INSTITUCION EDUCATIVA	TECNOLOGIA E INFORMATICA	GRADO 8°

GUIA INSTITUCIONAL DE APRENDIZAJE

PROGRAMACION CREATIVA CON SCRATCH

De las ideas a un videojuego: aprende a pensar, crear, probar y mejorar

RETO GENERAL: convertirte en creador digital y construir un proyecto interactivo que funcione gracias a tus propias instrucciones.

Institucion	San Vicente de Paúl	**Area**	Tecnologia e Informatica
Docente	Carlos Alberto Stuart Contreras	**Grado**	8°
Estudiante	_____	**Grupo**	1, 2, 3
Periodo	3	**Fecha**	__ / __ / ____
Duracion sugerida	8 a 10 horas de clase	**Guia No.**	1

Nombre del proyecto final: _____

Como usar esta guia

Lee la misión, sigue los pasos en Scratch, prueba lo que construyes y registra tus evidencias. Equivocarse es parte de programar: cada error es una pista para mejorar.

Acuerdo de ciudadanía digital

Crea contenidos respetuosos, usa nombres apropiados, no publiques datos personales y reconoce ideas o recursos de otras personas cuando los utilices.

1. Ruta de aprendizaje

Esta guía usa una metodología de aprendizaje por retos. Cada misión presenta un problema pequeño, una construcción en Scratch y una oportunidad para mejorar lo creado. El objetivo no es memorizar bloques: es aprender a pensar como programador.

Mision	Pregunta clave	Conceptos	Producto
0	Piensa como una máquina	Algoritmos e instrucciones precisas	Reto sin computador
1	Conoce tu laboratorio	Interfaz, objetos, escenario y eventos	Mini animacion
2	Haz que se mueva	Secuencias, coordenadas y dirección	Ruta animada
3	Repite con inteligencia	Bucles, espera y depuración	Animacion fluida
4	Toma decisiones	Sensores, condicionales y operadores	Interacción con teclado
5	Guarda datos	Variables, puntaje y temporizador	Juego Atrapa la estrella
6	Coordina personajes	Mensajes, escenas y sincronización	Historia interactiva
Final	Crea tu desafio	Integración de conceptos	Videojuego o historia interactiva

2. Lo que aprenderás

Pensamiento computacional

Descomponer problemas, crear algoritmos, reconocer patrones, probar soluciones y corregir errores.

Programación por bloques

Usar eventos, secuencias, bucles, condiciones, sensores, operadores, variables y mensajes.

Creacion digital

Diseñar animaciones, historias y juegos con una intención clara y una experiencia para el usuario.

Trabajo responsable

Organizar archivos, respetar contenidos de otras personas y comunicar decisiones del proyecto.

3. Metas de aprendizaje

- Explico con mis palabras que es un algoritmo y por qué una instrucción debe ser clara y ordenada.
- Construyó programas en Scratch que respondan a eventos y controlen personajes mediante bloques.
- Uso bucles y condicionales para evitar instrucciones repetitivas y permitir decisiones.
- Creo y modifico variables para almacenar datos como puntaje, tiempo, vidas o nivel.
- Pruebo mi proyecto, identifico errores y realizo cambios hasta que el resultado sea estable.
- Presentó un proyecto final funcional y explicó la lógica que utilice.

4. Conceptos esenciales antes de programar

Algoritmo

Es una serie de pasos ordenados para cumplir una meta. Una receta, las instrucciones de un juego o la ruta para llegar a un lugar pueden expresarse como algoritmos.

Programa

Es un conjunto de instrucciones que un computador puede ejecutar. En Scratch, esas instrucciones se construyen uniendo bloques como piezas de rompecabezas.

Evento

Es algo que inicia una acción. Por ejemplo: presionar la bandera verde, tocar una tecla o hacer clic sobre un personaje.

Secuencia

Es el orden en que se ejecutan las instrucciones. Cambiar el orden puede cambiar por completo el resultado.

