pueda ser descargado, usado y mejorado.
- Elaborar una propuesta pedagógica que acompañe a la herramienta BlockAr y ponerla en acción en el aula.
- Diseñar un caso de estudio en la Escuela N° 9, para la evaluación de BlockAr.

[3] Sitio oficial: https://github.com/

1.4. Resultados esperados

Se espera que al finalizar el desarrollo de esta tesina de grado, se logre:

- Disponer de la herramienta BlockAr destinada a los estudiantes y docentes de la Escuela Secundaria Nº 9 de Olmos para facilitar el acercamiento a conceptos y prácticas de programación, y su adopción para el desarrollo de proyectos escolares que vinculen diferentes áreas de conocimiento.
- Disponer de una guía de actividades orientada a docentes y estudiantes de la escuela sobre BlockAr y sus usos.
- Promover una comunidad en GitHub de BlockAr con el propósito de ser adoptada como herramienta de enseñanza sobre programación en diferentes escuelas.

1.5. Estructura de la tesina

La tesina se organiza de la siguiente manera:

- Capítulo 1: se analizan los temas que se trabajarán en la tesina indicando la motivación, los objetivos, la metodología de trabajo y los resultados esperados.
- Capítulo 2: se abordará el concepto de alfabetización digital y su importancia en el ámbito educativo, luego mencionaremos cómo el pensamiento computacional puede impactar en la capacidad de los estudiantes para resolver problemas. Continuaremos describiendo conceptos de gamificación, juegos serios, y analizaremos aplicaciones para aprender a programar que servirá como base para llevar adelante el desarrollo de BlockAr.
- Capítulo 3: se describirá el contexto educativo, las tecnologías utilizadas para el desarrollo de BlockAr, el diseño, la secuencia didáctica que acompañará al docente y a los estudiantes en el aprendizaje informático con la intervención de BlockAr. Por último se detallará cómo será el proceso de evaluación de BlockAr y se mostrarán los resultados.
- Capítulo 4: para finalizar esta tesina se explicará la conclusión y las extensiones que se pueden llevar adelante partiendo de la base de esta investigación.

Capítulo 2: Estado del arte y marco teórico

2.1 Introducción

Las tecnologías de la información y las comunicaciones nos atraviesan todo el tiempo, hoy en día nos encontramos en un mundo donde los trámites se hacen a través de Internet, muchas personas leen las noticias a través de sus navegadores web, ven el pronóstico del tiempo, revisan sus correos electrónicos, mandan mensajes a través de aplicaciones de celulares que les permite comunicarse con sus pares que se encuentran en diversos lugares del planeta. En este contexto social, la educación, como cualquier sector productivo o de servicios, se ve afectada por las tecnologías digitales (Toffler, 2001). Por otro lado, la alfabetización digital, el pensamiento computacional y los juegos serios, son elementos centrales para pensar una educación en donde la construcción con tecnologías digitales esté presente en el aula de la escuela en tanto colaboran en la promoción de habilidades cognitivas propias de la sociedad actual.

Con lo dicho al final del párrafo anterior, en este capítulo se caracterizarán las habilidades cognitivas que se promueven a través de la alfabetización digital, el pensamiento computacional y el juego.

2.2 Alfabetización digital

La alfabetización digital es un término que se está utilizando bastante en la actualidad. Paul Gilster, historiador y educador estadounidense, acuñó por primera vez el término "alfabetización digital", argumentando que la alfabetización digital iba más allá de las habilidades en el uso de la tecnología. Dijo que se trata de *“dominar las ideas, no las pulsaciones de teclas [de la computadora]”* (Gilster, 1997, p. 1). Además Gilster (1997) definió la alfabetización digital como *“la capacidad de comprender y utilizar información en múltiples formatos de una amplia gama de fuentes cuando se presenta a través de computadoras”* (p. 1). Para él, la alfabetización digital implica la capacidad de evaluar críticamente la información (presentada en diferentes formatos) y tomar decisiones sobre cómo utilizar esta información en diferentes contextos de la vida real.

La expresión "alfabetización digital" fue aplicada a lo largo de la década de 1990 por una serie de autores, que la utilizaron para significar esencialmente la capacidad de leer y comprender elementos de información en los formatos de hipertexto o multimedia (Bawden, 2001). No obstante, Lanham (1995), considera la alfabetización digital como una especie de "alfabetización multimedia" que propone que desde una fuente digital se podría generar muchas formas de texto, de informaciones, imágenes, sonidos, etc. De este modo surge la necesidad de una forma de alfabetización alternativa para interpretar y darle sentido a estas nuevas formas descritas anteriormente. Con el tiempo este concepto se fue desactualizando acorde al avance tecnológico. Glister propuso un enfoque más general, que fueron revisadas por Eshet (2002) quien señala que la alfabetización digital se debe considerar como la capacidad de utilizar las fuentes digitales de forma eficaz y lo vincula con un tipo de pensamiento.

Retomando a Gilster, en su libro de 1997 afirma que *"la alfabetización digital tiene que ver con el dominio de las ideas, no con las pulsaciones en el teclado.* Señala que *"no sólo hay que adquirir la habilidad de hallar las cosas, sobre todo se debe adquirir la capacidad de utilizar esas cosas en la vida del individuo"* (pp. 1-2). En este sentido, en un mundo globalmente conectado se debe poder desarrollar prácticas digitales socialmente responsables y también contribuir a las prácticas digitales en su propia vida personal, laboral y de aprendizaje.

La figura 2.2.1 describe el modelo de desarrollo de la alfabetización digital propuesto Sharpe y Beethan (2010) como una pirámide: en la base de la pirámide se ubica el tomar conciencia de la existencia de la tecnología y el acceso a ella, denominado acceso funcional. Sin embargo, el hecho de que pueda usar una pieza de hardware o software no significa que tenga la capacidad de usarla de manera eficaz. A medida que pasa más tiempo usando la tecnología, adquiere más confianza en sus habilidades técnicas, de información, de comunicación y de aprendizaje. Luego, puede comenzar a aplicar esas habilidades para tomar decisiones y elecciones informadas sobre cómo usar las diferentes tecnologías. A medida que avanza en este proceso, sus experiencias y prácticas contribuyen a su formación impulsando el uso creativo y apropiado de la tecnología.

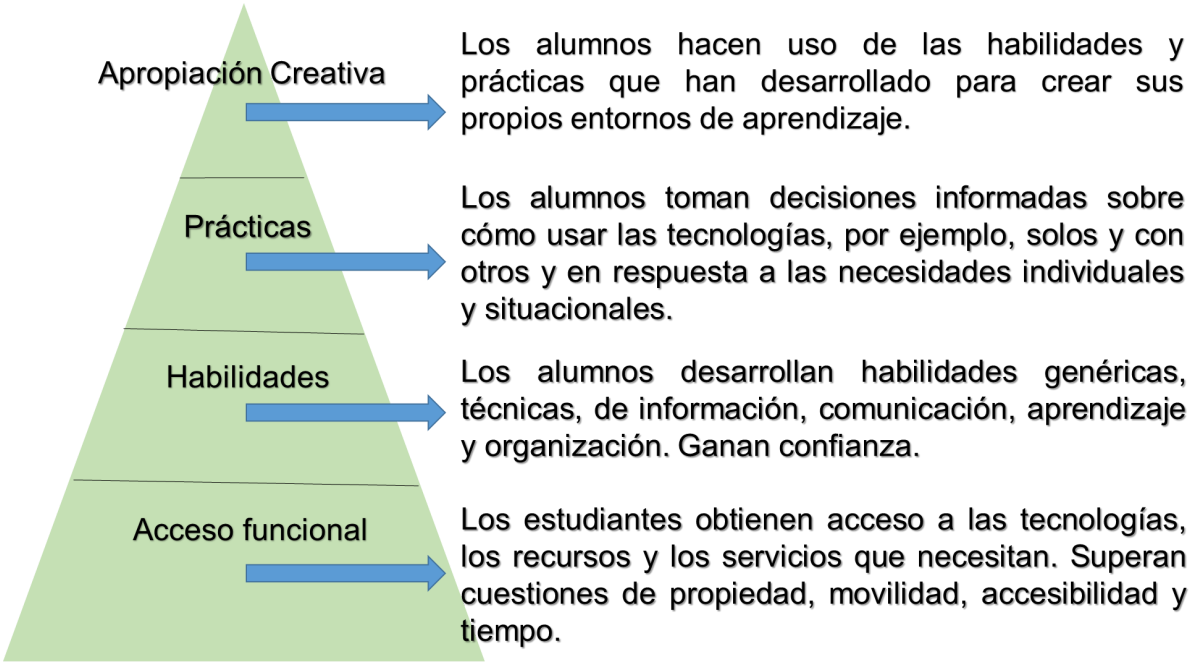


Figura 2.2.1. Modelo de desarrollo de la alfabetización digital basada en la pirámide de Sharpe y Beethan (2010)

Desde el significado de alfabetización digital, aportado por los autores estudiados, este tipo de alfabetización requiere del dominio de un nuevo lenguaje, el digital, y es este sentido que la escuela es una actor relevante en tanto formadora de ciudadanos que puedan intervenir de manera informada en la sociedad actual (Llorens-Largo, 2015). Esta formación relacionada con la informática debe entenderse en las dos vertientes de la educación: la informática educativa y la educación en informática (Sierra-Rodríguez & García-Peñalvo, 2015).

En este sentido surge la necesidad de preparar a los jóvenes para enfrentarse al mundo en el que les tocará vivir, brindándoles las herramientas cognitivas necesarias para desenvolverse adecuadamente en el mundo digital. Surge así el pensamiento computacional como enfoque de trabajo y la programación como práctica para resolver problemas (García-Peñalvo, 2016d; Wing, 2006, 2008; Zapata-Ros, 2015).

En tiempo de pandemia fue notable este tema debido a la adaptación de herramientas para crear un entorno digital, comprender el uso de diferentes plataformas que permitan crear clases virtuales, generar comunicación entre docentes y estudiantes, compartir documentos, crear videos, podcasts, etc.

2.3 Pensamiento computacional

Jeannette Wing (Wing, March 2006), vicepresidenta corporativa de Microsoft Research y profesora de Computer Science Department Carnegie Mellon University, popularizó el término pensamiento computacional en su artículo “Computational Thinking. It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use”.

Recuperando lo mencionado en la motivación de esta tesina, se entiende que el pensamiento computacional implica aplicar procesos de pensamiento lógico, sistémico, algorítmico, para representar soluciones a problemas como lo pueden ser las secuencias de instrucciones y algoritmos. Asimismo, la autora señala que el pensamiento computacional es un tipo de pensamiento al que deberían acceder todas las personas en el sistema educativo, no sólo los informáticos. A su vez, Resnick M, (2008) advierte que el pensamiento computacional fomenta el pensamiento analítico, sistemático, la creatividad y el trabajo colaborativo, todas ellas habilidades son fundamentales para la sociedad actual, a la que el autor denomina “Sociedad de la Creatividad”.

Zapata-Ros (2015) en su estudio publicado en la RED titulado “Pensamiento computacional: Una nueva alfabetizaciòn digital” realiza un análisis con la intención de contextualizar la definición del pensamiento computacional y la relación de habilidades asociadas, desde un análisis y una elaboración interdisciplinar basadas en las teorías del aprendizaje. Como resultado de dicha investigaciòn, el autor establece los siguientes componentes del pensamiento computacional:

- Análisis ascendente. En primer lugar se deben resolver los problemas más concretos, para luego resolver los más abstractos o generales. Pero muchas veces hay que situarse en primer lugar en un nivel abstracto y seleccionar los problemas concretos que nos ilustran para la resolución del problema general.
- Análisis descendente. La resolución de problemas se pueden descomponer en submetodos, o módulos de problemas distintos y auxiliares, o definir acciones concretas, etc.
- Heurística. Se basa en la sistematización de la experiencia de resolver problemas seguido de una serie de pautas como: analizar el problema,

concebir un plan, ejecutar el plan y utilizar técnicas recursivas descomponiendo problemas en problemas similares más sencillos.

- Pensamiento lateral y pensamiento divergente.
    - Pensamiento lateral: promover ideas creativas e innovadoras que se encuentran fuera del patrón del pensamiento habitual.
    - Pensamiento divergente: surge de ideas creativas a través de la imaginación.
- Creatividad. Mihály Csikszentmihalyi (1998) señala que los creativos, *"quienes producen una novedad aceptable en un campo, parecen capaces de usar bien dos formas opuestas de pensamiento: el convergente y el divergente"* El pensamiento convergente estructura conocimientos de manera lógica.
- Resolución de problemas.
- Pensamiento abstracto. Es la facultad del ser humano de abstraer las propiedades de los objetos que son necesarios para un estudio.
- Recursividad. Michael C. Corballis (2007) dice que *"La facultad de pensar sobre el pensar constituye el atributo crítico que nos distingue de todas las demás especies"*. Sostiene que pensar de forma recursiva es la característica principal que distingue a la mente humana de la de otros animales. Nos permite reflexionar sobre nuestra propia mente.
- Iteración. Es un elemento del pensamiento computacional que permite crear procedimientos repetitivos, con una proyección a representaciones cognitivas y en procedimientos complejos que son la base de importantes actividades y tareas en la resolución de problemas.
- Métodos por aproximaciones sucesivas. Ensayo–Error. Son métodos de resolución de problemas en donde se confrontan las ideas formadas de la realidad, en acciones, percepciones y la formación de modelos cognitivos.
- Contingencia e inmediatez. La ayuda del tutor debe hacerse de manera contingente. Además es importante hacerlo así porque el estudiante puede ser sensible a la pérdida de tiempo y perder la motivación cuando ésta sea excesiva o también porque la actividad sea confusa. Para evitar esta confusión, la respuesta debe ser inmediata a los errores de los estudiantes.
- Métodos colaborativos. A través de trabajos colaborativos en el momento en el cual dos o más personas trabajan juntas, compartiendo ideas,

experiencias, generando soluciones, agregando información clara, se trata de encontrar que la información fluya en un sentido de aprendizaje colectivo.

- Patrones. Un patrón (Alexander et al., 1977) "*describe un problema que ocurre una y otra vez en nuestro entorno y, a continuación, describe el núcleo de la solución de ese problema, de tal manera que el usuario puede utilizar esta solución un millón de veces más, sin tener que hacerlo de la misma manera dos veces "*.
- Sinéctica. (Gordon,1961) indica que la sinéctica *"estudia cómo organizar la integración de los diversos individuos que componen un grupo para la resolución de problemas. Es pues una teoría operacional orientada al uso consciente de los mecanismos psicológicos preconscientes que hay presentes en la actividad creadora humana".*
- Metacognición. El concepto de estrategias se incorpora a la psicología cognitiva, desde la perspectiva constructivista del conocimiento y del aprendizaje. Estas estrategias se tratan de la aplicación de una técnica concreta que implica habilidades, destrezas y una serie de técnicas que se aplican en función de las tareas a desarrollar, sobre las que tiene una intención de utilizar conscientemente. En definitiva existe la conciencia de los propios recursos cognitivos con los que cuenta el aprendiz. A este proceso se le denomina metacognición. Por lo tanto se puede ver que la metacognición no es sólo una estratégia o un conjunto de ellas, sino que es la condición necesaria para que pueda darse cualquier plan estratégico. La metacognición se encuentra ligada al estudio de los estilos de aprendizaje. En este sentido Keefe (1979) define los estilos de aprendizaje como la *"combinación de características cognitivas, afectivas, y de factores fisiológicos que sirven como indicadores relativamente estables de cómo un aprendiz percibe, interactúa con él, y responde al ambiente de aprendizaje" y* Stewart y Felicetti (1992) como lo que determina las *"condiciones educativas en las que un estudiante es más probable que aprenda".* Se puede ver como se presenta el factor de intencionalidad al analizar que ambos autores se enfocan en la manera en la cual los estudiantes prefieren aprender.

2.4 Gamificación

Comúnmente el ser humano juega para divertirse, para entretenerse; sin embargo, Crawford (1984) afirma que se juega para aprender, aunque ésta, sea una intención inconsciente. Para este autor las respuestas a las preguntas acerca de ¿por qué la gente juega? ¿qué los motiva? ¿qué hace que los juegos sean divertidos? son cruciales para un buen diseño de juegos. Por otro lado, hay quienes consideran al juego como una actividad social, teniendo relación con otros, aprendiendo pautas de comportamiento, valores y cultura (Huizinga, 1996; Gros, 2000; Gee, 2004;). Por otro lado Vélez (2005) sostiene que el juego promueve la imaginación, la fantasía, y la incertidumbre que abren nuevos caminos a la mente humana. Además Vélez (2005) señala que el juego, la lúdica y las inteligencias se unen para producir un estado de distensión abriendo paso al pensamiento metafórico y al impulso lúdico-emotivo para solucionar problemas de interioridad psíquica del mundo social-cultural. Por lo tanto, podemos afirmar que jugar constituye una estrategia para estimular el desarrollo integral de las personas.

Estas características se aplican tanto a los juegos tradicionales como a los videojuegos. Los videojuegos proporcionan a los jugadores habilidades y destrezas digitales, en tanto los sujetos se sumergen en la cultura digital, generan ambientes divertidos, admiten la capacidad de correr riesgos, de aprender acciones nuevas, de explorar contextos tridimensionales virtuales. Teniendo en cuenta lo mencionado anteriormente, el sujeto que juega con videojuegos establece una relación ventajosa con los objetos tecnológicos en comparación con las personas que no juegan videojuegos. Por otro lado, los sujetos que participan en el contexto del juego, se vinculan con otros y con la comunidad que se forma a través de las redes sociales del ambiente de juego, incorporando pautas de relación y convivencia (Gee, 2004). Al jugar se aprende del desafío, de la experiencia y de las propias acciones. Se han realizado numerosas investigaciones que demuestran las ventajas que tienen los videojugadores, teniendo en cuenta el desarrollo de sus habilidades y destrezas con respecto a los no-videojugadores, entre las que se encuentran: el desarrollo de la coordinación ojo-mano; mayor agudeza visual, rapidez de reacción, capacidad de atención a múltiples estímulos, facilidad para relacionarse con otros, alta motivación al logro, mayor tolerancia a la frustración, capacidad para tomar riesgos, resolver problemas y tomar decisiones (Green y Bavelier, 2006).

Existen mecánicas de juego propias de la gamificación, como técnica para aprender de forma entretenida, que generan sensaciones desafiantes, divertidas, satisfactorias, motivaciones entre otras. Recuperando el trabajo de Seniquel, V., Bakun, M. P., & Gómez Kennedy, M. I. (2015) manifiestan que estas mecánicas cuentan con los siguientes elementos:

- Misiones o retos
- Desafíos
- Premios
- Puntos
- Clasificaciones
- Niveles
- Regalos

2.5 Juegos serios

Las personas cuando juegan buscan diversión, pero cuando el contexto se vuelve competitivo, la diversión pasa a un plano más serio. Si observamos una partida de ajedrez en un torneo donde dos personas se encuentran jugando para lograr el triunfo, a estos sujetos se los verá concentrados para realizar movimientos de sus piezas para ganar el juego.

Los juegos serios según Michael y Chen (2006) son aquellos juegos que se usan para educar, entrenar e informar. El término fue introducido en la década de los 60 por Clark Abt para hacer referencia a juegos que simulaban eventos de la “I Guerra Mundial” y que recreaban las estrategias de guerra en el aula de clases. Sin embargo, en la actualidad, se le asigna este nombre a un grupo de videojuegos y simuladores cuyo objetivo principal es la formación. Esta área de desarrollo y creación de videojuegos ha surgido como una manera inteligente de combinar los beneficios de los videojuegos, su poder de penetración en la población y las necesidades de educación y formación efectiva en diferentes niveles, como el político-institucional el empresarial y el comercial.

2.6 Aplicaciones educativas para aprender a programar

Actualmente, existen herramientas, en su mayoría online, que buscan enseñar a programar y que están pensadas para su inclusión en la escuela o en espacios de formación alternativa destinados a jóvenes (clubes de barrio, eventos específicos, centros comunitarios, etc), como lo es Pilas Bloques, Scratch, entre otras. Si bien existen versiones para ser usadas en distintos contextos, la gran mayoría de ellas no están acompañadas de una secuencia didáctica que permita su uso y aplicación en las escuelas argentinas. La modalidad que utilizan estas herramientas es a través de bloques visuales que al encastrarse al estilo rompecabezas permiten crear programas, siendo una forma interesante para introducir a los estudiantes en el mundo de la programación. Cada instrucción o acción es una pieza de rompecabezas, que puede conectarse con otra pieza o acción para llevar adelante un objetivo. Esto lo hace entretenido y a la vez fácil de interactuar evitando enfrentarse a la sintaxis de un lenguaje de programación. A continuación se describirán algunas de las herramientas analizadas.

2.6.1 Scratch y ScratchJr

Scratch[4] es una herramienta para enseñanza de programación, está disponible de forma gratuita, fue fundada por Massachusetts Institute of Technology, National Science Foundation, Siegel Family Endowment, y LEGO Foundation. La aplicación se encuentra diseñada para ayudar a los jóvenes a desarrollar habilidades de programación a medida que crean proyectos y se enfrentan a temas matemáticos y computacionales, permite crear historias interactivas, animaciones, juegos, música y arte. El sitio oficial de Scratch contiene un apartado donde indica que está pensado para jóvenes entre 8 y 16 años pero la aplicación la utilizan personas de todas las edades. Presenta una interfaz amigable que permite crear scripts arrastrando y soltando bloques.

La figura 2.6.1.1 muestra las categorías que contiene Scratch con sus bloques correspondientes y un tablero donde se encastran dichos bloques para crear un programa.

[4] Sitio oficial de Scratch: https://scratch.mit.edu/

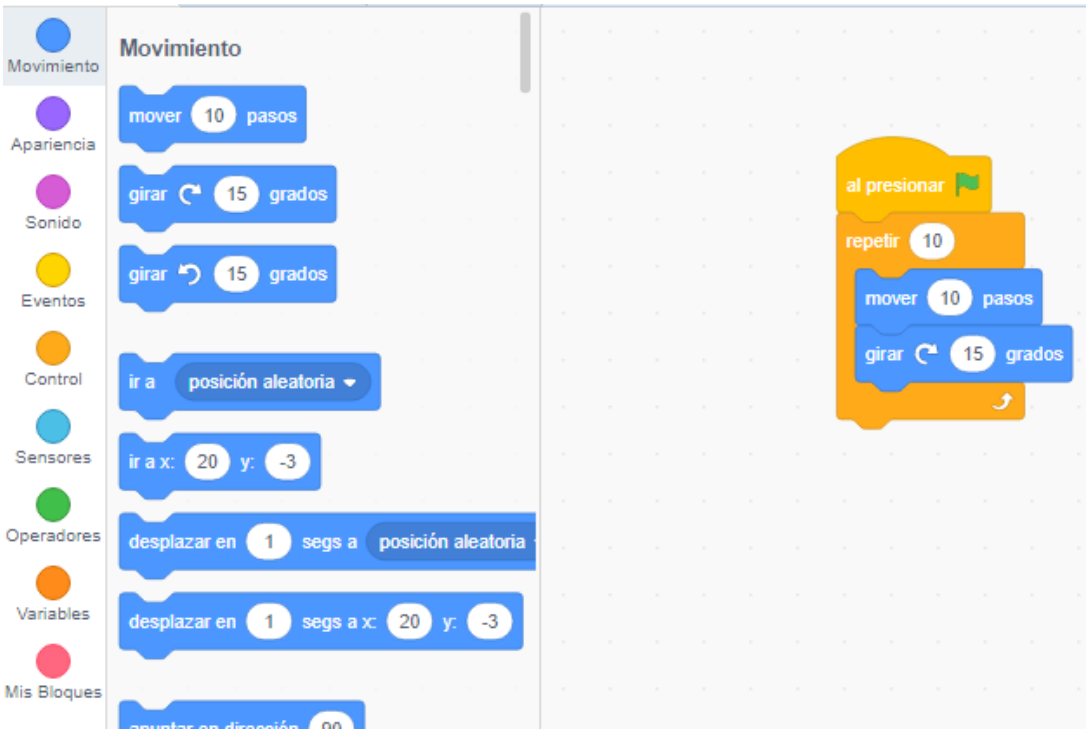


Figura 2.6.1.1 Categorías e instrucciones que contiene Scratch

El sitio oficial de Scratch cuenta con tutoriales, actividades y una amplia comunidad de docentes, sin embargo, es necesario contar con una pc de escritorio, notebook o netbook, no está pensado para celulares, por lo tanto, en escuelas que no cuentan con computadoras los estudiantes y docentes no podrán sacar provecho de Scratch.

