Fundamentos de Programación con Scratch: Introducción al Pensamiento Computacional

Tecnología e Informática | Pensamiento Computacional | para estudiantes de secundaria (12-15 años) | 16 semanas

Descripción del Curso

Este curso de 16 semanas está diseñado para estudiantes de secundaria entre 12 y 15 años, con el propósito de introducirlos en el mundo de la programación mediante la plataforma visual Scratch. A lo largo del curso, se explorarán los conceptos fundamentales del pensamiento computacional, incluyendo la descomposición de problemas, el diseño de algoritmos, la abstracción y la depuración.

El curso está dirigido a estudiantes que desean desarrollar habilidades en lógica y resolución de problemas a través de actividades prácticas y proyectos creativos. Se promueve un enfoque activo y constructivista, donde los estudiantes aprenden haciendo, experimentando y colaborando en la creación de programas interactivos.

Al finalizar, los estudiantes serán capaces de diseñar y programar historias animadas, juegos y simulaciones simples en Scratch, aplicando conceptos básicos de programación y pensamiento lógico que servirán como base para futuros aprendizajes en tecnología y ciencias de la computación.

Objetivos Generales

- Identificar y explicar los elementos fundamentales del entorno Scratch y su función en la programación visual.
- Diseñar algoritmos simples para resolver problemas cotidianos utilizando bloques de programación.
- Construir proyectos interactivos que integren secuencias, ciclos, condiciones y variables básicas.
- Aplicar técnicas de depuración para mejorar la funcionalidad de los programas desarrollados.
- Comunicar de manera clara y creativa los procesos y resultados de sus proyectos de programación.

Competencias

- Comprender y aplicar los conceptos básicos de programación visual en Scratch.
- Desarrollar habilidades para descomponer problemas en partes manejables y diseñar soluciones algorítmicas.
- Crear proyectos interactivos que integren animaciones, sonidos y controles mediante bloques de código.
- Utilizar estrategias de depuración para identificar y corregir errores en programas.
- Colaborar y comunicar ideas de manera efectiva durante el desarrollo de proyectos de programación.

Requerimientos

- Conocimientos básicos de informática y manejo de computadora.

- Acceso a una computadora con conexión a internet para usar la plataforma Scratch (scratch.mit.edu).
- Cuenta gratuita en Scratch para guardar y compartir proyectos.
- Interés por la resolución de problemas y la creatividad digital.

Unidades del Curso

Unidad 1: Introducción a la programación y el pensamiento computacional

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar y explicar los conceptos básicos de programación y pensamiento computacional mediante ejemplos cotidianos.
- Al finalizar la unidad, el estudiante será capaz de describir la importancia del pensamiento computacional en la resolución de problemas diarios a través de discusiones y actividades guiadas.
- Al finalizar la unidad, el estudiante será capaz de reconocer y nombrar los elementos fundamentales del entorno Scratch en un ejercicio práctico.
- Al finalizar la unidad, el estudiante será capaz de elaborar algoritmos simples en forma de pseudocódigo o diagramas de flujo para resolver problemas básicos planteados en clase.
- Al finalizar la unidad, el estudiante será capaz de comunicar de manera clara las etapas del pensamiento computacional aplicadas en la creación de algoritmos simples durante presentaciones orales o escritas.

Contenidos Temáticos

1. Conceptos básicos de programación y pensamiento computacional

- Definición de programación: qué es y para qué sirve.
- Introducción al pensamiento computacional: definición y componentes (descomposición, reconocimiento de patrones, abstracción, diseño de algoritmos).
- Ejemplos cotidianos donde aplicamos el pensamiento computacional (organizar tareas, seguir una receta, instrucciones para armar algo).

2. Importancia del pensamiento computacional en la vida diaria

- Cómo el pensamiento computacional ayuda a resolver problemas cotidianos.
- Discusión sobre situaciones comunes que requieren pensamiento lógico y estructurado.
- Beneficios de desarrollar habilidades de pensamiento computacional para el aprendizaje y la vida diaria.

3. Introducción al entorno Scratch

- Descripción general del entorno Scratch: escenario, sprites, bloques, área de scripts.
- Elementos fundamentales: bloques de movimiento, apariencia, control y eventos.

- Ejercicio práctico para identificar y nombrar estos elementos en la plataforma.

4. Elaboración de algoritmos simples

- Definición de algoritmo y su importancia.
- Formas de representar algoritmos: pseudocódigo y diagramas de flujo.
- Construcción de algoritmos simples para resolver problemas básicos (ejemplo: instrucciones para preparar un sándwich o indicar pasos para llegar a un lugar).

5. Comunicación de las etapas del pensamiento computacional

- Identificación y explicación de las etapas aplicadas al crear algoritmos.
- Presentación oral y escrita de los procesos seguidos para resolver problemas mediante algoritmos.
- Uso de vocabulario técnico y ejemplos claros para comunicar ideas.

Actividades

Actividad 1: Identificación de pensamiento computacional en la vida diaria

Objetivo: Identificar y explicar los conceptos básicos de programación y pensamiento computacional mediante ejemplos cotidianos.

Descripción:

- El docente presenta ejemplos comunes (como seguir una receta o armar un mueble) y pregunta a los estudiantes cómo organizarían esos pasos.
- En grupos pequeños, los estudiantes analizan una situación diaria y enumeran los pasos para resolverla, identificando patrones y descomponiendo el problema.
- Se comparte en plenaria para discutir y relacionar con los conceptos de pensamiento computacional.

Organización: Grupos de 3-4 estudiantes.

Producto esperado: Lista escrita de pasos y explicación de cómo aplicaron el pensamiento computacional.

Duración estimada: 45 minutos.

Actividad 2: Explorando el entorno Scratch

Objetivo: Reconocer y nombrar los elementos fundamentales del entorno Scratch en un ejercicio práctico.

Descripción:

- El docente muestra la interfaz de Scratch proyectada o en computadoras individuales.
- Los estudiantes exploran los principales elementos: escenario, sprites, bloques de código, áreas de trabajo.
- Realizan un cuestionario guiado para identificar y nombrar cada parte.

Organización: Individual o parejas.

Producto esperado: Cuestionario completado con las respuestas sobre elementos de Scratch.

Duración estimada: 40 minutos.

Actividad 3: Creación de un algoritmo simple en pseudocódigo y diagrama de flujo

Objetivo: Elaborar algoritmos simples en forma de pseudocódigo o diagramas de flujo para resolver problemas básicos planteados en clase.

Descripción:

- El docente plantea un problema sencillo (por ejemplo: cómo ordenar libros en una estantería por tamaño).
- Los estudiantes elaboran un pseudocódigo que describa los pasos para resolver el problema.
- Luego, crean un diagrama de flujo que represente ese pseudocódigo.
- Los alumnos presentan sus algoritmos y diagramas al grupo explicando el proceso.

Organización: Individual o parejas.

Producto esperado: Documento con pseudocódigo y diagrama de flujo, presentación oral breve.

Duración estimada: 1 hora.

Actividad 4: Presentación de las etapas del pensamiento computacional aplicadas

Objetivo: Comunicar de manera clara las etapas del pensamiento computacional aplicadas en la creación de algoritmos simples durante presentaciones orales o escritas.

Descripción:

- Cada estudiante o pareja selecciona el algoritmo que creó en la actividad anterior.
- Prepara una presentación escrita y oral donde explique las etapas del pensamiento computacional que aplicó (descomposición, reconocimiento de patrones, abstracción, diseño de algoritmos).
- Se realiza una ronda de presentaciones donde los compañeros pueden hacer preguntas.

Organización: Individual o parejas.

Producto esperado: Presentación oral y escrito explicativo.

Duración estimada: 50 minutos.

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre programación, pensamiento computacional y familiaridad con Scratch.

Cómo se evalúa: Mediante una encuesta corta y preguntas orales al inicio de la unidad.

Instrumento sugerido: Cuestionario simple con preguntas abiertas y de opción múltiple.

Evaluación formativa

Qué se evalúa: Progreso en la identificación de conceptos, uso del entorno Scratch, creación y comunicación de algoritmos.

Cómo se evalúa: Observación durante actividades, revisión de productos (listas de pasos, cuestionarios, pseudocódigos, diagramas, presentaciones).

Instrumento sugerido: Rúbrica para evaluar claridad, corrección y aplicación de conceptos en actividades prácticas.

Evaluación sumativa

Qué se evalúa: Capacidad para identificar y explicar conceptos básicos, describir importancia del pensamiento computacional, reconocer elementos de Scratch, elaborar algoritmos simples y comunicar etapas del pensamiento computacional.

Cómo se evalúa: Examen escrito con preguntas teóricas y prácticas; presentación oral individual o en parejas; entrega final de pseudocódigo y diagramas de flujo.

Instrumento sugerido: Prueba escrita, rúbrica para presentación oral y revisión de productos escritos.

Unidad 2: Conociendo el entorno Scratch

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar los componentes principales de la interfaz de Scratch, describiendo su función y ubicación en el entorno.
- Al finalizar la unidad, el estudiante será capaz de crear una cuenta en Scratch siguiendo los pasos indicados para comenzar a programar de manera segura.
- Al finalizar la unidad, el estudiante será capaz de navegar por las secciones básicas del entorno Scratch para acceder a bloques de programación y recursos multimedia.
- Al finalizar la unidad, el estudiante será capaz de explicar la utilidad de los diferentes tipos de bloques en Scratch, relacionándolos con su función en la creación de algoritmos simples.

Contenidos Temáticos

1. Introducción a Scratch y su entorno

- ¿Qué es Scratch? Presentación y propósito de la plataforma como herramienta educativa para aprender programación mediante bloques visuales.
- Beneficios de usar Scratch para el desarrollo del pensamiento computacional.

2. Creación de una cuenta en Scratch

- Importancia de crear una cuenta para guardar proyectos y colaborar.
- Pasos para crear una cuenta segura: acceso al sitio web, formulario de registro, elección de usuario y contraseña.
- Configuración básica de perfil y recomendaciones para la seguridad y privacidad del estudiante.

3. Exploración de la interfaz de Scratch

- Panel de escenario: ubicación, función y cómo visualizar la ejecución del proyecto.

- Área de bloques: descripción de las categorías principales de bloques y su organización.
- Zona de scripts o área de programación: espacio donde se ensamblan los bloques para crear algoritmos.
- Panel de sprites y objetos: manejo de personajes y objetos, agregar, eliminar y seleccionar.
- Barra de menús y herramientas: funciones para guardar, cargar proyectos y acceder a ayuda.

4. Navegación por las secciones básicas del entorno Scratch

- Cómo acceder a las diferentes categorías de bloques (Movimiento, Apariencia, Sonido, Eventos, Control, Sensores, Operadores y Variables).
- Uso del buscador y recursos multimedia: agregar sonidos, fondos e imágenes.
- Exploración de ejemplos y proyectos compartidos para inspirarse.

5. Tipos de bloques en Scratch y su utilidad básica

- Bloques de movimiento: instrucciones para desplazar y girar sprites.
- Bloques de apariencia: cambiar disfraces, mostrar y ocultar objetos.
- Bloques de sonido: reproducir sonidos y efectos.
- Bloques de eventos: iniciar acciones con clics, teclas o señales.
- Bloques de control: estructuras de repetición y condiciones.
- Bloques de sensores: interacción con el entorno y otros objetos.
- Bloques de operadores: realizar cálculos y operaciones lógicas.
- Bloques de variables: almacenar y modificar información.
- Relación entre tipos de bloques y creación de algoritmos simples para resolver problemas básicos.

Actividades

Actividad 1: Creando mi cuenta en Scratch

Objetivo: El estudiante será capaz de crear una cuenta en Scratch siguiendo los pasos indicados para comenzar a programar de manera segura.

Descripción:

- El docente guiará a los estudiantes para ingresar a la página oficial de Scratch.
- Se mostrará cómo acceder al formulario de registro y llenar los datos solicitados.
- Los estudiantes crearán su cuenta siguiendo las indicaciones, eligiendo un nombre de usuario seguro y una contraseña adecuada.
- Configurarán opciones básicas de perfil y revisarán recomendaciones para mantener su cuenta segura.

Organización: Individual

Producto esperado: Cuenta creada en Scratch y captura de pantalla del perfil configurado.

Duración estimada: 45 minutos

Actividad 2: Explorando la interfaz de Scratch

Objetivo: El estudiante será capaz de identificar los componentes principales de la interfaz de Scratch, describiendo su función y ubicación en el entorno.

Descripción:

- El docente realizará una presentación en vivo o con diapositivas sobre la interfaz de Scratch.
- Los estudiantes abrirán el editor de Scratch y explorarán cada componente señalado (escenario, bloques, sprites, menú).
- Se realizará una actividad práctica donde los estudiantes deberán identificar y nombrar cada zona en una captura de pantalla de la interfaz.

Organización: Individual o en parejas

Producto esperado: Imagen anotada con las zonas principales de la interfaz y una breve descripción escrita.

Duración estimada: 1 hora

Actividad 3: Navegando y clasificando bloques en Scratch

Objetivo: El estudiante será capaz de navegar por las secciones básicas del entorno Scratch para acceder a bloques de programación y recursos multimedia, y explicará la utilidad de los diferentes tipos de bloques.

Descripción:

- El docente mostrará las categorías de bloques y explicará brevemente la función de cada tipo.
- Los estudiantes abrirán el editor y explorarán cada categoría, arrastrando bloques a la zona de scripts para observar su función.
- Deberán completar una tabla clasificando bloques según su tipo y describiendo una posible utilidad o ejemplo de uso.

Organización: Individual

Producto esperado: Tabla con clasificación y descripción de bloques explorados.

