Domina las condiciones

Grado 7° Guía 1

TIC

Estudiantes

Apoya:

Educación

BRITISH COUNCIL

Colombia Programa
{EL CÓDIGO A TU FUTURO}

Domina las condiciones

Grado 7° Guía 1

TIC

Estudiantes

Educación

BRITISH COUNCIL

Colombia Programa
{EL CÓDIGO A TU FUTURO}

MINISTERIO DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES

Julián Molina Gómez
Ministro TIC

Luis Eduardo Aguiar Delgadillo
Viceministro (e) de Conectividad

Yeimi Carina Murcia Yela
Viceministra de Transformación Digital

Óscar Alexander Ballen Cifuentes
Director (e) de Apropiación de TIC

Alejandro Guzmán
Jefe de la Oficina Asesora de Prensa

Equipo Técnico
Lady Diana Mojica Bautista
Cristhiam Fernando Jácome Jiménez
Ricardo Cañón Moreno

Consultora experta
Heidy Esperanza Gordillo Bogota

BRITISH COUNCIL

Felipe Villar Stein
Director de país

Laura Barragán Montaña
Directora de programas de Educación, Inglés y Artes

Marianella Ortiz Montes
Jefe de Colegios

David Vallejo Acuña
Jefe de Implementación Colombia Programa

Equipo operativo
Juanita Camila Ruiz Díaz
Bárbara De Castro Nieto
Alexandra Ruiz Correa
Dayra Maritza Paz Calderón
Saúl F. Torres
Óscar Daniel Barrios Díaz
César Augusto Herrera Lozano
Paula Álvarez Peña

Equipo técnico
Alejandro Espinal Duque
Ana Lorena Molina Castro
Vanesa Abad Rendón
Raisa Marcela Ortiz Cardona
Juan Camilo Londoño Estrada

Edición y coautoría versiones finales
Alejandro Espinal Duque
Ana Lorena Molina Castro
Vanesa Abad Rendón
Raisa Marcela Ortiz Cardona

Edición
Juanita Camila Ruiz Díaz
Alexandra Ruiz Correa

British Computer Society – Consultoría internacional

Niel McLean
Jefe de Educación

Julia Adamson
Directora Ejecutiva de Educación

Claire Williams
Coordinadora de Alianzas

Asociación de facultades de ingeniería - ACOFI

Edición general
Mauricio Duque Escobar

Coordinación pedagógica
Margarita Gómez Sarmiento
Mariana Arboleda Flórez
Rafael Amador Rodríguez

Coordinación de producción
Harry Luque Camargo

Asesoría estrategia equidad
Paola González Valcárcel

Asesoría primera infancia
Juana Carrizosa Umaña

Autoría
Arlet Orozco Marbello
Harry Luque Camargo
Isabella Estrada Reyes
Lucio Chávez Mariño
Margarita Gómez Sarmiento
Mariana Arboleda Flórez
Mauricio Duque Escobar
Paola González Valcárcel
Rafael Amador Rodríguez
Rocío Cardona Gómez
Saray Piñerez Zambrano
Yimzay Molina Ramos

PUNTOAPARTE EDITORES

Diseño, diagramación, ilustración, y revisión de estilo

Impreso por Panamericana Formas e Impresos S.A., Colombia

Material producido para Colombia Programa, en el marco del convenio 1247 de 2023 entre el Ministerio de Tecnologías de la Información y las Comunicaciones y el British Council

"Esta guía corresponde a una versión preliminar en proceso de revisión y ajuste. La versión final actualizada estará disponible en formato digital y puede incluir modificaciones respecto a esta edición"

Prólogo

Estimados educadores, estudiantes y comunidad educativa:

En el Ministerio de Tecnologías de la Información y las Comunicaciones, creemos que la tecnología es una herramienta poderosa para incluir y transformar, mejorando la vida de todos los colombianos. Nos guia una visión de tecnología al servicio de la humanidad, ubicando siempre a las personas en el centro de la educación técnica.

Sabemos que no habrá progreso real si no garantizamos que los avances tecnológicos beneficien a todos, sin dejar a nadie atrás. Por eso, nos hemos propuesto una meta ambiciosa: formar a un millón de personas en habilidades que les permitan no solo adaptarse al futuro, sino construirlo con sus propias manos. Hoy damos un paso fundamental hacia este objetivo con la presentación de las guías de pensamiento computacional, un recurso diseñado para llevar a las aulas herramientas que fomenten la creatividad, el pensamiento crítico y la resolución de problemas.

Estas guías no son solo materiales educativos; son una invitación a imaginar, cuestionar y crear. En un mundo cada vez más impulsado por la inteligencia artificial, desarrollar habilidades como el pensamiento computacional se convierte en la base, en el primer acercamiento para que las y los ciudadanos aprendan a programar y solucionar problemas de forma lógica y estructurada.

Estas guías han sido diseñadas pensando en cada región del país, con actividades accesibles que se adaptan a diferentes contextos, incluyendo aquellos con limitaciones tecnológicas. Esta es una apuesta por la equidad, por cerrar las brechas y asegurar que nadie se quede atrás en la revolución digital. Quiero destacar, además, que son el resultado de un esfuerzo colectivo: más de 2.000 docentes colaboraron en su elaboración, compartiendo sus ideas y experiencias para que este material realmente se ajuste a las necesidades de nuestras aulas. Además, con el apoyo del British Council y su red de expertos internacionales, hemos integrado prácticas globales de excelencia adaptadas a nuestra realidad nacional.

Hoy presentamos un recurso innovador y de alta calidad, diseñado en línea con las orientaciones curriculares del Ministerio de Educación Nacional. Cada página de estas guias invita a transformar las aulas en espacios participativos, creativos y, sobre todo, en ambientes donde las y los estudiantes puedan desafiar estereotipos y explorar nuevas formas de pensar.

Trabajemos juntos para garantizar que cada estudiante, sin ·importar dónde se encuentre, tenga acceso a las herramientas necesarias para imaginar y construir un futuro en el que todos seamos protagonistas del cambio. Porque la tecnología debe ser un instrumento de justicia social, y estamos comprometidos a que las herramientas digitales ayuden a cerrar brechas sociales y económicas, garantizando oportunidades para todos.

Con estas guias, reafirmamos nuestro compromiso con la democratización de las tecnologías y el desarrollo rural, porque creemos en el potencial de cada región y en la capacidad de nuestras comunidades para liderar el cambio.