ScratchJr es muy parecido a Scratch pero adaptado para celulares y tablets, la figura 2.6.1.2 muestra una escena de la aplicación. Está enfocado a niños entre 5 y 7 años de edad. El sitio oficial no cuenta con actividades en español, lo que requiere tener un nivel de lectura de inglés para aprovechar el recurso.

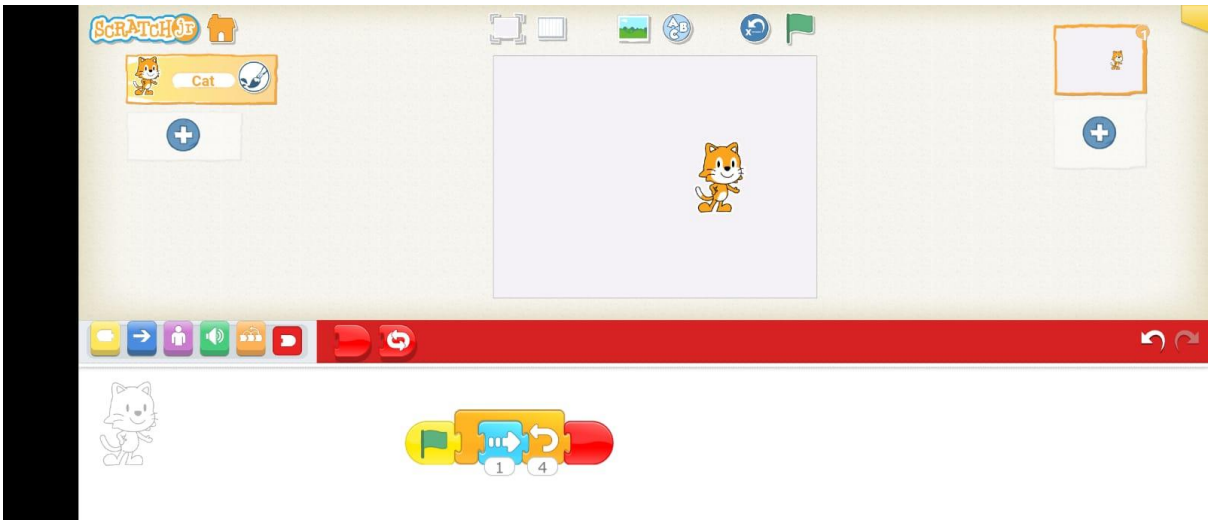


Figura 2.6.1.2 ScratchJr

2.6.2 RITA: Robot Inventor to Teach Algorithms

RITA (Robot Inventor to Teach Algorithms) es una herramienta desarrollada por Vannesa Aybar Rosales (Aybar Rosales V. et al, 2012) como tesis de Licenciatura en Informática en el año 2012, destinada a enseñar a programar en la escuela secundaria. RITA está desarrollado en Java e integra dos framewoks de código fuente abierto: OpenBlocks[5] y Robocode[6], Openblocks ofrece la funcionalidad de programación en bloques visuales y Robocode la de construcción de robots virtuales que compiten en un campo de batalla, la figura 2.6.2.1 muestra el ambiente de programación de RITA a través de bloques y la figura 2.6.2.2 visualiza el campo de batalla con los robots en acción. Con RITA es posible construir juegos de robots virtuales que compiten y para ello es necesario definir las estrategias que cada robot desarrollará en la competición. Los jugadores construyen sus estrategias arrastrando, soltando y encastrando bloques entre sí, los bloques están organizados por categorías de la misma manera que lo hace Scratch. En RITA la lógica de juego está muy presente: los robots consumen energía que es necesario controlar cuando se construyen estrategias, cuentan con sensores que les permite detectar otros robots, compiten por rondas. RITA es usado extensamente con estudiantes de las escuelas secundarias que participan del proyecto de extensión: “EXTENSIÓN en vínculo con escuelas secundarias” de la Facultad de Informática de la UNLP.

En el trabajo (Vanessa Aybar Rosales, Claudia Queiruga, Claudia Banchoff Tzancoff, Isabel Miyuki Kimura y Matías Brown Bartneche, 2017) se analiza si el uso de la incorporación de la comunicación en red mejora la experiencia en el uso de RITA, en relación a la jugabilidad durante las competencias de robots virtuales, y para ello se relevó la opinión de los estudiantes durante el juego.

[5] Massachusetts Institute of Technology. Openblocks: Librería Java open-source para crear bloques de programación. Recuperado en Agosto 2014 de http :// education. mit. edu/ about - old/ open - blocks - download - page/

[6] Mathew Nelson, Flemming N. Larsen. Robocode: Juego educativo open-source para la creación de estrategias para controlar robots en un campo de batalla. Recuperado en Marzo 2014 de http :// robocode. sourceforge. net/

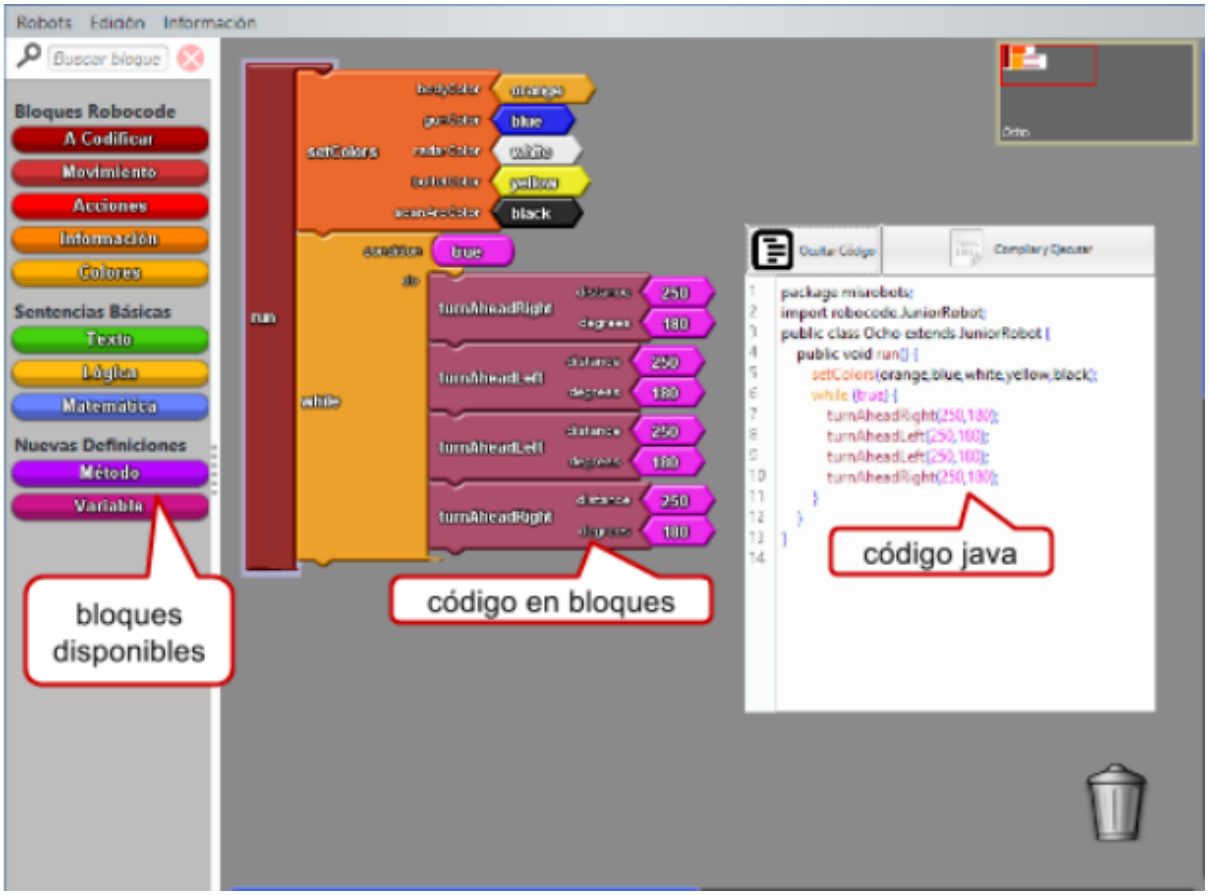


Figura 2.6.2.1 Ambiente de programación de RITA. Extraído de Vanessa Aybar Rosales et al (2017)

Figura 2.6.2.2 Campo de batalla de Robocode. Extraído de Vanessa Aybar Rosales et al (2017)

2.6.3 Pilas Bloques

Pilas Bloques es una herramienta online desarrollada por el proyecto Program.AR de la Fundación Sadosky con la colaboración del proyecto Huayra, está basada en la herramienta Pilas Engine Web desarrollada por Hugo Ruscitti. También

disponible para su descarga en diferentes sistemas operativos (windows, linux y mac os x).

Está organizada en desafíos para el primer y segundo ciclo de la escuela primaria, sin embargo también son adecuados para comenzar a trabajar en conceptos de programación con estudiantes de escuelas secundarias. A medida que se avanza en los desafíos la complejidad aumenta por ello es recomendable que el docente guíe la elección y secuenciación de los desafíos a resolver.

En la figura 2.6.3.1 se puede observar el entorno de Pilas Bloques.

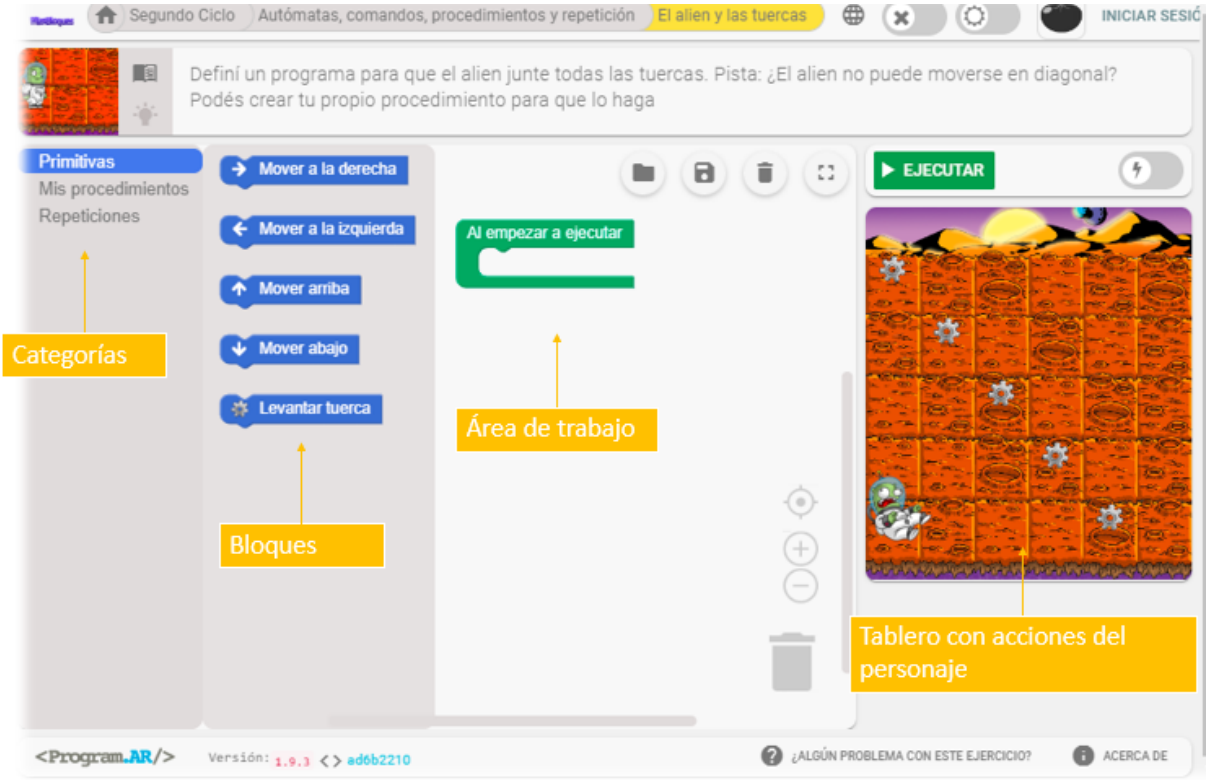


Figura 2.6.3.1 Entorno de Pilas Bloques

Pilas Bloques está diseñada para acompañar una secuencia didáctica, donde se definen varios temas progresivos seguidos de un hilo conductor para el aprendizaje de conceptos de programación.

A su vez el proyecto Program.AR junto con universidades nacionales ha desarrollado manuales para docentes de escuelas en los que se incluyen "secuencias didácticas"[7] que hacen uso de Pilas Bloques.

[7] Sitio de secuencias didácticas: https://program.ar/material-didactico/#manuales-docentes

2.6.4 Blockly Games

Blockly Games es una serie de juegos educativos que enseñan programación por medio de bloques. Está diseñado para niños que no han tenido experiencia previa con la programación de computadoras. Al final de estos juegos, los jugadores están listos para usar lenguajes convencionales basados en texto.

Las habilidades informáticas ayudan a los estudiantes a colaborar, crear y hacer que casi todas las materias parezcan más relevantes. Blockly Games fomenta el desarrollo de los programadores del mañana. Diseñado para ser a su propio ritmo, Blockly Games se puede descargar para su uso sin conexión, lo que garantiza la accesibilidad para todos los estudiantes y la tecnología. Todo el código es de código abierto, lo que significa que es gratuito y personalizable para satisfacer sus necesidades.

Un ejemplo de juego de Blockly Games se muestra en la figura 2.6.4.1 donde para cada animal debe elegir una imagen, la cantidad de patas y sus rasgos.

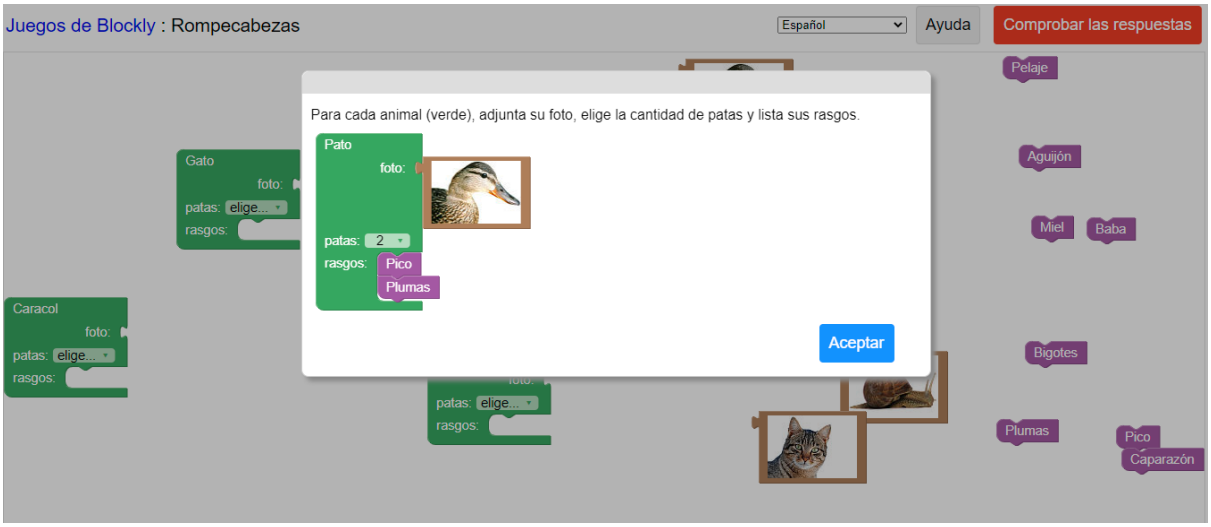


Figura 2.6.4.1 Uno de los juegos de Blockly Games

2.6.5 Sistematización de las herramientas evaluadas

Las herramientas elegidas cumplen con ciertos componentes que plantea Zapata-Ros (2015) con la intención de contextualizar la definición del pensamiento computacional. Como análisis de esta tesina se puede apreciar que las aplicaciones

manifiestan el análisis ascendente, pensamiento lateral, pensamiento divergente, creatividad, resolución de problemas y métodos por aproximaciones sucesivas.

Teniendo en cuenta los componentes del pensamiento computacional mencionados en el párrafo anterior y los elementos que mencionan Seniquel, V., Bakun, M. P., & Gómez Kennedy, M. I. (2015) sobre gamificación (puntos, niveles, premios, desafíos, retos, clasificaciones), se realiza un análisis de las herramientas evaluadas en este capítulo y otras que son de interés para la investigación de BlockAr que formarán parte de la creación de la misma como lo son la conectividad, la interfaz, si se puede instalar en celulares o tabletas, el resultado se puede ver en la tabla 2.6.5.1.

	Scratch	**ScratchJr**	**RITA**	**Pilas Bloques**	**Blockly Games**
Permite crear algoritmos a partir de arrastrar, soltar y conectar bloques.	si	si	si	si	si
Interfaz amigable	si	si	si	si	si
Se basa en resolución de desafíos	no	no	no	si	si
Contiene niveles progresivos de complejidad	no	no	si	si	si
Es código abierto o software libre	si	si	si	si	si
Está orientado a celular o tableta	no	si	no	no	no
Contiene puntuación o premios como técnica de gamificación.	no	no	si	no	no
Cuenta con tutoriales	si	si	si	si	si
Se puede jugar offline	no	si	si	no	si
Propone la enseñanza basada en juegos	no	no	si	si	si

Tabla 2.6.5.1. Comparativa de las herramientas analizadas para acercar la programación a la escuela.

Capítulo 3: Contexto educativo, tecnologías, diseño, implementación y resultados

3.1 Contexto educativo y disponibilidad de recursos tecnológicos

La Escuela Secundaria N° 9 de Lisandro Olmos que formó parte de mi investigación, pertenece a la gestión estatal. De esta escuela fui docente en un curso de 4F, es decir, habían varios cursos de 4 como así también de otros años. Contaba con una matrícula promedio de 28 estudiantes en el curso de NTIC de 4F. No incluye nivel primario, por lo que el espacio es solo para nivel secundario.

La comunidad pertenece generalmente a jóvenes de zonas aledañas a la escuela, también se integran estudiantes de Etcheverry, Abasto, entre otras localidades cercanas.

En cuanto a la disponibilidad de recursos tecnológicos para uso en la asignatura, la escuela cuenta con quince netbooks provenientes del plan federal “Conectar Igualdad”, que se comparten con el resto de las asignaturas de la escuela. Por otro lado, la escuela no dispone de una sala de computadoras para la materia de NTIC, y las netbooks hay que solicitarlas con tiempo porque puede pasar que no estén disponibles para una fecha, por lo tanto, no se consigan para las actividades planificadas. A su vez, la escuela no cuenta con conectividad a Internet, al momento que desarrolle la tesina.

La mayoría de los estudiantes disponen de teléfonos celulares personales con los que se podría trabajar en clase.

Al comenzar el ciclo lectivo en el año 2021, se comunicó a los estudiantes y al equipo de gestión de la escuela, la intención de utilizar los teléfonos celulares de los estudiantes como recurso tecnológico para la materia NTIC. Luego, se relevó la disponibilidad y características de los celulares de los estudiantes de la asignatura, se determinó que las especificaciones técnicas son variadas, por lo general no cuentan con la misma tecnología. A partir de este relevamiento, se tomó la decisión de desarrollar una aplicación móvil que no consuma muchos recursos del celular, para poder ser adaptada al celular de cada estudiante, tenga o no un celular potente, y que sea multiplataforma para poder ser instalado tanto en sistemas

operativos Android como iOS o usado a través de un navegador web desde una computadora.

La figura 3.1.1 describe las versiones de sistemas operativos de los celulares de 25 estudiantes que cursaron la asignatura NTIC en el año 2021.

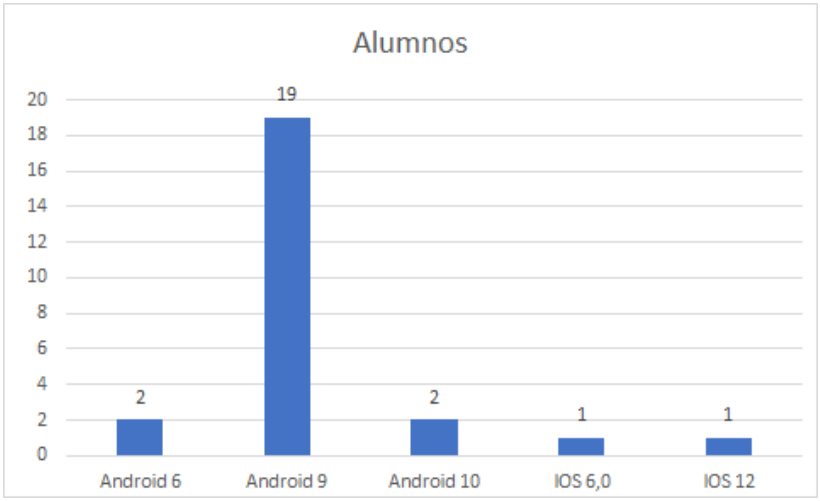


Figura 3.1.1. Cantidad de estudiantes por versión de Sistema Operativo.

La figura 3.1.2 muestra que la mayoría de los estudiantes cuentan con un celular de uso personal, aunque se detectó que 2 estudiantes no tienen celular. Estos datos se obtuvieron en clase al comenzar el ciclo lectivo en marzo del año 2021 al iniciar las clases presenciales, previo al aislamiento.

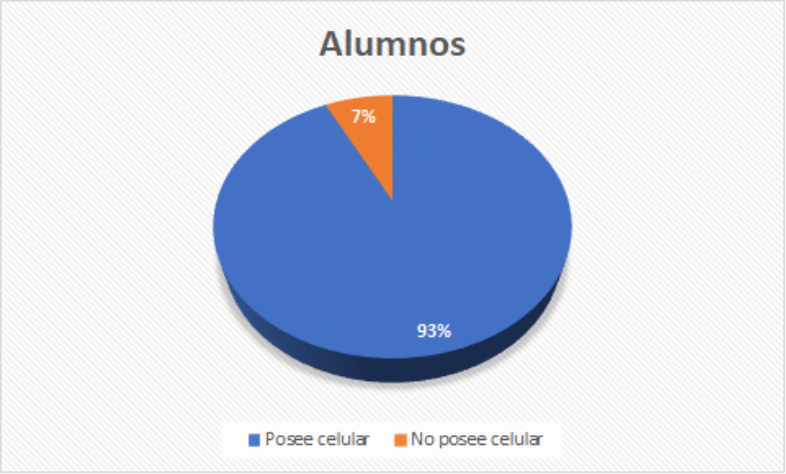


Figura 3.1.2. Cantidad de estudiantes que poseen celulares.

Este relevamiento permitió determinar que era posible incluir el uso de los teléfonos celulares de los estudiantes en las clases dado que la mayoría de los estudiantes cuentan con uno y así aprovecharlos como herramientas de estudio.

Se consultó sobre los planes de datos móviles y un porcentaje pequeño manifestó contar con datos móviles en sus celulares, a esto se suma que en el momento de realizar este relevamiento la escuela no disponía de conexión a Internet.

Como docente de la asignatura NTIC[8] de la escuela, a partir de la información obtenida en este relevamiento considero de suma utilidad poder abordar, desde la asignatura, el aprendizaje de programación trabajado a partir de problemáticas cotidianas, conocidas por los estudiantes, y para ello se propone desde este trabajo de tesina el desarrollo de una aplicación multiplataforma que integre el enfoque de programación con bloques visuales, al estilo puzzles, acompañado de secuencias didácticas que permitan alcanzar los objetivos de aprendizaje en torno a pensamiento lógico, pensamiento algorítmico, y lenguajes de programación.

La aplicación BlockAr se integra muy bien en celulares con sistema operativo android, tiene diferentes formas de ser descargadas, puede ser a través de bluetooth o desde cualquier lugar que se tenga conexión a internet, una vez instalada la aplicación almacena los datos localmente, por lo que no requiere conexión a internet para poder usarla.

3.2 BlockAr es software libre y código fuente abierto

El código de BlockAr fue desarrollado y escrito contemplando los propósitos del software libre y las características de código abierto. El código fuente se puede adquirir y redistribuir sin costo y puede ser modificado a criterio propio. Se puede descargar y compartir en cualquier ámbito, no es necesario que sea una institución educativa.