Duración estimada: 1 hora 15 minutos

Actividad 4: Primer algoritmo simple con bloques de Scratch

Objetivo: El estudiante será capaz de explicar la utilidad de los diferentes tipos de bloques en Scratch, relacionándolos con su función en la creación de algoritmos simples.

Descripción:

- El docente propondrá un pequeño reto: crear un programa que mueva un sprite, cambie su apariencia y reproduzca un sonido al hacer clic.
- Los estudiantes seleccionarán los bloques adecuados para cumplir el reto, integrando bloques de movimiento, apariencia, sonido y eventos.
- Probarán el proyecto y harán ajustes si es necesario.

- Finalmente, presentarán ante el grupo su proyecto explicando qué bloques usaron y para qué.

Organización: Individual o en parejas

Producto esperado: Proyecto funcional en Scratch y explicación oral o escrita del uso de bloques.

Duración estimada: 1 hora 30 minutos

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimiento previo sobre programación y familiaridad con plataformas digitales.

Cómo se evalúa: Preguntas orales o escritas sobre qué saben de programación y si han usado Scratch u otro entorno visual.

Instrumento sugerido: Cuestionario breve o lluvia de ideas en clase.

Evaluación formativa

Qué se evalúa: Progreso en la creación de cuenta, identificación de la interfaz, navegación por bloques y comprensión de su función.

Cómo se evalúa: Revisión de productos de las actividades (capturas de pantalla, tablas, proyectos) y observación del desempeño durante actividades prácticas.

Instrumento sugerido: Lista de cotejo para verificar cumplimiento de pasos en creación de cuenta, correcta identificación de interfaz y clasificación de bloques; rúbrica para evaluación del proyecto simple.

Evaluación sumativa

Qué se evalúa: Capacidad para identificar componentes de la interfaz, crear cuenta segura, navegar en el entorno y explicar el uso de bloques mediante la construcción de un algoritmo simple.

Cómo se evalúa: Presentación final de un proyecto sencillo en Scratch que integre varios tipos de bloques con explicación escrita u oral de su función.

Instrumento sugerido: Rúbrica de evaluación que considere funcionalidad del proyecto, diversidad de bloques usados y claridad en la explicación.

Unidad 3: Bloques básicos y secuencias

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar y describir los bloques básicos de movimiento, apariencia y sonido en Scratch para entender su función en la programación visual.
- Al finalizar la unidad, el estudiante será capaz de diseñar secuencias simples utilizando bloques de movimiento, apariencia y sonido para crear proyectos interactivos básicos.

- Al finalizar la unidad, el estudiante será capaz de construir y ejecutar un proyecto en Scratch que integre bloques básicos en una secuencia lógica para resolver un problema cotidiano.
- Al finalizar la unidad, el estudiante será capaz de analizar y corregir errores en las secuencias creadas para mejorar la funcionalidad del proyecto utilizando técnicas básicas de depuración.
- Al finalizar la unidad, el estudiante será capaz de comunicar de forma clara y creativa el proceso de creación de sus secuencias y los resultados obtenidos en su proyecto de Scratch.

Contenidos Temáticos

1. Introducción a los bloques básicos en Scratch

- **Descripción:** Se explicará qué son los bloques en Scratch, su importancia y cómo se utilizan para programar visualmente.
- **Subtemas:**
  - Concepto de bloques de programación en Scratch.
  - Categorías principales: Movimiento, Apariencia y Sonido.
  - Interfaz de Scratch: ubicación y uso de bloques básicos.

2. Bloques de movimiento

- **Descripción:** Se explorarán los bloques que permiten mover y posicionar los personajes o "sprites" en el escenario.
- **Subtemas:**
  - Bloques para mover (avanzar, retroceder, girar).
  - Bloques para posicionar (ir a posición x,y, ir a un objeto).
  - Bloques para cambiar dirección y orientación.

3. Bloques de apariencia

- **Descripción:** Se estudiarán los bloques que modifican la apariencia de los sprites para hacer la programación más visual e interactiva.
- **Subtemas:**
  - Cambiar disfraces.
  - Mostrar y ocultar sprites.
  - Decir o pensar mensajes.
  - Cambiar efectos visuales (color, brillo, etc.).

4. Bloques de sonido

- **Descripción:** Se analizarán los bloques que permiten agregar sonidos y música para enriquecer los proyectos.
- **Subtemas:**

- Reproducir sonidos predefinidos.
- Detener sonidos.
- Control de volumen y efectos.

5. Diseño y construcción de secuencias simples con bloques básicos

- **Descripción:** Se enseñará cómo combinar bloques de movimiento, apariencia y sonido para crear secuencias sencillas que den vida a proyectos interactivos.
- **Subtemas:**
  - Concepto de secuencia en programación.
  - Cómo ordenar bloques para lograr acciones lógicas.
  - Ejemplos básicos de secuencias (por ejemplo, un sprite que se mueve, cambia de disfraz y emite un sonido).

6. Construcción y ejecución de proyectos integrados

- **Descripción:** Aplicación práctica para crear un proyecto que integre los bloques básicos en una secuencia orientada a resolver un problema cotidiano o contar una historia.
- **Subtemas:**
  - Planificación del proyecto.
  - Construcción paso a paso del proyecto en Scratch.
  - Pruebas y ejecución del proyecto.

7. Análisis y corrección de errores (depuración básica)

- **Descripción:** Se aprenderá a identificar errores comunes en las secuencias y aplicar técnicas básicas para corregirlos y mejorar el proyecto.
- **Subtemas:**
  - Tipos de errores comunes en Scratch (bloques mal ordenados, faltantes o mal configurados).
  - Técnicas para revisar y probar el código.
  - Uso de la función "detener todo" y la ejecución paso a paso para detectar fallos.

8. Comunicación creativa del proceso y resultados

- **Descripción:** Estrategias para que el estudiante explique y comparta de manera clara y creativa el desarrollo y el resultado de su proyecto.
- **Subtemas:**
  - Presentación oral o escrita del proyecto.
  - Uso de capturas de pantalla o grabaciones para evidenciar el proceso.
  - Reflexión sobre los retos enfrentados y aprendizajes obtenidos.

Actividades

Actividad 1: Explorando los bloques básicos de Scratch

Objetivo: Identificar y describir los bloques básicos de movimiento, apariencia y sonido en Scratch.

Descripción paso a paso:

- El docente presenta la interfaz de Scratch y muestra dónde encontrar los bloques de movimiento, apariencia y sonido.
- Los estudiantes, de forma individual, exploran cada tipo de bloque creando pequeñas acciones con un sprite (por ejemplo, moverlo, cambiar su disfraz, emitir un sonido).
- Realizan una tabla donde anotan el nombre del bloque, su función y un ejemplo de uso.

Organización: Individual

Producto esperado: Tabla de bloques con descripción y ejemplo básico en Scratch.

Duración estimada: 50 minutos

Actividad 2: Creando una secuencia simple

Objetivo: Diseñar secuencias simples utilizando bloques de movimiento, apariencia y sonido para crear proyectos interactivos básicos.

Descripción paso a paso:

- En parejas, los estudiantes planifican una secuencia que involucre mover un sprite, cambiar su disfraz y reproducir un sonido.
- Construyen la secuencia en Scratch y la prueban para verificar que funciona correctamente.
- Comparten su proyecto con otra pareja para recibir retroalimentación.

Organización: Parejas

Producto esperado: Proyecto Scratch con secuencia simple funcionando.

Duración estimada: 60 minutos

Actividad 3: Proyecto integrado para resolver un problema cotidiano

Objetivo: Construir y ejecutar un proyecto en Scratch que integre bloques básicos en una secuencia lógica para resolver un problema cotidiano.

Descripción paso a paso:

- El docente plantea un problema cotidiano sencillo (por ejemplo, crear una animación que explique cómo reciclar).
- En grupos pequeños, los estudiantes diseñan el proyecto, definiendo qué bloques usarán y en qué orden.
- Construyen el proyecto en Scratch integrando movimiento, apariencia y sonido.
- Ejecutan y prueban el proyecto, haciendo ajustes necesarios.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Proyecto Scratch completo que resuelva el problema planteado.

Duración estimada: 2 sesiones de 50 minutos

Actividad 4: Depuración y presentación del proyecto

Objetivo: Analizar y corregir errores en las secuencias creadas y comunicar creativamente el proceso y resultados.

Descripción paso a paso:

- Cada grupo revisa su proyecto buscando posibles errores o mejoras en la secuencia.
- Aplica técnicas básicas de depuración para corregir fallas.
- Prepara una presentación donde expliquen el proceso de creación, los desafíos encontrados y cómo los solucionaron.
- Presentan su proyecto y reflexión al resto de la clase.

Organización: Grupos

Producto esperado: Proyecto corregido y presentación oral o multimedia.

Duración estimada: 1 sesión de 50 minutos

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre programación visual y familiaridad con Scratch.

Cómo se evalúa: Mediante una breve actividad práctica donde el estudiante identifica bloques básicos y sus funciones en Scratch.

Instrumento sugerido: Cuestionario interactivo en Scratch o cuestionario escrito con imágenes de bloques para identificar.

Evaluación formativa

Qué se evalúa: Diseño y construcción de secuencias simples, capacidad para integrar bloques básicos, y habilidades de depuración.

Cómo se evalúa: Observación directa durante las actividades, revisión de proyectos en Scratch, retroalimentación entre pares y autoevaluación guiada.

Instrumento sugerido: Lista de cotejo para observar uso correcto de bloques, orden lógico de secuencias y aplicación de correcciones.

Evaluación sumativa

Qué se evalúa: Capacidad para construir un proyecto integrado con bloques básicos, corregir errores y comunicar el proceso y resultados.

Cómo se evalúa: Presentación final del proyecto y entrega del archivo Scratch con secuencia funcional, acompañados de una explicación oral o escrita.

Instrumento sugerido: Rúbrica que valore la integración de bloques, la lógica de la secuencia, la calidad de la depuración y la claridad en la comunicación.

Unidad 4: Control de flujo: ciclos y eventos

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar y explicar el funcionamiento de los bloques de control de ciclos y eventos en Scratch mediante ejemplos prácticos.
- Al finalizar la unidad, el estudiante será capaz de diseñar algoritmos que utilicen ciclos para repetir acciones específicas en proyectos de Scratch, aplicando lógica secuencial y condicional.
- Al finalizar la unidad, el estudiante será capaz de construir proyectos interactivos en Scratch que respondan a eventos mediante el uso adecuado de bloques de control de eventos.
- Al finalizar la unidad, el estudiante será capaz de depurar programas que incluyan ciclos y eventos, identificando y corrigiendo errores para mejorar la funcionalidad del proyecto.
- Al finalizar la unidad, el estudiante será capaz de comunicar de forma clara y creativa el proceso de implementación de ciclos y eventos en sus proyectos, utilizando términos técnicos apropiados del entorno Scratch.

Contenidos Temáticos

1. Introducción al control de flujo en Scratch

- Concepto de control de flujo: explicación sencilla sobre cómo se controla la secuencia y repetición de acciones en un programa.
- Importancia de los ciclos y eventos para la interacción y automatización en proyectos de Scratch.
- Visión general de los bloques de control disponibles en Scratch.

2. Bloques de ciclos en Scratch

- Bloque "repetir (n) veces": estructura y uso para repetir un conjunto de instrucciones un número definido de veces.
- Bloque "repetir hasta que": ciclo basado en una condición que detiene la repetición al cumplirse.
- Bloque "por siempre": ciclo infinito que ejecuta continuamente las instrucciones dentro de él.
- Ejemplos prácticos de uso de ciclos para animaciones y movimientos repetitivos.

3. Bloques de eventos en Scratch

- Concepto de evento: definición y su papel en la programación interactiva.
- Bloque "Al presionar bandera verde": inicio del programa y su función en proyectos Scratch.
- Bloques "Al presionar tecla" y "Al hacer clic en sprite": respuestas a eventos de usuario.
- Otros bloques de eventos: recibir mensajes, sensores, y eventos de control.
- Ejemplos de proyectos que responden a eventos para interacción con el usuario.

4. Diseño de algoritmos con ciclos y eventos

- Definición y diseño de algoritmos simples que incluyan ciclos para repetir acciones.
- Incorporación de condiciones dentro de ciclos para controlar la ejecución (lógica condicional).
- Integración de eventos para iniciar o modificar el comportamiento del algoritmo.
- Diagramas de flujo básicos para representar ciclos y eventos en algoritmos.

5. Construcción de proyectos interactivos usando ciclos y eventos

- Planificación y armado de un proyecto en Scratch que utilice ciclos para acciones repetitivas.
- Incorporación de eventos para respuestas a acciones del usuario o del entorno.
- Uso combinado de ciclos y eventos para mejorar la interactividad y dinamismo del proyecto.

6. Depuración de programas con ciclos y eventos

- Identificación de errores comunes en el uso de ciclos: bucles infinitos, condiciones incorrectas, etc.
- Detección de problemas en eventos que no se activan o se activan en momento incorrecto.
- Estrategias para corregir errores y mejorar la funcionalidad de los programas.
- Pruebas y verificación de proyectos para asegurar que ciclos y eventos funcionen adecuadamente.

7. Comunicación y documentación del proceso de implementación

- Uso de terminología técnica apropiada de Scratch para describir ciclos y eventos.
- Presentación clara y creativa del proceso de diseño y construcción del proyecto.
- Elaboración de informes o exposiciones donde se expliquen los algoritmos y bloques usados.
- Compartir aprendizajes y dificultades encontradas en el desarrollo del proyecto.

Actividades

Actividad 1: Explorando y explicando bloques de ciclos y eventos

Objetivo: Identificar y explicar el funcionamiento de los bloques de control de ciclos y eventos en Scratch mediante ejemplos prácticos.