Bucle

Permite repetir instrucciones. Sirve para crear movimientos, animaciones o acciones que deben realizarse varias veces.

Condicion

Es una pregunta que solo puede responderse con verdadero o falso. Permite que el programa decida que hacer.

Variable

Es un espacio con nombre que guarda un dato que puede cambiar, como puntos = 0, vidas = 3 o tiempo = 30.

Depuracion

Es el proceso de encontrar y corregir errores. Un programador prueba, observa, formula una hipótesis, cambia algo y vuelve a probar.

5. El mapa de Scratch

Scratch organiza sus bloques por categorías. No necesitas memorizar todos los bloques; necesitas saber que tipo de acción buscas y en qué categoría podría estar.

Categoria	¿Para que sirve?
Movimiento	Ubicar, girar y desplazar objetos.
Apariencia	Cambiar disfraces, tamaño, texto y efectos.
Sonido	Reproducir y controlar sonidos.
Eventos	Indicar cuando inicia una acción.
Control	Repetir, esperar y tomar decisiones.
Sensores	Detectar teclas, contactos, ratón y otras entradas.
Operadores	Comparar, calcular y combinar condiciones.
Variables	Guardar y modificar datos.
Mis bloques	Crear instrucciones propias para organizar el programa.

MISION 0: PIENSA COMO UNA MÁQUINA

Proposito	Tiempo	Producto	Modo
Comprender por que un algoritmo debe ser preciso.	25 min	Algoritmo escrito	Individual o parejas

Desafio

Un compañero será un robot que solo puede obedecer instrucciones exactas. Tu misión es escribir un algoritmo para que dibujes un cuadrado en una hoja sin ver tu ejemplo.

Pasos

1	**Define el resultado:** Escribe una frase que explique que debe aparecer al final: un cuadrado de aproximadamente 5 cm por lado.
2	**Escribe instrucciones:** Redacta entre 5 y 8 pasos. Evita palabras ambiguas como "por ahí", "un poquito" o "más o menos".
3	**Prueba con otra persona:** Entrega las instrucciones sin explicar nada de forma oral. La otra persona debe seguirlas literalmente.
4	**Detecta fallas:** Marca los pasos que causaron confusión o un resultado diferente al esperado.
5	**Mejora:** Reescribe el algoritmo con instrucciones más claras, medibles y ordenadas.

Piensa y responde

¿Qué paso fue el más difícil de explicar?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

¿Qué ocurre cuando una instrucción era ambigua?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

Escribe una regla para redactar buenas instrucciones.

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

MISIÓN 1: CONOCE TU LABORATORIO

Proposito	Tiempo	Producto	Modo
Reconocer la interfaz de Scratch y crear una primera respuesta a un evento.	45-60 min	Mini animación de saludo	Individual o parejas

Abre Scratch en el navegador o en la aplicación instalada por tu institución. Identifica el escenario, la lista de objetos (sprites), la paleta de bloques, el área de programación y las pestañas Código, Disfraces y Sonidos.

Construccion guiada

1	**Elige un personaje:** Conserva el gato o selecciona otro objeto de la biblioteca.
2	**Cambia el fondo:** Elige un escenario que combine con tu personaje.
3	**Crea el inicio:** Arrastra el evento "al presionar bandera verde".
4	**Haz que hable:** Conecta un bloque de Apariencia para decir un saludo durante 2 segundos.
5	**Agrega movimiento:** Conecta un bloque para mover 40 pasos y después girar 30 grados.
6	**Prueba:** Presiona la bandera verde al menos tres veces. Observa si siempre sucede lo mismo.
7	**Personaliza:** Cambia el saludo para que incluya el nombre del personaje y el reto que quiere cumplir.

Modelo de logica

```
al presionar bandera verde
   decir "Hola, soy Byte" durante 2 segundos
   mover 40 pasos
   girar 30 grados
```

Prueba rapida

¿Qué bloque indica el momento en que comienza el programa?

Reto extra

Haz que el personaje cambie de disfraz antes de moverse.

