

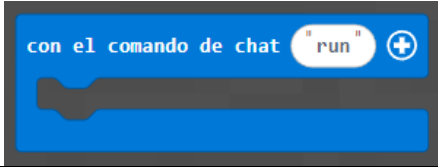
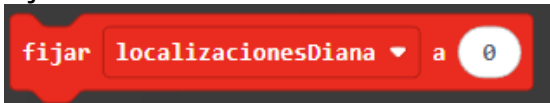
Práctica guiada 5. Entrenador de puntería

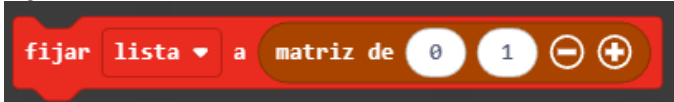


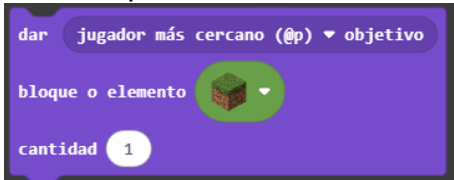
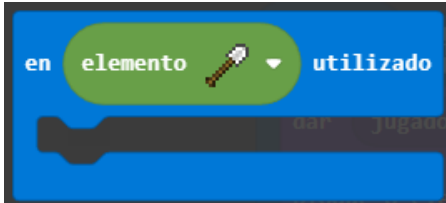
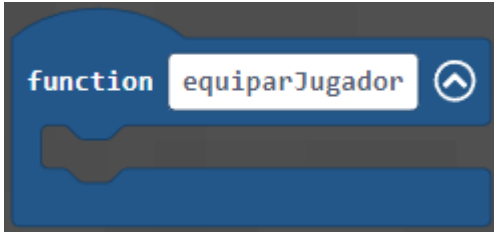
En esta práctica aprenderemos a diseñar y programar un *entrenador de puntería* dentro de Minecraft Education. A lo largo de las actividades, construiremos un sistema que genera objetivos aleatorios para que el jugador (o el Agente) practique su precisión. Trabajaremos conceptos como eventos de botón, generación aleatoria de posiciones, temporizadores, bucles y condiciones para detectar aciertos. Además, profundizaremos en la lógica de control del flujo del juego, gestionando puntuaciones y reinicios tras cada ronda.

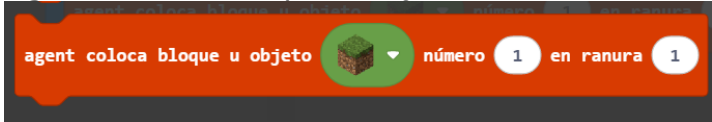

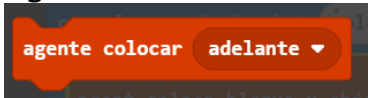
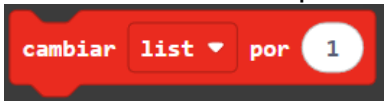

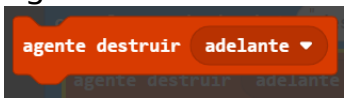

Mecánicas de Minecraft

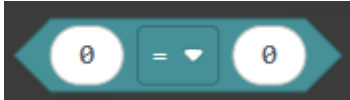
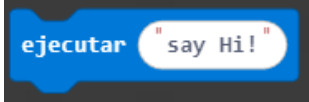
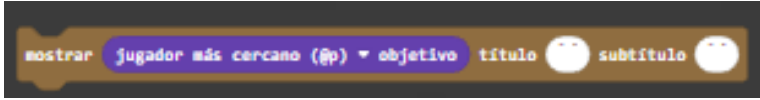

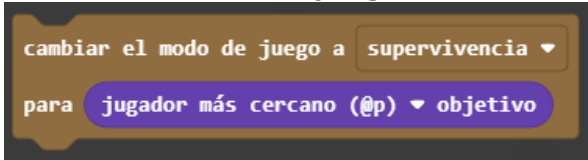
Mecánicas	Para ordenador
Abrir el editor de código	C
Abrir chat	T
Abrir el menú de pausa; cerrar los menús de juego	ESC
Minar un bloque	Clic Izquierdo
Interactuar con botones y con personajes	Clic derecho
Saltar	Barra espaciadora
Movimiento del jugador (Adelante/Atrás/Izquierda/Derecha)	W/S/A/D
Dirección del jugador	Movimiento del ratón

Bloques para esta práctica

Con el comando de chat 	Evento que se activa al escribir una palabra en concreto en el chat.
Fijar 	Establece el valor para una variable.