3.3 Tecnologías utilizadas para el desarrollo

Para el desarrollo de BlockAr se consideraron en una primera etapa herramientas que luego fueron descartadas por no adherir a los estándares de software libre y código fuente abierto, entre ella Unity[9], Unreal Engine[10], GameMaker Studio[11] y Construct 3[12].

[8] Diseño Curricular para la educación Secundaria. Nuevas Tecnologías de la Información y la Conectividad (2010): http://servicios.abc.gov.ar/lainstitucion/organismos/consejogeneral/disenioscurriculares/secundaria/materias_comunes_a_todas_las_orientaciones_de_4anio/tic_4_final_web.pdf

[9] Unity: https://unity.com/es

[10] Unreal Engine: https://www.unrealengine.com/en-US

[11] GameMaker Studio: https://gamemaker.io/es/gamemaker

[12] Construct 3: https://editor.construct.net/

Las herramientas que se detallan a continuación se eligieron porque no presentan restricción alguna para el proceso de desarrollo y uso del juego, además de adherir a los estándares de software libre y de código abierto.

El desarrollo de BlockAr cuenta con phaser 3, un framework de software libre y de código abierto para crear juegos. Al crear una instancia del juego se necesita pasarle una configuración. Actualmente la configuración se encuentra como muestra la figura 3.3.1, donde en type se indica el renderizador que se le debe asignar; al principio se optó por elegir WebGL pero durante las pruebas en celulares presentaba algunos problemas gráficos por lo que se cambió a CANVAS y se solucionaron. Luego en “scale” se indica el tamaño y la alineación del lienzo que tendrá el juego. En “scene” se van a definir los diferentes escenarios de BlockAr y por último se debe definir un tipo de física en “physics”, Phaser brinda soporte para 3 sistemas de física: estos sistemas son encargados de simular las leyes de la física en los videojuegos, por ejemplo, masa, volumen, densidad, gravedad, etc.

Phaser contiene tres sistemas de física:

- Arcade Physics, una biblioteca extremadamente liviana perfecta para dispositivos de baja potencia.
- Impact Physics para soporte de mosaico avanzado.
- Matter.js, un sistema de física de cuerpos y asignarle propiedades físicas como masa, área o densidad, también puede simular diferentes tipos de colisiones como la gravedad y la fricción.

Para BlockAr era suficiente elegir Arcade Physics ya que la gráfica y las funciones que se utilizan no son muy complejas, por otro lado se aprovecha que sea liviano, ideal para celulares y suficiente para los requerimientos de la aplicación.

```
config = {
    type: Phaser.CANVAS,
    scale: {
        width: 800,
        height: 480,
        mode: Phaser.Scale.FIT,
        parent: document.getElementById("gameRight"),
        autoCenter: Phaser.Scale.CENTER_BOTH,
    },
    physics: {
```

```
        default: 'arcade',
        arcade: {
            debug: false
        },
    },
    scene: [ SceneMenu, Scene1, Scene2, Scene3, Scene4, Scene2_1,
Scene2_2,Scene5_1]
};
```

Figura 3.3.1. Configuración de Phaser 3 que utiliza BlockAr

Las escenas en Phaser contienen un método constructor(), un método Preload() que carga un conjunto de archivos que van a ser utilizados, un método Create() que indica el comportamiento del juego, por ejemplo, en BlockAr se indica la interacción del personaje en el mapa con los objetos, éste personaje es un sprite sheet, es decir, es una imagen como muestra la figura 3.3.2. Una de las características de los sprite sheet es brindar animación. La animación le da un aspecto más divertido al juego, pero solo con la animación no es suficiente, por lo tanto, el personaje (como un objeto) debe interactuar con otros objetos, y para que esto suceda, el sistema de física de Phaser permite asignar a los objetos la función de colisionar con otros, de esta manera se puede conocer el momento en el cual el personaje toca otro objeto. Gracias a las funciones de colisiones es posible monitorear el estado del objeto dentro de un área restringida definida, puede ser un rectángulo, donde el personaje pueda desplazarse dentro del mismo y controlar que no se pase del perímetro.

Figura 3.3.2. Ejemplo de sprite sheet

La figura 3.3.3. muestra un ejemplo de lo mencionado anteriormente, en la primera línea, se crea un objeto en una posición dada, este objeto será nuestro personaje, en las siguientes líneas estamos indicando que el personaje pueda colisionar con los objetos y con el perímetro que se crea en la última línea. Entonces si el personaje toca algún límite, puede escuchar dicho evento e indicar que se está yendo del perímetro.

```
1. sprite = this.physics.add.sprite(initPosX, initPosY, 'dude');
2. sprite.setBounce(1,1);
3. sprite.setCollideWorldBounds(true);
4. sprite.body.setBoundsRectangle(new Phaser.Geom.Rectangle(width, 200 -
   2, width, height + 4));
```

Figura 3.3.3. Ejemplo de sprite habilitado para colisionar.

Hay una gran comunidad[13] que utiliza Phaser. Como resultado, está evolucionando y mejorando constantemente y se resuelven errores en base a los reportes de los usuarios que trabajan con Phaser. En mi caso me ayudó a encontrar la forma de integrar Blockly[14] con Phaser y que se adapte a Apache Cordova, entre otras cuestiones, mientras desarrollaba BlockAr que fueron bloqueantes pero me dieron soporte para continuar programando.

Otra de las tecnologías que formaron parte de BlockAr fue la integración de Blockly, es una biblioteca de código abierto desarrollada por Google que facilita la adición de programación visual basada en bloques a una aplicación. Está diseñado para ser flexible y admite un gran conjunto de funciones para diferentes aplicaciones. Blockly permite generar bloques específicos, por lo que se lo ha utilizado en aplicaciones que permiten programar personajes animados en una pantalla; crear guiones de historias; controlar robots; e incluso generar documentos legales. Los desarrolladores que usan Blockly crean sus propios lenguajes de bloque. Cuando los desarrolladores crean una aplicación usando Blockly, deben considerar cuidadosamente el estilo, los bloques a utilizar, qué API y qué características de lenguaje son adecuadas para su audiencia.

En BlockAr se utiliza Blockly para brindarle movimiento al personaje y para que interactúe con los objetos del mapa. En la figura 3.3.4 se muestra un ejemplo de código de un bloque para darle movimiento al personaje hacia la derecha.

```
/* Crea el bloque “Ir a derecha” */
Blockly.Blocks['move_right'] = {
  init: function() {
    this.appendDummyInput()
        .appendField("Ir a derecha");
    this.setInputsInline(false);
```

[13] Sitio de la comunidad de Phaser https://phaser.io/community.
[14] Sitio oficial: https://developers.google.com/blockly.

```
    this.setPreviousStatement(true, null);
    this.setNextStatement(true, null);
    this.setColour(330);
 this.setTooltip("");
 this.setHelpUrl("");
  }
};
/* Almacena el nombre de la función que se invoca al ejecutar “move_right”
*/
Blockly.JavaScript['move_right'] = function(block) {
  var code = 'ir_a_derecha();\n';
  return code;
};

/* Una vez que se ejecuta el bloque “move_right” se ejecuta la función
ir_a_derecha */
function initInterpreterGoRight(interpreter, scope) {
  Blockly.JavaScript.addReservedWords('ir_a_derecha');
   var wrapper = interpreter.createAsyncFunction(
    function(callback) {
      spriteRight();
      timeline.play();
      setTimeout(function(){
        timeline.destroy();
        timeline = yo.tweens.createTimeline();
        callback();
      },  timeSprite);
    });
  interpreter.setProperty(scope, 'ir_a_derecha', wrapper);
}

/* Llama a funciones de Phaser para generar los movimientos del personaje */
function spriteRight(){
    timeline.add({
        targets: sprite,
        x: {
            getEnd: function (sprite, key, value)
            {
                posX += moveX;
                return posX;
            },
```

```
            getStart: function (sprite, key, value)
            {
                sprite.anims.play('right');
                return posX;
            }
        },
        ease: 'Power1',
        duration: timeSprite
    });
}
```

Figura 3.3.4. Una porción de código que muestra cómo se vincula Blocky con Phaser

Con lo mencionado anteriormente, a modo resumen, conseguimos desarrollar un juego que permite programar a través de bloques, pero para lograr cumplir con el objetivo de esta tesina es necesario que el juego se pueda instalar en celulares, para esto se eligió Apache Cordova[15], un framework de desarrollo móvil de código abierto y permite utilizar tecnologías web estándar: HTML5, CSS3 y JavaScript para el desarrollo multiplataforma. Las aplicaciones se ejecutan dentro de contenedores dirigidos a cada plataforma y se basan en enlaces API que cumplen con los estándares para acceder a las capacidades de cada dispositivo, como sensores, datos, estado de la red, etc.

La Figura 3.3.5, obtenida del sitio oficial de Apache Cordova, muestra la arquitectura de una aplicación Cordova donde se puede observar cómo interactúa cada elemento.

[15] Sitio oficial: https://cordova.apache.org/.

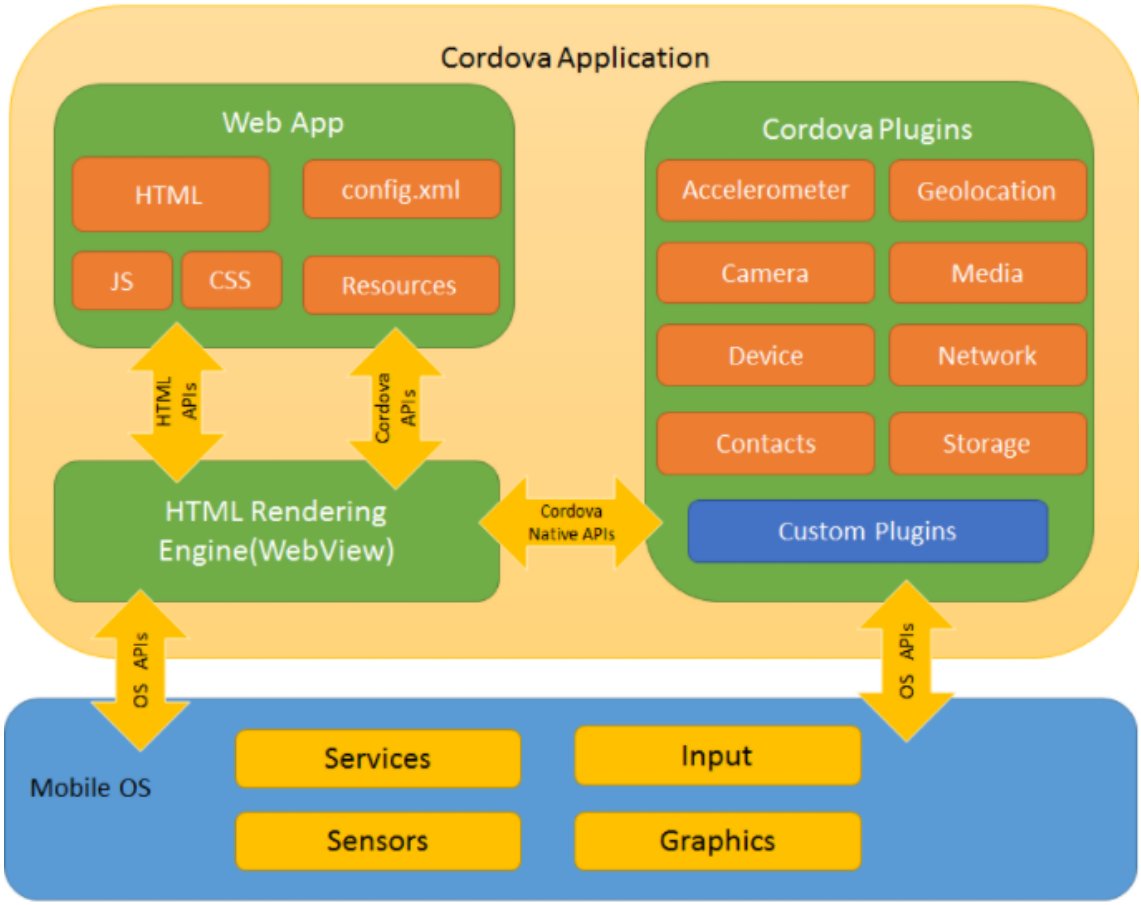


Figura 3.3.5 Arquitectura de la aplicación Cordova. Imagen obtenida del sitio oficial de Apache Cordova.

A continuación se detallan algunos elementos de la arquitectura propuesta por Apache Cordova:

- El WebView proporciona a la aplicación la interfaz de usuario. En algunas plataformas, también puede ser un componente dentro de una aplicación híbrida más grande que mezcla WebView con componentes de aplicaciones nativas.
- En la Web App reside el código de la aplicación. La aplicación en sí se implementa como una página web, de manera predeterminada, un archivo local llamado index.html, que hace referencia a CSS, JavaScript, imágenes, archivos multimedia u otros recursos necesarios para su ejecución. La aplicación se ejecuta en un WebView dentro del contenedor de aplicaciones nativo, que se distribuye a las tiendas de aplicaciones. Este contenedor tiene un archivo muy crucial, el archivo config.xml que proporciona información sobre la aplicación y especifica los parámetros que afectan su funcionamiento, como si responde a los cambios de orientación.

Los Plugins son una parte integral del ecosistema de Córdoba. Proporcionan una interfaz para que Cordova y los componentes nativos se comuniquen entre sí y los enlaces a las API de dispositivos estándar. Esto le permite invocar código nativo desde JavaScript.

Apache Cordova proporciona dos flujos de trabajo básicos para crear una aplicación móvil. Para comprender un poco cada flujo se dará una breve descripción a continuación:

Flujo de trabajo multiplataforma (CLI): la aplicación se ejecuta en tantos sistemas operativos móviles diferentes como sea posible, con poca necesidad de desarrollo específico de la plataforma. Este flujo de trabajo se centra en la Cordova CLI. La CLI es una herramienta de alto nivel que le permite crear proyectos para muchas plataformas a la vez, abstrayendo gran parte de la funcionalidad de los scripts de shell de nivel inferior. La CLI copia un conjunto común de web assets en subdirectorios para cada plataforma móvil, realiza los cambios de configuración necesarios para cada uno, ejecuta scripts de compilación para generar binarios de aplicaciones. La CLI también proporciona una interfaz común para aplicar complementos a su aplicación.

Flujo de trabajo centrado en la plataforma: este flujo de trabajo se utiliza si se quiere crear una aplicación para una única plataforma y necesita poder modificarla en un nivel inferior. Este enfoque se utiliza, por ejemplo, si se quiere que la aplicación mezcle componentes nativos personalizados con componentes Cordova basados en web. Como regla general, se debería utilizar este flujo de trabajo si se necesita modificar el proyecto dentro del SDK. Este flujo de trabajo se basa en un conjunto de scripts de shell de nivel inferior que se adaptan a cada plataforma compatible y una utilidad de Plugman, para administrar plugins, separada que le permite aplicar complementos. Si bien se puede usar este flujo de trabajo para crear aplicaciones multiplataforma, generalmente es más difícil porque la falta de una herramienta de nivel superior significa ciclos de compilación separados y modificaciones de complementos para cada plataforma.

Para esta tesina se optó por elegir el flujo de trabajo multiplataforma para que sea más sencillo lograr crear ejecutables para Android y Windows. Por otro lado se buscó crear una aplicación para IOS pero no se contó con los recursos necesarios.

En este caso se requería contar con una mac para poder crear el ejecutable tanto para IOS como para MacOs.

Como se mencionó anteriormente, Apache Cordova utiliza tecnologías HTML 5, CSS3, Javascript para el desarrollo multiplataforma, por lo tanto, daremos una definición breve de cada una de ellas para entender la estructura básica del código de BlockAr.

- HTML[16] 5 es la quinta revisión del lenguaje HTML. Esta versión define los nuevos estándares de desarrollo web, rediseñando el código para resolver problemas y actualizándose así a nuevas necesidades. No se limita solo a crear nuevas etiquetas o atributos, sino que incorpora muchas características nuevas.
- CSS[17] (Cascading Style Sheets) es el lenguaje de estilos utilizado para describir la presentación de documentos HTML o XML (en-US) (incluyendo varios lenguajes basados en XML como SVG, MathML o XHTML). CSS describe cómo debe ser renderizado el elemento estructurado en la pantalla, en papel, en el habla o en otros medios.

Desde CSS3, el alcance de las especificaciones se incrementó de forma significativa y el progreso de los diferentes módulos de CSS comenzó a mostrar varias diferencias, lo que hizo más efectivo desarrollar y publicar recomendaciones separadas por módulos.

- JS[18] JavaScript es un lenguaje ligero, interpretado y orientado a objetos con funciones de primera clase, y mejor conocido como el lenguaje de programación para las páginas Web, pero también se utiliza en muchos entornos que no son de navegador. Es un lenguaje de scripts que es dinámico, multiparadigma, basado en prototipos y admite estilos de programación orientados a objetos, imperativos y funcionales.

3.4 Diseño de BlockAr

BlockAr es un juego serio, como se menciona en el capítulo 2.5, para acercar al estudiante a la programación y además cuenta con desafíos que involucran

[16] https://dev.w3.org/html5/spec-LC/
[17] https://developer.mozilla.org/es/docs/Web/CSS
[18] https://developer.mozilla.org/es/docs/Web/JavaScript/About_JavaScript

conceptos que se enseñan en la materia NTIC, por ejemplo, memoria ram, disco duro, servidor, entre otros.

El desarrollo de BlockAr tiene como propósito la elaboración de una aplicación que pueda ejecutarse en diferentes plataformas, pero no sólo eso, sino que también toda la aplicación es de código abierto y libre, se menciona en el capítulo 3.2. En el Anexo 1 se puede encontrar la guía de configuración e instalación de las tecnologías mencionadas en el capítulo 3.3. En el siguiente enlace se encuentra el código de BlockAr y los APK para instalar en Android, dentro de la carpeta descargables, https://github.com/sebaspierini/blockar.

BlockAr integra varias tecnologías, entre ellas se encuentran: Blockly, Phaser 3, JS Interpreter, HTML5 y Javascript, pero para poder ejecutarse en diferentes plataformas es necesario de una herramienta como Apache Cordova, también de código abierto y libre, que se encarga de construir los ejecutables para Android, IOS, OsX, Windows 10, entre otros.

3.4.1 Interfaz y didáctica de BlockAr

En esta sección se presenta la interfaz de BlockAr y la didáctica que representa la forma en la cual los estudiantes se vinculan con el contenido del juego, de modo tal que la aplicación logre atraer a los estudiantes para que desarrollen el pensamiento analítico, sistemático, la creatividad y el trabajo colaborativo, como se mencionó en el capítulo 2.3. La secuencia didáctica se incluye en el Anexo 2.

Los datos, como los intentos, los puntajes, los desafíos completados, serán almacenados localmente en el dispositivo. De esta manera cada dispositivo podrá tener guardados los datos del progreso del juego.

El almacenamiento es mínimo, no requiere de mucha memoria. Para lograr esto se utiliza LocalStorage. El juego en sí no pesa más de 4MB, lo que lo hace ideal para dispositivos que no cuentan con mucha memoria. Este punto fue uno de los temas tratados para elegir las tecnologías para el desarrollo de BlockAr, ya que fue posible diseñar un juego para que la mayoría de los estudiantes pudieran jugar, porque no todos contaban con dispositivos nuevos y con capacidades grandes de almacenamiento.

La figura 3.4.1.1 muestra la pantalla de inicio de BlockAr, donde puede observarse que en la parte izquierda de la pantalla, hay un área de texto, dónde se muestran los intentos que el usuario fue acumulando al completar los desafíos. Por cada nivel el usuario deberá cumplir con los objetivos propuestos y según la cantidad de intentos que tenga para resolver el desafío se le otorgará un puntaje, con la estrategia que se muestra en la tabla 3.4.1.1 y se le sumará el total de cada desafío que fue completado correctamente. Los valores se quedan guardados permanentemente y se almacenan en el dispositivo como datos locales.

En la parte derecha de la pantalla se muestran los desafíos que el usuario deberá ir completando. Al seleccionar un número de desafío la aplicación lo llevará a la escena correspondiente que seleccionó.

En la figura 3.4.1.1 se puede ver en color verde los desafíos que se encuentran completados. Esto le permite a los estudiantes darse cuenta de los desafíos que aún no han completado.

Figura 3.4.1.1. Pantalla de inicio.

Intentos	Puntaje	Descripción
1	3	Resuelve el desafío en un intento
2	2	Resuelve el desafío en dos intentos
3	1	Resuelve el desafío en tres o más de tres intentos

o más		

Tabla 3.4.1.1. Puntajes.

3.4.2 Desafíos

Los desafíos de BlockAr están orientados al aprendizaje del estudiante para que logre fomentar el pensamiento computacional y desarrollar habilidades de alfabetización digital. Cada desafío cuenta con un grado de dificultad según el nivel de conocimiento que el estudiante va adquiriendo en las clases.

Al seleccionar un desafío se mostrará la descripción que indica lo que el personaje debe hacer: las puntuaciones, un botón para comenzar a jugar y otro botón por si quiere volver a la pantalla de inicio; esta información se puede ver en la figura 3.4.2.1.

Figura 3.4.2.1. Pantalla descriptiva del desafío antes de comenzar.

Una vez que comiencen el juego presionando el botón comenzar, se podrán ver las categorías y las acciones que puede realizar el personaje según la categoría (figura 3.4.2.2), de este modo se dará inicio a la creación de bloques que se deben conectar en el bloque “comenzar” creando el algoritmo adecuado para resolver el

desafío. A continuación la figura 3.4.2.3 muestra un ejemplo donde el estudiante arrastra y suelta bloques dentro del bloque comenzar, encastrando las piezas y construyendo el programa para darle instrucciones al personaje.

Figura 3.4.2.2. Vista del desafío una vez que se hace clic en comenzar.

Figura 3.4.2.3 Se observan las categorías, las acciones que puede realizar el personaje según la categoría movimientos, el bloque comenzar que tiene varias acciones y el recorrido que hará el personaje.

Para lograr cumplir el desafío el estudiante deberá comprender el enunciado que debe realizar y tendrá que poner a prueba sus habilidades para completar el objetivo del juego.

Existen distintos desafíos enumerados del 1 al 7 donde 1 es el más fácil y el 7 el más difícil. Cada uno de ellos contiene objetos como los de la tabla 3.4.2.1. que interactúan con el personaje de la figura 3.4.2.4. Cada desafío además cuenta con botones (tabla 3.4.2.2). que son necesarios para interactuar con el juego.

Nombre de objeto	Imagen	Nombre de objeto	Imagen
Celular		Memoria RAM	
PC		Disco Rígido	
Placa Madre		Placa de Video	
Servidor		Tienda	

Tabla 3.4.2.1. Objetos que se relacionan con los contenidos de la materia NTIC.

El personaje, mostrado en la figura 3.4.2.4. debe moverse en el tablero y responder a las acciones del usuario. También puede interactuar con objetos.

Figura 3.4.2.4. Personaje.

Nombre del botón	Imagen	Descripción
Info		Brinda información sobre el desafío y el puntaje.
Mostrar área de bloques		Muestra el área donde se crea el algoritmo a través de los bloques.
Aumentar velocidad		Acelera o reduce la velocidad del Personaje.
Ejecutar		Ejecuta el juego según los bloques que se encuentren dentro del bloque “Comenzar”.
Reiniciar		Reiniciar el juego. Ubica al personaje en la posición de comienzo, restablece las variables a su valor inicial.

Inicio		Te redirige a la pantalla principal.
Ampliar área de bloques		Amplía el área donde se crea el programa con bloques.
Reducir área de bloques		Reduce el área de bloques a su tamaño de inicio.

Tabla 3.4.2.2. Indica la función de cada botón del juego.

Cada desafío está pensado para que los estudiantes logren comprender los objetivos que se mencionan en la tabla 3.4.2.3, además contarán con la guía del docente para brindarle a los estudiantes las herramientas necesarias para profundizar en los procesos cognitivos para resolver los desafíos.