Julián Molina Gómez
Ministro de Tecnologías de la Información y las Comunicaciones
Gobierno de Colombia

Grado 7º Guía 1

TIC

Guía de íconos

Lógica, programación y depuración

Aprendizajes de la guía

Con las actividades de esta guía se espera que puedas avanzar en:

- Usar condicionales simples, dobles y anidados dentro de bucles para controlar el flujo de un programa.
- Seleccionar y establecer rangos en secuencias numéricas para realizar acciones específicas.
- Diseñar, depurar y optimizar algoritmos mediante el uso de operadores lógicos AND y OR, agrupando condiciones para determinar diferentes acciones.

Resumen de la guía

Esta guía presenta cinco sesiones de trabajo dedicadas al aprendizaje de condicionales simples, dobles y anidados, así como al uso de los operadores lógicos “y” (AND) y “o” (OR) para optimizar la ejecución de algoritmos que controlan objetos en el entorno de programación *Scratch*.

Incluye actividades tanto teóricas como prácticas, utilizando *Scratch* para aplicar condicionales y operadores lógicos en la creación de videojuegos.

Resumen de las sesiones

Sesión 1

Se introducen los conceptos de condicionales simples y dobles mediante actividades prácticas. Se aplican condicionales dentro de bucles “por siempre” y “repetir hasta que”, observando cómo una o dos acciones pueden depender de una sola condición.

Sesión 2

Se explora el uso de condicionales anidados. Se establecen situaciones donde una acción depende de varias condiciones. Se trabaja con ejemplos de la vida real para entender el flujo anidado de datos tratados en variables.

Aprendizajes de la guía

- Controlar la ejecución de un programa o algoritmo usando condicionales, variables numéricas y booleanas y operadores de comparación (mayor, menor, igual).

Sesión 3

Se seleccionan rangos en secuencias numéricas. Se establecen rangos de números consecutivos dentro de una secuencia, aplicando estos conceptos en situaciones prácticas y ejemplos relevantes.

Sesión 4

Se optimizan algoritmos mediante el uso de operadores lógicos AND y OR. Se aprende cómo agrupar varias condiciones para determinar diferentes acciones, haciendo más eficiente el flujo de datos en los programas.

Sesión 5

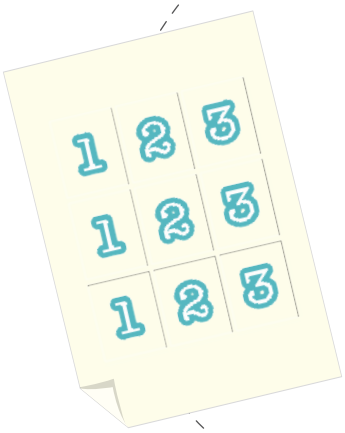
Se aplica lo aprendido en las sesiones anteriores al desarrollo de un videojuego. Se usan condicionales simples, dobles y anidados, así como operadores lógicos, para diseñar y programar un videojuego.

Conexión con otras áreas

A continuación se presenta la conexión con otras áreas:

Matemáticas

- Se da una conexión con la matemática en la utilización de rangos de secuencias numéricas y en el uso de operadores lógicos.



Sesión 1

Aprendizajes esperados

Al final de esta sesión verifica que puedas:

- Aplicar los conceptos de condicionales simples y dobles en la programación en *Scratch*.
- Demostrar usando *Scratch* cómo una o dos acciones pueden depender de una sola condición.
- Desarrollar actividades prácticas que demuestren el uso de condicionales simples y dobles dentro de bucles.

Duración sugerida

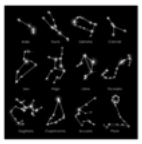
20% 40% 40%

Material para la clase

- ○ Anexo 1.1.
- ○ Acceso a *Scratch*.

Anexo

Anexo 1.1



El zodiaco utilizado por astrólogos y astrólogas es, esencialmente, de origen caldeo. Fue evolucionando de forma muy gradual y a menudo fortuita durante los dos milenios que precedieron a nuestra era, con aportes de numerosas civilizaciones (Mesopotamia, Babilonia, Grecia, Roma, Persia, entre otros) todas ellas con la necesidad de encontrar en lo que se veía en el firmamento una relación con las creencias del momento y crear calendarios. Interesarse por esta construcción lenta y progresiva y buscar las razones que empujó a la humanidad a modificar y perfeccionar el modelo establecido según los conocimientos de la época, ayuda a poner en perspectiva las afirmaciones no científicas de la astrología. Cada signo del zodiaco se asocia a imágenes imaginarias en el firmamento que se asemejan a figuras. Existen y han existido muchos horóscopos, esta actividad utiliza el denominado occidental, pero se pueden mencionar otros como el chino, el maya, el azteca o el árabe, entre muchos otros.

Tu reto es construir un programa que le permita saber a cada persona, sabiendo su mes y día de nacimiento, qué signo le corresponde según esta tradición del horóscopo occidental.

La siguiente tabla muestra la asignación del signo en los horóscopos actuales:

Entre el 21 de marzo y el 19 de abril	Aries
Entre el 20 de abril y el 20 de mayo	Tauro
Entre el 21 de mayo y el 20 de junio	Géminis
Entre el 21 de junio y el 22 de julio	Cáncer
Entre el 23 de julio y el 22 de agosto	Leo
Entre el 23 de agosto y el 22 de septiembre	Virgo
Entre el 23 de septiembre y el 23 de octubre	Libra
Entre el 24 de octubre y el 20 de noviembre	Escorpión
Entre el 22 de noviembre y el 21 de diciembre	Sagitario
Entre el 22 de diciembre y el 19 de enero	Capricornio
Entre el 20 de enero y el 18 de febrero	Acuario
Entre el 19 de febrero y el 20 de marzo	Piscis

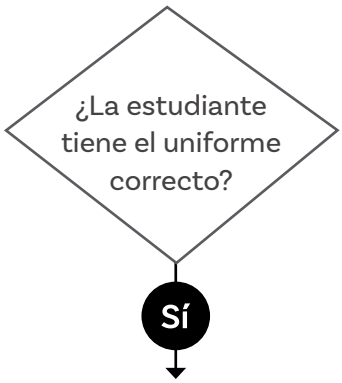
Lo que sabemos, lo que debemos saber

Esta sección corresponde al 20% de avance de la sesión

Empieza por leer el reto que te proponemos en esta guía, el cual se encuentra en el *Anexo 1.1*. Recuerda que lo resolverás hasta la sesión final.