<p>Fijar lista de matriz</p> 	<p>Establece el valor de una variable para que sea una matriz.</p>
<p>Coordenadas mundo</p> 	<p>Crea un nuevo valor de coordenadas en el mundo.</p>
<p>Agente se teletransporta</p> 	<p>Teletransporta al agente a unas coordenadas en concreto.</p>
<p>Obtener valor aleatorio</p> 	<p>Obtiene el valor aleatorio de una matriz.</p>
<p>Dar bloque o elemento</p> 	<p>Coloca el bloque o elemento seleccionado, la cantidad de veces seleccionada al inventario del jugador escogido.</p>
<p>Valor del elemento</p> 	<p>Devuelve un tipo concreto de elemento de Minecraft.</p>
<p>En elemento utilizado</p> 	<p>Un trozo de Código que solo se ejecuta en el momento en que el elemento seleccionado se utiliza.</p>
<p>Función</p> 	<p>Bloque de código que realiza una tarea en específico.</p>
<p>Llamada a una función</p> 	<p>De este modo se puede ejecutar el código de una función.</p>

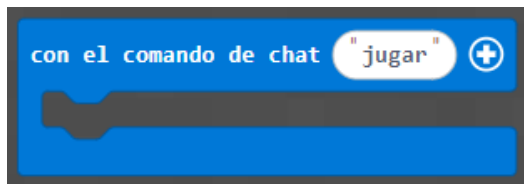
<p>Agente coloca bloque u objeto</p> 	<p>Coloca el número especificado de bloques de minecraft/objetos en la ranura especificada del inventario del agente.</p>
<p>Establecer ranura activa</p> 	<p>Establece que ranura del inventario va a utilizar el agente.</p>
<p>Agente colocar</p> 	<p>Coloca un objeto en el mundo desde la ranura seleccionada anteriormente del inventario del agente.</p>
<p>Cambiar constante por</p> 	<p>Cambiar el valor de una constante por lo indicado. En caso de ser constante numérica suma o resta según el valor puesto.</p>
<p>Agente detectar</p> 	<p>Comprueba si hay un bloque en la posición seleccionada respecto al agente.</p>
<p>Agente destruir</p> 	<p>Manda al agente la orden de destruir el bloque que se encuentre en la dirección indicada.</p>
<p>Si / si no</p> 	<p>Permite ejecutar acciones diferentes según si se cumple o no una condición.</p>

<p>Condicional</p> 	<p>Devuelve verdadero si los dos valores de la condición son iguales.</p>
<p>Ejecutar</p> 	<p>Ejecuta comandos de chat a través de código.</p>
<p>Mostrar</p> 	<p>Muestra un texto en pantalla al jugador seleccionado, con la opción de poner un título y un subtítulo.</p>
<p>Unir texto</p> 	<p>Une diferentes valores y los convierte en un solo valor de texto.</p>
<p>Mientras</p> 	<p>Mientras que no ocurra la condición, el bucle se seguirá ejecutando, una vez que ocurra, parará.</p>
<p>Cambiar el modo de juego</p> 	<p>Cambia el modo de juego entre supervivencia, creativo y aventura.</p>

Parte 1. Iniciando las variables

Como siempre, lo primero que vamos a hacer es crear un nuevo proyecto, recordando escoger un nombre adecuado. Lo segundo que debemos tener en cuenta es que, a lo largo del juego, vamos a tener que mantener el seguimiento de el número de diana que han sido acertadas, el tiempo que tarda el jugador en completar la partida y las diferentes localizaciones donde podrá aparecer la diana.

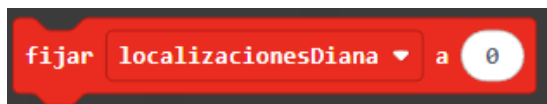
Cuando tenemos que llevar la cuenta de valores que van a cambiar a lo largo de la partida necesitamos variables para poder hacerlo. Es por esto por lo que vamos a crear nuevas variables, pero esta vez las vamos a iniciar en el bloque “con el comando de chat”.