Descripción:

- El docente presenta una breve demostración de los bloques "repetir (n) veces", "repetir hasta que", "por siempre" y eventos básicos como "al presionar bandera verde" y "al presionar tecla".
- Los estudiantes abren Scratch y exploran cada bloque creando pequeños scripts que muestren su funcionamiento.
- Cada estudiante escribe una breve explicación, con sus palabras, de cómo funciona cada bloque y crea un ejemplo sencillo en Scratch.
- Los estudiantes comparten sus explicaciones y ejemplos con la clase para discusión y retroalimentación.

Organización: Individual

Producto esperado: Documento breve con explicaciones escritas y capturas o enlaces a proyectos de Scratch.

Duración estimada: 1 hora

Actividad 2: Diseño y programación de un juego simple con ciclos y eventos

Objetivo: Diseñar algoritmos y construir proyectos que utilicen ciclos y eventos para repetir acciones y responder a interacciones.

Descripción:

- En grupos, los estudiantes diseñan un juego sencillo (por ejemplo, un laberinto, un juego de atrapar objetos o un dibujo interactivo) que requiera movimientos repetidos y respuesta a eventos de teclado o clic.
- El grupo desarrolla el algoritmo en papel o herramienta digital, identificando dónde usarán ciclos y eventos.
- Programan el proyecto en Scratch aplicando los bloques de control aprendidos.
- Prueban y ajustan el proyecto para asegurar el correcto funcionamiento de ciclos y eventos.
- Preparan una breve presentación para explicar su proyecto, el uso de ciclos y eventos, y las dificultades superadas.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Proyecto funcional en Scratch y presentación oral o escrita.

Duración estimada: 3 horas (puede distribuirse en varias sesiones)

Actividad 3: Depuración colaborativa de un proyecto con errores en ciclos y eventos

Objetivo: Identificar y corregir errores en proyectos que incluyen ciclos y eventos para mejorar su funcionalidad.

Descripción:

- El docente proporciona a cada grupo un proyecto Scratch con errores intencionales relacionados con ciclos (por ejemplo, bucles infinitos, condiciones mal formuladas) y eventos (eventos que no se activan o activan incorrectamente).
- Los estudiantes analizan el código, detectan los errores y discuten posibles soluciones.
- Realizan las correcciones necesarias y prueban el proyecto para verificar su correcto funcionamiento.
- Comparten con el grupo clase los errores encontrados y cómo fueron corregidos.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Proyecto corregido y reporte de errores y soluciones.

Duración estimada: 2 horas

Actividad 4: Presentación y documentación del proyecto final

Objetivo: Comunicar de forma clara y creativa el proceso de implementación de ciclos y eventos en sus proyectos, usando términos técnicos.

Descripción:

- Cada grupo prepara una presentación escrita o audiovisual donde expliquen el diseño del proyecto, el uso de ciclos y eventos, y los algoritmos implementados.
- Incluyen términos técnicos correctamente empleados y ejemplos de los bloques utilizados.

- Presentan su trabajo a la clase, respondiendo preguntas y comentarios.
- El docente y compañeros ofrecen retroalimentación constructiva.

Organización: Grupos

Producto esperado: Presentación clara y documentada del proyecto con terminología técnica adecuada.

Duración estimada: 1.5 horas

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre secuencias, repeticiones y eventos básicos en programación.

Cómo se evalúa: Cuestionario corto y discusión grupal para identificar el nivel inicial de comprensión.

Instrumento sugerido: Cuestionario de opción múltiple y preguntas abiertas; observación de participación en discusión.

Evaluación formativa

Qué se evalúa: Progreso en la comprensión y aplicación de bloques de ciclos y eventos, diseño de algoritmos, depuración y documentación.

Cómo se evalúa: Revisión continua de actividades prácticas, retroalimentación durante el trabajo en grupo y monitoreo de proyectos Scratch.

Instrumento sugerido: Rúbrica para evaluar proyectos y explicaciones parciales; listas de cotejo para seguimiento de tareas; observación directa del trabajo colaborativo.

Evaluación sumativa

Qué se evalúa: Capacidad para identificar, diseñar, construir, depurar y comunicar proyectos que usen ciclos y eventos en Scratch.

Cómo se evalúa: Presentación final del proyecto, entrega del código funcional, documentación escrita y demostración oral.

Instrumento sugerido: Rúbrica detallada que considere: funcionalidad del proyecto (uso correcto de ciclos y eventos), calidad del algoritmo, capacidad de depuración y claridad en la comunicación técnica.

Unidad 5: Condicionales y toma de decisiones

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar y explicar el funcionamiento de las estructuras condicionales en Scratch mediante ejemplos prácticos.
- Al finalizar la unidad, el estudiante será capaz de diseñar y construir programas en Scratch que utilicen condicionales para tomar decisiones basadas en diferentes condiciones.

- Al finalizar la unidad, el estudiante será capaz de aplicar condicionales compuestas para resolver problemas sencillos mediante la combinación de bloques lógicos en Scratch.
- Al finalizar la unidad, el estudiante será capaz de depurar programas que contengan estructuras condicionales para corregir errores y mejorar su funcionalidad.
- Al finalizar la unidad, el estudiante será capaz de comunicar de forma clara y creativa el proceso de toma de decisiones implementado en sus proyectos utilizando condicionales en Scratch.

Contenidos Temáticos

1. Introducción a las estructuras condicionales en Scratch

- ¿Qué es una estructura condicional?
  Descripción: Se explicará el concepto básico de condicionales como mecanismos para tomar decisiones en programación.
- Bloques condicionales en Scratch: "si", "si-sino"

  Descripción: Presentación de los bloques "si" y "si-sino" en Scratch y su funcionamiento básico.
- Ejemplos prácticos básicos de condicionales

  Descripción: Análisis y ejecución de pequeños programas en Scratch que usan condicionales para cambiar comportamiento según condiciones.

2. Diseño y construcción de programas con condicionales simples

- Uso de condicionales para tomar decisiones basadas en valores numéricos o eventos
  Descripción: Cómo usar condicionales para responder a comparaciones numéricas y eventos como presionar teclas o detectar colisiones.
- Implementación paso a paso de programas sencillos con condicionales

  Descripción: Guías para diseñar y construir proyectos Scratch que respondan a condiciones específicas.
- Práctica de modificación y extensión de proyectos existentes con condicionales

  Descripción: Ejercicios para modificar proyectos para incluir decisiones condicionales adicionales.

3. Condicionales compuestas y operadores lógicos en Scratch

- Concepto de condicional compuesta

  Descripción: Explicación de cómo combinar condiciones usando operadores lógicos (y, o, no) para tomar decisiones más complejas.

- Bloques lógicos en Scratch y su uso en condicionales

  Descripción: Introducción a los bloques de operadores lógicos y cómo se integran en condicionales.
- Ejemplos de programas con condicionales compuestas

  Descripción: Análisis y creación de proyectos que requieren múltiples condiciones para la toma de decisiones.

4. Depuración de programas con estructuras condicionales

- Errores comunes en condicionales y cómo identificarlos

  Descripción: Revisión de errores típicos como condiciones mal planteadas, bloques mal ubicados o falta de condiciones.
- Estrategias para depurar y corregir programas en Scratch

  Descripción: Técnicas para probar, detectar y corregir errores en condicionales usando el entorno de Scratch.
- Prácticas guiadas de depuración en proyectos con condicionales

  Descripción: Ejercicios para corregir errores en programas dados y mejorar su funcionalidad.

5. Comunicación y presentación de proyectos con condicionales

- Cómo explicar la lógica de decisiones en un programa Scratch

  Descripción: Técnicas para describir claramente el proceso de toma de decisiones implementado en proyectos.
- Uso de recursos visuales y narrativos para comunicar proyectos

  Descripción: Cómo utilizar presentaciones, demostraciones y documentación para compartir el trabajo realizado.
- Presentación creativa de proyectos finales

  Descripción: Propuestas para que los estudiantes muestren sus proyectos usando condicionales y expliquen su funcionamiento.

Actividades

Actividad 1: Explorando condicionales básicos en Scratch

Objetivo: Identificar y explicar el funcionamiento de las estructuras condicionales en Scratch mediante ejemplos prácticos.

Descripción paso a paso:

- El docente presenta un proyecto Scratch sencillo que usa un bloque "si" para responder a la presión de una tecla.
- Los estudiantes ejecutan el proyecto y observan cómo cambia el comportamiento según la condición.
- En parejas, los estudiantes modifican el proyecto para cambiar la condición o la acción.
- Se discute en grupo qué hace cada bloque condicional y cómo influye en el programa.

Organización: Parejas

Producto esperado: Proyecto modificado con condicionales y explicación oral o escrita del funcionamiento.

Duración estimada: 45 minutos

Actividad 2: Creando un juego de decisiones con condicionales simples

Objetivo: Diseñar y construir programas en Scratch que utilicen condicionales para tomar decisiones basadas en diferentes condiciones.

Descripción paso a paso:

- El docente explica cómo usar condicionales para responder a eventos y valores.
- Los estudiantes planifican un juego sencillo (por ejemplo, atrapar objetos o evitar obstáculos) que dependa de decisiones condicionales.
- Construyen el juego en Scratch, implementando condicionales para controlar el comportamiento.
- Prueban y ajustan el juego para asegurar que las condiciones funcionen correctamente.

Organización: Individual

Producto esperado: Juego funcional en Scratch con uso de condicionales simples.

Duración estimada: 90 minutos

Actividad 3: Resolviendo problemas con condicionales compuestas

Objetivo: Aplicar condicionales compuestas para resolver problemas sencillos mediante la combinación de bloques lógicos en Scratch.

Descripción paso a paso:

- El docente introduce los operadores lógicos y explica cómo combinarlos en condiciones.
- Los estudiantes reciben un problema que requiere múltiples condiciones (por ejemplo, un juego donde se debe presionar dos teclas para activar una acción).
- Diseñan y programan la solución utilizando condicionales compuestas en Scratch.
- Comparten sus proyectos y explican cómo funcionan las condiciones compuestas.

Organización: Grupos pequeños (3-4 estudiantes)

Producto esperado: Proyecto Scratch con condicionales compuestas y explicación grupal.

Duración estimada: 90 minutos

Actividad 4: Depurando y mejorando proyectos con condicionales

Objetivo: Depurar programas que contengan estructuras condicionales para corregir errores y mejorar su funcionalidad.

Descripción paso a paso:

- El docente presenta un proyecto Scratch con errores relacionados a condicionales (condición mal planteada o bloque fuera de lugar).
- Los estudiantes analizan el proyecto, identifican errores y proponen correcciones.
- Realizan las correcciones, prueban el programa y verifican mejoras.
- Discuten en grupo las estrategias usadas para depurar y los resultados obtenidos.

Organización: Individual o parejas

Producto esperado: Proyecto corregido y informe breve de las correcciones realizadas.

Duración estimada: 60 minutos

Actividad 5: Presentación creativa del proyecto final con condicionales

Objetivo: Comunicar de forma clara y creativa el proceso de toma de decisiones implementado en proyectos usando condicionales en Scratch.

Descripción paso a paso:

- Cada estudiante o grupo prepara una presentación breve de su proyecto final que usa condicionales.
- Incluyen explicación de la lógica de las decisiones, mostrando el código y el resultado.
- Presentan ante el grupo clase, usando recursos visuales o narrativos para facilitar la comprensión.
- Se realiza una sesión de preguntas y retroalimentación.

Organización: Individual o grupos

Producto esperado: Presentación oral con soporte visual y proyecto Scratch funcional.

Duración estimada: 60 minutos

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre condicionales y lógica básica en programación.

Cómo se evalúa: Preguntas orales o escritas y pequeña actividad de reconocimiento de bloques condicionales en Scratch.

Instrumento sugerido: Cuestionario breve y observación directa durante una actividad inicial.

Evaluación formativa

Qué se evalúa: Progreso en la comprensión y aplicación de condicionales simples y compuestas, capacidad para depurar y modificar programas.

Cómo se evalúa: Revisión continua de proyectos, retroalimentación durante actividades prácticas, observación de participación en discusiones y resolución de problemas.

Instrumento sugerido: Rúbrica para evaluación de proyectos y listas de cotejo para seguimiento de habilidades de depuración y diseño.

Evaluación sumativa

Qué se evalúa: Dominio integral de las estructuras condicionales: identificación, diseño, aplicación, depuración y comunicación del proceso.

Cómo se evalúa: Proyecto final completo con condicionales (individual o grupal) acompañado de presentación explicativa.

Instrumento sugerido: Rúbrica detallada que incluya criterios de funcionalidad del programa, uso correcto de condicionales, calidad de la depuración y claridad en la presentación.

Unidad 6: Variables y almacenamiento de información

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de definir qué es una variable y describir su función en un proyecto de Scratch.
- Al finalizar la unidad, el estudiante será capaz de crear variables en Scratch y asignarles valores adecuados para almacenar diferentes tipos de datos.
- Al finalizar la unidad, el estudiante será capaz de utilizar variables para modificar y controlar el flujo de un programa mediante la manipulación de sus valores.
- Al finalizar la unidad, el estudiante será capaz de diseñar y construir un proyecto interactivo que integre variables para almacenar información y mostrar resultados dinámicos.
- Al finalizar la unidad, el estudiante será capaz de aplicar técnicas básicas de depuración para identificar y corregir errores relacionados con el uso de variables en sus programas.

Contenidos Temáticos

1. Introducción a las Variables en Programación

- **¿Qué es una variable?**
  Se explicará el concepto de variable como un espacio de almacenamiento en la memoria del programa que puede contener datos que pueden cambiar durante la ejecución.
- **Función de las variables en Scratch**
  Se describirá cómo las variables permiten guardar información temporal para que el programa la utilice y modifique, facilitando la interacción y el control del flujo.