Desafío	Objetivos
1	• Comprender el significado de programar. • Comprender el significado de instrucciones en un programa. • Comprender entre instrucción simple y secuencia de instrucciones. • Analizar el resultado de las instrucciones. • Introducir al pensamiento computacional. • Comprender el uso del tablero donde el personaje puede moverse. • Comprender que el personaje puede interactuar con objetos en el tablero.
2, 3 y 4	• Comprender la diferencia entre instrucciones simples e iteradores. • Identificar que los bloques que se encuentran dentro del bloque “repetir” se pueden repetir varias veces. • Comprender que las computadoras utilizan este tipo de iteradores para ejecutar instrucciones.

	• Utilizar el término pc.
5 y 6	• Comprender el uso de procedimientos. • Diferenciar entre procedimientos e instrucciones simples. • Identificar nombres cortos y claros para los procedimientos. • Utilizar términos cómo, placa de video, placa madre, memoria ram, disco duro.
7	• Comprender el significado de condicional en programación. • Utilizar el término servidor.

Tabla 3.4.2.3. Objetivos de los desafíos

En la figura 3.4.2.4 se pueden ver procedimientos con sus respectivos nombres, llamados a procedimientos a través de sus nombres, iteradores, condicionales, secuencias, y operaciones.

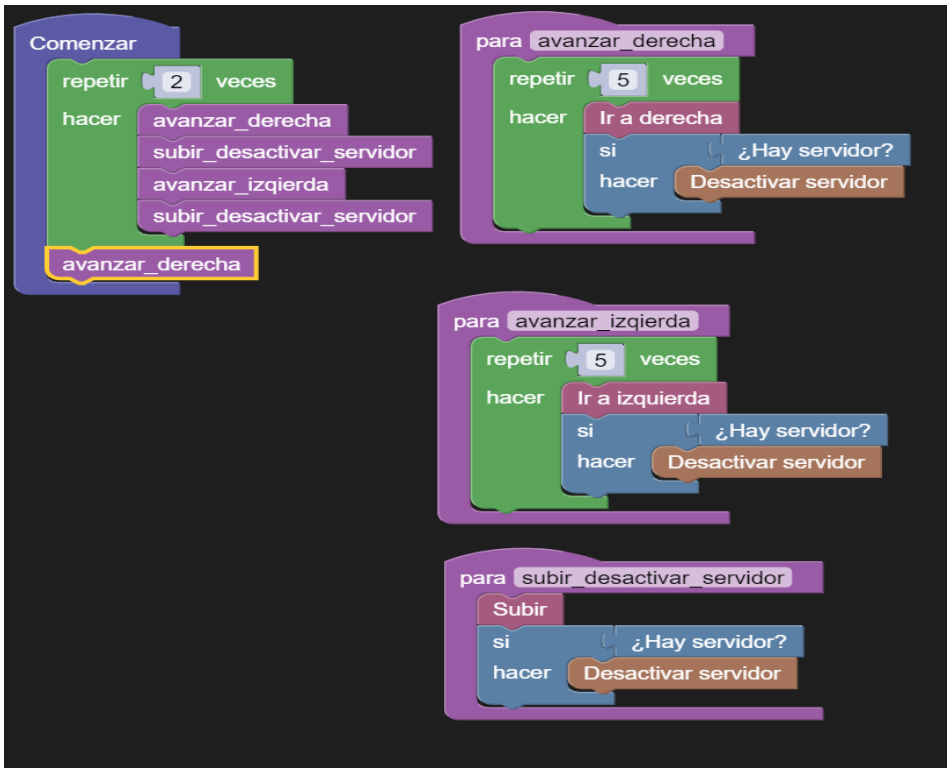


Figura 3.4.2.4. Posible solución para el desafío 7 a modo de ejemplo.

3.4.3 Secuencia didáctica diseñada para BlockAr

La Secuencia didáctica diseñada exclusivamente para BlockAr surge de la necesidad de fortalecer los contenidos de la materia NTIC junto a nuevos conceptos de programación y lograr transmitir contenido adicional a través de explicaciones que dará el docente, captando la atención y sosteniendo la participación de los estudiantes, utilizando metodologías propias e intuitivas, tomando como eje los temas que se encuentran en dicha secuencia. Esta misma será una guía para el docente que lo ayudará a llevar adelante las clases junto con el uso de la aplicación BlockAr. Contará de 5 clases de 2 horas cada una, donde se dividen en inicio (explicación teórica), desarrollo (actividades que los estudiantes deben realizar) y cierre (actividades de refuerzo), además dispondrá de imágenes ilustrativas que ayudará al docente para comprender la secuencia y que lo ayude a transmitir dichos conceptos, iniciando en la comprensión de los significados de programación en un nivel básico (principiante) desde el concepto de instrucciones, objetos, sensores, hasta lograr ampliar el conocimiento a secuencia, iteraciones, rutinas y en un nivel más elevado, lograr comprender los condicionales.

En el Anexo 2 se encuentra la secuencia didáctica.

3.5 Evaluación de BlockAr

Al comienzo las pruebas se realizaron ante tres docentes, dos licenciados en sistemas, un ingeniero industrial, dos doctoras en sistemas. Algunos con experiencia en aplicaciones para celulares. De las pruebas informales se obtuvieron resultados positivos y, las críticas recibidas permitieron realizar algunas modificaciones. Por lo tanto, surgieron algunos cambios en base a la experiencia de los usuarios. Uno de los cambios realizados está relacionado con un defecto de gráficos, visual, que se pudo arreglar configurando webgl por canva. Otro punto a considerar fue que el panel de bloques inicial era muy pequeño cuando se trataba armar un algoritmo más amplio, por lo que se dió la posibilidad de ampliar el área de bloques. Por último, otra modificación realizada está relacionada a la ejecución del juego: cuando el juego se reiniciaba seguía corriendo, lo que llenaba la memoria y se tildaba el celular. Esto fue solucionado borrando la instancia que se estaba ejecutando al reiniciar el juego.

Una vez realizadas las modificaciones y luego de probarlas nuevamente por los mismos usuarios con feedback exitoso, se decidió probar BlockAr con los estudiantes.

Las clases fueron guiadas, en principio se les solicitó que descarguen la aplicación en sus casas ya que en la escuela no cuenta con Internet y pocos estudiantes tenían datos móviles. Aquellos estudiantes que no hicieron esta tarea, la completaron en la clase para lo cual se les compartió el juego por bluetooth logrando que todos aquellos que disponían de un celular con el sistema operativo Android pudieran instalar el juego. Se elaboró una secuencia didáctica para que el docente guíe a sus estudiantes desde la instalación de la aplicación en sus celulares, hasta enseñarles los conceptos de programación, utilizar conceptos de la materia a través de los objetos con los que el personaje interactúa, desarrollar el pensamiento computacional, comprender cómo construir un algoritmo y como un autómata puede interactuar dentro de un tablero con objetos.

En esta sección se muestran los resultados obtenidos en esta experiencia, en la que participaron los estudiantes de la Escuela de Educación Secundaria Nº9 acerca del uso de BlockAr en la clase de NTIC. Al atravesar un momento de pandemia y en una modalidad de burbujas fue un poco difícil que participaran todos los estudiantes ya que muchos de ellos faltaban a clase.

. Como se mencionó anteriormente en la tesina, en el año 2021, la escuela contaba con algunas máquinas pero no llegaba a cubrir la matrícula, si bien no asistieron la mayoría a clase por ser una ocasión inusual por la pandemia, entonces las netbooks alcanzaban, pero en el caso donde todos deben volver a la presencialidad ya no se cubría la capacidad total, por lo tanto se decide trabajar con celulares.

De los 10 estudiantes que participaron en la experiencia el 90 % contaban con celulares con el sistema operativo Android y el 10% con IOS.

Inicialmente 3 estudiantes tuvieron inconvenientes para instalar el juego, debido a que contaban con una versión de Android menor a 7 o. como en el caso de una alumna, que utilizaba IOS. En el caso de los celulares Android, se pudo compilar el juego a versiones anteriores de Android 7 y de esta manera poder instalarlo y participar de la clase. En el caso de IOs, no fue posible resolver el problema.

Las consignas del desafío 1 al 4 les resultaron bastante claras. Pero del desafío 5 al 7, si bien la mayoría comprendió la consigna, algunos no lo hicieron. Los resultados pueden verse en la figura 3.5.1.

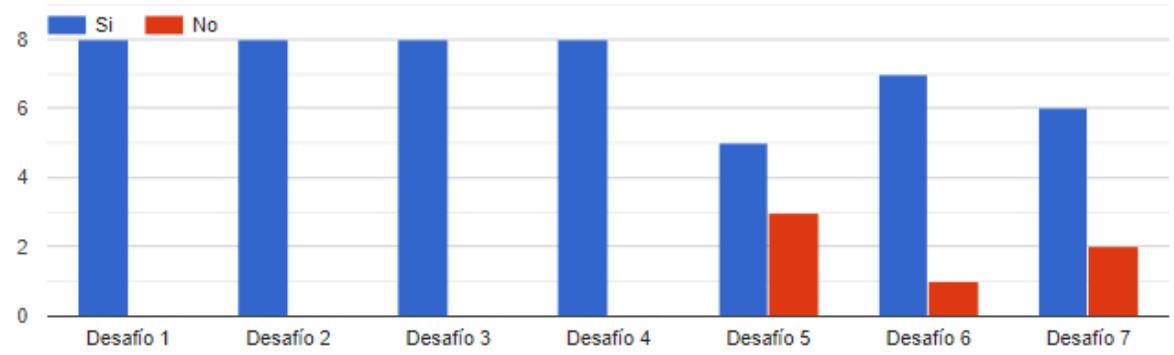


Figura 3.5.1. Estudiantes que comprendieron las consignas de los desafíos.

Luego de la clase se les preguntó a los estudiantes si consideraban necesaria las explicaciones del docente y cuán difícil les resultó completar cada desafío. Las figuras 3.5.2 y 3.5.3 muestran las respuestas a estas preguntas lo cual confirma que es necesario contar con un docente y una secuencia didáctica que sirva de soporte para que guíe a los estudiantes, sobre todo a aquellos que no lograron comprender el enunciado o no logran completar el desafío. Es decir, que las herramientas a utilizar para trabajar estas temáticas deben estar acompañadas de instrumentos que acompañen a los estudiantes.

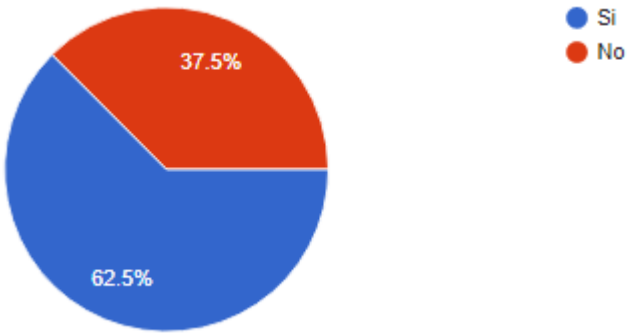


Figura 3.5.2. Porcentaje de estudiantes que consideran necesaria las explicaciones del docente.

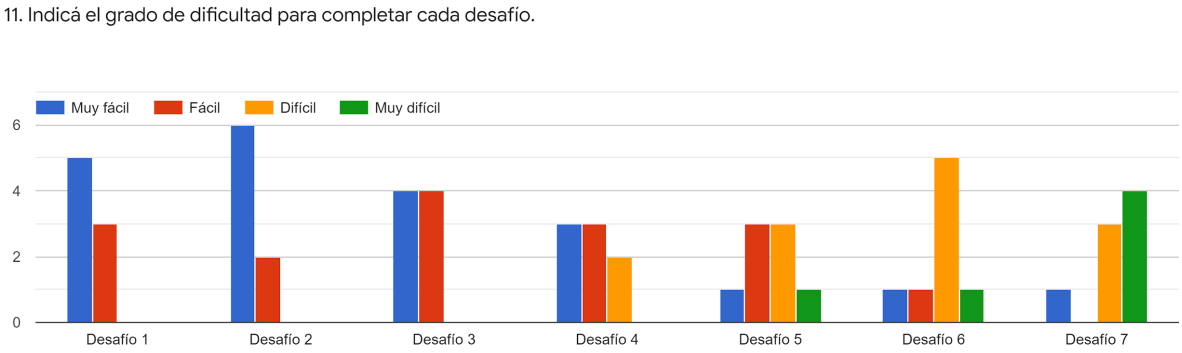
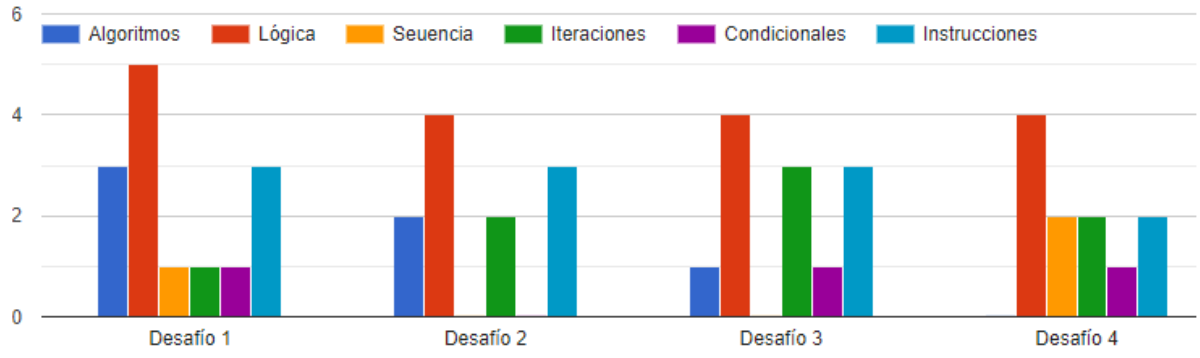


Figura 3.5.3. Indica el grado de dificultad para completar cada desafío.

En la encuesta realizada se pueden ver según el resultado que indica la figura 3.4.4. los conceptos trabajados. De la misma puede observarse en términos generales que adquirieron los términos siguientes:

- Algoritmos
- Lógica
- Secuencia
- Iteraciones
- Instrucciones

No se observa en en la figura 3.5.4 la mayoría que indiquen el término condicional, por lo tanto habría que profundizar un poco más en ese tema y agregar más desafíos con condicionales.



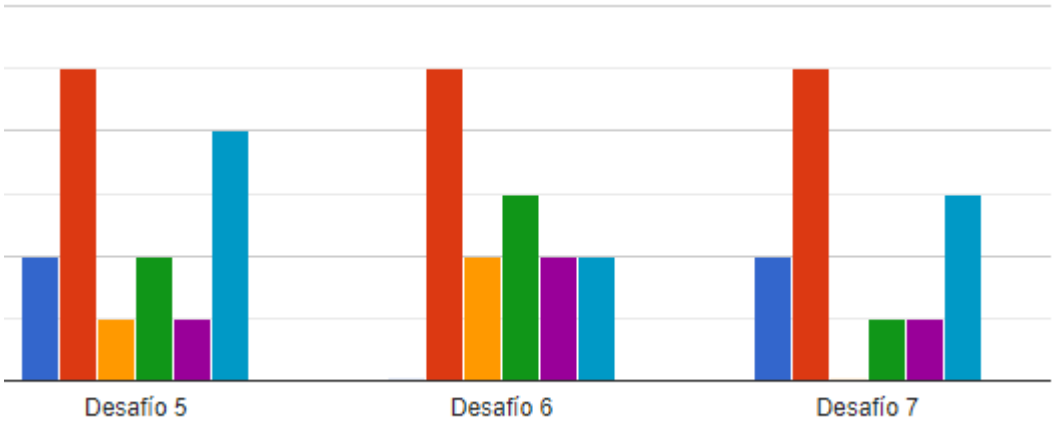


Figura 3.5.4. Indicador sobre los conceptos comprendidos en clase.

Si bien la clase fue realizada con pocos estudiantes, las actividades realizadas y los datos recabados permiten llegar a la conclusión que la experiencia fue exitosa. Es importante destacar que el 50 % de los estudiantes utilizaron la aplicación BlockAr en sus casas sin pedirles que lo hagan, al 75% les resultó entretenida y el 100% recomendarían BlockAr a otras personas.

El texto de la encuesta realizada se encuentra disponible en el Anexo 3.

Capítulo 4: Conclusiones y trabajos futuros

4.1 Conclusión

Como resultado de la investigación aplicando las técnicas de gamificación en la aplicación se logró atraer la atención de los estudiantes en los distintos desafíos propuestos en BlockAr, cuya interacción con el juego, favoreció la comprensión de los conceptos que se intentaron enseñar y el desarrollo de habilidades cognitivas para enfrentarse en la era digital en la que nos encontramos.

Es notable que la secuencia didáctica y el acompañamiento del docente fue necesaria para el aprendizaje de los estudiantes en los desafíos con mayor dificultad. También se requiere agregar más desafíos para que los estudiantes tengan un proceso más largo de aprendizaje y consoliden lo aprendido.

La mayoría de los estudiantes no contaron con dispositivos móviles de las generaciones más actuales, pero esto no fue un impedimento para instalar la app, por lo tanto, fue exitoso el desarrollo de una aplicación, que consume pocos recursos de hardware, ya que un gran porcentaje de estudiantes tuvieron la oportunidad de interactuar con la aplicación gracias a la utilización de los celulares que disponen.

4.2 Trabajos futuros

- Diseñar escenarios de aplicación que articulen los contenidos de programación con propuestas curriculares de otras áreas de conocimiento, que se trabajan en la escuela.
- Agregar otros desafíos para que los estudiantes y el docente puedan trabajar más contenidos de NTIC.
- Crear una funcionalidad nueva en la aplicación que permita crear escenarios para agregar otros desafíos y objetos. Por ejemplo: un constructor de escenarios.
- Darle la posibilidad al estudiantes de elegir otros personajes y que le puedan editar el nombre y/u otras características
- Crear un módulo para que las soluciones resueltas por los estudiantes se puedan integrar con Arduino.

Referencias

Alexander et al., 1977, A Pattern Language, Oxford University Press, px.

Aybar Rosales Vanessa del Carmen (2012). Aplicaciones complementarias a ROBOCODE que faciliten el aprendizaje de programación en escuelas secundarias. (Tesis de grado) Universidad Nacional de La Plata, La Plata.

Balanskat, A., & Engelhardt, K. (2015). Computing our future. Computer programming and coding Priorities, school curricula and initiatives across Europe. Retrieved from Brussels, Belgium: http://fcl. eun.org/documents/10180/14689/Computing+our+future_final.pdf/746e36b1-e1a6-4 bf1-8105- ea27c0d2bbe0

Borrás Gené, O. (2015). Fundamentos de gamificación.

Crawford, C. (1984). The art of computer game design.

Corballis, M. C. (2007). Pensamiento recursivo. Mente y cerebro, 27, 78-87.

http://amscimag.sigmaxi.org/4Lane/ForeignPDF/2007-05CorballisSpanish.pdf

Huizinga, J. H. (1949). Homo Ludens: A study of the play-element in culture. London: Routledge & Kegan Paul.

García-Peñalvo, F. J. (2016d). What Computational Thinking Is. Journal of Information Technology Research, 9(3), v-viii.

Csikszentmihalyi, M. (1998). Creatividad: el fluir y la psicología del descubrimiento y

la invención. Ed. Paidós.

Gee, J. P. (2004). *Lo que nos enseñan los videojuegos sobre el aprendizaje y el alfabetismo*. Málaga: Aljibe.

Gordon, W. J. (1961). Synectics: The development of creative capacity.

Green, C. S., & Bavelier, D. (2006). Effect of action video games on the spatial distribution of visuospatial attention. *Journal of experimental psychology: Human perception and performance*, *32*(6), 1465.

Gros, B. (2000). Del software educativo a educar con software. *Revista Quaderns Digital*, *24*, 440-482.

Keefe, J.W. (1979) Learning style: An overview. NASSP's Student learning styles: Diagnosing and proscribing programs (pp. 1-17). Reston, VA. National Association of Secondary School Principles..

Latorre, A. J. (2003). *Juego y educación: Aplicación de la construcción y uso de juegos educativos a los procesos de enseñanza y aprendizaje*. Madrid: Comunidad de Madrid, Dirección General de Promoción Educativa, 2003.

Llorens-Largo, F. (2015). Dicen por ahí. . . que la nueva alfabetización pasa por la programación. ReVisión, 8(2), 11-14.

Marcano Lárez, B. E. (2008). Juegos serios y entrenamiento en la sociedad digital.

Piaget, J., & Buey, F. J. F. (1983). *Psicología y pedagogía*. Barcelona, España: Ariel.

Resnick, M. (2008). Cultivando las semillas para una sociedad más creativa. *Revista Electrónica" Actualidades Investigativas en Educación"*, *8*(1), 0.

Seniquel, V., Bakun, M. P., & Gómez Kennedy, M. I. (2015). Gamificación: mecánicas y dinámicas de juego en el proceso de enseñanza-aprendizaje en la Universidad.

Sharpe, R. y Beetham, H. (2010) Comprender los usos de la tecnología por parte de los estudiantes para el aprendizaje: Hacia la apropiación creativa. En R. Sharpe, H. Beetham y S. de Freitas (Eds.) Repensar el aprendizaje para una era digital: cómo los estudiantes dan forma a sus experiencias , (págs. 85-99). Routledge Falmer, Londres y Nueva York. Obtenido de https://radar.brookes.ac.uk/radar/items/4887c90b-adc6-db4f-397f-ea61e53739e0/1/

Sierra-Rodríguez, J. L., & García-Peñalvo, F. J. (2015). Informática Educativa y Educación en Informática. Education in the Knowledge Society (EKS), 16(4), 25-31. doi: http://dx.doi.org/10.14201/eks20151642531

Stallman, R. (2004). Software libre para una sociedad libre. Madrid: Traficantes de Sueños, 2004.

Stewart, K.L., Felicetti, L.A. (1992). Learning styles of marketing majors. Educational Research Quarterly, 15(2), 15-23.

Toffler, A. (2001). Conmociones, oleadas y poder en la Era Digital. In A. Leer (Ed.), La visión de los líderes en la era digital (pp. 22-30). México: Prentice Hall

Unidad Académica Escuela Normal Superior N° 1 "Mary O. Graham". (s.f.). Especialización Docente de Nivel Superior en Didáctica de las Ciencias de la Computación. Recuperado el 26 de Diciembre de 2020 de https://uanormal1-bue.infd.edu.ar/sitio/especializacion-docente-de-nivel-superior-en-didactica-de-las-ciencias-de-la-computacion/

Universidad Pedagógica Nacional Pública y Federal. (s.f.). Profesorado en Informática. Recuperado el 26 de Diciembre de 2020 de https://unipe.edu.ar/formacion/grado/profesorado-en-informatica.

Vanessa Aybar Rosales, Claudia Queiruga, Claudia Banchoff Tzancoff, Isabel Miyuki Kimura y Matías Brown Bartneche (2017). Programming Competitions in High School Classrooms: RITA en RED. En Computer Conference (CLEI), 2017 XLIII Latin American. Córdoba, 4 al 8 de septiembre de 2017. Editorial: IEEE. ISBN: 978-1-5386-3057-0. Indexada: DBLP, IEEE Xplore. Soporte y/o medio de difusión de los proceedings: Digital.

Vélez, C. A. J. (2005). *La inteligencia lúdica: juegos y neuropedagogía en tiempos de transformación*. COOP. EDITORIAL MAGISTERIO.

Wing, J.M. (March 2006). Computational Thinking. It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use. COMMUNICATIONS OF THE ACM /Vol. 49, No. 3. https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf

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. doi: http://dx.doi.org/10.1098/rsta.2008.0118

Zapata-Ros, M. (2015). Pensamiento computacional: Una nueva alfabetización digital. *Revista de Educación a distancia*, (46)

Anexo 1: Configuración e instalación de BlockAr

Phaser 3

Se puede descargar desde el sitio oficial[19] o utilizando la consola.

Para realizar las pruebas con Phaser 3 es necesario ejecutarlo en un servidor web por razones de seguridad. Tiene que ver con el protocolo utilizado para acceder a los archivos, los navegadores se bloquean para que javascript no acceda a la ruta de archivos del sistema operativo. Para esto el juego necesitará cargar recursos: imágenes, archivos de audio, datos JSON. Por lo dicho anteriormente debe ejecutarse en http://, para más información puede leer el blog de Chromium[20].

La aplicación se desarrollará en Windows, por lo tanto, se eligió XAMPP[21] que contiene un servidor web HTTP Apache. Una vez descargado el proyecto de Phaser 3 se agrega en la carpeta htdocs, y allí se procede a crear la aplicación.