Antes de continuar, verifica:

- ☐ El proyecto inicia con la bandera verde.
- ☐ Mi personaje realiza al menos tres acciones.
- ☐ Puedo señalar donde estan escenario, objetos y area de programacion.

MISIÓN 2: HAZ QUE SE MUEVA CON PRECISIÓN

Proposito	Tiempo	Producto	Modo
Usar secuencias, coordenadas(x/y) y dirección para controlar una ruta.	60 min	Ruta animada con destino	Individual o parejas

Concepto: coordenadas

El escenario de Scratch usa un plano. El centro está cerca de x = 0, y = 0. x cambia de izquierda a derecha y (y) cambia de abajo hacia arriba.

Desafío: lleva a tu personaje por tres estaciones

1	**Prepara el escenario:** Elige un fondo y define tres lugares: Inicio, Punto A y Meta.
2	**Fija una salida:** Al presionar bandera verde, usa "ir a x: y:" para colocar siempre el personaje en el mismo punto.
3	**Primera ruta:** Usa "desplazar en ... segundos a x: y:" para llegar al Punto A.
4	**Segunda ruta:** Conecta otro deslizamiento para llegar a la Meta.
5	**Agrega información:** En cada punto, haz que el personaje diga el valor aproximado de x e y.
6	**Reinicia y prueba:** Ejecuta varias veces. Si el personaje no regresa al inicio, revisa el primer bloque de posición.
7	**Hazlo tuyo:** Agrega una cuarta estación o cambia la dirección del personaje durante el recorrido.

Experimento de coordenadas

Punto	x	y	¿Donde aparece?
A	-150	100	_____
B	150	100	_____
C	-150	-100	_____
D	150	-100	_____

Pista de depuracion

Si el objeto se mueve bien una vez pero comienza mal en la segunda prueba, probablemente falta un bloque que restablezca su posición inicial.

MISIÓN 3: REPITE CON INTELIGENCIA

Proposito	Tiempo	Producto	Modo
Aplicar bucles para reducir codigo repetido y crear animaciones fluidas.	60 min	Personaje animado	Individual o parejas

¿Por qué usar bucles?

Si una acción debe repetirse 10 veces, no conviene copiar 10 veces los mismos bloques. Un bucle hace el programa mas corto, ordenado y fácil de modificar.

Desafío: crea una caminata animada

1	**Escoge dos disfraces:** Usa un objeto que tenga por lo menos dos disfraces o agrega un segundo disfraz.
2	**Crea el evento:** Inicia con la bandera verde y coloca al personaje en una posición inicial.
3	**Agrega un bucle:** Usa "repetir 12".
4	**Dentro del bucle:** Mueve 10 pasos, cambia al siguiente disfraz y espera 0,1 o 0,2 segundos.
5	**Controla los bordes:** Agrega "si toca un borde, rebotar" si quieres que el personaje permanezca en pantalla.
6	**Compara:** Duplica el proyecto o toma nota: ¿que cambia si reemplazas repetir 12 por "por siempre"?
7	**Depura:** Si la animación es demasiado rápida, ajusta el tiempo de espera; si no se nota el disfraz, verifica que el objeto tenga más de uno.

```
al presionar bandera verde
   ir a x: -180 y: 0
   repetir 12
      mover 10 pasos
      siguiente disfraz
      esperar 0.15 segundos
```

Mini laboratorio

¿Qué valor de espera produce una animación más natural?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

¿Que diferencia hay entre repetir un número de veces y repetir por siempre?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

MISIÓN 4: HAZ QUE EL PROGRAMA DECIDA

Proposito	Tiempo	Producto	Modo
Combinar sensores, operadores y condicionales para responder al usuario.	75 min	Control con teclado y zonas interactivas	Individual o parejas

Condicional

Una estructura "si... entonces" ejecuta algo solo cuando una condicion es verdadera. La forma "si... entonces / si no" permite escoger entre dos caminos.