Entender cómo funcionan las condicionales y los operadores lógicos es muy importante en programación porque te ayuda a tomar decisiones dentro de tu código. En *Scratch*, estas herramientas son esenciales para que puedas controlar lo que pasa en tus programas. Con ellas, puedes controlar lo que hacen los personajes y los objetos, haciendo que respondan de diferentes maneras según lo que esté ocurriendo, consiguiendo que todo funcione tal como lo habías imaginado.

Volvamos a analizar el siguiente ejemplo que alguna vez trabajamos en la Guía 3 de grado 5 donde se vio el concepto de condicionales:



Puede ingresar al colegio

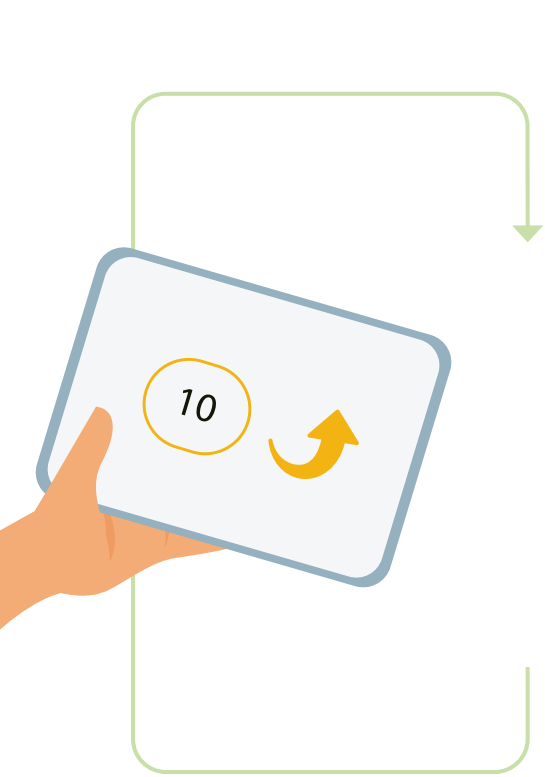
También puedes expresar estas condicionales con expresiones que usan las palabras "***Si***" y "***entonces***" como en el ejemplo:

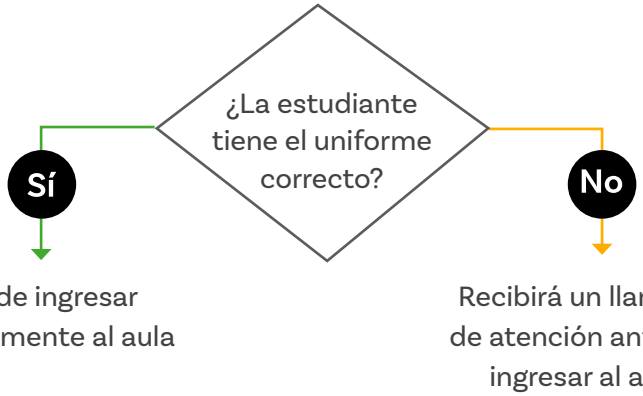
- *Si* la estudiante tiene el uniforme correcto, *entonces* puede entrar al aula.

Fíjate que en este caso hay una única acción "puede entrar al aula" como consecuencia afirmativa de la condición "¿La estudiante tiene el uniforme correcto?"

A este tipo de condicionales se les conoce como condicionales simples y son aquellas a las que, frente a una pregunta de "Si" o "No" únicamente tienen consecuencia o acción en el caso que la respuesta sea afirmativa.

Por otra parte, existen las condicionales dobles en cuyo caso se toma también acción en el caso que la respuesta sea negativa.





Estas condicionales al igual que las condicionales simples se pueden expresar de la siguiente manera:

- *Si* la estudiante tiene el uniforme correcto, *entonces* puede entrar directamente al aula; *si no, entonces* recibirá un llamado de atención antes de ingresar al aula.

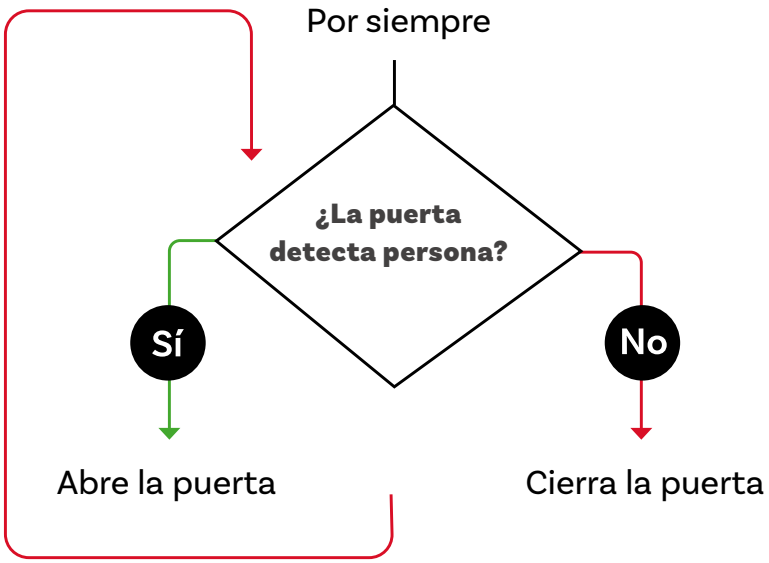
Por otra parte, las condicionales simples y dobles también se aplican a los bucles o ciclos de repetición, por ejemplo, se pueden repetir acciones relacionadas con una condicional. Analiza el comportamiento de las puertas automáticas de los centros comerciales que permanentemente funcionan sin asistencia humana.

- *Si* la puerta detecta personas, *entonces* la puerta se abre; *si no, entonces* la puerta se cierra.

En este caso esta instrucción solamente se estaría ejecutando una sola vez. Si queremos que la puerta funcione permanentemente deberemos combinar la condicional con el bucle o ciclo "Por Siempre", como se ve en la *Figura 1*.

- *Por Siempre*
  Si la puerta detecta personas, *entonces* la puerta se abre; *si no, entonces* la puerta se cierra.

Figura 1. Diagrama de flujo para detector de personas



Vamos a programar este ejemplo usando *Scratch* donde usaremos los ciclos “Repetir” y “Por siempre”, como se muestra en la ver *Figura 2*.