2. Creación y Asignación de Variables en Scratch

- **Creación de variables en Scratch**
  Explicación paso a paso para crear variables desde el entorno de Scratch, incluyendo variables para todos los objetos y para objetos específicos.
- **Tipos de datos y asignación de valores**
  Se analizarán los tipos de datos básicos que pueden almacenar (números, textos) y cómo asignarles valores iniciales o modificarlos durante la ejecución.

3. Uso de Variables para Controlar el Flujo del Programa

- **Modificar variables durante la ejecución**
  Uso de bloques para cambiar el valor de una variable, incrementar, decrementar y almacenar resultados intermedios.
- **Condiciones y decisiones basadas en variables**
  Cómo utilizar variables en bloques condicionales para tomar decisiones, controlar bucles y modificar el comportamiento del programa.

4. Diseño y Construcción de Proyectos Interactivos con Variables

- **Planificación del proyecto**
  Definir qué información se debe almacenar y cómo será utilizada para la interacción con el usuario.
- **Implementación en Scratch**
  Integrar variables para mostrar resultados dinámicos, como puntajes, contadores o información ingresada por el usuario.
- **Pruebas y ajustes**
  Ejecutar el proyecto para verificar el correcto funcionamiento de las variables y la interacción.

5. Depuración y Corrección de Errores Relacionados con Variables

- **Identificación de errores comunes**
  Reconocer errores tales como variables no inicializadas, valores incorrectos o uso inapropiado en condiciones.
- **Técnicas básicas de depuración**
  Uso de impresión en pantalla, seguimiento paso a paso y ajustes para corregir errores en el uso de variables.

Actividades

Actividad 1: Explorando Variables en Scratch

Objetivo: Definir qué es una variable y describir su función en un proyecto de Scratch.

Descripción:

- El docente presenta ejemplos visuales de variables en Scratch y explica su función.
- Los estudiantes abren Scratch y crean una variable llamada "contador".
- Utilizan bloques para aumentar el contador cada vez que presionan una tecla.

- Discuten en grupo cómo la variable almacena la información y permite contar eventos.

Organización: Individual

Producto esperado: Proyecto Scratch con una variable "contador" que se incrementa con una acción.

Duración estimada: 40 minutos

Actividad 2: Creación y Uso de Variables para Control de Flujo

Objetivo: Crear variables en Scratch y utilizarlas para controlar el flujo del programa.

Descripción:

- Los estudiantes diseñan un programa simple donde una variable controla cuándo un sprite se mueve o cambia de color.
- Crean una variable llamada "vida" que disminuye al presionar un botón.
- Usan bloques condicionales para que cuando "vida" llegue a cero, el sprite desaparezca o muestre un mensaje.
- Prueban y ajustan el programa para que funcione correctamente.

Organización: Parejas

Producto esperado: Programa Scratch con variable que controla el estado y acciones del sprite.

Duración estimada: 50 minutos

Actividad 3: Proyecto Interactivo con Variables

Objetivo: Diseñar y construir un proyecto interactivo que integre variables para almacenar información y mostrar resultados dinámicos.

Descripción:

- Los estudiantes planifican un juego simple o actividad interactiva (por ejemplo, un juego de preguntas con puntaje).
- Crean variables para almacenar el puntaje o respuestas del usuario.
- Implementan el proyecto en Scratch, utilizando las variables para actualizar y mostrar resultados.
- Presentan el proyecto a sus compañeros y reciben retroalimentación.

Organización: Grupos pequeños (3-4 estudiantes)

Producto esperado: Proyecto Scratch interactivo que usa variables para almacenar y mostrar información dinámica.

Duración estimada: 2 horas

Actividad 4: Depuración de Variables en Scratch

Objetivo: Aplicar técnicas básicas de depuración para identificar y corregir errores relacionados con el uso de variables.

Descripción:

- El docente entrega un proyecto Scratch con errores comunes en el uso de variables.
- Los estudiantes analizan el código, identifican dónde están los errores y sugieren correcciones.

- Proponen y aplican cambios para corregir los errores y prueban el proyecto corregido.
- Discuten en grupo las estrategias utilizadas para depurar.

Organización: Individual o parejas

Producto esperado: Proyecto corregido y reporte breve de los errores encontrados y soluciones aplicadas.

Duración estimada: 45 minutos

Evaluación

Evaluación Diagnóstica

Qué se evalúa: Conocimiento previo sobre variables y almacenamiento de información.

Cómo se evalúa: Preguntas orales o escritas cortas para definir qué es una variable y su función.

Instrumento sugerido: Cuestionario breve o discusión guiada en clase.

Evaluación Formativa

Qué se evalúa: Progreso en la creación y uso de variables, aplicación en proyectos y habilidades de depuración.

Cómo se evalúa: Observación directa durante actividades, revisión de proyectos parciales y retroalimentación continua.

Instrumento sugerido: Rúbrica de desempeño para actividades prácticas y registro anecdótico del docente.

Evaluación Sumativa

Qué se evalúa: Capacidad para diseñar y construir un proyecto completo con variables, y aplicar depuración para corregir errores.

Cómo se evalúa: Entrega y presentación del proyecto final, acompañado de un informe breve explicando el uso de variables y el proceso de depuración.

Instrumento sugerido: Rúbrica de evaluación del proyecto final que incluya criterios sobre uso correcto de variables, funcionalidad, interacción y calidad de la depuración.

Unidad 7: Operadores y cálculos

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar y describir los diferentes bloques de operadores matemáticos y lógicos en Scratch para comprender su función en la programación visual.
- Al finalizar la unidad, el estudiante será capaz de utilizar bloques de operadores para realizar cálculos matemáticos simples dentro de un programa, aplicando operaciones básicas como suma, resta, multiplicación y división.
- Al finalizar la unidad, el estudiante será capaz de construir programas que integren operadores lógicos para controlar el flujo del programa mediante condiciones, demostrando comprensión de comparaciones y operadores booleanos.

- Al finalizar la unidad, el estudiante será capaz de diseñar y depurar algoritmos que utilicen operadores para resolver problemas cotidianos, evaluando la precisión y funcionalidad de los cálculos realizados.
- Al finalizar la unidad, el estudiante será capaz de explicar el uso de operadores y cálculos en sus proyectos interactivos, comunicando claramente cómo estos contribuyen a la lógica y resultados esperados del programa.

Contenidos Temáticos

1. Introducción a los operadores en Scratch

- Concepto de operadores en programación visual: definición y función básica.
- Tipos de operadores en Scratch: matemáticos, lógicos y comparativos.
- Exploración del entorno de Scratch: ubicación y apariencia de los bloques de operadores.

2. Operadores matemáticos básicos

- Bloques para suma, resta, multiplicación y división: uso y sintaxis visual.
- Ejemplos prácticos de cálculos simples con operadores matemáticos.
- Precisión y limitaciones en operaciones matemáticas en Scratch (división, números decimales).

3. Operadores lógicos y de comparación

- Concepto de operadores lógicos: AND, OR, NOT.
- Operadores de comparación: igual a, mayor que, menor que, mayor o igual, menor o igual, diferente.
- Cómo funcionan los operadores booleanos en condiciones para controlar el flujo del programa.

4. Construcción de programas con operadores para control de flujo

- Uso de operadores en bloques condicionales (si, si-sino).
- Integración de operadores matemáticos y lógicos en la toma de decisiones dentro de Scratch.
- Ejercicios para crear programas que respondan a condiciones usando operadores.

5. Diseño y depuración de algoritmos con operadores

- Resolución de problemas cotidianos mediante algoritmos que utilizan operadores.
- Identificación y corrección de errores relacionados con cálculos y condiciones.
- Pruebas y ajustes para asegurar precisión y funcionalidad en los resultados.

6. Comunicación y explicación del uso de operadores en proyectos

- Presentación de proyectos que incluyen operadores y cálculos.
- Descripción clara de cómo los operadores contribuyen a la lógica del programa.
- Reflexión sobre la importancia de los cálculos en la interacción y resultados del proyecto.

Actividades

Actividad 1: Explorando los bloques de operadores en Scratch

Objetivo: Identificar y describir los diferentes bloques de operadores matemáticos y lógicos en Scratch.

Descripción:

- El docente abre el editor de Scratch y muestra la categoría "Operadores".
- Los estudiantes, en sus computadoras, exploran cada bloque de operador, identificando su nombre y función.
- Se realiza una lluvia de ideas para que cada estudiante describa con sus palabras qué hace cada bloque.
- Se crea un cuadro en la pizarra o digital con los bloques y sus funciones descritas por los estudiantes.

Organización: Individual

Producto esperado: Lista escrita o digital con los bloques de operadores y su función.

Duración estimada: 45 minutos

Actividad 2: Programa de calculadora básica con operadores matemáticos

Objetivo: Utilizar bloques de operadores para realizar cálculos matemáticos simples dentro de un programa.

Descripción:

- Los estudiantes diseñan un programa en Scratch que permita ingresar dos números y seleccionar una operación (suma, resta, multiplicación, división).
- Usan los bloques de operadores matemáticos para realizar el cálculo seleccionado.
- El programa muestra el resultado en pantalla.
- Se prueba el programa con diferentes valores para verificar su funcionamiento.

Organización: Parejas

Producto esperado: Programa funcional de calculadora básica en Scratch.

Duración estimada: 90 minutos

Actividad 3: Creación de un juego con condiciones usando operadores lógicos

Objetivo: Construir programas que integren operadores lógicos para controlar el flujo mediante condiciones.

Descripción:

- Se propone un juego simple donde el personaje debe tomar decisiones según ciertas condiciones (por ejemplo, si la puntuación es mayor a 10 y el tiempo menor a 30 segundos, ganar).
- Los estudiantes utilizan operadores lógicos y de comparación para definir estas condiciones en bloques “si” o “si-sino”.
- Prueban y ajustan las condiciones para que el juego funcione correctamente.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Juego interactivo con condiciones lógicas implementadas.

Duración estimada: 2 horas

Actividad 4: Depuración y explicación del uso de operadores en un proyecto

Objetivo: Diseñar y depurar algoritmos con operadores y explicar su uso en proyectos interactivos.

Descripción:

- Los estudiantes reciben un proyecto en Scratch con errores en cálculos o condiciones.
- Identifican y corrigen los errores utilizando los operadores de forma correcta.
- Preparan una presentación breve explicando cómo usaron los operadores y qué lógica aplicaron para lograr el resultado esperado.
- Comparten su explicación con el grupo para fomentar el aprendizaje colaborativo.

Organización: Individual o parejas

Producto esperado: Proyecto corregido y presentación oral o escrita explicando el uso de operadores.

Duración estimada: 90 minutos

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre operadores y lógica básica.

Cómo se evalúa: Cuestionario rápido y práctica exploratoria en Scratch para identificar bloques de operadores.

Instrumento sugerido: Formulario digital o papel con preguntas de opción múltiple y ejercicios cortos dentro del editor Scratch.

Evaluación formativa

Qué se evalúa: Progreso en la identificación, uso y aplicación de operadores matemáticos y lógicos en actividades prácticas.

Cómo se evalúa: Observación directa durante actividades, revisión de proyectos en desarrollo y retroalimentación continua.

Instrumento sugerido: Lista de cotejo para evaluar el uso correcto de bloques de operadores, participación en actividades y calidad del código en Scratch.

Evaluación sumativa

Qué se evalúa: Dominio integral de los operadores y cálculos en la creación de un proyecto final funcional que incluya cálculos y condiciones lógicas.

Cómo se evalúa: Presentación y demostración del proyecto final junto con una explicación oral o escrita sobre el uso y propósito de los operadores.

Instrumento sugerido: Rúbrica de evaluación que contemple precisión del código, uso adecuado de operadores, funcionalidad del proyecto y claridad en la explicación.

Unidad 8: Diseño y planificación de proyectos

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de descomponer problemas simples en partes más pequeñas para facilitar su análisis y solución.
- Al finalizar la unidad, el estudiante será capaz de diseñar algoritmos básicos utilizando diagramas de flujo o pseudocódigo que representen la lógica de un proyecto antes de programarlo en Scratch.
- Al finalizar la unidad, el estudiante será capaz de identificar y organizar las secuencias, ciclos y condiciones necesarias para planificar un proyecto interactivo en Scratch.
- Al finalizar la unidad, el estudiante será capaz de evaluar y modificar su plan de proyecto para asegurar que el diseño del algoritmo cumpla con los objetivos propuestos.
- Al finalizar la unidad, el estudiante será capaz de comunicar de manera clara y estructurada el plan y diseño de su proyecto a sus compañeros o profesores, utilizando términos básicos del pensamiento computacional.

Contenidos Temáticos

1. Introducción al diseño y planificación de proyectos

- Concepto de proyecto en programación: definición y ejemplos simples.
- Importancia de planificar antes de programar.
- Relación entre pensamiento computacional y diseño de proyectos.

2. Descomposición de problemas

- Qué es la descomposición: dividir un problema complejo en partes pequeñas.
- Ejemplos prácticos de descomposición en proyectos simples.
- Ejercicios para identificar subproblemas y tareas.

3. Diseño de algoritmos básicos

- Definición y propósito de un algoritmo en la programación.
- Introducción a diagramas de flujo: símbolos básicos y cómo usarlos.
- Introducción al pseudocódigo: estructura y ejemplos simples.
- Cómo representar la lógica de un proyecto con diagramas de flujo y pseudocódigo.

4. Secuencias, ciclos y condiciones en la planificación

- Concepto de secuencia: pasos que se ejecutan uno tras otro.
- Ciclos o repeticiones: cuándo y cómo usarlos.
- Condiciones o decisiones: uso de estructuras condicionales básicas.
- Identificación y organización de secuencias, ciclos y condiciones para un proyecto en Scratch.

5. Evaluación y modificación del plan de proyecto

- Revisión crítica del algoritmo y plan de proyecto.
- Identificación de errores o mejoras en la lógica.
- Cómo modificar y optimizar el plan antes de programar.