Blockly

Se puede descargar desde GitHub[22] o simplemente a través de la consola de comandos.:

Blockly se va a integrar con phaser 3 para trabajar en conjunto.

Apache Cordova

Para instalar apache cordova es necesario tener instalado Node.js[23], luego se debe escribir en consola, utilizando el símbolo del sistema (cmd en caso de correrlo en windows), lo siguiente:

```
C:\>npm install -g cordova
```

Ahora, ya se puede ejecutar el comando cordova en el símbolo del sistema. A continuación, se procede a crear la estructura de directorio requerida para la aplicación cordova. Para crear la aplicación ejecutamos el comando

```
cordova create path_de_myapp com.app_game.myapp nombre_de_myapp
```

[19] Sitio oficial: https://phaser.io/download/stable
[20] https://blog.chromium.org/2008/12/security-in-depth-local-web-pages.html
[21] https://www.apachefriends.org/es/index.html
[22] https://github.com/google/blockly
[23] https://nodejs.org/es/

Una vez creada la estructura básica de la aplicación, añado las plataformas dentro del proyecto path_de_myapp, en las cuales voy a correr la aplicación, vamos a seleccionar para android y para ios:

```
cd path_de_myapp
cordova platform add ios
cordova platform add android
```

La **estructura** que se crea es de la siguiente manera:

```
path_de_myapp/
|-- config.xml
|-- www/
|-- platforms/
| |-- android/
| |-- ios/
```

config.xml

Es un archivo de configuración global en el cual se pueden controlar muchos aspectos del comportamiento de una aplicación. Está basado en la especificación de W3C.

Ejemplo de configuración luego de ejecutar el comando anterior:

```
<?xml version='1.0' encoding='utf-8'?>
<widget id="com.app_game.myapp" version="1.0.0" xmlns="http://www.w3.org/ns/widgets" xmlns:cdv="http://cordova.apache.org/ns/1.0">
    <name>nombre_de_myapp</name>
    <description>
        A sample Apache Cordova application that responds to the deviceready event.
    </description>
    <author email="dev@cordova.apache.org" href="http://cordova.io">
        Apache Cordova Team
    </author>
    <content src="index.html" />
    <access origin="*" />
    <allow-intent href="http://*/*" />
    <allow-intent href="https://*/*" />
    <allow-intent href="tel:*" />
    <allow-intent href="sms:*" />
    <allow-intent href="mailto:*" />
    <allow-intent href="geo:*" />
    <platform name="android">
        <allow-intent href="market:*" />
    </platform>
    <platform name="ios">
        <allow-intent href="itms:*" />
        <allow-intent href="itms-apps:*" />
    </platform>
</widget>
```

Los siguientes elementos de configuración aparecen en el archivo config.xml de primer nivel y se admiten todas las plataformas soportadas Cordova:

- El atributo id del elemento <widget> proporciona el identificador del dominio de la aplicación, y la versión, el número de versión.
  La etiqueta widget también puede tener atributos que especifican las versiones alternativas.
- El elemento <name> especifica el nombre formal de la aplicación, como aparece en la pantalla principal del dispositivo y dentro de la tienda de aplicaciones.
- Los elementos <description> y <author> especifican metadatos e información de contacto que puede aparecer en anuncios de la tienda de aplicaciones.
- El elemento <content> define la página de inicio de la aplicación en el directorio web de alto nivel de activos. El valor predeterminado es index.html, que habitualmente aparece en el directorio de nivel superior www de un proyecto.
- Los elementos <access> definen el conjunto de dominios externos que puede comunicarse con la aplicación. El valor predeterminado que se muestra arriba le permite acceder a cualquier servidor.

Para que la aplicación inicie de manera vertical y quede fija la orientación vamos a agregar en el archivo lo siguiente.

```
<preference name="Orientation" value="landscape" />
```

Los valores posibles son default , landscape o portrait.

www/

Contiene los artefactos web del proyecto, como archivos .html, .css y .js. La mayoría de su código se copia en este directorio.

```
cordova prepare
```

Ahora cuando se ejecuta el comando anterior se reproduce un directorio www dentro del subdirectorio de cada plataforma, apareciendo, por ejemplo, en **platforms/ios/www** o **platforms/android/assets/www**. Debido a que la CLI copia constantemente los archivos de la carpeta www de origen, solo debe editar estos archivos y no los que se encuentran en los subdirectorios de las plataformas.

Esto ocurre debido a que anteriormente se ejecutó:

```
cordova platform add ios
cordova platform add android
```

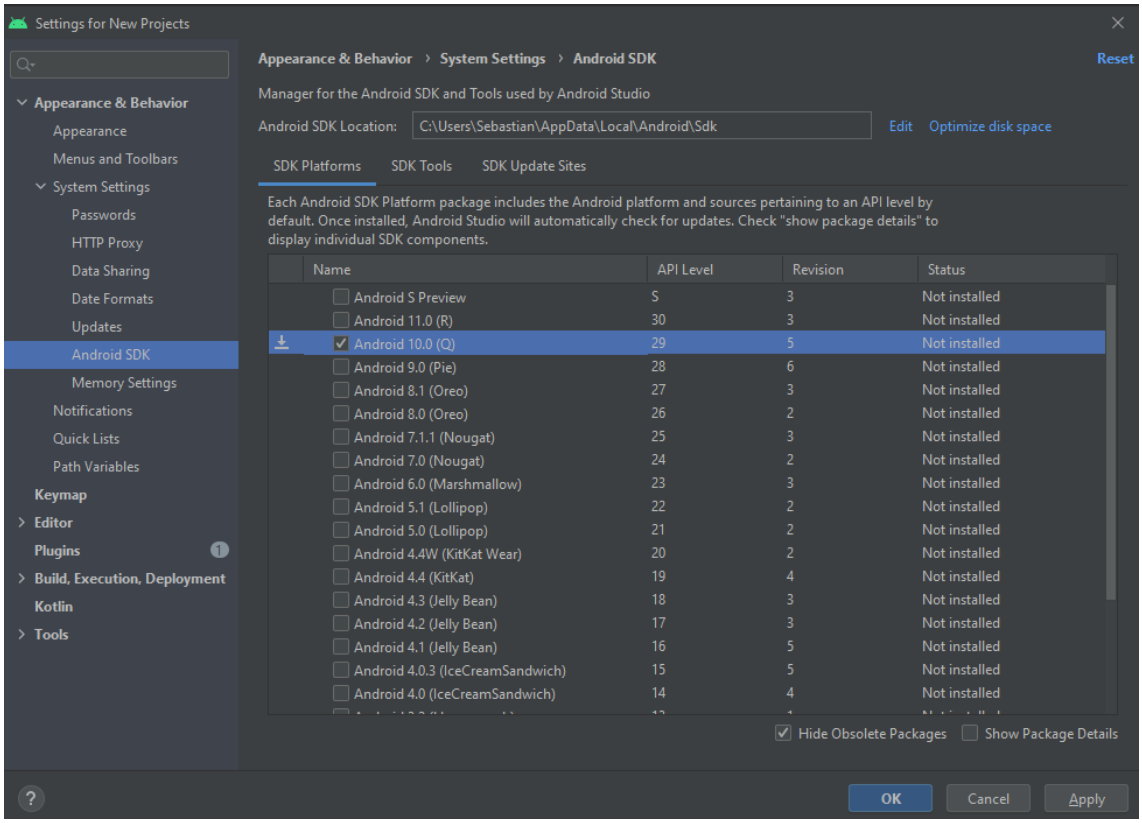
Preparación de plataforma para android

Requerimientos

- Instalar Java Development Kit (JDK) 8
- Instalar Gradle
- Instalar SDK de Android (Android Studio)

Después de instalar el SDK de Android, también debe instalar los paquetes para cualquier nivel de API que desee apuntar. Se recomienda que instale la versión de SDK más alta que admita su versión de cordova-android. Esta información se puede consultar desde el sitio web de cordova.

Al abrir Android Studio versión 4.2 se debe hacer click en Configure y luego en SDK Manager vamos a seleccionar la plataforma cuyo nivel de api es 29 y la vamos a instalar aplicando los cambios.



Establecer variables del sistema

- Establecer JAVA_HOME apuntando al directorio \jdk del JDK de java

- Establecer ANDROID_HOME apuntando al directorio de Android\Sdk de Android Studio
- Establecer ANDROID_SDK_ROOT apuntando al directorio de Android\Sdk de Android Studio
- Agregar a Path el directorio \bin de gradle
- Agregar a Path el directorio Android\Sdk\tools de Android Studio
- Agregar a Path el directorio Android\Sdk\platform-tools de Android Studio
- Agregar a Path el directorio Android\Sdk\tools\bin de Android Studio

Nota: el directorio Android\Sdk suele encontrarse en el directorio de usuario AppData\Local\ por ejemplo C:\Users\Sebastian\AppData\Local\Android\Sdk\

Construir la aplicación

El script cordova create, crea de manera predeterminada una estructura basada en web cuya página de inicio es www/index.html, este index.html va a contener el archivo de inicialización de la aplicación.

Una vez que se haya construido la aplicación, se procede a compilarla ejecutando el comando cordova build. Esto creará un archivo apk que luego se puede instalar en el celular.

Si se quiere compilar una plataforma, por ejemplo android, se debe ejecutar:

```
cordova build android
```

Probar la aplicación

Los SDK para plataformas móviles a menudo vienen con emuladores que ejecutan una imagen del dispositivo, de modo que puede iniciar la aplicación desde la pantalla de inicio y ver cómo interactúa con muchas funciones de la plataforma. Entonces en primer lugar se debe abrir Android Studio y luego abrir el AVD Manager para emular un dispositivo Android. Una vez que se está ejecutando el dispositivo android, a través de una terminal ejecutar un comando como el siguiente para reconstruir la aplicación y verla dentro del emulador de una plataforma específica:

```
cordova emulate android
```

Instalación en Windows 10

En el sistema operativo windows 10 al ejecutar el comando

```
cordova build windows10
```

Se genera un archivo .appx que es un instalador para windows 10.

En la siguiente figura se puede ver la aplicación corriendo en windows 10.

Instalación en Android

El siguiente comando genera un archivo instalador .apk que se ejecuta en los dispositivos móviles android.

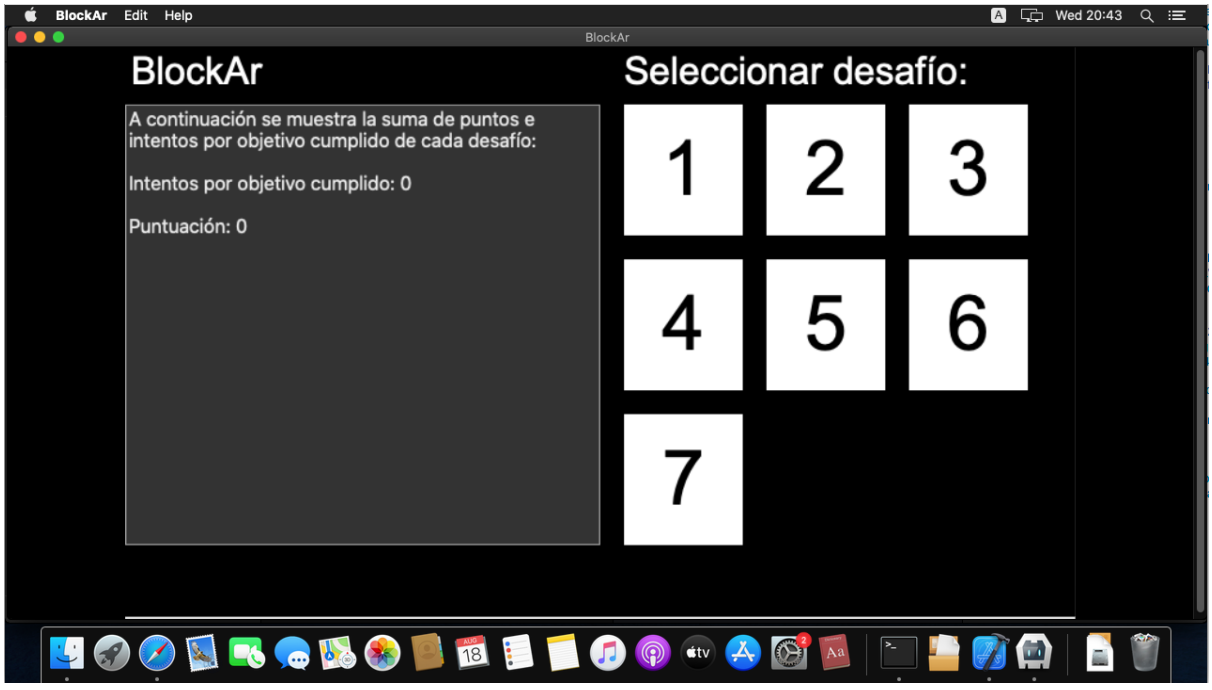
```
cordova build android
```

Para saber donde se guarda el ejecutable se puede acceder al sitio de Apache Cordova que indica según la versión que tengas instalada donde se ubica el archivo .apk.

En la siguiente figura se observa BlockAr corriendo en un emulador con android 9 a través de la aplicación de Android Studio.

Instalación en Os X

A continuación se muestra BlockAr corriendo sobre el sistema operativo Catalina de Apple



Anexo 2: Secuencia didáctica

Introducción

Las siguientes clases están ordenadas con números para darle un sentido a la secuencia didáctica para llevar a cabo el aprendizaje de programar a través de BlockAr. Cada docente la puede integrar según su cronograma y/o diseño curricular.

Clase 1 (2hs)

Objetivos

- Comprender el significado de programar.
- Comprender el significado de instrucciones en un programa.
- Comprender entre instrucción simple y secuencia de instrucciones.
- Analizar el resultado de las instrucciones.
- Introducir al pensamiento computacional.
- Utilizar términos tecnológicos.

Inicio (30 min)

En ésta primera clase el docente deberá explicar los conceptos de programación, instrucciones, secuencia de instrucciones y pensamiento computacional, por lo tanto se puede exponer a través de un video, presentaciones o como el docente considere necesario, los contenidos correspondientes para llevar la clase adelante.

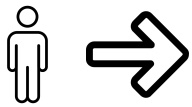
Desarrollo

Actividad (30 min)

El docente deberá dibujar un tablero de 3 x 3 celdas en el pizarrón del tamaño un poco mayor a una hoja de cuaderno (23 cm x 19 cm), una bandera dentro de una celda, en una hoja dibujar un personaje asignándole un nombre para ubicarlo en una celda de inicio. Más adelante se muestra como puede ser el ejemplo que estoy describiendo.

El docente indicará que existen algunas instrucciones (avanzar, girar izquierda 90º, girar derecha 90º). Para que se comprenda bien, es necesario explicar con algún ejemplo lo que hace cada instrucción y como se refleja dentro del tablero la ejecución de una secuencia de instrucciones para el personaje creado.

Suponiendo que el personaje es la flecha y arranca en la dirección que está la flecha dentro de la tabla (tablero).

Necesitamos crear una pila donde se acumulan las instrucciones (acciones).

Si el personaje que está en el tablero inicia donde está la flecha en sentido derecha, considerando las instrucciones que mencionamos anteriormente (avanzar, girar izquierda 90º, girar derecha 90º). La instrucción “avanzar” avanza de a un casillero. ¿Qué instrucciones pueden llevar al personaje a la bandera?

Los estudiantes deberán indicarle al docente que secuencia de instrucciones se deben agregar en una pila para que se ejecuten y logren cumplir el objetivo de llevar al personaje a la posición de la bandera.

Entonces el docente puede guiar a los estudiantes para crear la pila como se muestra a continuación.

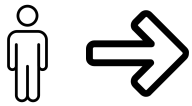
Puede haber más de un camino para llegar a la bandera.

	avanzar
	avanzar
	girar izquierda 90º
	avanzar
	avanzar

Ahora mostraré por etapas como va cambiando el tablero a medida que avanza según la secuencia escrita arriba. A medida que el personaje avanza irá dejando con flechas el camino que fue haciendo según la dirección en la que fue recorriendo los casilleros.

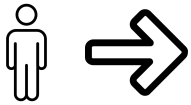
El personaje se encontraba en dirección derecha, por lo tanto se mueve a la derecha y se mantiene en esa dirección hasta que reciba una orden distinta a “avanzar”.

En la línea 1 dice avanzar, avanza un casillero a la derecha.

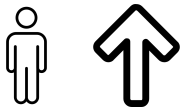
		
		

La línea 2 dice avanzar, entonces el personaje avanza un casillero a la derecha.

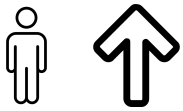
		

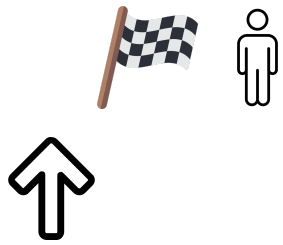
La línea 3 dice girar izquierda 90º, entonces gira 90 grados a la izquierda por lo tanto en el tablero que se muestra a continuación con una flecha se indicará el sentido del personaje que va a pasar a tener.

La línea 4 dice avanzar, pero ahora como le dijimos que gire 90 grados entonces va a avanzar hacia arriba.

La línea 5 dice avanzar, entonces va a avanzar hacia arriba.

Una pregunta para los estudiantes ¿Qué sucede si el personaje recibe cuatro veces avanzar desde el comienzo y en sentido derecha?

Actividad (30 min)

Al terminar la actividad anterior los estudiantes se deberán organizar en grupos de dos, el docente a cada grupo le ofrecerá un dibujo en papel que represente un objeto como la bandera del ejemplo anterior y un personaje de papel. Un integrante deberá dibujar una tabla en una hoja y tendrá que colocar el objeto en una celda de la tabla y al personaje en otra celda alejado del objeto, por último, el compañero deberá crear las instrucciones para que el personaje llegue al objeto.

Cierre (30 min)

Manteniendo el mismo grupo deberán indicar los pasos aproximados para salir del aula desde el punto donde se encuentra ubicado y sentado, de modo que puedan descomponer esa acción en varias subtareas. Por ejemplo: pararse, caminar hasta la puerta, girar picaporte, abrir puerta, salir.

Para terminar pueden pensar en otras actividades sencillas y ver cómo la pueden descomponer a través de instrucciones. Por ejemplo:

Actividad: hablar por teléfono.

Subtareas: Acá debe indicar cómo se descompone la actividad. “agarrar el teléfono”, “encender telefono”, “introducir clave de desbloqueo”, “presionar botón de llamada”, “marcar número de teléfono”, “presionar el botón para iniciar la llamada”.

Clase 2 (2 hs)

Objetivos

- Repasar el significado de programar.
- Repasar el significado de instrucciones en un programa.
- Repasar las diferencias entre instrucción simple y secuencia de instrucciones.
- Analizar el resultado de las instrucciones.
- Descargar BlockAr.
- Instalar BlockAr en el celular con un sistema Android o en una pc con windows 10.
- Comprender la interfaz de BlockAr.
- Utilizar términos tecnológicos.

Inicio (1 hora 15 min)

El docente explicará cómo pueden adquirir BlockAr y como instalarlo. El docente puede compartir la app a través de su celular por diferentes vías (Bluetooth, Whatsapp, NFC, etc), en caso de ser una aplicación para instalar en una netbook de conectar igualdad o alguna computadora de la escuela que posea Windows 10, podrá compartir un ejecutable a través de un pendrive. Luego cada estudiante deberá instalar la aplicación en el dispositivo que tenga a su alcance.

Se dará una introducción para comprender cómo interactuar con BlockAr.

En la figura siguiente se muestra la interfaz de la pantalla principal de BlockAr, en la parte izquierda se indican los puntos totales que va acumulando por cada desafío que completa, en la parte derecha se indican los desafíos que tiene para seleccionar.

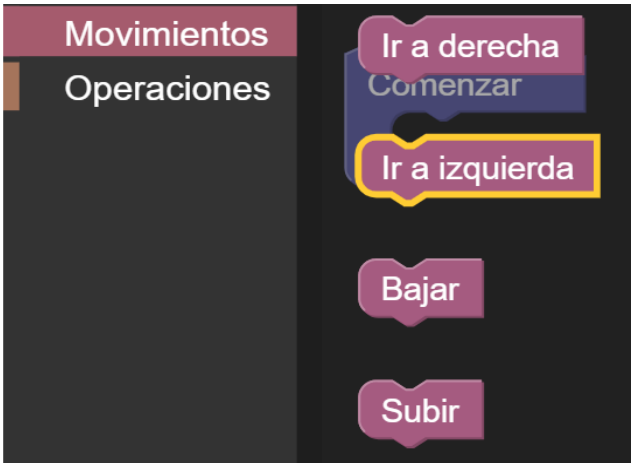
En esta clase los estudiantes deberán comenzar con el desafío Nº1.

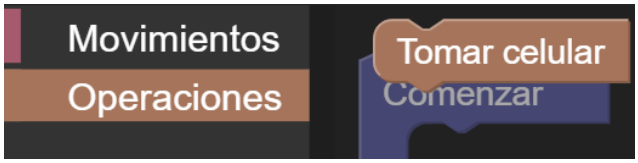
A continuación se muestra la pantalla del desafío Nº1 con una descripción que indica donde se encuentra el texto del desafío que debe cumplir y el puntaje que acumulará, a la derecha se puede ver un botón “Comenzar” que al hacer clic dará inicio a programar.

Luego de hacer clic en comenzar se muestra la siguiente imágen:

En el área de bloques deberá introducir el programa (la secuencia de instrucciones o la secuencia de bloques en este caso) que generen las acciones del personaje, dichas acciones se dividen en categorías como se muestra en las dos imágenes siguientes:

- Movimientos: “Ir a derecha”, “Ir a izquierda”, “Bajar”, “Subir”.
- Operaciones: “Tomar celular”.

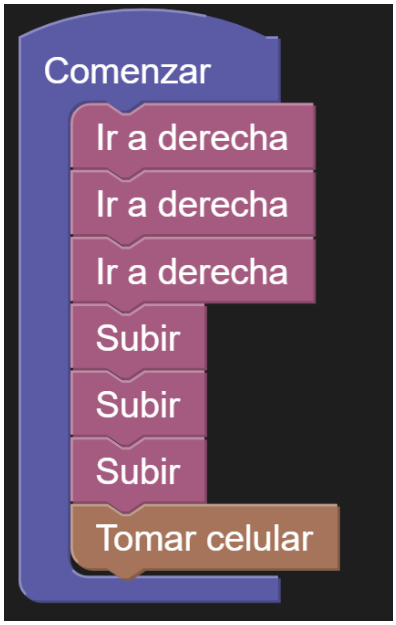




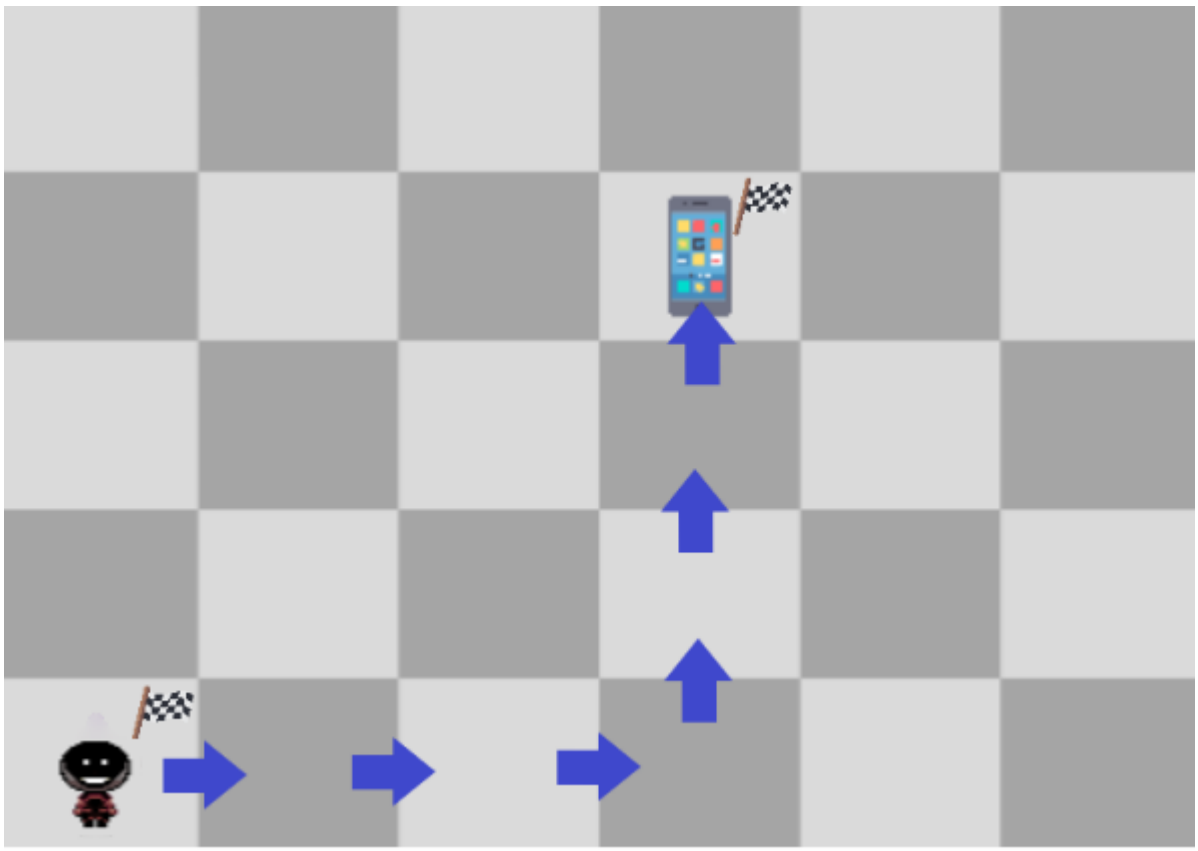
Las Operaciones pueden ir cambiando en diferentes desafíos.