Figura 2. Bloques de control “repetir” y “por siempre”

Nota

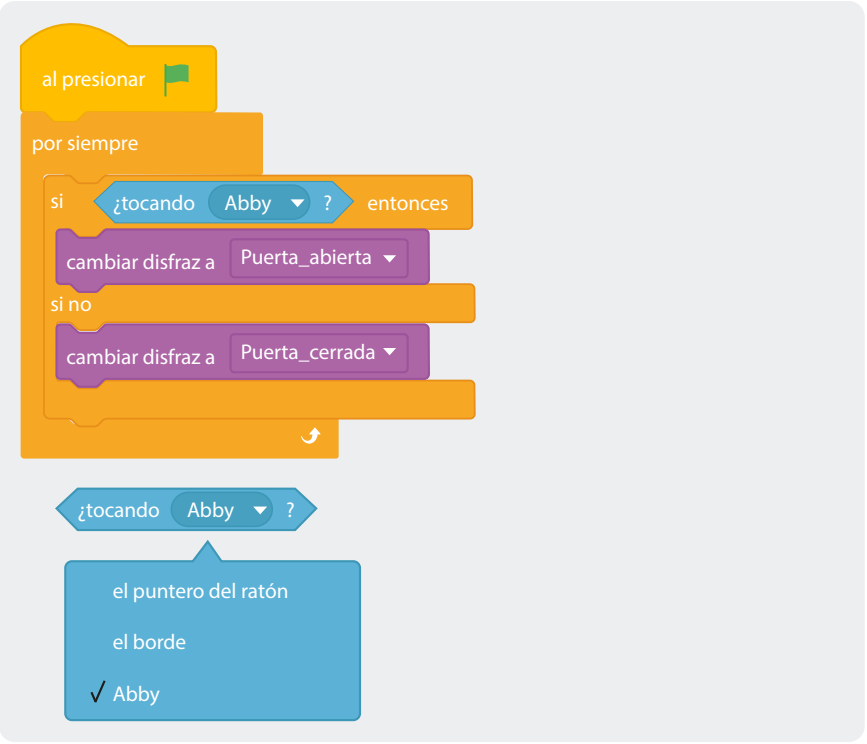
Ubica la pregunta condicional en la paleta de sensores.

Enlace

Acceso a *Scratch*.

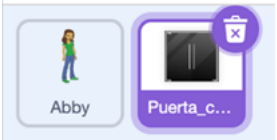
Una vez abierto *Scratch* encontrarás dos objetos "Abby" y "Puerta", haz clic en el objeto "Puerta" y replica el código, como se ve en la *Figura 3*.

Figura 3. Programación para abrir puerta al tocar



Tan pronto repliques el código haz clic en la bandera verde. Posteriormente, arrastra a Abby hacia la puerta usando el ratón, como se ve en la *Figura 4*.

Figura 4. Ícono de Aby y puerta



¿Qué ocurre con la puerta cuando Abby la toca? ¿Qué ocurre con la puerta cuando Abby deja de tocarla? ¿Qué ocurre con la puerta cuando eliminamos de la programación el ciclo "Por Siempre" y solo dejamos la condicional? ¿Sigue funcionando el programa?

Manos a la obra

Conectadas

Esta sección corresponde al 60% de avance de la sesión

Organízate en grupos siguiendo las recomendaciones de tu docente.

En grupos, van a usar *Scratch* para controlar con condicionales un gato que atrapa una manzana. Una parte la harán siguiendo instrucciones y luego deben completar el código para conseguir que el programa funcione correctamente. Sigan las instrucciones y luego hagan las variaciones sugeridas al código.

Instrucciones: en grupo, utilicen el ciclo "Repetir hasta que" para hacer un juego en *Scratch* donde apliquen lo aprendido.

1. Crear un nuevo proyecto desde *Scratch* siguiendo la ruta Archivo --> Nuevo.

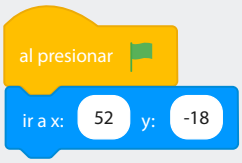
2. Elegir un nuevo objeto, por ejemplo, Gato 2 "Cat 2", como se ve en la *Figura 5*.

Figura 3. Eleccion de objeto en *Scratch*

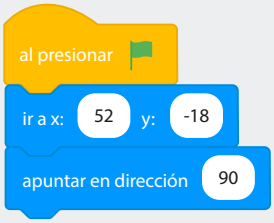
3. Eliminar el gato "Objeto 1" que viene por defecto.

4. En la pestaña código del objeto Gato 2 “Cat 2” vayan a la paleta “Eventos” y arrastren al lienzo de programación la opción “Al presionar bandera verde”.

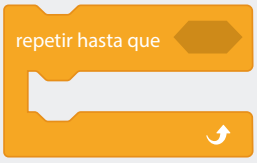
5. Ir a la paleta “Movimiento” y fijar la posición de “Cat 2” en el centro de la pantalla con la opción “ir a”.



6. En esta misma paleta fijar la posición de “Cat 2” a 90 grados con la opción “Apuntar en dirección”.

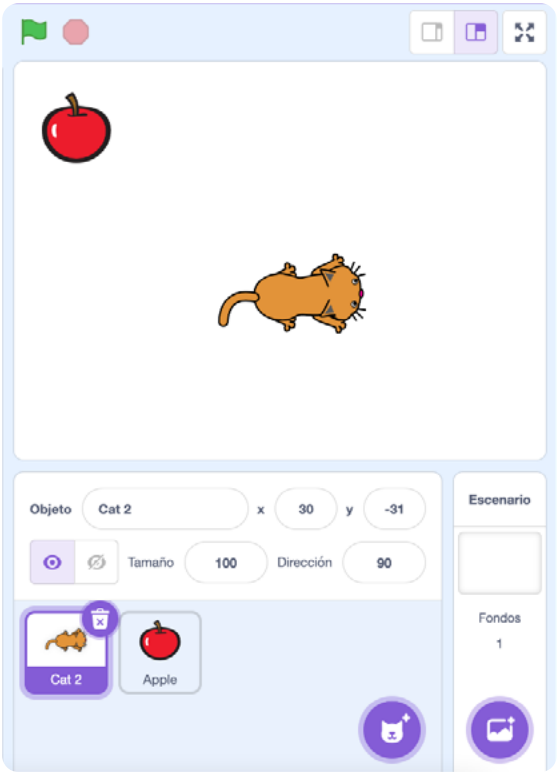


7. Ir a la paleta “Control” y ensamblar la opción “Repetir hasta que”.



8. Ir por un nuevo objeto, por ejemplo, Manzana, “Apple” y arrastrar con el cursor a alguna esquina de la pantalla, como se ve en la *Figura 6*.

Figura 6. Agregar objetos en zona de programación



9. Dar clic nuevamente en Gato 2 “Cat 2” y continuar codificando. Ahora ir a la paleta “Sensores” y ensamblar la opción “¿Tocando Apple?” en el ciclo “Repetir hasta que”.