6. Comunicación clara y estructurada del diseño del proyecto

- Uso de términos básicos del pensamiento computacional para explicar el plan.
- Técnicas para presentar el algoritmo y la planificación a compañeros o profesores.
- Ejemplos prácticos de comunicación oral y escrita del diseño.

Actividades

Actividad 1: "Descomponiendo un problema: el juego del laberinto"

Objetivo: Descomponer problemas simples en partes más pequeñas para facilitar su análisis y solución.

Descripción:

- Se presenta a los estudiantes un problema sencillo: diseñar un juego de laberinto en Scratch.
- En grupos, identificarán las partes del problema (por ejemplo: crear el laberinto, programar el movimiento del personaje, detectar colisiones, establecer el objetivo).
- Escribirán los subproblemas o tareas en una lista ordenada.
- Discutirán cómo cada parte contribuye al proyecto general.

Organización: Grupos de 3-4 estudiantes.

Producto esperado: Lista con las partes descompuestas del problema y breve explicación de cada una.

Duración estimada: 50 minutos.

Actividad 2: "Diseñando algoritmos con diagramas de flujo"

Objetivo: Diseñar algoritmos básicos utilizando diagramas de flujo que representen la lógica del proyecto.

Descripción:

- Los estudiantes recibirán un subproblema sencillo (por ejemplo: mover un personaje en cuatro direcciones con teclas).
- Con material impreso o digital, dibujarán el diagrama de flujo que represente la lógica para controlar el movimiento.
- Se revisarán los símbolos básicos y su significado antes de iniciar.
- Al finalizar, cada grupo presentará su diagrama y explicará el flujo lógico.

Organización: Parejas.

Producto esperado: Diagrama de flujo completo y presentación oral breve.

Duración estimada: 60 minutos.

Actividad 3: "Pseudocódigo para decisiones y ciclos"

Objetivo: Identificar y organizar secuencias, ciclos y condiciones en un algoritmo usando pseudocódigo.

Descripción:

- Se explicará la estructura básica de pseudocódigo y ejemplos de secuencias, ciclos y condiciones.
- Individualmente, los estudiantes escribirán pseudocódigo para un proyecto simple, por ejemplo, un juego donde el personaje debe recoger objetos y evitar obstáculos.
- El pseudocódigo debe incluir al menos una condición y un ciclo.
- Luego, compartirán su pseudocódigo en grupos pequeños para recibir retroalimentación y proponer mejoras.

Organización: Individual y luego grupos de 3.

Producto esperado: Documento con pseudocódigo que incluya secuencias, ciclos y condiciones.

Duración estimada: 70 minutos.

Actividad 4: "Evaluando y comunicando nuestro plan de proyecto"

Objetivo: Evaluar y modificar el plan de proyecto y comunicarlo claramente a otros utilizando términos de pensamiento computacional.

Descripción:

- Cada grupo revisará el plan de proyecto y algoritmo desarrollado anteriormente.
- Identificarán posibles mejoras o errores en la lógica y propondrán modificaciones.
- Prepararán una presentación oral o escrita donde expliquen el plan, la lógica y las mejoras realizadas usando vocabulario básico de pensamiento computacional.
- Presentarán su proyecto a la clase para recibir retroalimentación final.

Organización: Grupos de 3-4 estudiantes.

Producto esperado: Plan modificado y presentación clara del diseño del proyecto.

Duración estimada: 90 minutos.

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre descomposición de problemas, algoritmos y estructuras básicas de programación.

Cómo se evalúa: Breve cuestionario oral o escrito con preguntas abiertas y de opción múltiple sobre conceptos básicos y ejemplos simples.

Instrumento sugerido: Cuestionario inicial con 5 preguntas.

Evaluación formativa

Qué se evalúa: Progreso en la capacidad de descomponer problemas, diseñar algoritmos, identificar estructuras lógicas y comunicar el plan.

Cómo se evalúa: Observación directa durante actividades, revisión de diagramas de flujo y pseudocódigo, retroalimentación entre pares, y autoevaluaciones guiadas.

Instrumento sugerido: Rúbricas para diagramas de flujo y pseudocódigo, listas de cotejo para participación y comunicación.

Evaluación sumativa

Qué se evalúa: Capacidad integral para descomponer un problema, diseñar y modificar un algoritmo básico, y presentar su planificación con claridad.

Cómo se evalúa: Entrega y presentación final de un plan de proyecto que incluya descomposición, algoritmo (diagrama de flujo o pseudocódigo), y explicación clara del diseño.

Instrumento sugerido: Rúbrica de evaluación que considere precisión en la descomposición, calidad del algoritmo, uso correcto de estructuras lógicas, y claridad en la comunicación.

Unidad 9: Creación de animaciones simples

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de seleccionar y modificar sprites y disfraces para crear personajes animados que representen una historia simple.
- Al finalizar la unidad, el estudiante será capaz de diseñar y organizar secuencias de bloques en Scratch para controlar el movimiento y las acciones de los sprites en una animación.
- Al finalizar la unidad, el estudiante será capaz de aplicar efectos visuales y sonidos en Scratch para enriquecer la presentación de una historia animada.
- Al finalizar la unidad, el estudiante será capaz de identificar y corregir errores básicos en sus animaciones para asegurar un funcionamiento correcto y fluido.
- Al finalizar la unidad, el estudiante será capaz de comunicar de forma clara y creativa el proceso y resultado de su animación mediante una presentación o explicación oral o escrita.

Contenidos Temáticos

1. Introducción a los sprites y disfraces en Scratch

- Definición y función de los sprites en Scratch: personajes y objetos animados.
- Selección de sprites desde la biblioteca y carga de sprites personalizados.
- Modificación y creación de disfraces para un mismo sprite: edición básica, cambio de color, rotación y duplicación.
- Relación entre sprites y disfraces para representar diferentes estados o acciones en la animación.

2. Diseño y organización de secuencias de bloques para animar sprites

- Revisión de categorías de bloques relevantes: movimiento, apariencia, control y sonido.

- Construcción de secuencias básicas para mover un sprite: desplazamientos, giros y uso de coordenadas.
- Programación de cambios de disfraces para simular animación dentro de un sprite.
- Uso de bloques de control para organizar la secuencia: eventos, repeticiones y esperas.

3. Aplicación de efectos visuales y sonidos

- Introducción a los efectos visuales disponibles en Scratch: cambio de color, brillo, fantasma, etc.
- Programación para activar y modificar efectos durante la animación.
- Incorporación de sonidos predefinidos y grabados para enriquecer la presentación.
- Sincronización de efectos visuales y sonidos con las acciones de los sprites.

4. Detección y corrección de errores básicos en animaciones

- Identificación de errores comunes: bloque mal colocado, secuencia incorrecta, animación que no inicia o se detiene.
- Uso de la función "probar" y "depurar" para ejecutar y observar el comportamiento del proyecto.
- Estrategias para corregir errores: reorganizar bloques, revisar condiciones y tiempos de espera.
- Importancia de la revisión iterativa y pruebas constantes para asegurar fluidez.

5. Comunicación y presentación del proyecto de animación

- Organización de una presentación oral o escrita clara y creativa del proceso de creación.
- Elementos clave a comunicar: idea de la historia, selección de sprites y disfraces, decisiones de programación.
- Uso de recursos visuales y demostración en vivo de la animación.
- Fomento de la retroalimentación entre compañeros y autoevaluación.

Actividades

Actividad 1: "Creando mi personaje animado"

Objetivo: Seleccionar y modificar sprites y disfraces para crear personajes animados que representen una historia simple.

Descripción paso a paso:

- Acceder al editor de Scratch y explorar la biblioteca de sprites.
- Seleccionar un sprite que represente un personaje para una historia sencilla.
- Modificar el disfraz del sprite usando las herramientas básicas: cambiar colores, añadir elementos o crear un nuevo disfraz.
- Guardar el sprite con sus disfraces modificados para su uso posterior.

Organización: Individual

Producto esperado: Un sprite con al menos dos disfraces diferentes que representen estados o acciones del personaje.

Duración estimada: 45 minutos

Actividad 2: "Secuencia de movimiento y animación"

Objetivo: Diseñar y organizar secuencias de bloques en Scratch para controlar el movimiento y las acciones de los sprites en una animación.

Descripción paso a paso:

- Utilizar el sprite creado en la actividad anterior.
- Programar movimientos básicos: desplazarse de un punto a otro y girar.
- Incorporar cambio de disfraces para simular animación (por ejemplo, caminar o saltar).
- Agregar bloques de control para repetir la animación o responder a eventos (clic o tecla).
- Probar la secuencia y hacer ajustes para lograr fluidez.

Organización: Parejas

Producto esperado: Animación programada con movimientos y cambios de disfraces funcionando correctamente.

Duración estimada: 1 hora

Actividad 3: "Enriqueciendo la historia con efectos y sonidos"

Objetivo: Aplicar efectos visuales y sonidos en Scratch para enriquecer la presentación de una historia animada.

Descripción paso a paso:

- Agregar efectos visuales a los sprites durante la animación, por ejemplo, cambio de color o brillo al realizar una acción.
- Insertar sonidos pregrabados o grabar sonidos propios relacionados con la historia.
- Programar la activación sincronizada de efectos y sonidos con las acciones de los sprites.
- Ejecutar la animación completa para verificar que los efectos y sonidos se integren adecuadamente.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Animación con efectos visuales y sonidos integrados que complementan la historia.

Duración estimada: 1 hora 15 minutos

Actividad 4: "Presentando mi animación"

Objetivo: Comunicar de forma clara y creativa el proceso y resultado de la animación mediante una presentación oral o escrita.

Descripción paso a paso:

- Preparar una breve explicación o presentación escrita sobre la historia creada, el proceso de selección/modificación de sprites y disfraces, y la programación realizada.
- Incluir detalles sobre los efectos visuales y sonidos aplicados y cómo contribuyen a la presentación.
- Exponer o compartir la animación con el grupo o docente, explicando el proyecto y respondiendo preguntas.
- Recibir retroalimentación y realizar una autoevaluación del trabajo realizado.

Organización: Individual o en parejas

Producto esperado: Presentación oral o escrita acompañada de la animación funcional en Scratch.

Duración estimada: 45 minutos

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre el uso de Scratch, familiaridad con sprites y bloques básicos.

Cómo se evalúa: Cuestionario corto y práctica inicial para seleccionar y modificar un sprite simple.

Instrumento sugerido: Lista de cotejo para observar habilidades básicas en el entorno Scratch y breve cuestionario escrito.

Evaluación formativa

Qué se evalúa: Progreso en la creación y programación de animaciones, aplicación de efectos, corrección de errores y participación en actividades.

Cómo se evalúa: Observación directa durante actividades, revisión de proyectos en desarrollo, retroalimentación entre pares y autoevaluación guiada.

Instrumento sugerido: Rúbrica de desempeño con criterios para selección y modificación de sprites, diseño de secuencias, aplicación de efectos, corrección de errores y comunicación.

Evaluación sumativa

Qué se evalúa: Producto final: animación completa que incluya personajes con disfraces modificados, secuencia funcional de bloques, efectos visuales y sonidos aplicados, corrección de errores y presentación clara del proyecto.

Cómo se evalúa: Revisión del proyecto en Scratch y presentación oral o escrita.

Instrumento sugerido: Rúbrica detallada que valore creatividad, funcionalidad técnica, integración de efectos y sonidos, calidad de la presentación y capacidad de corrección de errores.

Unidad 10: Desarrollo de juegos básicos I

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de diseñar juegos sencillos en Scratch incorporando controles básicos para la interacción del usuario.
- Al finalizar la unidad, el estudiante será capaz de implementar condiciones que permitan la respuesta del juego a las acciones del jugador usando bloques condicionales.
- Al finalizar la unidad, el estudiante será capaz de combinar secuencias y ciclos para crear dinámicas de juego que faciliten la interacción continua del usuario.
- Al finalizar la unidad, el estudiante será capaz de identificar y corregir errores simples en el diseño de juegos para mejorar su funcionalidad y experiencia de usuario.

- Al finalizar la unidad, el estudiante será capaz de comunicar el proceso de diseño y funcionamiento de su juego básico utilizando un lenguaje claro y estructurado.

Contenidos Temáticos

1. Introducción al diseño de juegos sencillos en Scratch

- Conceptos básicos de un juego: objetivo, reglas y controles
- Exploración de la interfaz de Scratch para crear juegos
- Creación y configuración de sprites y escenarios

2. Incorporación de controles básicos para la interacción del usuario

- Uso de eventos para detectar acciones del usuario (teclado, ratón)
- Bloques para mover sprites: movimiento con teclas y clics
- Configuración de variables para controlar estados y puntajes

3. Implementación de condiciones con bloques condicionales

- Introducción a los bloques "si", "si...entonces...sino"
- Uso de operadores lógicos y comparaciones para tomar decisiones
- Condiciones para detectar colisiones y eventos en el juego

4. Combinación de secuencias y ciclos para dinámicas continuas

- Estructuras de control: secuencias, bucles "repetir" y "por siempre"
- Creación de ciclos de juego para actualizaciones constantes
- Sincronización de movimientos y respuestas mediante ciclos

5. Identificación y corrección de errores simples en juegos

- Tipos comunes de errores en Scratch: lógica, secuencia y sintaxis visual
- Uso de la depuración paso a paso y mensajes para detectar errores
- Pruebas de funcionalidad y ajustes para mejorar la experiencia de usuario

6. Comunicación del proceso de diseño y funcionamiento del juego

- Documentación básica del diseño: objetivos y reglas del juego
- Presentación oral o escrita explicando la lógica y funcionamiento
- Uso de vocabulario técnico claro y estructurado para describir el proyecto

Actividades

Actividad 1: Creación de un juego sencillo con controles básicos

Objetivo: Diseñar un juego sencillo en Scratch con controles básicos para la interacción del usuario.