Las variables que vamos a necesitar serán las siguientes:

- “tiempo”: nos servirá para llevar la cuenta del tiempo que tarda el jugador en conseguir dar a todas las dianas
- “dianasRestantes”: comenzará con un valor concreto que representará el número de dianas totales a las que deberá darle el jugador para finalizar la partida e irá restándose cada vez que el jugador le dé a una diana.
- “localizacionesDiana”: nos servirá para almacenar las posibles localizaciones en las que aparecerá la diana, cada vez aparecerá en una diferente al azar.

Una vez creadas las variables, debemos darles los valores con los que comienzan, este proceso se llama inicializar una variable, y como se explicaba anteriormente, se hará en el bloque “con el comando de chat”, usando el bloque “fijar” de las variables.



Los valores de inicio de cada una de las variables serán: para el tiempo, 0, puesto que vamos a ir incrementándolo poco a poco; para las dianas restantes, deberá ser el número de dianas que queremos que acierte el jugador para ganar, podemos ponerlo a 20 para comenzar, pero siempre podemos cambiarlo y poner el número que creamos conveniente; y finalmente, localizaciones diana que implicará utilizar una lista de localizaciones, para crear esta lista vamos a necesitar usar una matriz, recordamos lo que hemos visto en la parte de teoría.

Las matrices son una colección de datos similares que pueden ser referenciados en el código, en nuestro caso, una lista de localizaciones en el mundo que puedan ser seleccionadas de forma aleatoria.

Las matrices las podréis encontrar en el apartado AVANZADO y ARREGLOS, utilizaremos el primer bloque de todos, llamado “fijar lista”



Cambiaremos la variable “lista” por nuestra variable “localizacionesDiana”. Al observar la matriz, podemos fijarnos en que tiene ahora mismo espacio para almacenar 2 valores diferentes, si clicamos el más o el menos, podemos añadir y quitar valores a esta matriz. Si nos fijamos en el mapa, nos podemos dar cuenta de que la diana va a tener 5 posibles espacios sobre los que aparecer.

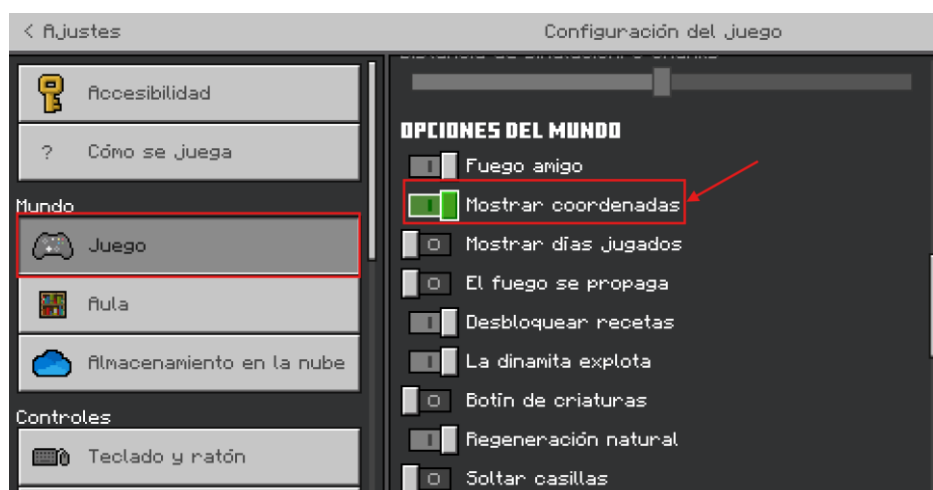


Por tanto, vamos a añadir 5 espacios para valores en nuestra matriz y los vamos a rellenar con bloques de valor de posición en el mundo. Luego marcaremos que posición concreta tiene cada uno.



Más tarde, vamos a hacer que nuestro agente se teleporte de forma aleatoria al centro de uno de los 5 espacios de diana y coloque una justo debajo suyo. Lo que haremos a continuación es analizar cuáles son las coordenadas donde deberá ir cada uno de los bloques.

Para ello, lo primero será ir a ajustes, al apartado de juego y le diremos que muestre las coordenadas.



Ahora ya podremos visualizar nuestras coordenadas y las de nuestro agente, en la esquina superior izquierda. Por el momento, solo necesitaremos nuestras coordenadas.