10. De la misma paleta “Control” seleccionar y ensamblar dentro del ciclo “Repetir hasta que” una condicional simple “Si Entonces”.

11. Ir a la paleta “Sensores”, seleccionar y ensamblar en la condicional la opción “¿Tecla (Flecha arriba) Presionada?”.

12. Dentro de la condicional apuntar a Gato 2 “Cat 2” en dirección a 0 grados y ensamblar “Mover (10) pasos”.

13. Ir a la paleta “Apariencia” y ensamblar al finalizar el ciclo “Repetir hasta que” el mensaje “Decir (“GANASTE”) durante (5) segundos”, como se ve en la *Figura 7*.

Figura 7. Agregar bloque de mensaje

Ejecutar el código oprimiendo la bandera verde y luego mover el Gato 2 “Cat 2” oprimiendo la flecha de arriba del teclado.

Ahora, a modificar el código que hicieron:

¿Qué tendrían que hacer al código para que Gato 2 “Cat 2” llegue a Manzana “Apple”?

¡Completen el código para ganar!, tal y como se ve en la *Figura 8.*

Figura 8. Agregar condicional

Completen el código para que, sin importar dónde ubiquen a Manzana "*Apple*" en pantalla, puedan llevar a Gato 2 "Cat 2" con las flechas del teclado.

Para ir más lejos

Si aún tienes tiempo y quieres profundizar, te proponemos completar la siguiente actividad:

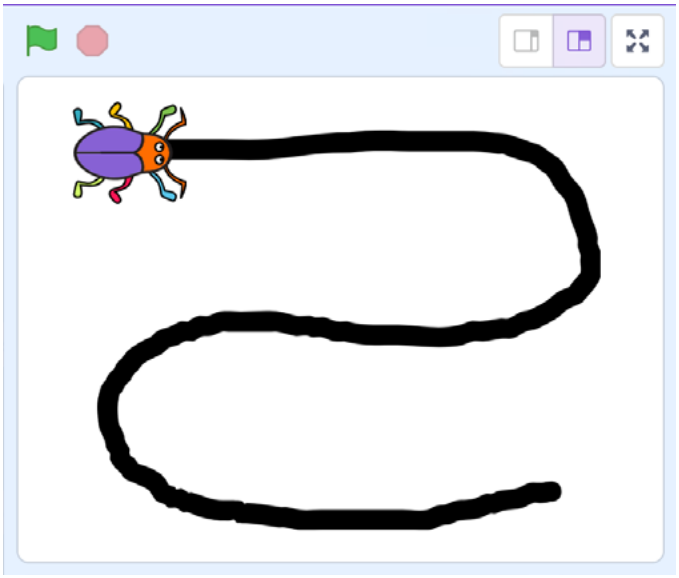
Hasta este momento hemos trabajado en el uso de las condicionales para controlar el funcionamiento de una puerta automática y también para controlar los movimientos de un gato que persigue una manzana, el control con el uso de condicionales no solamente es utilizado en programación de juegos de computadora, sino que también es ampliamente utilizado en el mundo de la robótica. Hay competencias mundiales de robots que siguen líneas.

Enlace

Acceso al proyecto

Ahora es tu turno de crear un programa para controlar los movimientos de un insecto que debe seguir un recorrido marcado por una línea negra. Programa el insecto sigue líneas usando la misma lógica de los ejercicios hasta ahora realizados en esta Guía. En la *Figura 9* puedes ver un ejemplo.

Figura 9. Programar insecto que sigue línea



Antes de irnos

Esta sección corresponde al 100% de avance de la sesión

De forma individual, regresa a revisar los aprendizajes esperados. Elije la opción de respuesta que mejor describa lo que alcanzaste.

1. ¿Puedes usar condicionales simples para cuando solo ocurre un evento o acción que dependa de la respuesta afirmativa o negativa a una pregunta en *Scratch*?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ Aún no

2. ¿Puedes hacer que sucedan uno o dos eventos o acciones diferentes en *Scratch* dependiendo de la respuesta afirmativa o negativa de una pregunta condicional?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ No

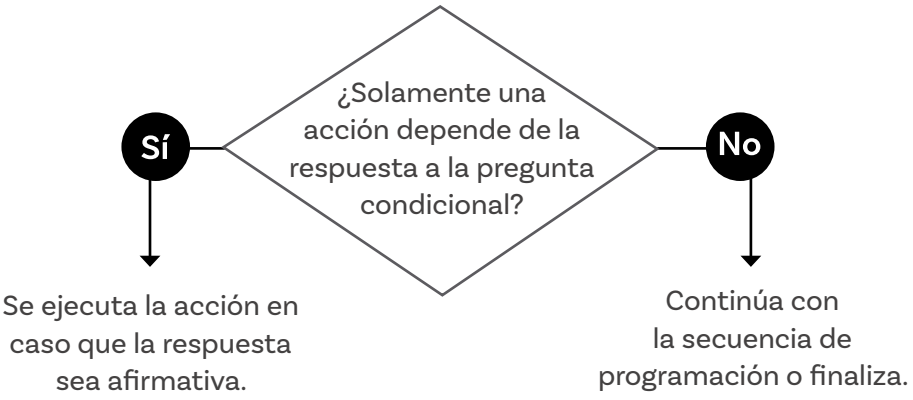
3. ¿Puedes usar condicionales entre ciclos para que una o varias acciones se repitan de acuerdo con el resultado de la respuesta afirmativa o negativa de dichos condicionales?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ No