Desafío: controla al personaje con el teclado

1	**Crea el control continuo:** Usa un bloque "por siempre" para revisar las teclas durante todo el juego.
2	**Movimiento horizontal:** Dentro del bucle, agrega: si tecla flecha derecha presionada, cambiar x por 5. Repite para la flecha izquierda con -5.
3	**Movimiento vertical:** Agrega condiciones para flecha arriba (cambiar y por 5) y flecha abajo (cambiar y por -5).
4	**Crea una zona objetivo:** Agrega otro objeto o usa un color del escenario como meta.
5	**Detecta contacto:** Si el personaje toca la meta, haz que diga "¡Llegue!" y reproduzca un sonido.
6	**Agrega una regla:** Si toca un obstáculo, debe volver al punto de inicio.
7	**Prueba casos:** Prueba moverte en las cuatro direcciones, tocar la meta y tocar el obstáculo.

Combina condiciones

Los operadores permiten comparar y unir preguntas. Ejemplos: puntaje > 10; tecla espacio presionada Y tocando color rojo; vidas = 0.

AND / Y

Se cumple solo si las dos condiciones son verdaderas.

OR / O

Se cumple si al menos una condición es verdadera.

Reto extra

Crea una puerta que solo se abra cuando el personaje toque la puerta y tenga una llave. Puedes representar la llave con una variable que valga 0 o 1.

MISION 5: CONSTRUYE UN JUEGO CON PUNTAJE

Proposito	Tiempo	Producto	Modo
Crear y actualizar variables para registrar datos durante un juego.	90 min	Juego: Atrapa la estrella	Individual o parejas

Variable

Piensa en una variable como una caja con etiqueta. La caja "Puntaje" puede comenzar en 0 y aumentar cada vez que el jugador consigue un objeto.

Parte A. Prepara el jugador

1	**Crea el objeto jugador:** Elige un personaje y programa su movimiento con las flechas usando lo aprendido en la misión 4.
2	**Crea la estrella:** Agrega un segundo objeto pequeño que sera la meta que debes atrapar.
3	**Crea la variable:** En Variables, crea una variable llamada Puntaje para todos los objetos.
4	**Reinicia el dato:** Al presionar bandera verde, fija Puntaje en 0.

Parte B. Programa la estrella

1	**Movimiento inicial:** Al presionar bandera verde, envía la estrella a una posición al azar.
2	**Espera el contacto:** Usa un bucle por siempre.
3	**Suma puntos:** Si la estrella toca al jugador, cambia Puntaje por 1.
4	**Cambia de lugar:** Después de sumar, mueve la estrella a una nueva posición al azar.
5	**Evita dobles puntos:** Agrega una espera muy corta, por ejemplo 0,2 segundos, si observas que suma varias veces por un solo contacto.

Parte C. Agrega tiempo

1	**Crea Tiempo:** Crea una variable Tiempo y fijala en 30 al iniciar.
2	**Cuenta regresiva:** Repite 30 veces: espera 1 segundo y cambia Tiempo por -1.
3	**Fin del juego:** Cuando termine el conteo, envía un mensaje "fin".
4	**Detén el control:** Al recibir "fin", muestra el puntaje final y detiene todos los programas o bloquea el movimiento.

Meta minima

El juego debe permitir mover al jugador, aumentar Puntaje solo cuando hay contacto, cambiar la posición de la estrella y terminar después de un tiempo definido.

Reto de nivel avanzado

Agrega una variable Vidas. Si el jugador toca un enemigo, resta 1. Cuando Vidas = 0, termina el juego.

Misión 5: Construye un juego con puntaje

Grado 8.º
90 minutos
Individual o parejas

Tecnología e Informática — Programación en Scratch

¿Qué vamos a construir?

Estudiantes, en esta misión vamos a crear un videojuego llamado «Atrapa la estrella», utilizando Scratch.

Nuestro personaje se moverá por la pantalla utilizando las flechas del teclado. Cada vez que atrape una estrella, ganará un punto y la estrella aparecerá en otro lugar. Tendremos 30 segundos para conseguir la mayor cantidad de puntos posible.