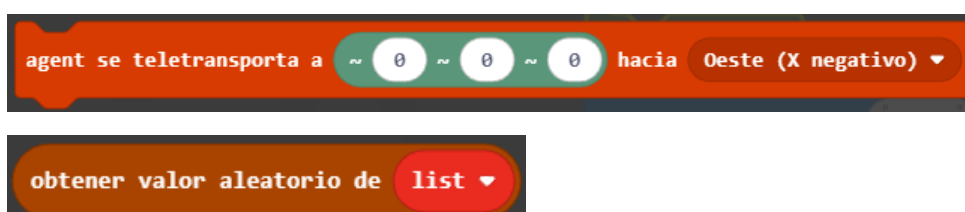


Para hacer más fácil la obtención de coordenadas, vamos a ir a nuestro inventario y vamos a coger un bloque, lo vamos a colocar en el centro de cada una de las zonas de diana, recordad que podéis volar en el modo creativo si pulsáis dos veces seguidas la barra espaciadora. Obtendréis algo así:



Ahora, debéis colocar vuestro personaje encima de cada uno de esos bloques y anotar las posiciones, ya que esas van a ser las posiciones en las que deberá colocarse aleatoriamente nuestro agente para colocar las dianas. Una vez las tengamos todas anotadas, las podemos pasar a nuestra matriz de coordenadas. Recordad eliminar los bloques de madera una vez terminéis de recoger las coordenadas.

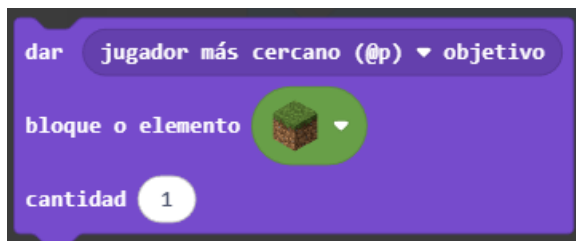
Ahora que ya tenemos las posibles coordenadas, queremos que el agente se teletransporte a alguna de ellas de forma aleatoria. Así que vamos a añadir justo debajo de la creación de nuestra matriz, utilizaremos el bloque de “agente se teletransporta a” y le daremos como valor “obtener valor aleatorio de” del apartado ARREGLOS.



Para este último tendremos que seleccionar la variable donde tenemos guardadas todas nuestras coordenadas. Y además le diremos que mire hacia el norte, de forma que se quede mirando hacia el jugador. Puedes probarlo poniendo “jugar” en el chat, cada vez el agente estará en un sitio diferente de los seleccionados, ten en cuenta que al ser aleatorio puede que no se mueva porque ha salido dos veces seguidas la misma coordenada.

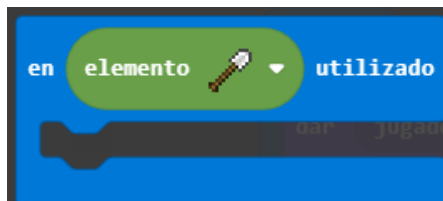
Parte 2. Equipando al jugador

Vamos a crear una nueva función que equipe a nuestro jugador con todo lo necesario para jugar. La llamaremos “equiparJugador”, como hemos hecho en anteriores prácticas, vamos a darle al jugador el equipo que necesita, para ello, usaremos el bloque “dar”, pero deberemos cambiar un pequeño detalle, ya que el valor que tiene ahora en dar solo permite bloques y nosotros queremos darle un arco, así que tendremos que cambiarlo por elemento, y seleccionar el arco.



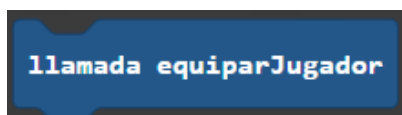
Ahora, duplicaremos ese mismo bloque y le añadiremos al jugador unas flechas, que las necesitará para disparar el arco, de momento le daremos 20, pero es muy posible que necesite más si el jugador falla alguno de los objetivos. Así que vamos a solucionar este problema.

En el apartado de JUGADOR, nos encontramos con un bloque llamado “en elemento utilizado”, este bloque no se puede añadir a ninguna parte de código y funciona por si solo, activándose cada vez que se utiliza un elemento.



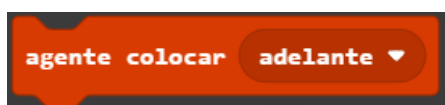
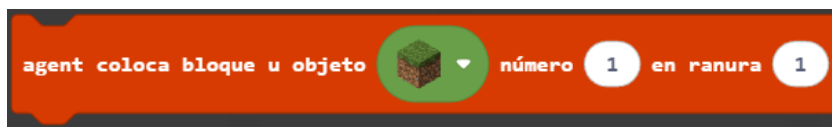
En nuestro caso, lo que vamos a hacer es que cada vez que el jugador utilice el arco le devolveremos una flecha, con el bloque “dar”, pero solamente una esta vez.