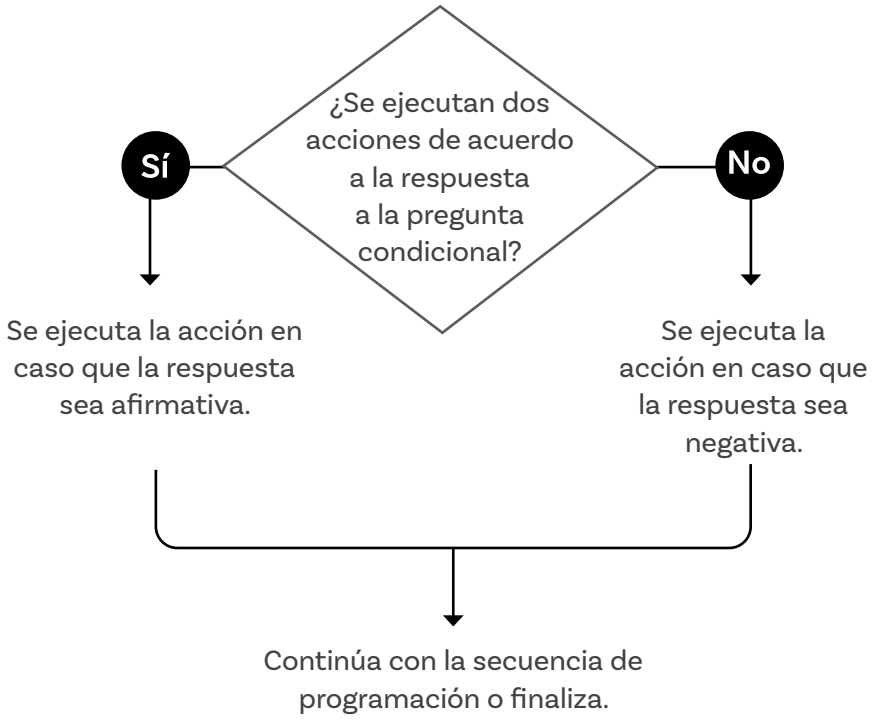
Si tus respuestas fueron "Parcialmente" o "Aún no", vuelve a las actividades propuestas en Scratch. Luego, discute con tus compañeras y compañeros de grupo lo que se hizo en cada momento de la actividad y el rol al que correspondía. Si todavía te quedan dudas, consúltale a tu docente.

Ahora revisa lo que has aprendido y pregúntate, ¿en qué te servirá para resolver el reto que se plantea en el *Anexo 1.1*?

Condicionales simples



Condicionales dobles



Sesión 2

Aprendizajes esperados

Al final de esta sesión verifica que puedas:

- Usar condicionales anidados para ejecutar más de dos acciones específicas en *Scratch*.
- Identificar situaciones donde una acción depende de varias condiciones.
- Implementar ejemplos prácticos requieran el uso de condicionales anidados.

Duración sugerida

20% 40% 40%

Material para la clase

- ○ Anexo 2.1.
- ○ Anexo 2.2.
- ○ Anexo 2.3.
- ○ Acceso a *Scratch*.

Lo que sabemos, lo que debemos saber

Esta sección corresponde al 20% de avance de la sesión

Las condicionales anidadas son estructuras de control donde se coloca una condición dentro de otra. Esto significa que la ejecución de un evento o acción depende de múltiples condiciones. Las condicionales anidadas permiten evaluar más de una condición y ejecutar diferentes acciones basadas en los resultados de estas evaluaciones.

Piensa en la siguiente situación cotidiana que puede describirse con el uso de condicionales anidadas, para que se ejecute la acción "Fuiste promovido de sexto a séptimo año escolar", deben darse una serie de circunstancias que pueden plantearse como preguntas cuyo resultado son "Sí" o "No" o "Afirmativo" o "Negativo".

Condiciones:

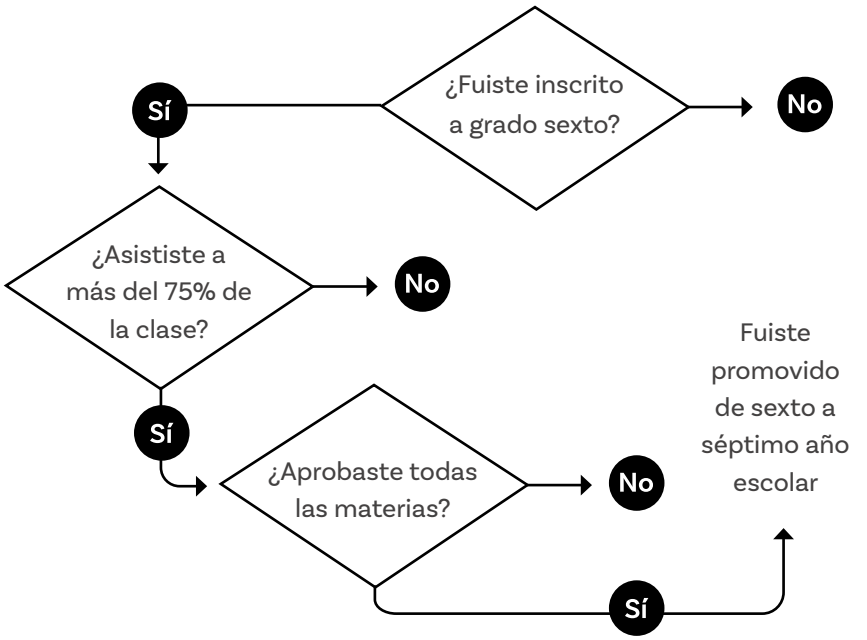
- ¿Aprobaste todas las asignaturas?
- ¿Fuiste inscrito a grado sexto?
- ¿Asististe más del 75% de las clases?

Estas preguntas pueden organizarse en un orden lógico en el cual una se plantea dentro de la otra; es decir no tendría sentido plantear algunas preguntas antes que otras, pues dependen de una respuesta anterior. En este ejemplo no tiene sentido preguntar "¿aprobaste todas las asignaturas?" sin antes preguntar "¿fuiste inscrito a grado sexto?" o "¿asististe más del 75% de las clases?" debido a que si la respuesta de las últimas dos preguntas es negativa es lógico que no apruebe todas las asignaturas.

Observemos en este diagrama el control de flujo que demuestra la lógica de este caso.

Anexo

Anexo 2.1



Manos a la obra
Desconectadas

Esta sección corresponde al 60% de avance de la sesión

Organízate en grupos según las indicaciones de tu docente. Requerirás de las cartas recortadas del *Anexo 2.1*.

Juguemos "Adivina el número secreto", el objetivo es descubrir un número oculto que un repartidor ha seleccionado al azar entre 1 y 5. A través de varias rondas, harán suposiciones y comparaciones para encontrar el número correcto, utilizando la lógica y las pistas que te dé el repartidor. A la primera carta que sacan de un mazo con los números entre 1 y 5 la llamaremos "suposición" y si es menor al número secreto cada jugador podrá sumar otra carta para adivinar el número. ¡Es un desafío divertido que pone a prueba tu habilidad para pensar de forma estratégica! Ahora, agrúpense en equipos de cinco y sigan las instrucciones para comenzar.

Sigue los pasos:

1. Agruparse de a 5 participantes, escoger al o la repartidor(a).
2. Cada estudiante del grupo debe contar con las cartas marcadas con los números del 1 al 5.