Dentro del bloque comenzar se debe agregar la secuencia de bloques que le ordenan al personaje a realizar las acciones.

Por ejemplo, si quiere indicarle al personaje que avance 3 casilleros a la derecha, suba 3 casilleros del tablero y por último tome el celular entonces se debe hacer lo siguiente:



En la siguiente imágen se indica con flecha de color azul los pasos que hace el personaje en el tablero y las banderas indican donde comienza el personaje y donde termina.



Éste camino representa la secuencia de bloques que están en el bloque “Comenzar”, pero hay más de un camino para llegar al celular.

En el desafío 1 el objeto “celular” aparecerá ubicado en el tablero de forma aleatoria, por lo tanto, la secuencia de instrucciones no será siempre la misma para llevar al personaje al celular.

A continuación se detallan los botones del juego:

 Muestra el área de desafío y de puntuación.

 Muestra el área de bloques.

 Disminuir o aumentar la velocidad del personaje.

 Ejecuta el juego según la secuencia de bloques que se encuentren dentro del bloque “Comenzar”.

 Resetear juego. Ubica al personaje en la posición de comienzo, restablece las variables a su valor inicial.

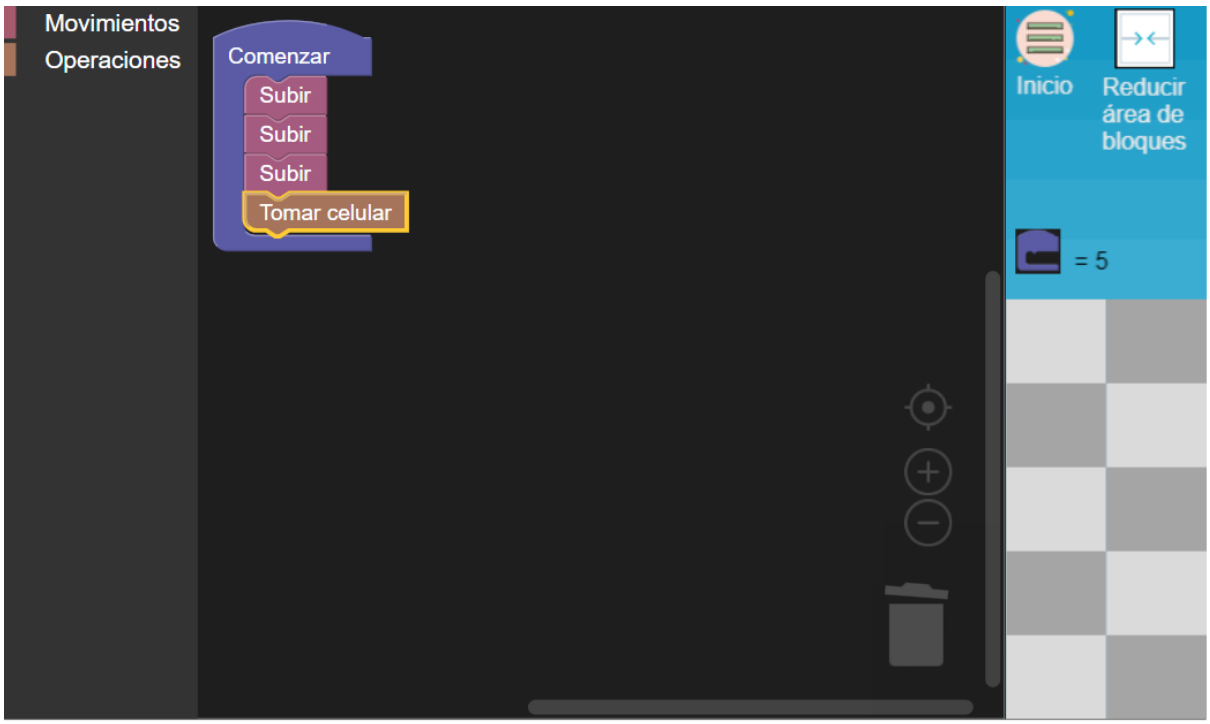
 Menú. Al hacer clic se vuelve a la pantalla de inicio.

 Alarga el área de bloques para mayor comodidad al momento de insertar bloques.

 Reduce el área de bloques a su tamaño de inicio.

 Objeto celular que aparece en el tablero. Según el desafío pueden aparecer objetos diferentes y el personaje puede interactuar con ellos. Los objetos serán elementos tecnológicos.

Luego de hacer clic en el botón “Ampliar Área de bloques” sucede lo siguiente:

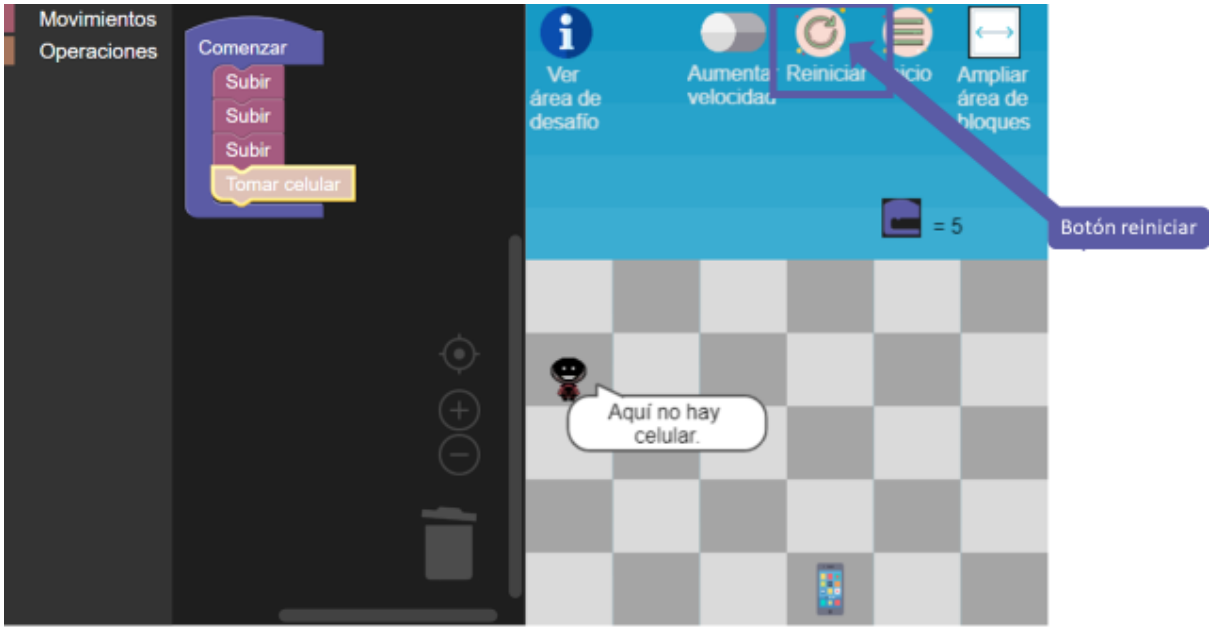


En el caso de querer volver al tamaño original se puede hacer clic en el botón “Reducir área de bloques”.

Si el estudiante logra cumplir el objetivo se mostrará el puntaje obtenido y luego podrá ir al menú para seleccionar el siguiente desafío.

En color verde se mostrarán los desafíos terminados.

Si el estudiante no logra cumplir el objetivo puede reiniciar la partida



Al finalizar la explicación se deja un tiempo para responder consultas.

Desarrollo

Actividad (20 min)

Una vez que cada estudiante descargó la aplicación, la instaló y conoce la interfaz, deberá realizar el desafío 1 que consiste en agregar los bloques correspondientes dentro del bloque “Comenzar” para que el personaje se mueva hasta el celular con los bloques que se encuentran en la categoría “Movimientos” y luego utilice el bloque “Tomar celular” que se encuentra en la categoría “Operaciones”.

Objetivo del desafío 1: El personaje debe ir a buscar el celular para llamar a un técnico para que arregle su pc. Para lograrlo deberá recorrer los casilleros que sean necesarios para llegar al celular y tomarlo. No se puede salir del tablero.

Cierre (25 min)

Responder las siguientes preguntas:
¿Qué ocurre si el personaje intenta salir del tablero?
¿Qué ocurre si el personaje intenta tomar un celular donde no hay?
¿Qué ocurre con el contador de bloques a medida que se agregan bloques?

Para reflexionar:

En el caso que el celular aparezca en un casillero a la derecha de la posición donde inicia el personaje y en el bloque “Comenzar” agrego lo siguiente: “Mover derecha”, “Tomar celular”, “Mover derecha”. ¿Se ejecuta el último bloque?¿Por qué?

Clase 3 (2 hs)

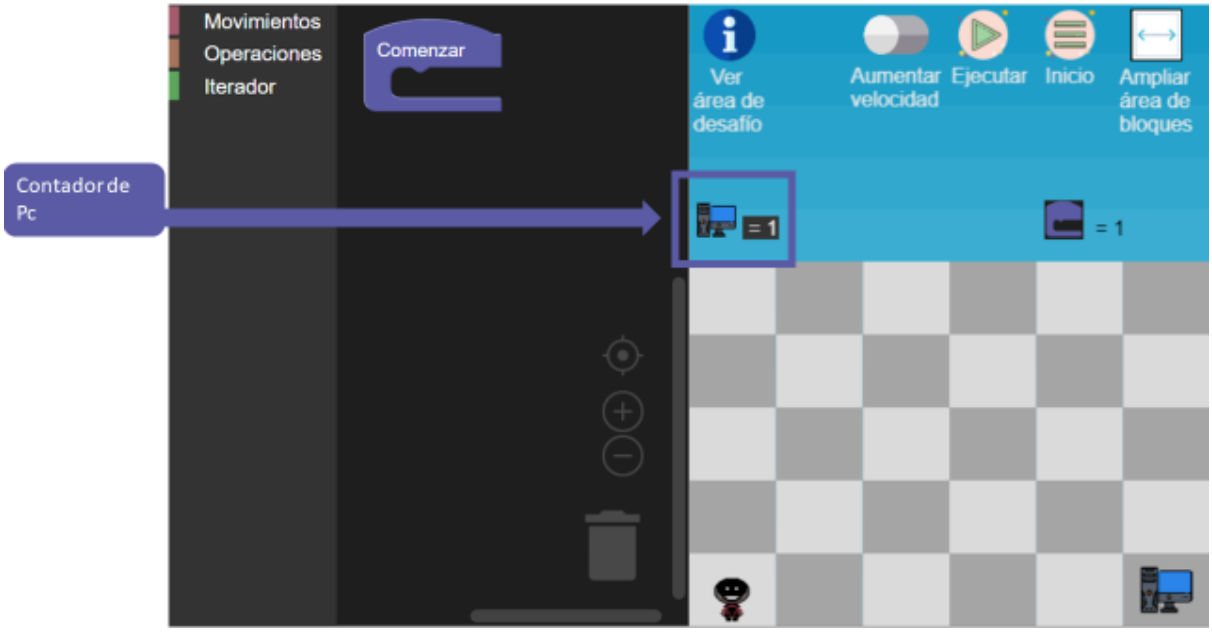
Objetivos

- Comprender la diferencia entre instrucciones simples e iteradores.

- Identificar que los bloques que se encuentran dentro del bloque “repetir” se pueden repetir varias veces.
- Comprender que las computadoras utilizan este tipo de iteradores para ejecutar instrucciones.
- Comprender a través de pseudocódigo cómo funciona el iterador.
- Utilizar términos tecnológicos.

Inicio (30 min)

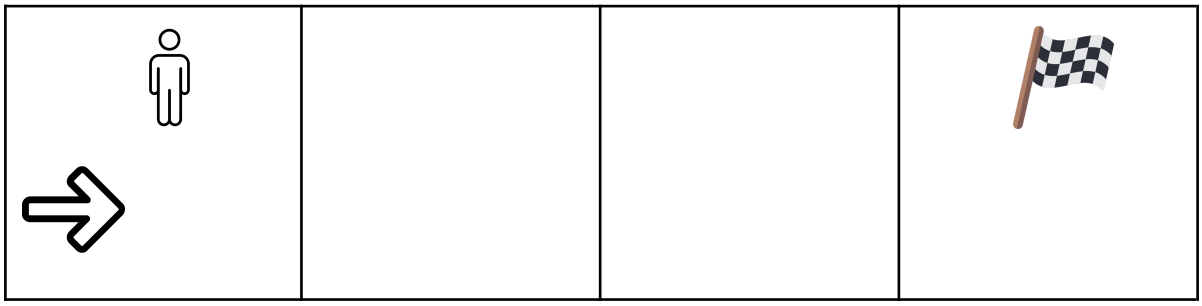
Hacer un repaso de la clase anterior indicando que existían solo dos categorías, y que para avanzar más de dos casilleros había que agregar varias veces el mismo bloque. En cambio, ahora, en el desafío 2, se agrega una categoría Iterador que incluye un bloque repetir que permite repetir varias veces los bloques que se encuentran dentro de él.



 Objeto Pc.

En la pantalla se pueden ver dos objetos pc iguales, uno dentro del tablero y otro fuera del tablero que indica la cantidad de objetos pc que hay (Contador de pc). A medida que toma un objeto pc, éste irá decrementando su cantidad.

Es necesario que el docente escriba en el pizarrón un ejemplo sencillo, por ejemplo, dibujar cuatro celdas continuas en horizontal, tener un personaje de papel y dibujar un objeto en la última celda.



Ahora el docente le comenta a los estudiantes, si se tiene un bloque avanzar, que avanza de a un casillero, ¿Cuántos bloques voy a necesitar para llegar a la bandera?. La respuesta correcta es que va a necesitar tres bloques “avanzar”.

Entonces luego de que los estudiantes respondan y se termine de entender el tema, se procede a explicar lo que ocurre si agregamos un bloque repetir que puede contener muchos bloques. para ésto, utilizaremos pseudocódigo (una forma de representar los distintos pasos que va a realizar un programa), como se indica a continuación.

```
repetir ( 4 ) veces
{
“avanzar”
}
```

Las llaves representan lo que va dentro del bloque repetir.

Pregunta para los estudiantes ¿Que creen que hace el pseudocódigo que se muestra?. Si no logran descifrarlo, se les debe explicar cómo funciona.

Ahora se quiere agregar una instrucción “tomar bandera” por lo que el docente debe escribir en el pizarrón los casos que se muestran a continuación, con una serie de preguntas para guiar a los estudiantes a entender lo que sucede en cada caso.

¿Qué sucede si se pone dentro del bloque, antes de “avanzar”?

repetir (4) veces {
“tomar bandera”
“avanzar”
}

¿Qué sucede si se agrega después de “avanzar”?

repetir (4) veces {
“avanzar”
“tomar bandera”
}

¿Qué sucede si se agrega antes por fuera del bloque?

“tomar bandera”
repetir (4) veces {
“avanzar”
}

¿Qué sucede si se agrega después, por fuera del bloque?

repetir (4) veces {
“avanzar”
}
“tomar bandera”

Concluyendo estas explicaciones e intervenciones con los estudiantes se puede continuar con el desarrollo de la actividad.

Desarrollo

Actividad (30 min)

Objetivo del desafío 2: El personaje se encuentra en su tienda y debe tomar la pc que puede ver para encenderla. Para ir a la pc puede usar el iterador. No puede usar más de cinco bloques. El iterador cuenta como dos bloques.

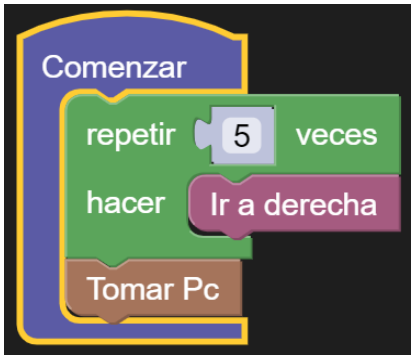
El contador de bloques indicará la cantidad que se están usando para evitar problemas.

El contador de pc indica la cantidad de pc que hay en el tablero.

Este desafío es trivial para que los estudiantes puedan ver cómo funciona el bloque repetir.

Podrían probar lo que sucede al colocar “Tomar Pc” antes del bloque repetir o dentro del bloque repetir antes y después del bloque “Ir a derecha” como se planteó al inicio de ésta clase con los ejemplos de pseudocódigo.

Solución para el desafío 2:



Actividad (30 min)

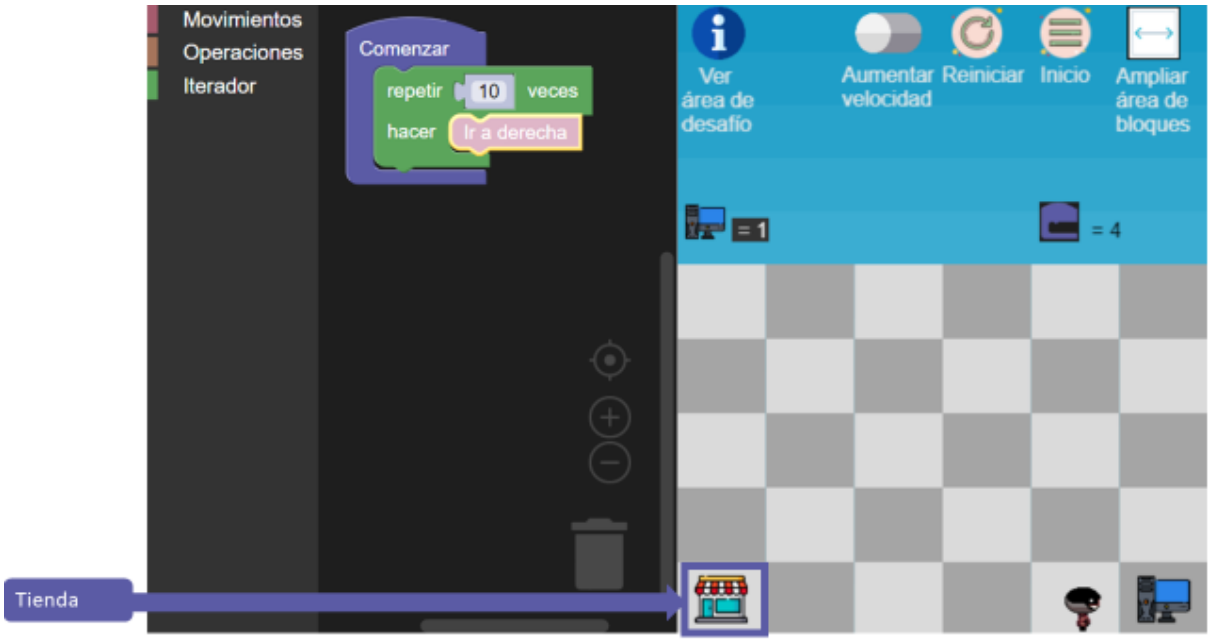
Una vez que resolvieron la actividad anterior se continuará con el desafío 3 que le suma un poco de complejidad, la cual consiste en ir hasta la pc, tomarla “Tomar Pc”, llevarla a la tienda, y colocarla con la operación “Colocar pc en la tienda”.

Objetivo del desafío 3: El personaje recibió una llamada del técnico para que retire la pc que dejó a reparar. Debe tomar la pc y llevarla a la tienda. No puede usar más de nueve bloques. El iterador cuenta como dos bloques.

El contador de bloques indicará la cantidad que se están usando para evitar problemas.

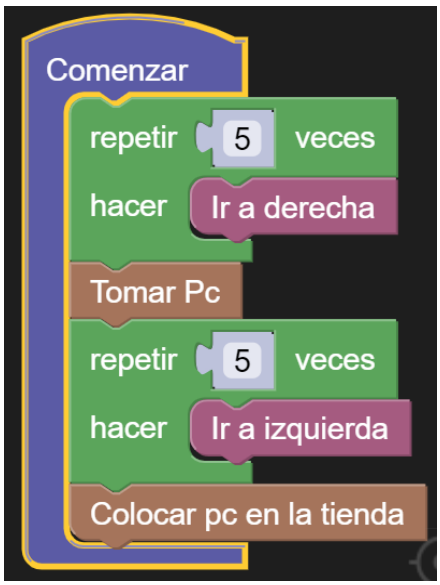
El contador de pc indica la cantidad de pc que hay en el tablero.

En ésta instancia de juego deberán utilizar dos Iteradores, ya que se le restringe la cantidad de bloques que pueden utilizar para que enfoquen sus energías en los iteradores.



 Objeto tienda.

Solución para el desafío 3:



Cierre (30 min)

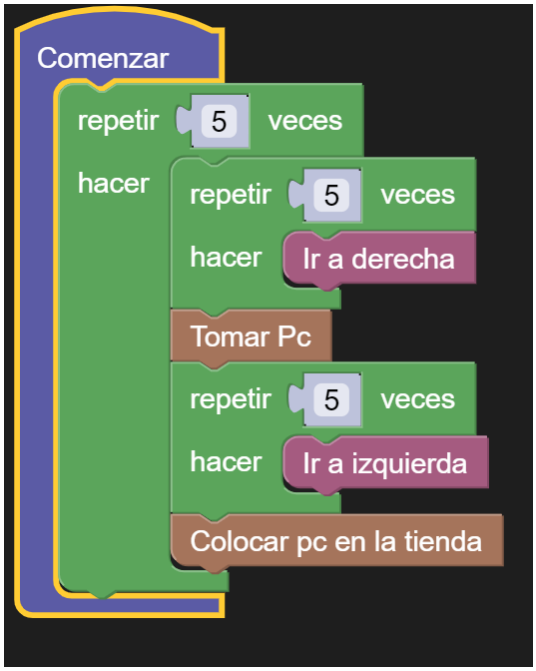
Para cerrar ésta clase y fortalecer el uso del bloque repetir se irá al desafío 4 que le suma mayor complejidad que el desafío anterior.

Objetivo del desafío 4: El personaje compró cinco pc para su tienda. Debe ir a buscar de a una, tomarla y llevarla a la tienda con el fin de armar un ciber. No puede usar más de once bloques. Los iteradores cuentan como dos bloques.

El contador de bloques indicará la cantidad que se están usando para evitar problemas.

El contador de pc indica la cantidad de pc que hay en el tablero.