Descripción:

- Explicar brevemente la estructura de un juego simple en Scratch.
- Guiar a los estudiantes para crear un escenario y al menos dos sprites.
- Programar el sprite principal para moverse con las teclas de flecha o WASD.
- Agregar un objetivo sencillo, como recoger objetos o evitar obstáculos.
- Probar el juego y ajustar el control de movimiento para que sea fluido.

Organización: Individual

Producto esperado: Juego básico funcional con controles de usuario implementados.

Duración estimada: 2 horas

Actividad 2: Implementación de condiciones para respuestas del juego

Objetivo: Utilizar bloques condicionales para que el juego responda a las acciones del jugador.

Descripción:

- Introducir bloques "si" y "si...entonces...sino" con ejemplos prácticos.
- Programar condiciones para detectar colisiones entre sprites.
- Crear respuestas condicionadas, como cambiar el puntaje o mostrar mensajes.
- Agregar condiciones para terminar el juego o pasar a un siguiente nivel.
- Realizar pruebas y ajustes para asegurar que las condiciones funcionen correctamente.

Organización: Parejas

Producto esperado: Juego con condiciones que respondan a eventos del jugador.

Duración estimada: 2 horas

Actividad 3: Creación de dinámicas continuas usando secuencias y ciclos

Objetivo: Combinar secuencias y ciclos para crear interacción continua en el juego.

Descripción:

- Explicar y demostrar el uso de bucles "repetir" y "por siempre".
- Programar movimientos automáticos de un sprite enemigo o de objetos en el escenario.
- Sincronizar ciclos para que el juego tenga una dinámica constante y fluida.
- Integrar estas dinámicas con las condiciones y controles ya implementados.
- Probar y optimizar el flujo del juego para evitar bloqueos o pausas.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Juego con movimientos y respuestas continuas mediante ciclos y secuencias.

Duración estimada: 2 horas

Actividad 4: Depuración y presentación del juego

Objetivo: Identificar y corregir errores simples y comunicar el diseño y funcionamiento del juego.

Descripción:

- En equipo, probar el juego para identificar errores de lógica o funcionalidad.
- Usar herramientas de depuración de Scratch para encontrar problemas.
- Corregir errores y mejorar la experiencia de usuario.
- Preparar una breve presentación (oral o escrita) explicando el proceso de diseño y las características del juego.
- Compartir el juego y la presentación con el grupo o clase para recibir retroalimentación.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Versión corregida del juego y presentación clara del proceso y funcionamiento.

Duración estimada: 2 horas

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre Scratch, conceptos básicos de programación y experiencia previa en diseño de juegos.

Cómo se evalúa: Cuestionario breve y una actividad práctica inicial donde los estudiantes exploren y describan bloques básicos de Scratch.

Instrumento sugerido: Cuestionario en papel o digital, actividad de exploración guiada en Scratch.

Evaluación formativa

Qué se evalúa: Avance en la implementación de controles, condiciones, uso de ciclos, y capacidad para detectar y corregir errores.

Cómo se evalúa: Observación directa durante las actividades, revisión de proyectos en desarrollo, retroalimentación continua y autoevaluación entre pares.

Instrumento sugerido: Rúbrica para evaluar funcionalidades específicas del juego, listas de cotejo para seguimiento de etapas, y registros de observación docente.

Evaluación sumativa

Qué se evalúa: Juego final funcional con controles, condiciones, dinámicas continuas, además de la presentación clara del proceso y funcionamiento.

Cómo se evalúa: Presentación y entrega del proyecto final, revisión del código y la documentación del diseño, evaluación oral o escrita del proceso.

Instrumento sugerido: Rúbrica de evaluación que incluya criterios de funcionalidad, creatividad, claridad en la presentación y corrección de errores.

Unidad 11: Desarrollo de juegos básicos II

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de implementar variables para puntajes en un juego básico utilizando bloques de programación en Scratch, asegurando que el puntaje se actualice correctamente durante la ejecución.
- Al finalizar la unidad, el estudiante será capaz de diseñar y aplicar condiciones lógicas que controlen el flujo del juego, mejorando la interacción y la experiencia del usuario.
- Al finalizar la unidad, el estudiante será capaz de depurar y corregir errores relacionados con el manejo de variables y condiciones en proyectos de juegos desarrollados en Scratch.
- Al finalizar la unidad, el estudiante será capaz de comunicar de manera clara la lógica y el funcionamiento de su juego, describiendo el uso de variables y estructuras condicionales empleadas para mejorar la experiencia del usuario.

Contenidos Temáticos

1. Introducción a las variables en Scratch

- Concepto de variable: definición y ejemplos simples.
- Tipos de variables en Scratch: variables para todos los objetos y variables para un solo objeto.
- Creación y uso básico de variables: cómo crear una variable para puntaje.

2. Implementación de puntajes en un juego básico

- Incrementar y decrementar el puntaje mediante bloques de código.
- Mostrar el puntaje en pantalla: uso del monitor de variables.
- Reiniciar el puntaje al iniciar o reiniciar el juego.

3. Condiciones lógicas para controlar el flujo del juego

- Repaso de operadores lógicos: igual, mayor que, menor que, y, o, no.
- Estructuras condicionales en Scratch: bloques "si", "si-entonces", "si-entonces-sino".
- Aplicación de condiciones para mejorar la interacción: por ejemplo, incrementar puntaje solo si se cumple una condición, cambiar niveles o estados del juego.

4. Depuración y corrección de errores en proyectos con variables y condiciones

- Identificación de errores comunes relacionados con variables y condiciones.
- Uso de herramientas de Scratch para depurar: bloques para mostrar valores, pausas y mensajes.
- Estrategias para corregir errores lógicos y de programación.

5. Comunicación de la lógica y funcionamiento del juego

- Cómo describir el uso de variables para puntajes.

- Explicación de condiciones lógicas implementadas y su función en el juego.
- Presentación oral o escrita del proyecto con énfasis en la lógica de programación.

Actividades

Actividad 1: Creación e implementación de la variable puntaje

Objetivo: Implementar variables para puntajes en un juego básico utilizando bloques de programación en Scratch, asegurando que el puntaje se actualice correctamente durante la ejecución.

Descripción:

- El docente presenta un proyecto base de un juego simple (por ejemplo, atrapar objetos que caen).
- Los estudiantes crean una variable llamada "puntaje".
- Programan que cada vez que se atrape un objeto, el puntaje aumente en 1.
- Configuran el juego para que el puntaje se muestre en pantalla durante la ejecución.
- Prueban el juego para verificar que el puntaje se actualice correctamente.

Organización: Individual.

Producto esperado: Juego básico con variable puntaje implementada y funcional.

Duración estimada: 1 hora.

Actividad 2: Uso de condiciones para controlar el flujo del juego

Objetivo: Diseñar y aplicar condiciones lógicas que controlen el flujo del juego, mejorando la interacción y la experiencia del usuario.

Descripción:

- El docente explica el uso de bloques condicionales y operadores lógicos en Scratch.
- Los estudiantes modifican su juego para que el puntaje solo aumente si el objeto atrapado es de un color específico.
- Incorporan una condición para mostrar un mensaje de "Nivel completado" cuando el puntaje llegue a un valor determinado.
- Prueban y observan cómo las condiciones afectan la dinámica del juego.

Organización: Parejas.

Producto esperado: Juego con condiciones implementadas que controlan el puntaje y el flujo del juego.

Duración estimada: 1.5 horas.

Actividad 3: Depuración y corrección de errores en el juego

Objetivo: Depurar y corregir errores relacionados con el manejo de variables y condiciones en proyectos de juegos desarrollados en Scratch.

Descripción:

- El docente presenta ejemplos de errores comunes: variable que no se actualiza, condiciones mal configuradas.

- Los estudiantes revisan su proyecto y utilizan bloques para mostrar valores de variables durante la ejecución.
- Identifican posibles errores lógicos y los corrigen con apoyo del docente y compañeros.
- Documentan los errores encontrados y las soluciones implementadas.

Organización: Grupos de 3-4 estudiantes.

Producto esperado: Juego corregido y reporte escrito o digital con errores y soluciones.

Duración estimada: 1.5 horas.

Actividad 4: Presentación y explicación de la lógica del juego

Objetivo: Comunicar de manera clara la lógica y el funcionamiento de su juego, describiendo el uso de variables y estructuras condicionales empleadas.

Descripción:

- Cada estudiante o grupo prepara una breve presentación oral o escrita.
- Describen cómo implementaron la variable para el puntaje y cómo la actualizaron.
- Explican las condiciones lógicas aplicadas y su efecto en la jugabilidad.
- Comparten dificultades encontradas y cómo las resolvieron.
- Reciben retroalimentación del docente y compañeros.

Organización: Individual o grupos según preferencia.

Producto esperado: Presentación oral o documento explicativo.

Duración estimada: 1 hora.

Evaluación

Evaluación diagnóstica

Se evalúa el conocimiento previo sobre variables y condiciones lógicas en Scratch.

- **Cómo:** Cuestionario corto con preguntas de opción múltiple y preguntas abiertas sobre conceptos básicos de variables y condicionales.
- **Instrumento sugerido:** Formulario digital o impreso con 10 preguntas.

Evaluación formativa

Se evalúa el progreso durante las actividades mediante observación y revisión continua.

- **Qué se evalúa:** Implementación correcta de variables, aplicación de condiciones lógicas, capacidad para depurar errores y participación en actividades.
- **Cómo:** Revisión directa de proyectos en Scratch, preguntas guiadas, y revisión de reportes de depuración.
- **Instrumento sugerido:** Listas de cotejo para cada actividad y notas de observación del docente.

Evaluación sumativa

Se evalúa el logro de los objetivos al final de la unidad mediante la entrega y presentación del proyecto final.

- **Qué se evalúa:** Juego con variable para puntaje funcional, condiciones lógicas aplicadas correctamente, evidencia de depuración y explicación clara de la lógica del juego.
- **Cómo:** Presentación del proyecto y defensa oral o escrita de su funcionamiento.
- **Instrumento sugerido:** Rúbrica que incluya criterios de funcionalidad, uso de variables, aplicación de condiciones, depuración y claridad en la explicación.

Unidad 12: Depuración y mejora de proyectos

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar errores comunes en proyectos de Scratch mediante la revisión sistemática del código.
- Al finalizar la unidad, el estudiante será capaz de aplicar técnicas básicas de depuración para corregir errores que afectan la ejecución de sus programas.
- Al finalizar la unidad, el estudiante será capaz de optimizar la lógica de sus proyectos utilizando ciclos y condiciones para mejorar su eficiencia.
- Al finalizar la unidad, el estudiante será capaz de evaluar y modificar variables y bloques para mejorar la funcionalidad y la interacción de sus programas.
- Al finalizar la unidad, el estudiante será capaz de documentar los cambios realizados durante la depuración y mejora para comunicar claramente el proceso y los resultados.

Contenidos Temáticos

1. Introducción a la depuración en Scratch

- Concepto de depuración: qué es y por qué es importante en programación.
- Tipos comunes de errores en proyectos Scratch: sintaxis, lógica y ejecución.
- Herramientas básicas para revisar y analizar proyectos en Scratch.

2. Identificación sistemática de errores en proyectos Scratch

- Revisión paso a paso del código: cómo leer y entender los bloques.
- Errores frecuentes: bloques mal conectados, variables mal usadas, condiciones incorrectas.
- Uso del modo “ver en vivo” y seguimiento de la ejecución para detectar fallas.

3. Técnicas básicas de depuración y corrección de errores

- Pruebas parciales: probar partes pequeñas del código para aislar errores.
- Uso de bloques “decir” o “mostrar” para depurar (imprimir valores y estados).
- Corrección de errores detectados y verificación de funcionamiento.

4. Optimización de la lógica mediante ciclos y condiciones

- Revisión de estructuras repetitivas: uso correcto de “repetir”, “por siempre” y bucles anidados.
- Condiciones eficientes: simplificación y combinación de bloques “si”, “si-sino”.
- Ejemplos de mejora en proyectos para reducir código repetitivo y mejorar el rendimiento.

5. Evaluación y modificación de variables y bloques para mejorar funcionalidad

- Revisión del uso de variables: nombres claros, alcance y actualización correcta.
- Modificación de bloques para mejorar la interacción con el usuario y la respuesta del programa.
- Incorporación de comentarios y organización del código para facilitar futuras mejoras.

6. Documentación de cambios y comunicación del proceso de mejora

- Importancia de documentar el proceso de depuración y mejora.
- Formatos sencillos para registrar errores encontrados, soluciones aplicadas y resultados.
- Presentación oral o escrita para explicar las mejoras realizadas en el proyecto.

Actividades

Actividad 1: Explorando y detectando errores comunes en un proyecto Scratch

Objetivo: Identificar errores comunes en proyectos de Scratch mediante la revisión sistemática del código.

Descripción:

- El docente proporcionará un proyecto Scratch con errores intencionales (bloques mal conectados, variables mal usadas, condiciones erróneas).
- Los estudiantes abrirán el proyecto y realizarán una inspección paso a paso del código.
- Deberán listar al menos cinco errores detectados y explicar por qué son errores.

Organización: Individual

Producto esperado: Lista escrita con errores detectados y explicación.

Duración estimada: 45 minutos

Actividad 2: Depuración guiada con bloques de diagnóstico

Objetivo: Aplicar técnicas básicas de depuración para corregir errores que afectan la ejecución de sus programas.

Descripción:

- Los estudiantes trabajan con un proyecto que presenta fallas en su ejecución.
- Se les enseña a usar bloques “decir” para imprimir valores de variables en tiempo real y detectar problemas.
- Guía paso a paso para corregir errores encontrados y probar el programa tras cada cambio.