Lekce Scratch
Chase Game Tutorial in Scratch : 4 Steps - Instructables
Using Variables and Data in Scratch | Envato Tuts+
6

Al terminar aprenderemos a utilizar las variables, que permiten guardar y actualizar información mientras funciona un programa.

Parte A. Prepara el jugador (25 minutos)

Paso 1. Crea tu proyecto

Ingresa a Scratch y crea un proyecto nuevo. Cambia su nombre por Atrapa la estrella.

Escoge el personaje que quieras controlar. Puede ser el gato de Scratch u otro personaje de la biblioteca.

Paso 2. Programa el movimiento

Selecciona al jugador y organiza los siguientes bloques:

Objeto: Jugador

Movimiento

Recuerda:

- Cambiar x mueve el personaje horizontalmente: derecha (+5) e izquierda (-5).
- Cambiar y lo mueve verticalmente: arriba (+5) y abajo (-5).
- Por siempre permite revisar continuamente si estás presionando una tecla.

Paso 3. Agrega la estrella

En la parte inferior derecha de Scratch, selecciona Elegir un objeto y busca una estrella (Star).

Hazla más pequeña que el jugador. Esta será la meta que debemos atrapar.

Paso 4. Crea la variable Puntaje

Selecciona la categoría naranja Variables, haz clic en Crear una variable y escribe `Puntaje`.

Marca Para todos los objetos. Deja visible la variable en el escenario.

En el jugador, crea este programa:

⚑ al hacer clic en bandera verde

fijar [Puntaje] a (0)

¿Por qué? Porque cada partida debe comenzar con cero puntos, sin importar cuántos puntos conseguimos anteriormente.

Parte B. Programa la estrella (25 minutos)

Paso 5. Ubica la estrella al azar

Selecciona la estrella y agrega:

⚑ al hacer clic en bandera verde

ir a [posición aleatoria]

Con este programa, la estrella aparecerá en diferentes lugares cada vez que comience el juego.

Paso 6. Detecta cuando el jugador atrapa la estrella

Debajo de los bloques anteriores, añade un ciclo por siempre y un bloque si... entonces.

Dentro de la condición, utiliza el bloque `¿tocando [Jugador]?` de la categoría Sensores.

Programa completo de la estrella

⚑ al hacer clic en bandera verde

ir a [posición aleatoria]

por siempre

si

¿Tocando [Jugador]?

entonces

cambiar [Puntaje] por (1)

ir a [posición aleatoria]

esperar (0.2) segundos

¿Qué sucede? Cuando el jugador toca la estrella, el puntaje aumenta en 1 y la estrella cambia de posición. La pequeña espera ayuda a evitar que se sumen varios puntos por un mismo contacto.

Parte C. Agrega el tiempo (25 minutos)

Paso 7. Crea la variable Tiempo

En Variables, crea otra variable llamada `Tiempo` para todos los objetos. Deja visible el contador en el escenario.

Selecciona el Escenario y construye el siguiente programa:

Programa del cronómetro

⚑ al hacer clic en bandera verde

fijar [Tiempo] a (30)

repetir (30)

esperar (1) segundos

cambiar [Tiempo] por (-1)

enviar [fin]

El juego comienza con 30 segundos. Cada segundo, el valor de Tiempo disminuye en 1 hasta llegar a 0.

Para crear el mensaje `fin`, utiliza el bloque enviar [mensaje1], selecciona Nuevo mensaje y escribe `fin`.

Paso 8. Programa el final del juego

Selecciona al jugador y crea un nuevo programa independiente:

al recibir [fin]

detener [otros programas en este objeto]

decir (unir [¡Fin! Puntaje:] (Puntaje)) durante (3) segundos

detener [todos]

En la estrella, crea también este programa, para impedir que se sigan sumando puntos después de terminar el tiempo:

al recibir [fin]

detener [otros programas en este objeto]

Así, cuando finaliza el cronómetro, el jugador deja de moverse, la estrella deja de sumar puntos y aparece un mensaje con el resultado final.