Solución para el desafío 4:



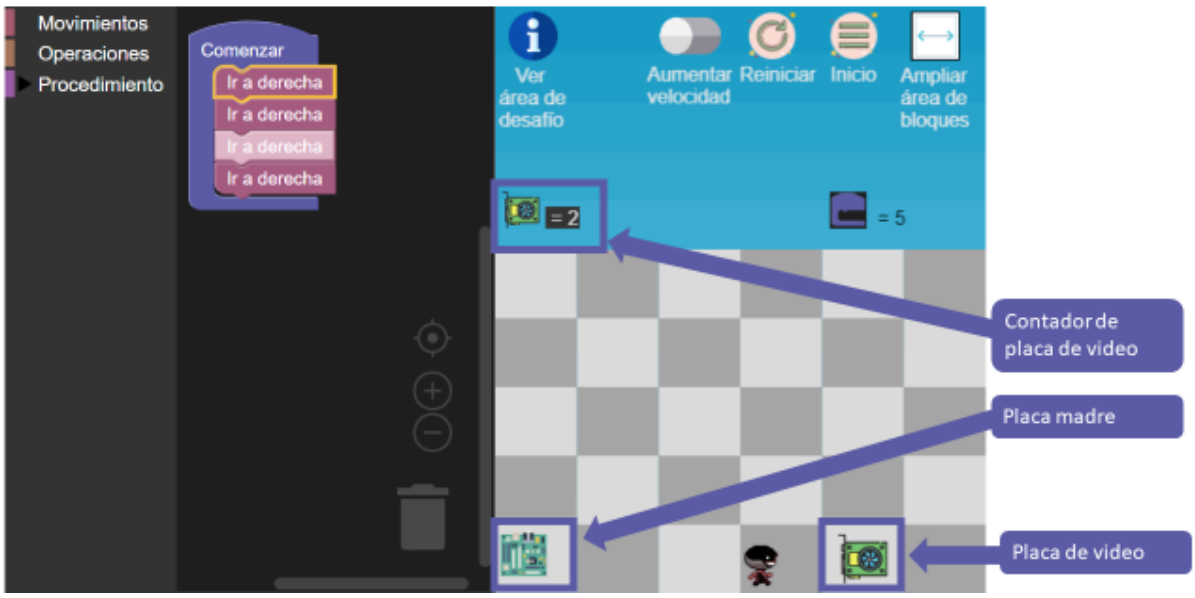
Clase 4 (2 hs)

Objetivos

- Comprender el uso de procedimientos.
- Diferenciar entre procedimientos e instrucciones simples.
- Identificar nombres cortos y claros para los procedimientos.
- Utilizar términos tecnológicos.

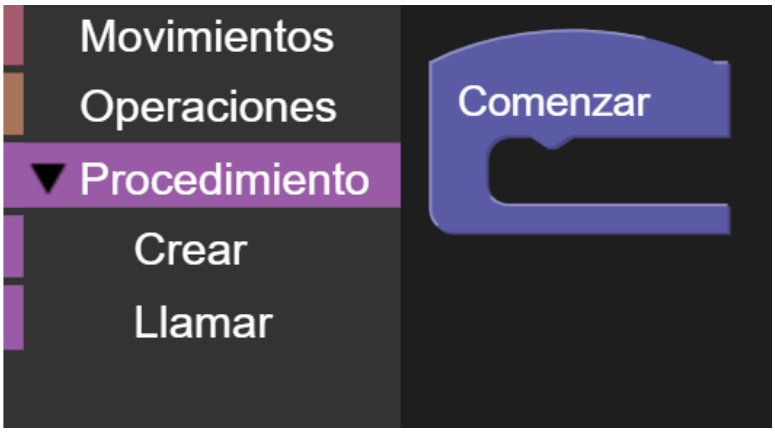
Inicio (1 hora)

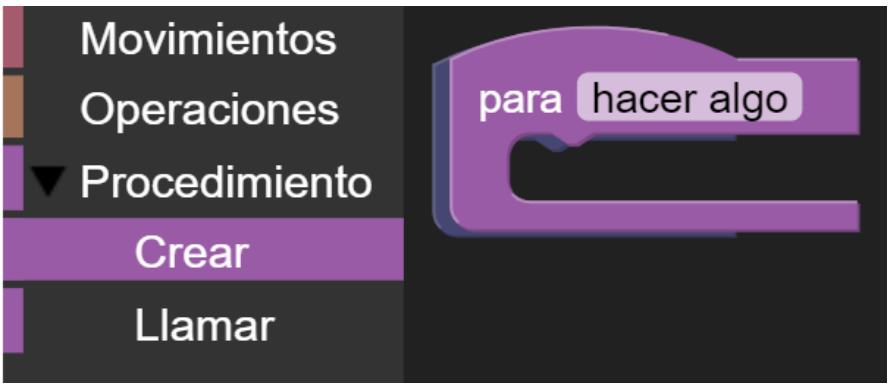
En la clase anterior se introdujo el concepto de iterador, pero ahora para introducir el concepto de procedimientos en el desafío 5 se quitará la categoría Iterador para que se comprenda el rol del procedimiento en un programa.



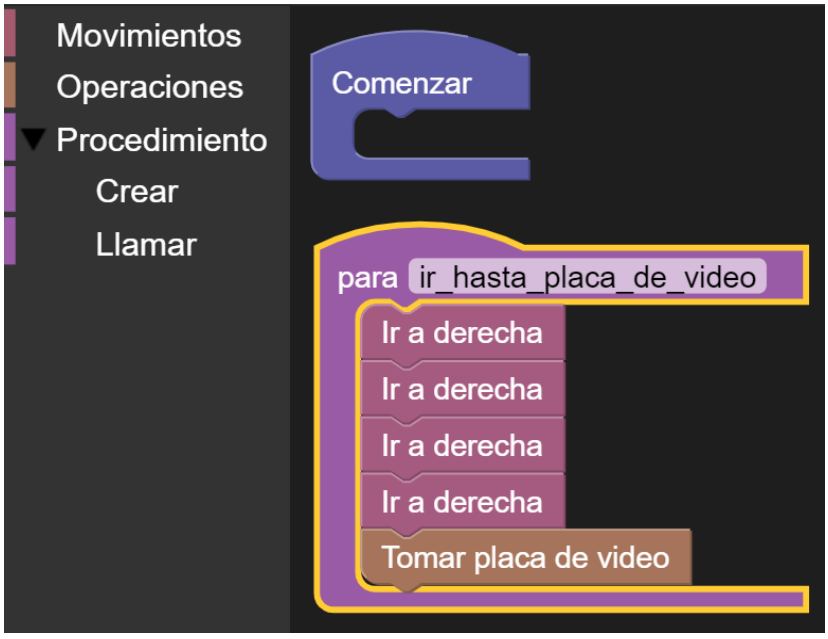
Es necesario que el estudiante entienda que el procedimiento es un subprograma que puede ser invocado a través de una instrucción y que la secuencia de instrucciones que se encuentra dentro del procedimiento se ejecutará tantas veces como sea invocado el procedimiento. Es posible que se comprenda mejor al explicarlo a través de BlockAr.

Se agrega una categoría llamada “Procedimiento” la cual dispondrá de dos sub-categorías, una para crear el procedimiento “Crear” y otra para llamarlo “Llamar”.

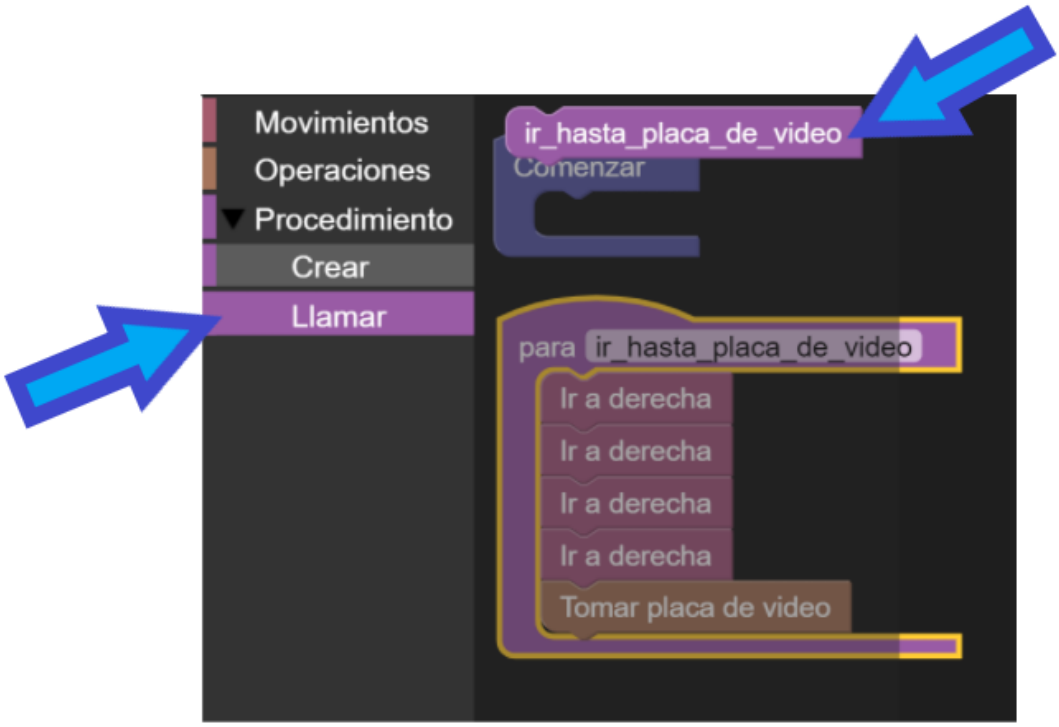




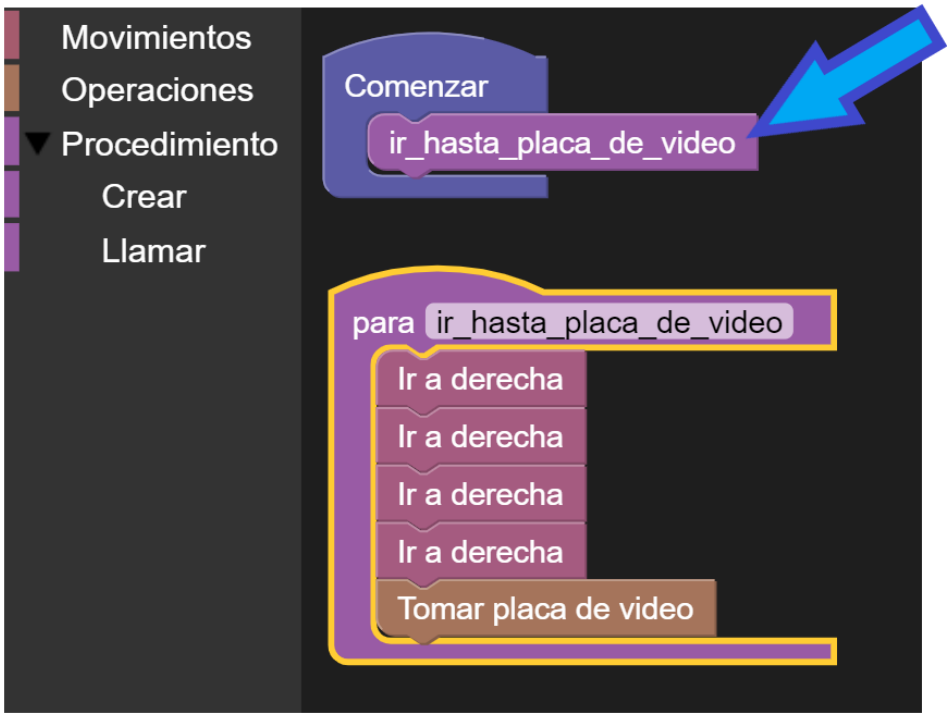
Al hacer clic en crear aparecerá a la derecha un bloque “para hacer algo”, se debe hacer clic en ese bloque para crearlo, una vez creado se mostrará en el espacio de trabajo y luego se sugiere renombrar “hacer algo” por un nombre característico de la acción que se quiere realizar, por ejemplo, en la figura siguiente se muestra un procedimiento llamado “ir_hasta_placa_de_video” que tiene una secuencia de instrucciones que se ejecutarán cuando el procedimiento es llamado.



Después de crear el bloque y asignarle un nombre, puede hacer clic en “Llamar” y se podrán ver los procedimientos que creó. En la categoría llamar aparecerán todos los procedimientos creados por el usuario.



Para llamar el procedimiento creado, en este caso, “ir_hasta_placa_de_video” es necesario ubicar el bloque dentro del bloque “Comenzar”, de esta manera se llamará ese procedimiento y al ejecutar el juego se ejecutará la secuencia contenida en el procedimiento. Desarrolle el siguiente caso en el juego. Luego presione ejecutar para ver lo que sucede.



En la clase puede acompañar a los estudiante a crear este procedimiento, y llamarlo de tal manera que el estudiante pueda ver como funciona el procedimiento. Aún no se debe resolver el ejercicio completo, ya que es parte de la actividad.

Desarrollo

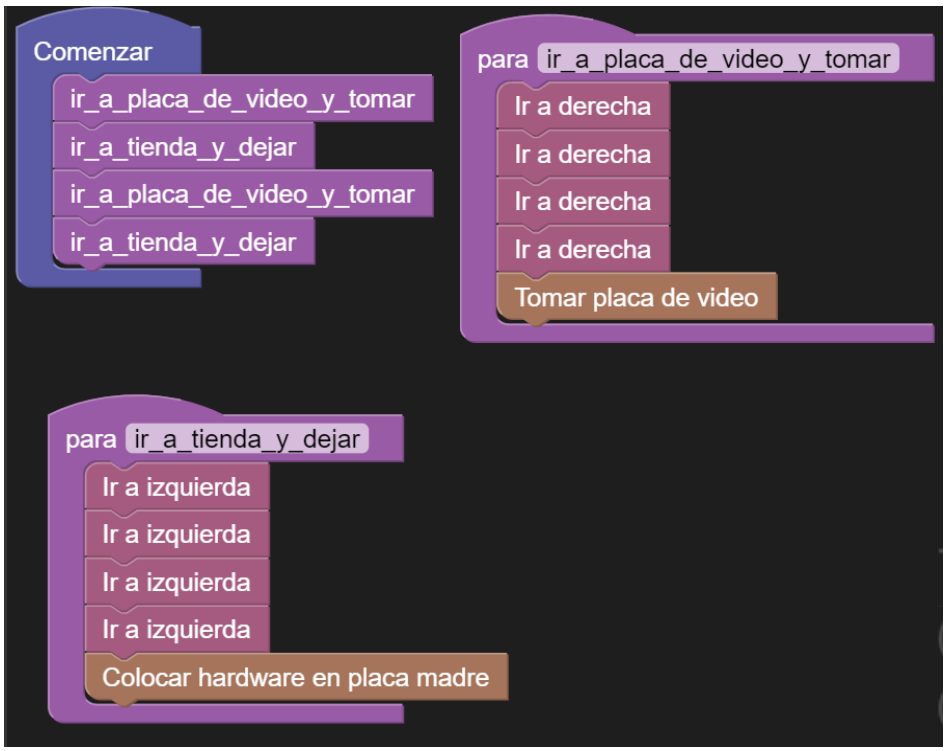
Actividad (30 min)

En la clase anterior se le sugiere al docente que acompañe a los estudiantes a comprender cómo crear un procedimiento y cómo llamarlo, así como también se acompaña al estudiante a comprender cómo es la dinámica del procedimiento.

Ahora dejaremos que el estudiante pueda resolver el desafío 5 completo por su cuenta.

Objetivo del desafío 5: El personaje quiere armar una pc para diseño gráfico, por el momento sólo pudo comprar dos placas de video de última generación y potentes. Debe ir a buscar las placas de video que compró, tomar de a una y llevarla a la placa madre para instalarla como corresponde. En éste desafío se recomienda utilizar procedimientos. No puede usar más de diecisiete bloques.

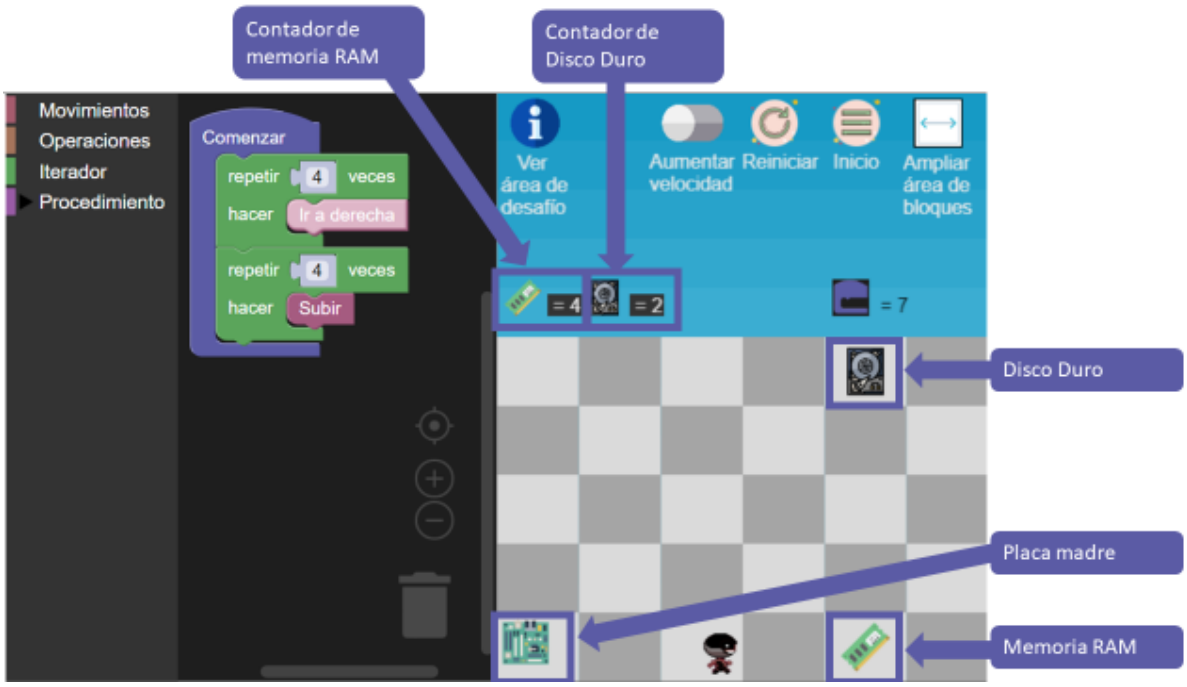
Solución para el desafío 5:



Cierre (30 min)

Una vez que se comprende a utilizar procedimientos, se propone que los estudiantes resuelvan el desafío 6.

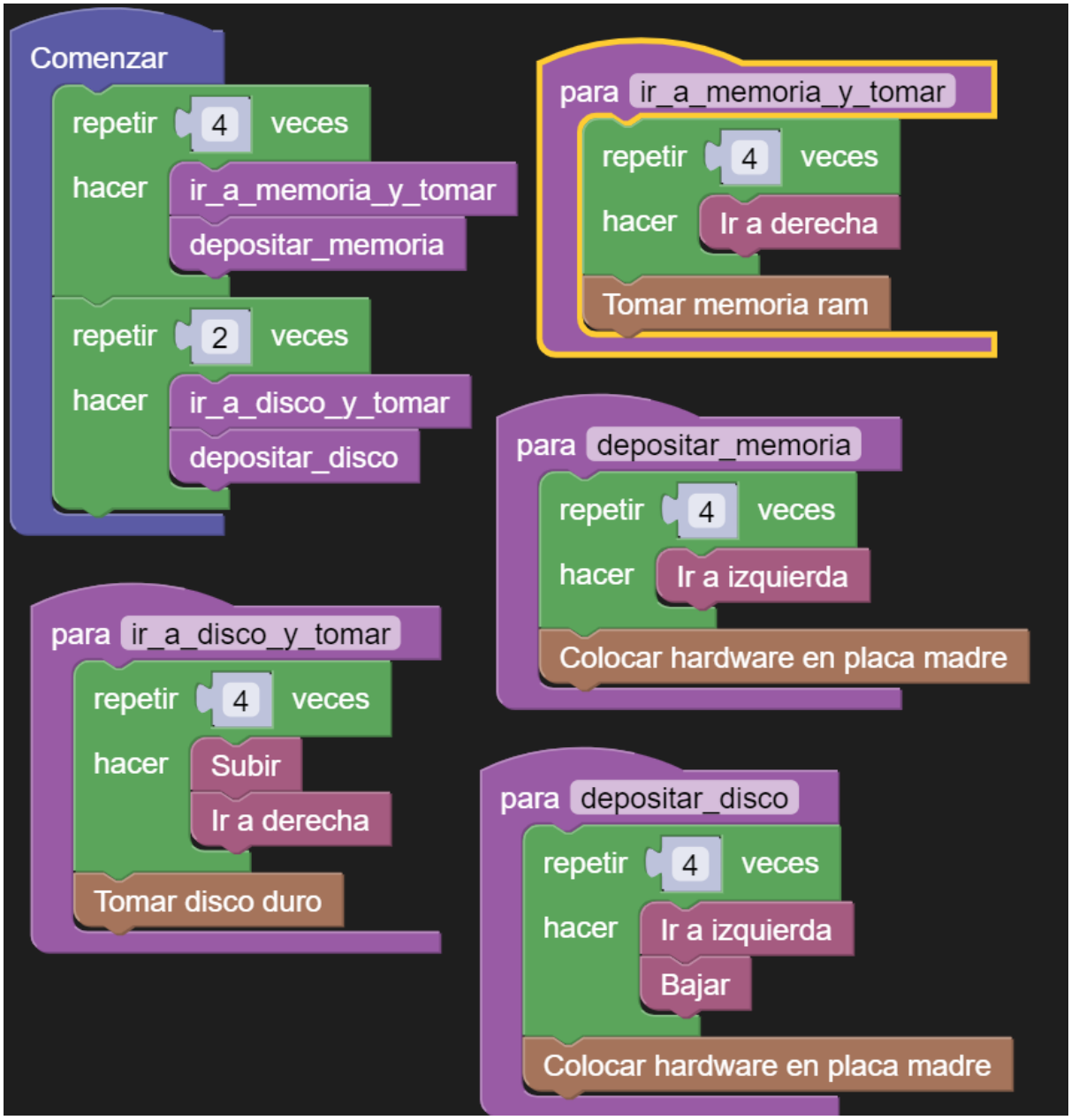
Ahora pueden hacer uso de todas las categorías aprendidas hasta el momento y la resolución es libre, no hay límite de bloques, hay muchas formas de resolverlo.



Objetivo del desafío 6: El personaje quiere armar una pc para diseño gráfico, para eso necesita bastante memoria ram y memoria de almacenamiento. Debe recolectar cuatro memorias ram y dos discos duros, no puede tomar más de un hardware a la vez, luego debe llevarlo a la placa madre para instalarlo como corresponde.

Si los estudiantes no logran terminarlo en la clase, pueden llevarse la tarea para resolverla en sus casas y se puede aprovechar la próxima clase para consultar.

Posible solución para el desafío 6:



Clase 5 (2 hs)

Objetivos

- Comprender el significado de condicional en programación.
- Utilizar términos tecnológicos.

Inicio (15 min)

Se le pregunta a los estudiantes si pudieron resolver el ejercicio de cierre de la clase anterior, si no pudieron, se retoma la clase anterior para ayudarles a cerrar el ejercicio.

Una vez que logran resolverlo se da inicio a un tema nuevo que trata de condicionales y operadores lógicos simples.

Desarrollo

Actividad (45 min)

Se procede a dibujar en el pizarrón tres celdas consecutivas, se recorta un personaje en papel del tamaño menor a la celda para que pueda caber en ella, se recorta una pc de papel del tamaño del personaje.

El docente deberá indicarle a los estudiantes que el personaje de papel va en la primera celda de la izquierda, y la pc de papel puede ir en la segunda o tercera celda, pero no se coloca hasta que el programa de inicio. Además se debe aclarar que hay una instrucción “avanzar” que avanza de a una celda a la vez en sentido derecha. El objetivo será que el personaje llegue a la bandera y si encuentra alguna pc en el camino la encienda, pero no se sabe si habrá pc o en caso que si hay una, no se sabe en qué celda estará, y si no hay pc y el personaje intenta encenderla termina perdiendo, por lo tanto, será necesario preguntar si hay pc antes de encenderla.

Entonces ¿Cómo puedo hacer eso?

En programación existen condicionales, algo parecido a:

si **“¿Hay Pc?”** entonces **“Enciendo pc”**

La estructura sería algo como:

si (hay_pc)
{
enciendo_pc
}

“hay_pc” es una operación lógica que va a determinar verdadero o falso según sea el caso. Por ejemplo, si el personaje se encuentra parado sobre una pc y el personaje pregunta si ¿hay pc? la respuesta sería: ¡sí!, entonces el personaje puede encender la pc a través de la instrucción “enciendo_pc”. Ahora si no hay pc, no se ejecutará lo que se encuentra dentro del bloque, es decir, lo que está dentro de las llaves { **enciendo_pc** }. Se pueden armar otros ejemplos en el pizarrón e interactuar con los estudiantes para despejar dudas. De ser necesario se puede ampliar la clase 2 para que involucre otra clase. Dependerá de la capacidad del docente para explicar y de los estudiantes para comprender.