Organización: Parejas

Producto esperado: Proyecto corregido y breve informe de los errores corregidos y cómo se solucionaron.

Duración estimada: 1 hora

Actividad 3: Optimización del proyecto mediante ciclos y condiciones

Objetivo: Optimizar la lógica de sus proyectos utilizando ciclos y condiciones para mejorar su eficiencia.

Descripción:

- Cada estudiante selecciona un proyecto propio o uno proporcionado.
- Identifican partes del código donde se repite lógica o donde las condiciones pueden simplificarse.
- Modifican el código para usar ciclos adecuados y condiciones optimizadas.
- Comparan el antes y después en funcionamiento y eficiencia.

Organización: Individual

Producto esperado: Proyecto optimizado y un documento breve que explique los cambios realizados.

Duración estimada: 1.5 horas

Actividad 4: Documentando y presentando mejoras en el proyecto

Objetivo: Documentar los cambios realizados durante la depuración y mejora para comunicar claramente el proceso y los resultados.

Descripción:

- Cada estudiante prepara una breve documentación escrita o presentación que incluya:
  - Errores iniciales detectados.
  - Técnicas utilizadas para corregirlos.
  - Mejoras de optimización aplicadas.
  - Resultados obtenidos tras las mejoras.
- Presentan su documentación o exposición al grupo para recibir retroalimentación.

Organización: Individual o en parejas

Producto esperado: Documento o presentación que explique el proceso de depuración y mejora.

Duración estimada: 1 hora

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre errores comunes, lectura y análisis de código Scratch.

Cómo se evalúa: Presentación de un proyecto con errores y preguntas para identificar posibles fallas.

Instrumento sugerido: Cuestionario corto y análisis guiado en clase.

Evaluación formativa

Qué se evalúa: Progreso en la identificación y corrección de errores, aplicación de técnicas de depuración y optimización.

Cómo se evalúa: Observación directa durante actividades, revisión de proyectos corregidos y optimizados, retroalimentación continua.

Instrumento sugerido: Rúbrica de desempeño para actividades prácticas y revisión de informes parciales.

Evaluación sumativa

Qué se evalúa: Capacidad para identificar errores, aplicar técnicas de depuración, optimizar la lógica, modificar variables y documentar el proceso.

Cómo se evalúa: Entrega de un proyecto final corregido y optimizado junto con documentación explicativa y presentación oral o escrita.

Instrumento sugerido: Rúbrica que contemple calidad del proyecto, efectividad de correcciones, claridad en la documentación y presentación.

Unidad 13: Introducción a la colaboración en Scratch

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar y describir las funciones básicas para compartir proyectos en la plataforma Scratch, demostrando comprensión de las opciones de privacidad y acceso.
- Al finalizar la unidad, el estudiante será capaz de colaborar con sus compañeros mediante la creación conjunta de un proyecto en Scratch, aplicando técnicas de comunicación y división de tareas.
- Al finalizar la unidad, el estudiante será capaz de evaluar y proporcionar retroalimentación constructiva sobre proyectos de sus compañeros en Scratch, utilizando criterios definidos de funcionamiento y creatividad.
- Al finalizar la unidad, el estudiante será capaz de modificar un proyecto compartido en Scratch incorporando sugerencias recibidas, mejorando la funcionalidad y la presentación del mismo.

Contenidos Temáticos

1. Introducción a la colaboración en Scratch

- Concepto de colaboración y su importancia en proyectos de programación
- Beneficios de trabajar en equipo en la plataforma Scratch

2. Compartir proyectos en Scratch

- Funciones básicas para compartir proyectos: publicar y enviar enlaces
- Opciones de privacidad y acceso: público, privado y compartido con grupos
- Configuración de permisos para colaboración y comentarios

3. Creación conjunta de proyectos en Scratch

- Técnicas de comunicación efectiva entre compañeros
- División de tareas y asignación de roles dentro del equipo

- Uso de la plataforma para coordinar y combinar aportes

4. Evaluación y retroalimentación constructiva

- Criterios para evaluar proyectos: funcionamiento, creatividad y presentación
- Cómo dar retroalimentación clara, respetuosa y constructiva
- Ejemplos prácticos de comentarios útiles y sugerencias

5. Modificación y mejora de proyectos compartidos

- Incorporación de sugerencias recibidas para mejorar funcionalidad
- Mejoras en la presentación visual y narrativa del proyecto
- Proceso de revisión y actualización continua del proyecto

Actividades

Actividad 1: Explorando las opciones para compartir proyectos en Scratch

Objetivo: Identificar y describir las funciones básicas para compartir proyectos en Scratch, comprendiendo opciones de privacidad y acceso.

Descripción paso a paso:

- El docente muestra la interfaz de Scratch destacando el botón para compartir proyectos.
- Los estudiantes crean un proyecto sencillo.
- Guían a los estudiantes para explorar las opciones al compartir: hacer público, privado o compartir con grupos.
- Discusión grupal sobre ventajas y desventajas de cada opción.

Organización: Individual

Producto esperado: Captura de pantalla o breve informe describiendo cómo se compartió el proyecto y las opciones exploradas.

Duración estimada: 45 minutos

Actividad 2: Proyecto colaborativo por equipos con división de tareas

Objetivo: Colaborar con compañeros mediante la creación conjunta aplicando técnicas de comunicación y división de tareas.

Descripción paso a paso:

- Formar equipos de 3-4 estudiantes.
- Elegir un tema para crear un proyecto en Scratch (por ejemplo, un juego simple o una animación).
- Definir roles y tareas para cada integrante (programador, diseñador de personajes, narrador, probador).
- Comunicar y coordinar el trabajo mediante reuniones breves o chat de Scratch.
- Crear y combinar las partes del proyecto dentro de la misma cuenta o compartiendo archivos.

Organización: Grupos

Producto esperado: Proyecto colaborativo publicado en Scratch con evidencia de trabajo en equipo.

Duración estimada: 2 sesiones de 60 minutos cada una

Actividad 3: Evaluación entre pares y retroalimentación constructiva

Objetivo: Evaluar y proporcionar retroalimentación constructiva utilizando criterios de funcionamiento y creatividad.

Descripción paso a paso:

- Cada estudiante o grupo presenta su proyecto.
- Se entregan rúbricas sencillas con criterios claros (funcionamiento, creatividad, presentación).
- Los estudiantes revisan proyectos de otros compañeros y llenan la rúbrica.
- Preparan comentarios escritos o verbales con sugerencias respetuosas y concretas.
- Se realiza una sesión de retroalimentación donde se comparten comentarios y se discuten posibles mejoras.

Organización: Individual y grupos

Producto esperado: Rúbrica completada y lista de comentarios para compañeros.

Duración estimada: 60 minutos

Actividad 4: Mejorando un proyecto a partir de sugerencias recibidas

Objetivo: Modificar un proyecto compartido incorporando sugerencias para mejorar funcionalidad y presentación.

Descripción paso a paso:

- Cada estudiante o grupo revisa la retroalimentación recibida.
- Planifican cambios específicos a implementar para mejorar el proyecto.
- Realizan modificaciones en Scratch, mejorando código, diseño o narrativa.
- Publican la versión mejorada y preparan una breve explicación de los cambios realizados.
- Compartir con el docente y compañeros la versión final y reflexionar sobre el proceso.

Organización: Individual o grupos

Producto esperado: Proyecto modificado publicado con explicación de mejoras realizadas.

Duración estimada: 60 minutos

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre compartir y colaborar en proyectos digitales.

Cómo se evalúa: Preguntas orales o cuestionario breve sobre experiencias previas con Scratch o plataformas similares.

Instrumento sugerido: Cuestionario de preguntas cerradas y abiertas, o dinámica grupal de lluvia de ideas.

Evaluación formativa

Qué se evalúa: Progreso en la comprensión y aplicación de funciones para compartir, colaboración y retroalimentación.

Cómo se evalúa: Observación directa durante actividades, revisión de productos intermedios como proyectos compartidos y retroalimentación dada.

Instrumento sugerido: Listas de cotejo para seguimiento de participación, rúbricas para evaluación parcial del proyecto.

Evaluación sumativa

Qué se evalúa: Dominio de los objetivos de la unidad: compartir, colaborar, evaluar y mejorar proyectos en Scratch.

Cómo se evalúa: Análisis del proyecto final modificado, calidad de la retroalimentación proporcionada y reflejo de mejoras implementadas.

Instrumento sugerido: Rúbrica integral que incluya criterios de funcionalidad, creatividad, colaboración y capacidad para incorporar sugerencias.

Unidad 14: Presentación de proyectos

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de organizar y estructurar la información relevante de su proyecto Scratch para elaborar una presentación clara y coherente.
- Al finalizar la unidad, el estudiante será capaz de utilizar herramientas digitales básicas para diseñar diapositivas o materiales visuales que complementen la explicación de su proyecto.
- Al finalizar la unidad, el estudiante será capaz de explicar verbalmente los procesos de programación y las funcionalidades de su proyecto, utilizando un lenguaje adecuado para su audiencia.
- Al finalizar la unidad, el estudiante será capaz de responder preguntas y retroalimentar comentarios relacionados con su proyecto, demostrando comprensión de los conceptos y técnicas aplicadas.
- Al finalizar la unidad, el estudiante será capaz de aplicar técnicas creativas para comunicar los resultados y aprendizajes obtenidos durante el desarrollo de su proyecto en Scratch.

Contenidos Temáticos

1. Organización y estructura de la información del proyecto Scratch

- Identificación de los elementos clave del proyecto: objetivo, funcionalidades, etapas de desarrollo.
- Cómo seleccionar la información relevante para la presentación.
- Estructura básica de una presentación efectiva: introducción, desarrollo y conclusión.

2. Uso de herramientas digitales para la creación de materiales visuales

- Introducción a herramientas digitales básicas para presentaciones (Google Slides, PowerPoint, Canva).

- Diseño de diapositivas: uso de texto, imágenes, colores y animaciones.
- Inserción de capturas de pantalla y videos del proyecto Scratch.

3. Comunicación oral del proyecto y explicación de procesos de programación

- Técnicas para explicar con claridad y lenguaje adecuado.
- Cómo describir las funcionalidades y lógica del proyecto Scratch.
- Prácticas de exposición oral y uso adecuado del tiempo.

4. Interacción con la audiencia: preguntas y retroalimentación

- Estrategias para responder preguntas de forma clara y precisa.
- Cómo recibir y utilizar comentarios para mejorar el proyecto.
- Práctica de diálogo y retroalimentación constructiva entre compañeros.

5. Técnicas creativas para comunicar resultados y aprendizajes

- Uso de recursos visuales y narrativos para hacer la presentación atractiva.
- Incorporación de anécdotas o ejemplos personales relacionados con el proyecto.
- Exploración de formatos creativos: storytelling, infografías, demostraciones en vivo.

Actividades

Actividad 1: "Organizando mi proyecto Scratch para la presentación"

Objetivo: Organizar y estructurar la información relevante del proyecto Scratch para elaborar una presentación clara y coherente.

Descripción:

- El docente explica los elementos clave para organizar la información del proyecto.
- Los estudiantes revisan sus proyectos Scratch y anotan el objetivo, funcionalidades y etapas de desarrollo.
- En grupos pequeños, comparan y seleccionan la información más relevante para comunicar.
- Individualmente, los estudiantes diseñan un esquema o guion para su presentación.

Organización: Individual con trabajo en grupos pequeños

Producto esperado: Esquema o guion escrito para la presentación del proyecto.

Duración estimada: 1 hora

Actividad 2: "Creando diapositivas para mi presentación"

Objetivo: Utilizar herramientas digitales básicas para diseñar diapositivas o materiales visuales que complementen la explicación del proyecto.

Descripción:

- El docente presenta brevemente las funciones principales de una herramienta digital de presentación (p. ej. Google Slides).
- Los estudiantes importan capturas de pantalla o videos de su proyecto Scratch.
- Diseñan diapositivas que incluyan título, texto breve, imágenes y colores, aplicando principios básicos de diseño.
- Al finalizar, cada estudiante guarda y comparte su presentación con el docente.

Organización: Individual

Producto esperado: Presentación digital con diapositivas que complementen su explicación.

Duración estimada: 2 horas

Actividad 3: "Ensayando la exposición oral de mi proyecto"

Objetivo: Explicar verbalmente los procesos de programación y funcionalidades del proyecto usando lenguaje adecuado.

Descripción:

- Los estudiantes preparan una breve exposición usando su guion y presentación digital.
- En parejas, practican la explicación y se dan retroalimentación sobre claridad y lenguaje.
- El docente ofrece consejos para mejorar la comunicación verbal y el uso del tiempo.

Organización: Parejas

Producto esperado: Exposición oral ensayada y mejorada.

Duración estimada: 1.5 horas

Actividad 4: "Presentación final y sesión de preguntas"

Objetivo: Presentar el proyecto, responder preguntas y aplicar técnicas creativas para comunicar resultados y aprendizajes.

Descripción:

- Cada estudiante presenta su proyecto frente al grupo, utilizando sus diapositivas y guion.
- Los compañeros y docente hacen preguntas relacionadas con el proyecto y su desarrollo.
- El expositor responde y recibe retroalimentación para mejorar.
- Se fomenta el uso de técnicas creativas para hacer la presentación atractiva.

Organización: Individual frente al grupo

Producto esperado: Presentación oral completa con sesión de preguntas y respuestas.

Duración estimada: 2 horas

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimiento previo sobre organización de información y uso de herramientas digitales para presentaciones.

Cómo se evalúa: Preguntas orales y breve actividad escrita para identificar expectativas y nivel inicial.

Instrumento sugerido: Cuestionario breve y discusión grupal inicial.

Evaluación formativa

Qué se evalúa: Progreso en la organización de la información, diseño de diapositivas, habilidades de comunicación oral y respuesta a preguntas.