Ahora solo tenemos que llamar a la función desde el bloque “con el comando de chat” y listo, podremos probar si el jugador recibe el arco y las flechas y si estas se recargan cada vez que usa una.



Parte 3. Colocando las dianas

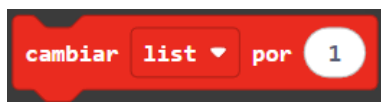
Crearemos una nueva función llamada “crearDiana”. Cuando iniciamos el juego, el agente ya se coloca en una de las posibles posiciones para la diana, así que vamos a pedirle con esta función, que coloque justo debajo suya un bloque llamado “target”. Recordad, primero debemos darle al agente el bloque y luego le podemos pedir que lo ponga en el mundo, asegurándonos antes de que la ranura activa es la adecuada.



Para finalizar esta función, la llamaremos al final de todo del bloque “con el comando de chat”, justo después de la función anterior.

Parte 4. Programando la jugabilidad

Crearemos una nueva función, llamada jugabilidad. Mientras que esta función esté en marcha, queremos que el tiempo vaya pasando y contabilizarlo, para ello vamos a utilizar “cambiar variable”, seleccionando nuestra variable de tiempo.



Lo ponemos de esta forma porque cuando llamemos a esta función estaremos haciéndolo de forma constante y repetida, por lo que el tiempo irá incrementando.

Después de sumar uno al tiempo, queremos comprobar si el jugador ha conseguido darle al objetivo, para ello utilizaremos un “si”, ya que el código va a comprobar Si el bloque de “target” ha sido golpeado. ¿Cómo hará esto? Pues el bloque de “target” manda una señal de redstone cuando es golpeado, como se muestra [en este vídeo](#), y podemos usar eso para comprobar si ha sido golpeado, necesitaremos usar el bloque de “agente detectar”, pero en lugar de detectar un bloque le diremos que detecte una señal de redstone, y que la detecte debajo.



En caso de que detecte una señal de redstone, querrá decir que el jugador ha golpeado el objetivo, y, por tanto, queremos que destruya ese bloque.



Además, también queremos que se reduzca en uno el número de dianas que le quedan por golpear a nuestro jugador, por lo que usando el mismo bloque de antes de “cambiar variable” le daremos el valor “-1” a nuestra variable “dianasRestantes”.

Ahora que el jugador ha golpeado una diana, pueden ocurrir dos cosas, la primera es que el agente se mueva y genere otra diana o la segunda, que es que el jugador ya ha llegado al número de dianas que debía llegar y, por tanto, se ha terminado el juego. Así que tendremos que usar un bloque de condición de tipo “sí/sí no”, y lo colocaremos debajo pero dentro de nuestro actual “sí”.

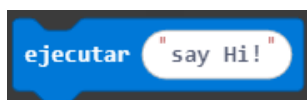


Lo más sencillo es primero comprobar si todavía quedan dianas para que el juego se acabe, o lo que es lo mismo, si la variable “dianasRestantes” es mayor que 0. Para ello necesitaremos usar un bloque de comparación, adaptalo según creas conveniente.

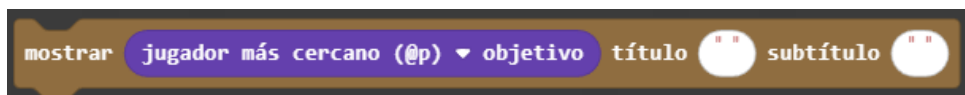


Si las “dianasRestantes” es mayor que 0, es que el juego debe continuar, y que, por tanto, el agente debe escoger una nueva posición aleatoria, para ello, podemos duplicar el bloque que tenemos en el bloque “con el comando de chat” llamado “agente teletransportar...” y colocarlo en nuestro bloque de condición, y también deberemos duplicar y añadir para que el agente cree una nueva diana, así que después del bloque de teletransporte, pondremos el bloque de “llamada crearDiana”.

Y ahora, ¿qué pasa si no quedan objetivos a los que dar? Pues que nuestro juego debe finalizar, comenzaremos por quitarle al jugador el arco y las flechas, ¿recuerdas como se borran todos los elementos del inventario del jugador? Era usando el siguiente bloque, trata de recordar el resto o revísalo en prácticas anteriores.