3. El o la repartidora, quien también cuenta con un mazo de cartas, escoge una carta aleatoriamente.

4. Todas las y los jugadores ponen su "Suposición" sobre la mesa al mismo tiempo.

5. El o la repartidora indica qué jugadores siguen en el juego.

6. Al mismo tiempo las y los jugadores que continúan ponen su segundo número en la mesa.

7. El o la repartidora les indica a los jugadores quiénes adivinaron el número secreto o no.

8. Posteriormente intercambian el rol de repartidor(a) y el juego termina cuando todos hayan pasado por el rol de repartidor(a).

Reflexiona sobre esto después de jugar:

¿?

¿Qué pregunta condicional depende de la respuesta de la otra?

- ○ *¿Es mayor tu suposición que el número secreto?*
- ○ *¿Es la suma de ambos números igual al número secreto?*

Manos a la obra

Conectadas

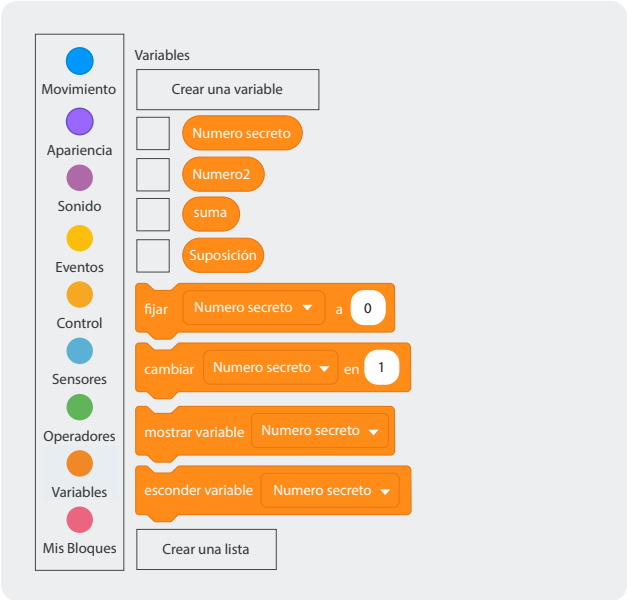
Organízate en grupos siguiendo las indicaciones de tu docentes.

En grupos, van a programar en *Scratch* una versión solitaria del juego "Adivina el número secreto" donde participan como jugador(a) y el objeto Gato "Cat" será el repartidor.

Sigan los pasos:

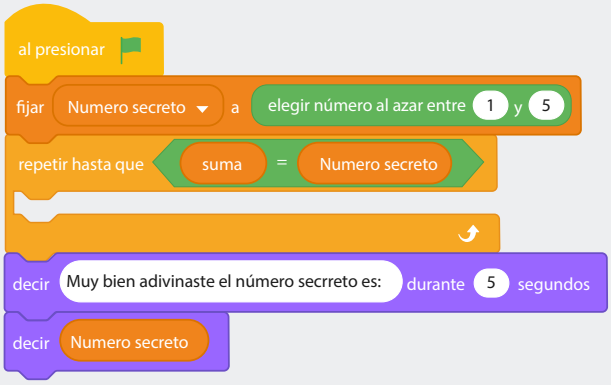
1. Crear desde la paleta “Variables” las variables llamadas “Número secreto”, “Numero2”, “suma” y “Suposición”. En la *Figura 1* pueden ver la creación de variables.

Figura 1. Creación de variables



2. Arrastrar al lienzo de programación desde la paleta “Eventos” la bandera verde. Luego, pasar a la paleta “Variables” y ensamblar “fijar (Número secreto) a”. Para obtener la opción de “elegir número al azar entre” ir a la paleta “Operadores”. Pueden ver este ejemplo en la *Figura 2*.

Figura 2. Asignación de número al azar

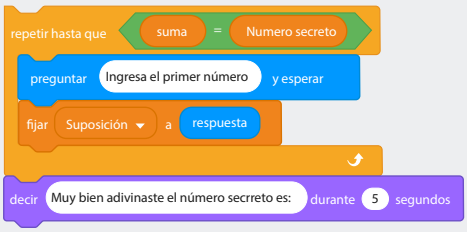


3. Arrastrar desde la paleta “Control” el ciclo “repetir hasta que” y, como pregunta condicional del ciclo ensamblar, una igualdad desde la paleta de “Operadores” donde se pregunte si la variable “suma” es igual a la variable “Número secreto”.

4. Por fuera del ciclo, es decir, cuando la suma sea igual al número secreto ensambla desde la paleta de “Apariencia” los mensajes “decir () durante () segundos” y “decir ()”.

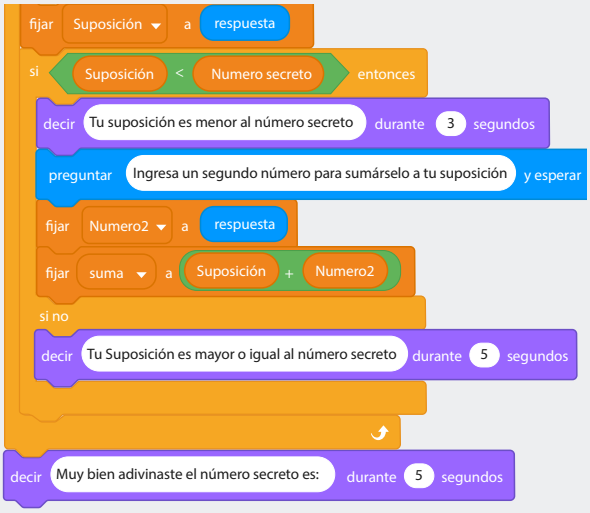
5. Dentro del ciclo, ensamblar desde la paleta “Sensores” la opción “preguntar () y esperar”. Luego, dentro del ciclo, ensamblar también desde la paleta de “Variables” la opción “Fijar (Suposición) a (respuesta)”. Ver *Figura 3.*

Figura 3. Ensamblar “preguntar () y esperar”



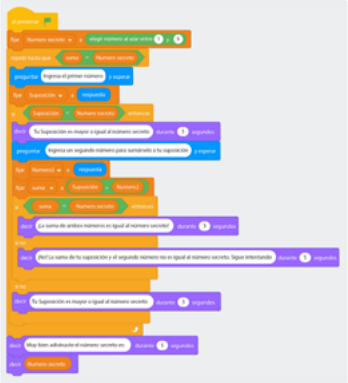
6. Dentro del ciclo ensamblar, también, una condicional doble “Si - si no” y para la pregunta condicional usar el operador “menor que”. Arrastrar a los espacios del operador las variables “Suposición” y “Número secreto”. Fíjense que la pregunta condicional sería: *¿La suposición es menor al número secreto?*