Parte D. Prueba tu juego (15 minutos)

Antes de entregar tu proyecto, comprueba que todo funciona correctamente.

Lista de verificación

0 de 6 pruebas

El jugador se mueve con las cuatro flechas del teclado.
El puntaje comienza en cero al presionar la bandera verde.
Solo se gana un punto cuando se atrapa una estrella.
La estrella aparece en una nueva posición al ser atrapada.
El tiempo comienza en 30 y disminuye hasta cero.
Al terminar el tiempo, el juego se detiene y muestra el puntaje final.

Reto final

Si ya terminaste, mejora tu videojuego agregando un sonido cada vez que atrapes la estrella, un fondo personalizado o un personaje diferente.

¿Cómo se evaluará tu misión?

Criterio	Puntos
El jugador se mueve correctamente	20
La variable Puntaje funciona	20
La estrella cambia de posición	20
El cronómetro funciona y finaliza el juego	20
El juego está organizado y personalizado	20
Total	100

Producto que debes entregar: un proyecto de Scratch llamado Atrapa la estrella, con el jugador, la estrella, el puntaje y el cronómetro funcionando correctamente.

Pregunta de reflexión: ¿Qué diferencia existe entre usar el bloque `fijar Puntaje a 0` y el bloque `cambiar Puntaje por 1`? Explica en tu cuaderno cuándo utilizarías cada uno.

Recuerda: programar no es solamente unir bloques, sino dar instrucciones claras para que un programa realice las acciones que imaginamos.

MISIÓN 6: COORDINA PERSONAJES Y ESCENAS

Proposito	Tiempo	Producto	Modo
Usar mensajes para sincronizar objetos y organizar una historia interactiva.	60-75 min	Historia de tres escenas	Individual o parejas

Mensaje

Un mensaje permite que un objeto avise a otros que algo ocurrio. Es una forma de coordinar acciones sin conectar todos los bloques en un solo programa.

Desafío: historia interactiva en tres escenas

1	**Planea:** Escribe una historia sencilla con inicio, problema y solución. Usa dos personajes como mínimo.
2	**Crea tres fondos:** Prepara un fondo para cada escena.
3	**Escena 1:** Con la bandera verde, muestra el fondo 1. Un personaje habla y luego envía el mensaje "escena2".
4	**Escena 2:** Al recibir "escena2", cambia el fondo. El segundo personaje responde y envía "escena3".
5	**Escena 3:** Al recibir "escena3", cambia el fondo y muestra el desenlace.
6	**Interactividad:** Antes de pasar a la escena 3, permite que el usuario presione una tecla o responda una pregunta.
7	**Prueba sincronización:** Ejecuta desde el inicio. Corrige personajes que aparecen antes de tiempo usando mostrar/ocultar.

```
al recibir
[escena2]
   cambiar fondo
a [Laboratorio]
   mostrar
   decir
"Tenemos un
problema"
durante 2
segundos
   enviar
[escena3]
```

Reto extra: clones

Si ya dominas las misiones anteriores, explora clones para crear copias temporales de un objeto, por ejemplo varias estrellas o enemigos. Primero logra que una copia funcione y después aumenta la cantidad.

7. Proyecto final: Laboratorio de creadores

Elige una de estas dos rutas. En ambas debes demostrar que comprendes la lógica del programa y que puedes explicar por que elegiste cada bloque importante.

Ruta A: Videojuego

Crea un juego con objetivo, controles, puntaje o vidas, una condición de victoria o final y retroalimentación para el jugador.

Ruta B: Historia interactiva

Crea una historia con al menos tres escenas, dos personajes, mensajes, decisiones del usuario y un desenlace que dependa de una acción.