Cómo se evalúa: Observación durante actividades, retroalimentación docente y compañeros, revisión de guion y presentaciones digitales.

Instrumento sugerido: Rúbrica de evaluación formativa con criterios para cada objetivo, lista de cotejo y observación directa.

Evaluación sumativa

Qué se evalúa: Calidad y coherencia de la presentación final, dominio del contenido, uso adecuado de herramientas digitales, claridad en la explicación oral y manejo de preguntas.

Cómo se evalúa: Evaluación mediante rúbrica que considere organización, diseño visual, expresión oral, interacción con la audiencia y creatividad.

Instrumento sugerido: Rúbrica de presentación final con indicadores claros para cada criterio.

Unidad 15: Proyecto final: diseño y desarrollo

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de planificar un proyecto integrador en Scratch que incluya secuencias, ciclos, condiciones y variables básicas, organizando las etapas de desarrollo de manera clara y coherente.
- Al finalizar la unidad, el estudiante será capaz de construir un prototipo funcional de su proyecto integrador en Scratch aplicando los bloques de programación aprendidos y demostrando la integración de conceptos fundamentales.
- Al finalizar la unidad, el estudiante será capaz de aplicar técnicas de depuración para identificar y corregir errores en su proyecto, mejorando su funcionalidad y rendimiento.
- Al finalizar la unidad, el estudiante será capaz de comunicar de forma clara y creativa el proceso de diseño y desarrollo de su proyecto final, utilizando términos técnicos adecuados y soportes visuales dentro del entorno Scratch.
- Al finalizar la unidad, el estudiante será capaz de justificar las decisiones de diseño y programación tomadas durante el desarrollo del proyecto, relacionándolas con los objetivos y problemas planteados.

Contenidos Temáticos

1. Planificación del proyecto integrador en Scratch

- Definición del tema y objetivos del proyecto: Identificación del problema o historia a desarrollar.
- Diseño del storyboard o guion gráfico: Representación visual de las escenas y acciones.
- Organización de las etapas de desarrollo: Desglose en tareas con secuencias, ciclos, condiciones y variables.
- Selección de personajes, escenarios y recursos multimedia: Elección acorde al tema y funcionalidad.

2. Construcción del prototipo funcional en Scratch

- Creación y configuración de sprites: Modificación y personalización.
- Implementación de secuencias de comandos: Orden lógico de instrucciones.
- Uso de estructuras de control: Incorporación de ciclos (repeticiones) y condiciones (decisiones).
- Introducción y manipulación de variables básicas: Almacenamiento y actualización de datos.
- Integración de sonidos y efectos visuales para mejorar la experiencia del usuario.

3. Técnicas de depuración en Scratch

- Identificación de errores comunes en programación visual.
- Uso de herramientas de depuración en Scratch: Observación paso a paso y bloques de prueba.
- Corrección y optimización del código para mejorar funcionalidad y rendimiento.
- Prueba continua del proyecto para asegurar el correcto funcionamiento de cada parte.

4. Comunicación del proceso de diseño y desarrollo

- Documentación clara del proyecto: Explicación del propósito y etapas del desarrollo.
- Uso de términos técnicos adecuados: Secuencia, ciclo, condición, variable, depuración.
- Creación de presentaciones visuales dentro de Scratch: Uso de fondos, mensajes y disfraces para explicar el proyecto.
- Presentación oral y/o escrita del proyecto destacando el proceso creativo y técnico.

5. Justificación de las decisiones de diseño y programación

- Análisis de cómo cada bloque y estructura contribuye al cumplimiento de los objetivos.
- Reflexión sobre los problemas enfrentados y soluciones implementadas.
- Relación entre las funcionalidades implementadas y el problema o historia planteada.
- Evaluación crítica del proyecto para futuras mejoras y aprendizajes.

Actividades

Actividad 1: Planificación detallada del proyecto integrador

Objetivo: Planificar un proyecto integrador en Scratch que incluya secuencias, ciclos, condiciones y variables básicas, organizando las etapas de desarrollo de manera clara y coherente.

Descripción paso a paso:

- Elegir un tema para el proyecto (por ejemplo, un juego sencillo, una historia animada o una simulación).
- Realizar un storyboard que describa las escenas principales, personajes y acciones.
- Listar las funcionalidades que el proyecto deberá tener, identificando el uso de secuencias, ciclos, condiciones y variables.
- Dividir el desarrollo en etapas y asignar tareas específicas para cada parte.

Organización: Individual o parejas.

Producto esperado: Documento o presentación con storyboard, lista de funcionalidades y plan de trabajo organizado.

Duración estimada: 2 horas.

Actividad 2: Construcción del prototipo funcional en Scratch

Objetivo: Construir un prototipo funcional de su proyecto integrador en Scratch aplicando los bloques de programación aprendidos y demostrando la integración de conceptos fundamentales.

Descripción paso a paso:

- Crear los sprites y escenarios definidos en la planificación.
- Programar las secuencias básicas para las acciones de los personajes.
- Incluir ciclos para repetir acciones y condiciones para tomar decisiones en la interacción.
- Agregar variables para manejar puntuaciones, estados o cualquier dato relevante.
- Incorporar sonidos y efectos visuales para mejorar la experiencia.

Organización: Individual o parejas.

Producto esperado: Prototipo funcional en Scratch con integración de secuencias, ciclos, condiciones y variables.

Duración estimada: 4 horas.

Actividad 3: Depuración y mejora del proyecto

Objetivo: Aplicar técnicas de depuración para identificar y corregir errores en su proyecto, mejorando su funcionalidad y rendimiento.

Descripción paso a paso:

- Probar el proyecto para detectar fallos o comportamientos inesperados.
- Usar la función de ejecución paso a paso para observar el flujo del programa.
- Identificar errores lógicos o de sintaxis visual.
- Realizar las correcciones necesarias y optimizar el código.
- Registrar los cambios realizados y justificar las mejoras.

Organización: Individual.

Producto esperado: Proyecto corregido con documentación de errores y soluciones aplicadas.

Duración estimada: 2 horas.

Actividad 4: Presentación y justificación del proyecto final

Objetivo: Comunicar de forma clara y creativa el proceso de diseño y desarrollo de su proyecto final, utilizando términos técnicos adecuados y soportes visuales dentro del entorno Scratch, además de justificar las decisiones de diseño y programación tomadas.

Descripción paso a paso:

- Preparar una presentación que incluya el propósito del proyecto, etapas de desarrollo y funcionalidades implementadas.
- Explicar el uso de secuencias, ciclos, condiciones y variables en el proyecto.
- Mostrar dentro del proyecto Scratch cómo funcionan las partes clave del código.
- Justificar las decisiones tomadas en diseño y programación, relacionándolas con los objetivos y problemas planteados.
- Responder preguntas del docente o compañeros para demostrar comprensión.

Organización: Individual o parejas.

Producto esperado: Presentación oral y visual del proyecto con soporte en Scratch.

Duración estimada: 2 horas.

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre planificación de proyectos en Scratch y uso básico de secuencias, ciclos, condiciones y variables.

Cómo se evalúa: Cuestionario corto y discusión guiada sobre experiencias previas con proyectos en Scratch.

Instrumento sugerido: Cuestionario digital o en papel con preguntas de opción múltiple y abiertas.

Evaluación formativa

Qué se evalúa: Progreso en la planificación, construcción, depuración y documentación del proyecto.

Cómo se evalúa: Revisión continua de los productos parciales (storyboard, prototipo, registro de depuración) y retroalimentación durante las actividades.

Instrumento sugerido: Lista de cotejo que incluya criterios de planificación clara, correcta implementación de bloques, identificación y corrección de errores, y calidad de la documentación.

Evaluación sumativa

Qué se evalúa: Producto final completo, presentación y justificación del proyecto integrador.

Cómo se evalúa: Evaluación del proyecto en Scratch, presentación oral y escrita, y defensa de las decisiones de diseño y programación.

Instrumento sugerido: Rúbrica que valore funcionalidad, integración de conceptos (secuencias, ciclos, condiciones, variables), calidad de la presentación, uso de terminología técnica y capacidad de justificación.

Unidad 16: Proyecto final: presentación y evaluación

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de presentar su proyecto integrador utilizando los elementos fundamentales de Scratch de manera clara y organizada.
- Al finalizar la unidad, el estudiante será capaz de evaluar su propio proyecto y el de sus compañeros aplicando criterios de funcionalidad, creatividad y uso correcto de secuencias, ciclos, condiciones y variables.
- Al finalizar la unidad, el estudiante será capaz de proporcionar retroalimentación constructiva y sugerencias de mejora basadas en la observación y análisis de los proyectos presentados.
- Al finalizar la unidad, el estudiante será capaz de reflexionar sobre su proceso de aprendizaje y describir las estrategias utilizadas para resolver problemas durante el desarrollo del proyecto.

Contenidos Temáticos

1. Presentación del proyecto integrador en Scratch

- Preparación para la presentación: organización del proyecto y elementos a destacar
- Uso de elementos fundamentales de Scratch en la presentación: secuencias, ciclos, condiciones y variables
- Técnicas para comunicar el funcionamiento y propósito del proyecto de manera clara y ordenada

2. Evaluación de proyectos: criterios y aplicación

- Definición y explicación de criterios de evaluación: funcionalidad, creatividad y uso correcto de conceptos de programación
- Autoevaluación: análisis crítico del propio proyecto según los criterios establecidos
- Evaluación entre pares: observación y valoración de proyectos de compañeros aplicando los mismos criterios

3. Retroalimentación constructiva y sugerencias de mejora

- Principios para proporcionar retroalimentación respetuosa y constructiva
- Técnicas para identificar fortalezas y áreas de mejora en los proyectos observados
- Formulación de sugerencias prácticas para mejorar aspectos técnicos y creativos del proyecto

4. Reflexión sobre el proceso de aprendizaje

- Identificación de estrategias utilizadas para resolver problemas durante el desarrollo del proyecto
- Reconocimiento de aprendizajes adquiridos y dificultades superadas
- Elaboración de un breve informe o presentación personal sobre la experiencia de aprendizaje

Actividades

Presentación oral y demostración del proyecto integrador

Objetivo: Presentar el proyecto integrador utilizando elementos fundamentales de Scratch de manera clara y organizada.

Descripción:

- Cada estudiante prepara una presentación de 5 minutos explicando su proyecto.
- Debe mostrar cómo se utilizan secuencias, ciclos, condiciones y variables en su proyecto.
- Demuestra el proyecto en Scratch y explica su funcionamiento y propósito.
- Los demás estudiantes escuchan y toman notas para la evaluación entre pares.

Organización: Individual

Producto esperado: Presentación oral y demostración funcional del proyecto.

Duración estimada: 45 minutos (dependiendo del número de estudiantes, 5 minutos por presentación).

Evaluación entre pares con rúbrica

Objetivo: Evaluar el propio proyecto y el de los compañeros aplicando criterios de funcionalidad, creatividad y uso correcto de conceptos de programación.

Descripción:

- Se entrega a cada estudiante una rúbrica con criterios claros para evaluar proyectos: funcionalidad, creatividad, uso de secuencias, ciclos, condiciones y variables.
- Después de cada presentación, los estudiantes completan la rúbrica evaluando el proyecto presentado.
- Se realiza una discusión grupal para comentar observaciones y consolidar la evaluación.

Organización: Individual y grupal

Producto esperado: Rúbricas evaluadas y resumen de puntos fuertes y áreas de mejora para cada proyecto.

Duración estimada: 60 minutos.

Sesión de retroalimentación constructiva

Objetivo: Proporcionar retroalimentación constructiva y sugerencias de mejora basadas en la observación y análisis de los proyectos presentados.

Descripción:

- En grupos pequeños, cada estudiante comparte las observaciones y sugerencias que anotó durante la evaluación entre pares.
- Se guía a los estudiantes para formular comentarios positivos y sugerencias concretas para mejorar los proyectos.
- Cada estudiante recibe retroalimentación y puede anotar ideas para mejorar su proyecto.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Lista de sugerencias de mejora para cada proyecto.

Duración estimada: 40 minutos.

Reflexión individual sobre el proceso de aprendizaje

Objetivo: Reflexionar sobre el proceso de aprendizaje y describir las estrategias utilizadas para resolver problemas durante el desarrollo del proyecto.

Descripción:

- Los estudiantes redactan un informe breve o preparan una presentación digital donde describen:
  - Los desafíos encontrados durante la creación del proyecto.
  - Las estrategias que usaron para solucionar problemas.
  - Los aprendizajes más importantes que obtuvieron.
- Se pueden incluir ejemplos específicos de código o etapas del proyecto.

Organización: Individual

Producto esperado: Informe o presentación de reflexión personal.

Duración estimada: 45 minutos.

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimiento previo sobre presentación de proyectos y criterios básicos de evaluación en Scratch.

Cómo se evalúa: Preguntas orales y discusión grupal para identificar expectativas y experiencias previas.

Instrumento sugerido: Lista de cotejo o preguntas guía para discusión.

Evaluación formativa

Qué se evalúa: Progreso en la presentación del proyecto, aplicación de criterios de evaluación y calidad de retroalimentación.

Cómo se evalúa: Observación directa durante las presentaciones, revisión de rúbricas completadas y la calidad de las sugerencias dadas en grupos.

Instrumento sugerido: Rúbrica para presentación y evaluación entre pares, registro de observación docente.

Evaluación sumativa

Qué se evalúa: Calidad final del proyecto presentado, autoevaluación, evaluación de pares y reflexión individual.

Cómo se evalúa: Calificación basada en la rúbrica de evaluación del proyecto, revisión del informe de reflexión y análisis de la autoevaluación.

Instrumento sugerido: Rúbrica detallada para el proyecto integrador, formato de autoevaluación y plantilla para reflexión escrita.