Una vez limpio el inventario, vamos a mostrar un mensaje al jugador de que ha terminado, utilizando el bloque “mostrar”.



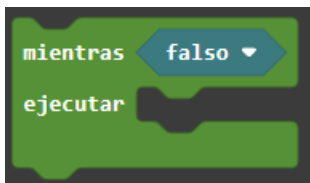
Para el título queremos mostrarle un mensaje de “¡SE ACABÓ!”, o algo parecido, pero para el subtítulo, queremos darle información al jugador de la puntuación que ha obtenido, para ello vamos a necesitar un nuevo bloque del apartado AVANZADO y TEXTO.



El bloque unir nos permite mostrar diferentes valores seguidos en formato de texto, de forma que podemos poner un primer valor que diga “Tiempo:”, un segundo valor que muestre nuestra variable de “tiempo”, simplemente añadiéndola a su correspondiente hueco, y, finalmente, añadiremos un tercer valor para indicar en qué se ha medido el tiempo, puesto que minecraft no funciona con segundos, si no con “ticks”. Puesto que es un bloque complejo de explicar, a continuación, se muestra cómo debería quedar:



Finalmente, solo nos quedaría llamar a la función, puesto que queremos que esta se ejecute mientras que queden objetivos, la añadiremos con un bucle de “mientras” al final del bloque “con el comando de chat”.



Y añadiremos como condición la misma que hemos utilizado para el bloque “si/si no”, es decir, mientras que el valor de “dianasRestantes” sea mayor que 0, entonces, llamaremos a la función “jugabilidad”.

Ahora ya está el juego casi completo, solo falta solucionar algunos posibles fallos.

Parte 5. Arreglando fallos y mejorando el juego

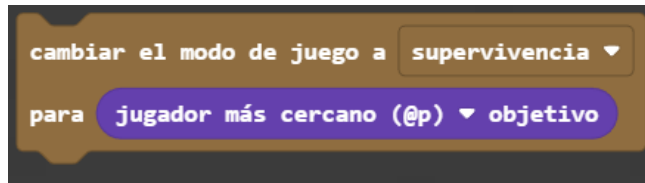
Tras finalizar el programa nos queda un último fallo por solucionar, y es que la flechas no desaparecen de mapa después de golpear al objetivo lo que produce que se acumulen muchas flechas en el suelo tras cada partida, y que, si da la casualidad de que se genera un nuevo objetivo justo debajo del que acabas de destruir, es posible que caiga la flecha de arriba y golpee el nuevo objetivo sin haberlo hecho el jugador.

Para arreglar esto, no tenemos ningún bloque concreto que nos pueda ayudar, pero si tenemos un comando de chat, por lo que vamos a utilizar el bloque “ejecutar” y lo vamos a colocar justo el primero dentro de nuestro “si el agente detecta redstone”. El comando que queremos ejecutar será el siguiente “remove @e[type=arrow]”, podéis copiarlo y pegarlo para que os resulte más sencillo.

Además, podemos añadir interactividad a nuestro juego si añadimos justo después de lo anterior, un sonido, es decir, que, al dar a un objetivo, suene algo, para dar un poco de vida a nuestro juego. Para ello, como tampoco tenemos un bloque específico, necesitaremos de nuevo el bloque “execute”, esta vez, el comando será el siguiente “playsound random.orb”.

Como último cambio, vamos a hacer como en el juego del parkour y vamos a poner al jugador en modo supervivencia, esta vez, lo pondremos simplemente al inicio del juego,

cuando lanzamos el comando por chat de “jugar”, aunque, si lo crees conveniente, también puede buscar otro lugar para colocarlo.



Ahora sí, ya está finalizado nuestro juego de puntería, ya podéis lanzar el código y probarlo. ¿En cuánto tiempo lográis finalizar el juego? ¿Cómo podrías mejorar el juego?

Entrega

Una vez hayas finalizado el juego, asegúrate de:

1. Guardar el mundo correctamente.
2. Guarda el proyecto y realiza una captura de pantalla del código y del circuito



3. Presenta de forma oral el proyecto o haz un documento con las capturas y la explicación del código realizado.
4. Sube el archivo del proyecto y el documento, si es el caso, a Moodle.

ANTES DE LA PRÓXIMA PRÁCTICA.