Requisitos obligatorios

Tu proyecto final debe incluir:

- ☐ Un evento de inicio claro.
- ☐ Al menos un bucle.
- ☐ Al menos una condición.
- ☐ Al menos un sensor o entrada del usuario.
- ☐ Al menos una variable que tenga una función real en el proyecto.
- ☐ Al menos un mensaje entre objetos o escenas.
- ☐ Una mejora visual o sonora que ayude a comprender lo que ocurre.
- ☐ Pruebas realizadas y corrección de por lo menos dos errores.

Fase 1. Diseña antes de programar

Nombre del proyecto	_____
Objetivo para el usuario	_____
Personajes / objetos	_____
Reglas principales	_____
Variables que necesito	_____
Mensajes que necesito	_____

Fase 2. Escribe el algoritmo en lenguaje natural

1. _____
2. _____
3. _____
4. _____
5. _____
6. _____

Fase 3. Construye por capas

- Primero haz que funcione el movimiento o la secuencia principal.
- Después agrega reglas, puntaje o decisiones.
- Luego agrega mensajes, sonidos y detalles visuales.
- Al final prueba el inicio, el juego normal, los errores posibles y el final.

8. Taller de depuración: detective de errores

Cuando algo no funciona, evita borrar todo. Usa un proceso ordenado. El objetivo es encontrar la causa del problema con la menor cantidad de cambios posible.

Metodo P.R.O.B.A.R.

P = Pausa y observa. R = Reproduce el error. O = Observa que bloque se ejecuta. B = Busca una causa posible. A = Ajusta una sola cosa. R = Repite la prueba.

Errores frecuentes y pistas

Sintoma	Posible causa	Prueba / solucion
El personaje empieza en lugares distintos	Falta restablecer x/y al iniciar.	Agrega una posición inicial después de la bandera verde.
El puntaje no vuelve a cero	La variable conserva el valor de una prueba anterior.	Fija Puntaje en 0 al iniciar.
Suma demasiados puntos	El contacto se detecta varias veces mientras los objetos siguen tocándose.	Mueve la meta, espera un instante o controla el contacto.
Una escena aparece antes de tiempo	Un objeto no se oculta o responde a un evento incorrecto.	Revisa mostrar/ocultar y los mensajes recibidos.
El personaje no responde a las teclas	La comprobación de teclas ocurre una sola vez.	Coloca las condiciones dentro de un bucle por siempre.
El juego nunca termina	La condición de fin no se alcanza o no existe una acción al finalizar.	Revisa el contador y programa el mensaje o detención final.

Bitácora de errores del proyecto final

Error observado	Hipotesis	Cambio realizado	Resultado

9. Evaluación del aprendizaje

La evaluación combina el proceso y el producto. El proyecto no necesita ser el mas complejo: debe funcionar, mostrar comprensión y evidenciar mejoras realizadas a partir de las pruebas.

Rubrica del proyecto final

Criterio	4 - Avanzado	3 - Satisfactorio	2 - En proceso	1 - Inicial
Lógica y funcionamiento	El programa funciona de principio a fin y las reglas se cumplen.	Funciona casi siempre; presenta fallas menores.	Funciona parcialmente y requiere varias correcciones.	No hay una secuencia funcional todavía.
Uso de conceptos	Integra eventos, bucles, condiciones, sensores, variables y mensajes con sentido.	Integra la mayoría de los conceptos con sentido.	Usa algunos conceptos, pero varios no aportan a la solución.	Usa pocos conceptos o están desconectados.
Depuracion	Registra errores, explica causas y mejora el programa con pruebas.	Corrige errores y puede explicar algunas decisiones.	Corrige con ayuda y registra poca evidencia.	No demuestra un proceso de prueba y corrección.
Diseno y experiencia	La interacción es clara; imágenes, textos y sonidos apoyan el objetivo.	La interacción se comprende con pocos ajustes necesarios.	La experiencia es confusa en algunos momentos.	Es difícil comprender cómo usar el proyecto.