Cierre (1 hora)

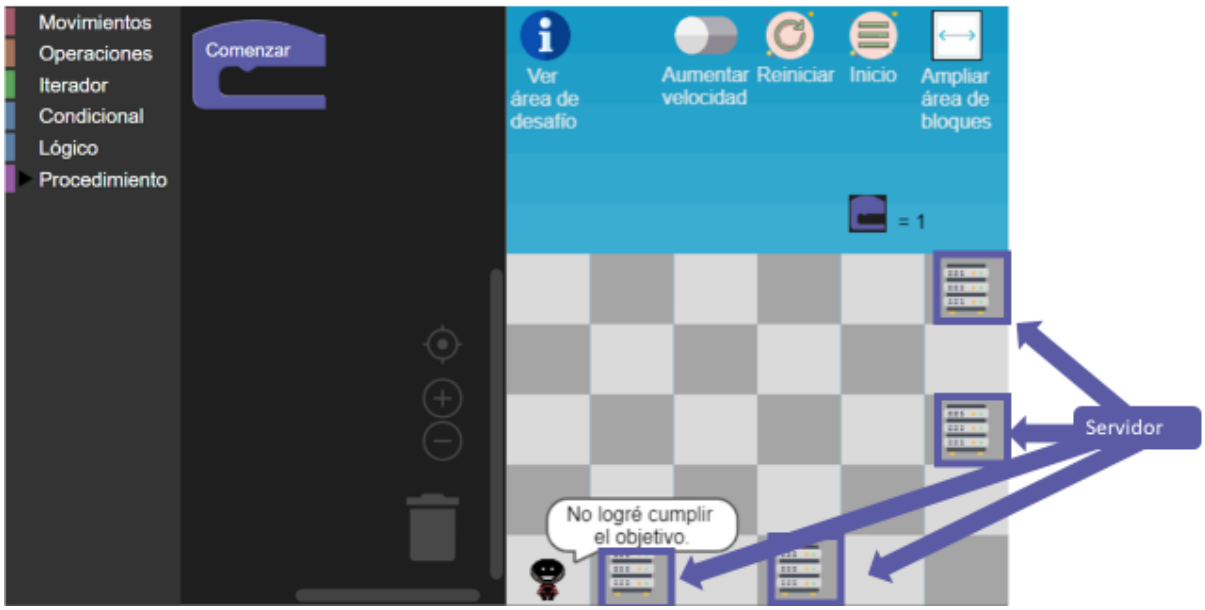
Deberán realizar el desafío 7 para poner en práctica los conocimientos que obtuvieron. Previo se deberá explicar que al principio no verán los servidores porque el personaje no sabe dónde están, por lo tanto deberán utilizar el bloque que se encuentra dentro de la categoría “Condicional” y el bloque que se encuentra dentro de la categoría “Lógico”.

Objetivo del desafío 7: La refrigeración de la sala de servidores dejó de funcionar. El personaje debe recorrer la sala en busca de servidores y desactivarlos antes que se quemen.

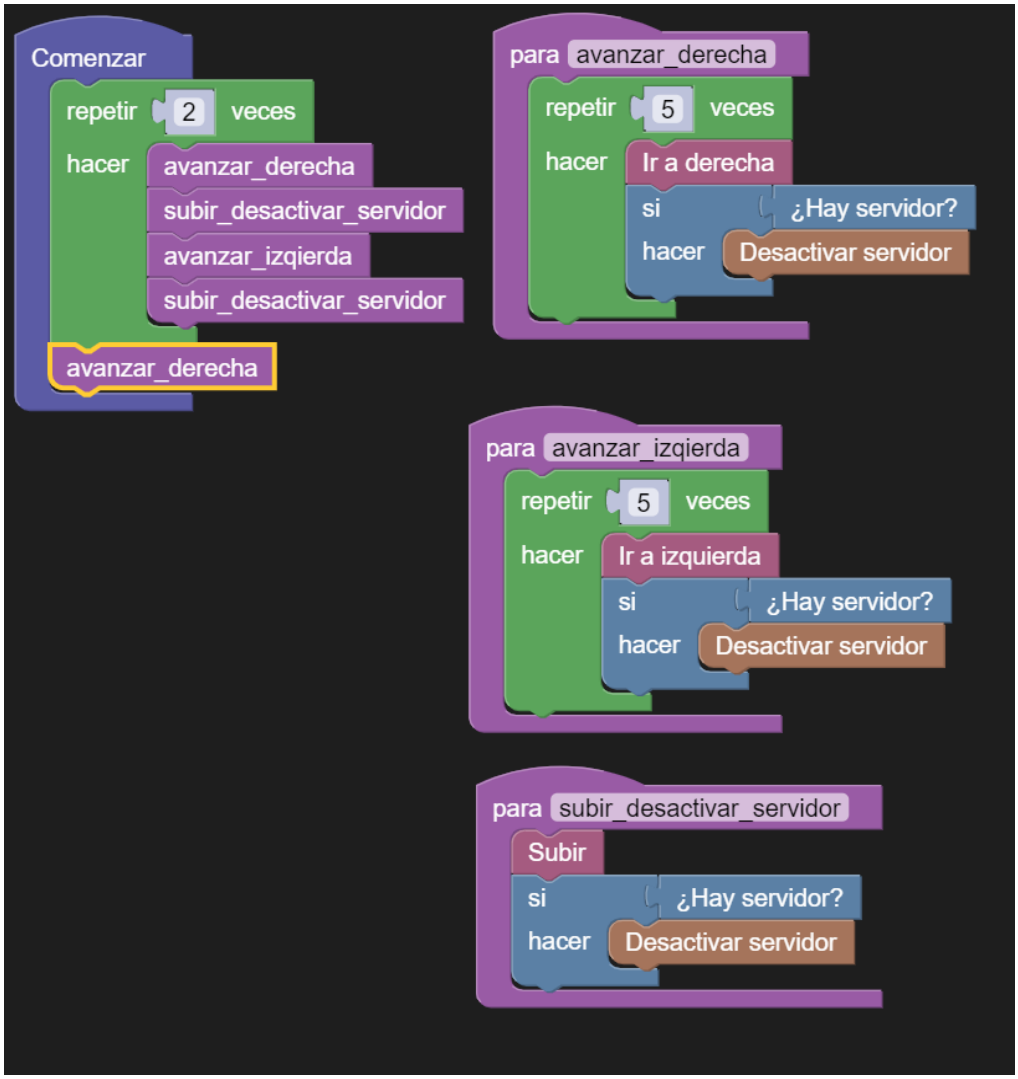
Es el desafío más avanzado, puede requerir ayuda del docente.

Al hacer clic en ejecutar se mostrarán los servidores.

Si no logra cumplir el objetivo del desafío entonces se verán los servidores y algo como muestra la figura



Posible solución para el desafío 7:


Comenzar
repetir 2 veces
hacer
avanzar_derecha
subir_desactivar_servidor
avanzar_izqierda
subir_desactivar_servidor
avanzar_derecha
para avanzar_derecha
repetir 5 veces
hacer
Ir a derecha
si ¿Hay servidor?
hacer Desactivar servidor
para avanzar_izqierda
repetir 5 veces
hacer
Ir a izquierda
si ¿Hay servidor?
hacer Desactivar servidor
para subir_desactivar_servidor
Subir
si ¿Hay servidor?
hacer Desactivar servidor

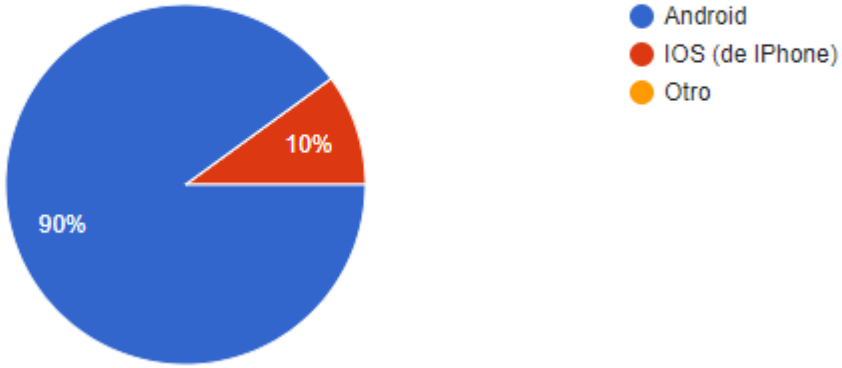
Anexo 3: Encuestas

1. ¿Tenes celular?

10 respuestas

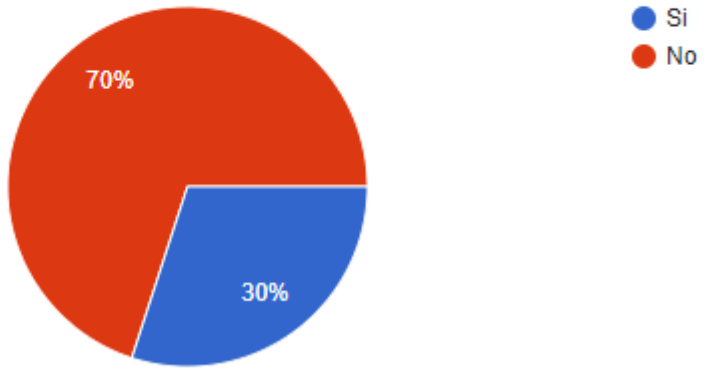
2. ¿Qué sistema operativo tiene tu celular?

10 respuestas



3. ¿Tuviste inconvenientes para instalar el juego?

10 respuestas



4. Si respondiste que Si la pregunta anterior, te pedimos que nos cuentes el motivo: ¿por qué crees que ocurrió?

3 respuestas

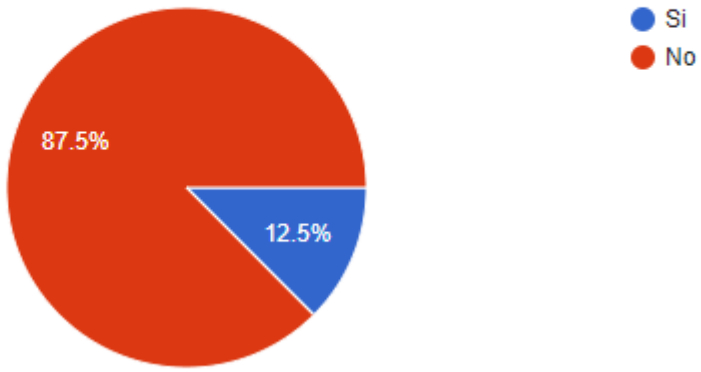
Por lo que me dijo el profe era por el modelo del celular

Según el profe era por el modelo del celular,

Porque era para androi y no para iPhones

5. ¿Tuviste inconvenientes con el celular para jugar?

8 respuestas



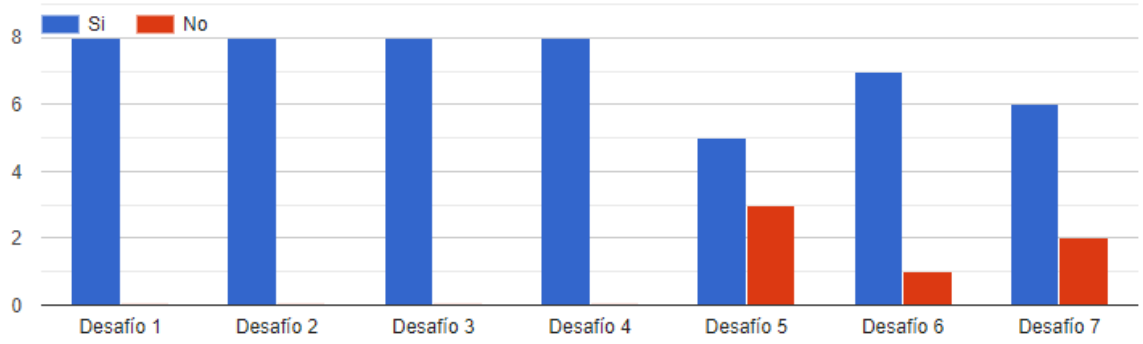
6. Si respondiste Si ¿Por qué crees que fueron los inconvenientes?

1 respuesta

Porque es para androi y no para iPhones

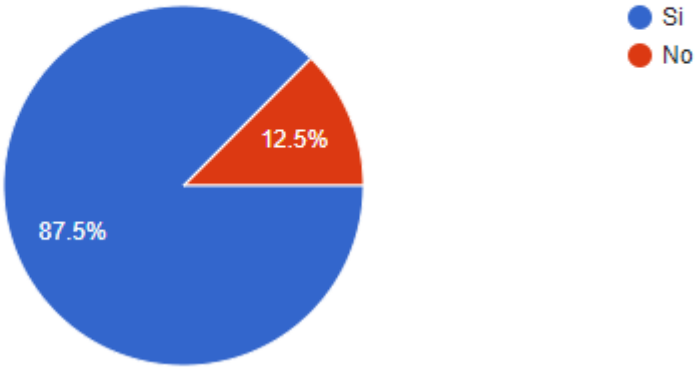
7. ¿Entendiste las consignas de los desafíos?

Copiar



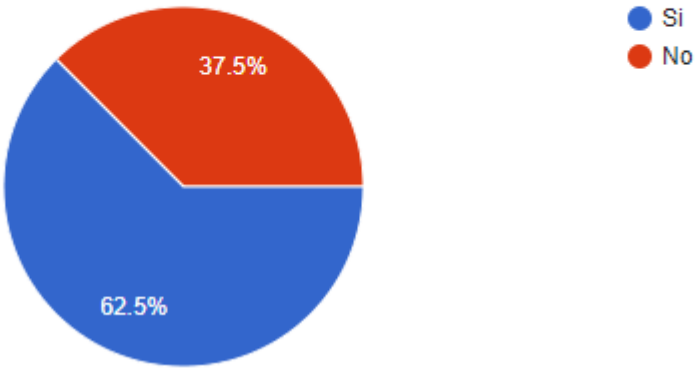
8. ¿Lograste completar todos los desafíos?

8 respuestas



9. ¿Consideras que sin la explicación en clase podrías haber resuelto los desafíos propuestos?

8 respuestas

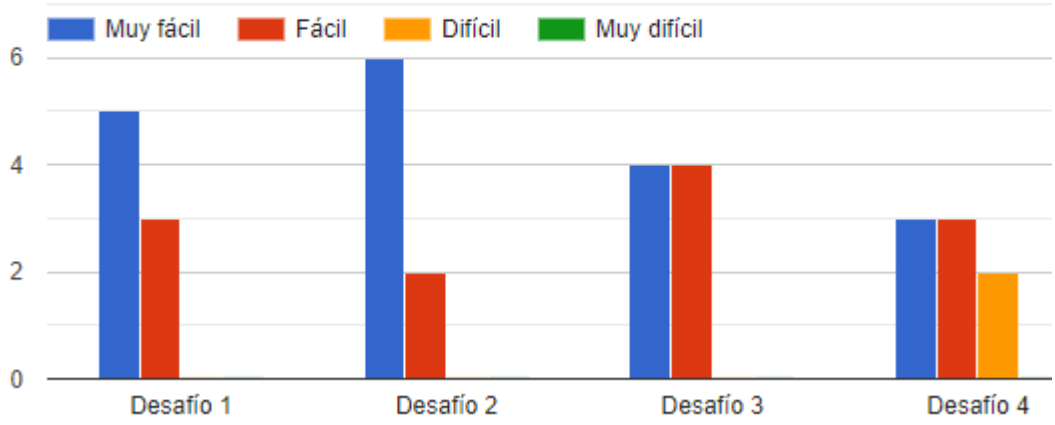


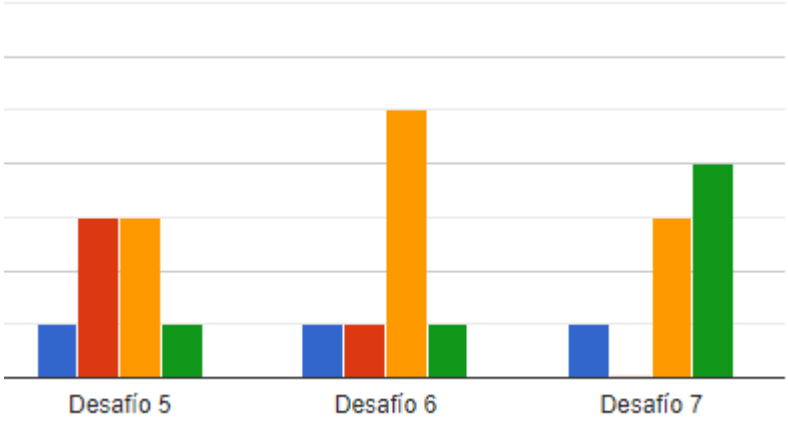
10. Si respondiste No, nos podrás contar cuál pensás que es el motivo.

3 respuestas

No entendía una de las consignas del juego hasta que lo explico
Era fácil pero a la vez era difícil
no sabia la forma de moverlo

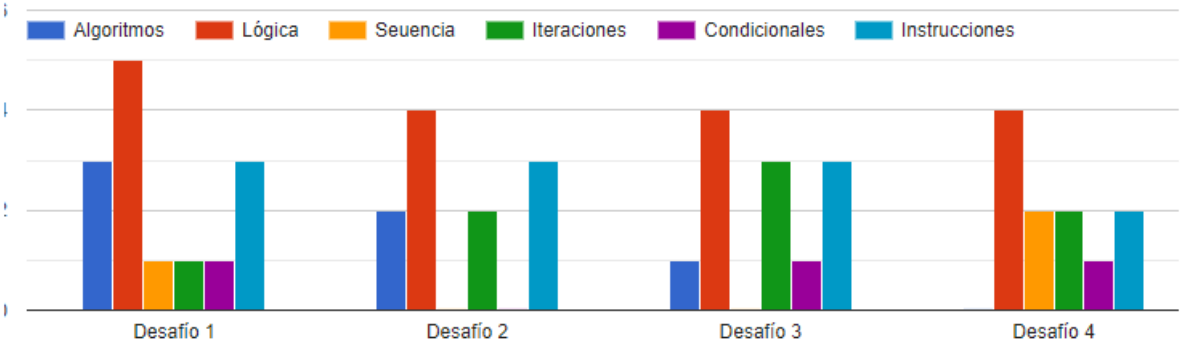
11. Indicá el grado de dificultad para completar cada desafío.

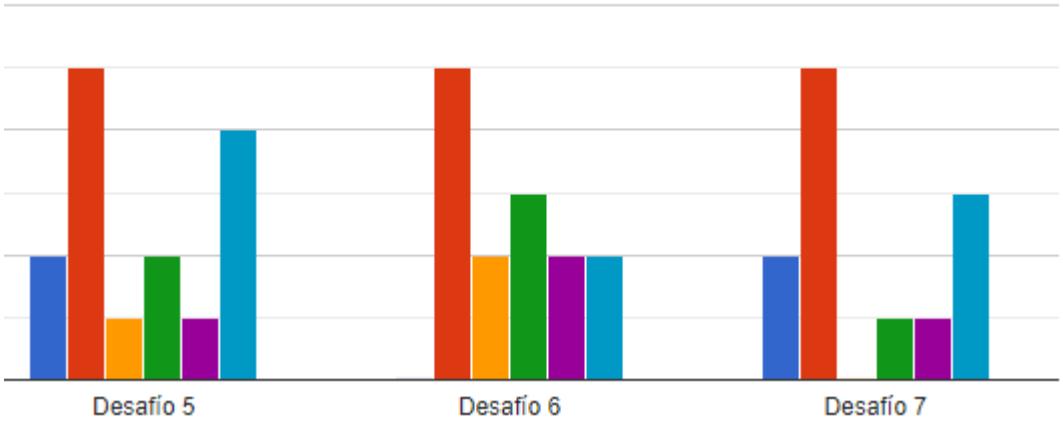




12. ¿Qué conceptos crees que aprendiste jugando? Podes seleccionar más de una opción por desafío.

Copiar





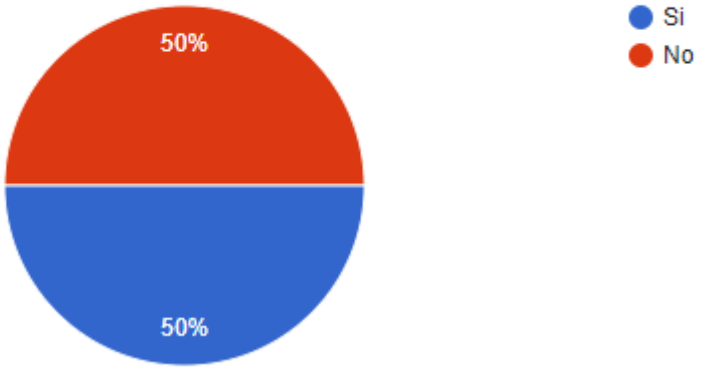
13. ¿Qué otras cosas aprendiste?

3 respuestas

En ele proceso del juego aprendí a seguir instrucciones
Tiempo
a carburar la mente un poco mas

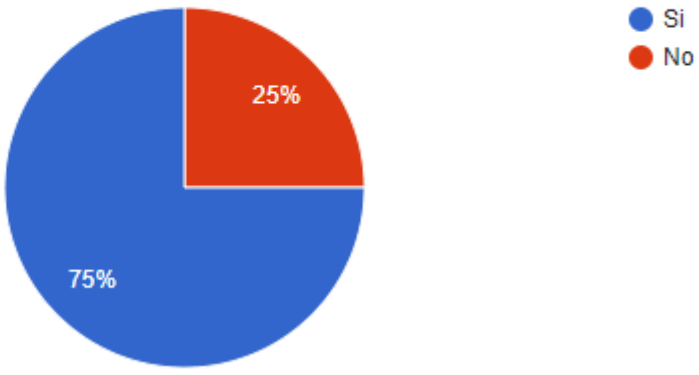
14. ¿Jugaste en otro lugar que no sea en la clase?

8 respuestas



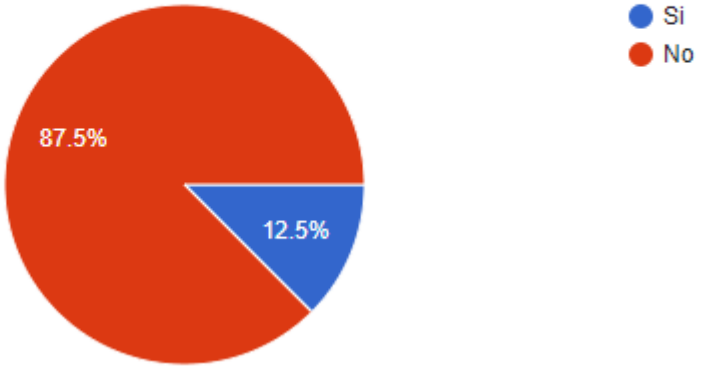
15. ¿Te resultó entretenido jugar?

8 respuestas



16. ¿Encontraste errores en la aplicación mientras jugabas?

8 respuestas



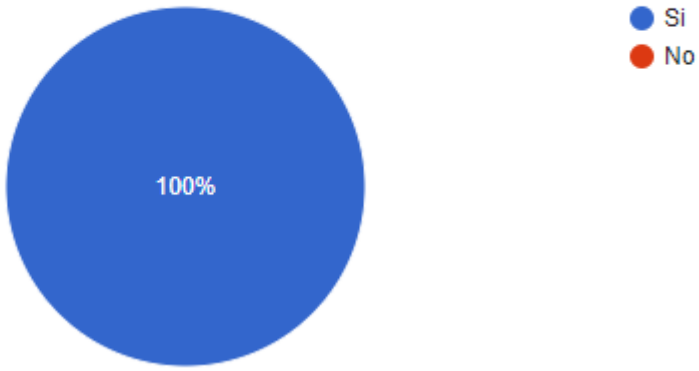
17. ¿Qué consideras que le podríamos agregar a la aplicación en cuanto a operaciones, funciones, desafíos, tutoriales, entre otros temas?

3 respuestas

Nada
Desafíos y tutoriales
mas desafios y clacificacion de rango

18. ¿Recomendarías jugar este juego a otras personas?

8 respuestas



19. Si respondiste No, ¿nos podrás contar el motivo?.

0 respuestas

Todavía no hay respuestas para esta pregunta.