Criterio	4 - Avanzado	3 - Satisfactorio	2 - En proceso	1 - Inicial
Explicacion	Explica con claridad la lógica, las variables y una decisión técnica.	Explica la mayoría de elementos principales.	Explica parcialmente y necesita preguntas de apoyo.	No logra relacionar los bloques con el comportamiento del proyecto.

Autoevaluación: semaforo del programador

VERDE - Lo domino

Puedo hacerlo sin ayuda y explicarlo a otra persona.

AMARILLO - Estoy practicando

Lo hago con una pista o revisando un ejemplo.

ROJO - Necesito apoyo

Todavia no comprendo como iniciar o por que funciona.

Marca tu estado al finalizar

Eventos y secuencias: ☐ Verde ☐ Amarillo ☐ Rojo

Coordenadas y movimiento: ☐ Verde ☐ Amarillo ☐ Rojo

Bucles: ☐ Verde ☐ Amarillo ☐ Rojo

Condicionales y sensores: ☐ Verde ☐ Amarillo ☐ Rojo

Variables: ☐ Verde ☐ Amarillo ☐ Rojo

Mensajes: ☐ Verde ☐ Amarillo ☐ Rojo

Depuracion: ☐ Verde ☐ Amarillo ☐ Rojo

10. Cierre: explica lo que construiste

Completa estas preguntas antes de presentar tu proyecto. No basta con mostrar que funciona: un creador digital también puede explicar su solución.

1. ¿Cual era el problema o reto que querías resolver?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

2. ¿Que parte de tu programa se repite y que bucle usaste?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

3. ¿Que decisión toma tu programa y cual es la condición?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

4. ¿Que dato guarda tu variable y cuando cambia?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

5. ¿Para que usaste mensajes?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

6. Describe un error que encontraste y como lo corregiste.

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

7. Si tuvieras una hora mas, ¿que mejora agregarias?

EVIDENCIA / RESPUESTA DEL ESTUDIANTE

11. Orientaciones para el docente

Sugerencia de mediación: presentar cada misión con una demostración corta (5 a 8 minutos), permitir exploración guiada, detener el grupo para una puesta en común y cerrar con una evidencia breve. Evite resolver el proyecto completo frente al estudiante; modele la forma de pensar y de depurar.

Preguntas que ayudan sin dar la respuesta	Estrategia de apoyo
¿Qué esperabas que ocurriera? ¿Qué ocurrió realmente? ¿Qué bloque se ejecutó antes? ¿Qué valor tiene la variable en ese momento?	Trabaje con parejas temporales: una persona conduce el teclado y la otra explica la lógica. Cambien roles cada 10 minutos.

Evidencias sugeridas

Archivo .sb3 o enlace del proyecto, captura del escenario final, bitácora de errores, explicación oral de 2 minutos y rubrica diligenciada.

Adaptación y extensión

Para estudiantes que requieren apoyo, entregue coordenadas y estructuras base. Para quienes avanzan mas rápido, proponga niveles, clones, bloques personalizados, listas o un sistema de dificultad progresiva.

Lista de verificación docente

- ☐ El estudiante comprende que los bloques forman una secuencia ejecutable.
- ☐ El estudiante puede modificar un programa sin depender de copiar exactamente un ejemplo.

- ☐ El estudiante prueba y corrige errores de forma intencional.
- ☐ El producto final incluye interacción, datos y una regla de finalización.
- ☐ El estudiante explica al menos una decision de programacion con sus propias palabras.

12. Vocabulario rápido

Termino	Significado sencillo
Algoritmo	Pasos ordenados para resolver una tarea.
Bloque	Instruccion visual que se conecta con otras.
Objeto / sprite	Personaje u objeto programable.
Escenario	Área donde ocurre el proyecto.
Evento	Situación que inicia una acción.
Bucle	Estructura que repite instrucciones.
Condicion	Pregunta de verdadero o falso.
Sensor	Bloque que detecta entradas o situaciones.
Variable	Dato con nombre que puede cambiar.
Mensaje	Senal que coordina scripts u objetos.
Script	Grupo de bloques conectados.
Depurar	Encontrar y corregir errores.