Figura 4. Bloque condicional "Si - Si no" en Scratch



Anexo

Anexo 2. 2



Nota

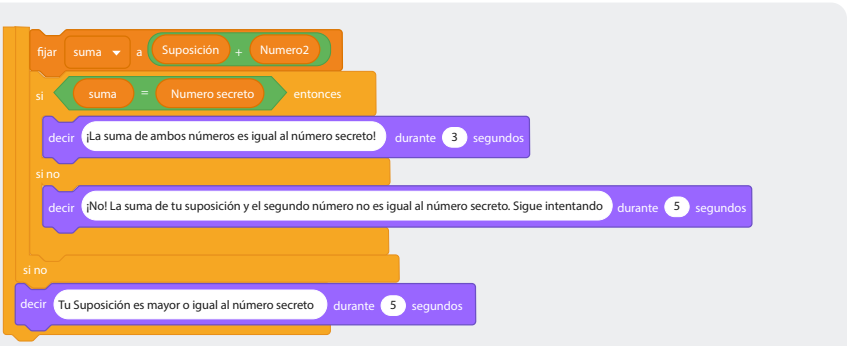
A partir de este paso relaciona la lógica del juego cuando lo hiciste con tu grupo con las cartas, pues vamos a anidar una condicional entre otra de acuerdo con las preguntas ¿La suposición es menor al número secreto? y ¿La suma de la suposición y el segundo número que ingresaste es igual al número secreto?

7 Ahora, programen las acciones a seguir en caso de que la respuesta a la pregunta condicional sea afirmativa. Arrastrar la opción "Decir () durante () segundos" y comunicar, al jugador, a través de Gato "Cat" que la suposición es menor al número secreto. Posteriormente, pedir que ingrese un segundo número para sumarlo a su suposición.

8 Fijar la respuesta a la variable "Numero2" y luego sumar las variables "Suposición" y "Numero2". Luego retomaremos, nuevamente, el lado afirmativo de la condicional.

9 Por otra parte, en caso de que la respuesta a la pregunta ¿La suposición es menor al número secreto? sea negativa, programar a Gato "Cat" para que le informe al jugador(a) que su suposición es mayor o igual al número secreto.

Figura 5. Mensaje de Gato en respuesta a la suposición en *Scratch*

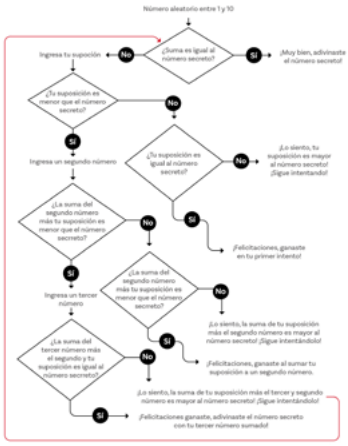


10 Volvamos nuevamente al lado afirmativo de la pregunta *¿La suposición es menor al número secreto?*. Por ahora, programaron a Gato "Cat" para que, en caso de afirmativo, le informe al jugador(a) que su suposición es menor al número secreto y luego le pidiera un segundo número para sumarlo a su suposición. Ahora, vamos a programar a Gato "Cat" para que, al plantearnos la pregunta *¿La suma de la suposición y el segundo número que ingresaste es igual al número secreto?* le diga al jugador(a): en caso afirmativo, que la suma es igual al número secreto, lo que significa que ha ganado. Por lo contrario, en caso de que la suma no sea igual al número secreto le diga que ¡No!, la suma de su suposición y el segundo número que ingresó no son iguales al número secreto y que siga intentando.

En el *Anexo* 2.2 encontrarán un ejemplo que podrá ayudarles.

Anexo

Anexo 2. 3



Para ir más lejos

Vamos a aumentar la incertidumbre en el juego “Adivina el número secreto”. Ahora, vas a modificar la programación del juego para incluir una tercera oportunidad de ganar.

En la versión anterior se tenía la posibilidad de ganar en dos oportunidades: la primera, ingresando una suposición y, si esta era igual al número secreto, ganaba. Otra forma de ganar era cuando la suposición era menor al número secreto y podía sumar un segundo número a la suposición, si la suma de ambos era igual al número secreto, también se ganaba.

El reto consiste en variar el juego en dos aspectos:

1. El primero es que vamos a subir la incertidumbre aumentando el valor aleatorio del número secreto pasándolo del rango de 1 a 5 al rango de 1 a 10.

2. El segundo aspecto por modificar es incluir un tercer número en caso de que la suma entre el valor de tu suposición y el segundo número siga siendo menor al valor del número secreto.

Usar el diagrama de flujo del *Anexo 2.3* como una guía para modificar o rehacer la programación del juego.

Antes de irnos

Esta sección corresponde al 100% de avance de la sesión

Revisa los aprendizajes esperados de forma individual respondiendo las preguntas de forma que mejor reflejen tu progreso:

1. ¿Puedes utilizar condicionales anidados para ejecutar acciones que dependan de más de un condicional?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ Aún no

2. ¿Puedes programar juegos sencillos en *Scratch* en los que se evalúen resultados a través de condicionales anidadas?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ No

3. ¿Puedes seguir secuencias de programación que involucran condicionales anidados descritos desde diagramas de flujo?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ No

> **Si tus respuestas fueron "Parcialmente" o "Aún no", vuelve a las actividades propuestas en *Scratch*. Luego, discute con tus compañeras y compañeros de grupo lo que se hizo en cada momento de la actividad y el rol al que correspondía. Si todavía te quedan dudas, consúltale a tu docente.**

Revisa el reto para examinar en dónde te servirá lo que has aprendido.

¿?

¿Cuándo usar condicionales simples, dobles o anidadas?

Pregunta condicional

De la respuesta de afirmativa de esta depende ejecutar una sola acción.	De la respuesta de esta depende ejecutar dos acciones diferentes.	De la respuesta de esta depende ejecutar más de dos acciones diferentes.
Condicional simple	**Condicional doble**	**Condicional anidada**

Grado 7º **Guía 1**

TIC

Sesión 3

Aprendizajes esperados

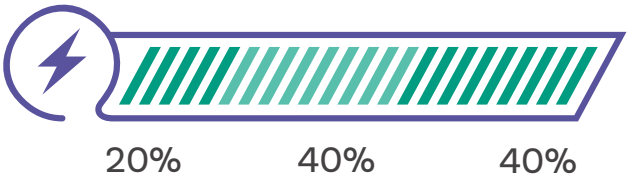
Al final de esta sesión verifica que puedas:

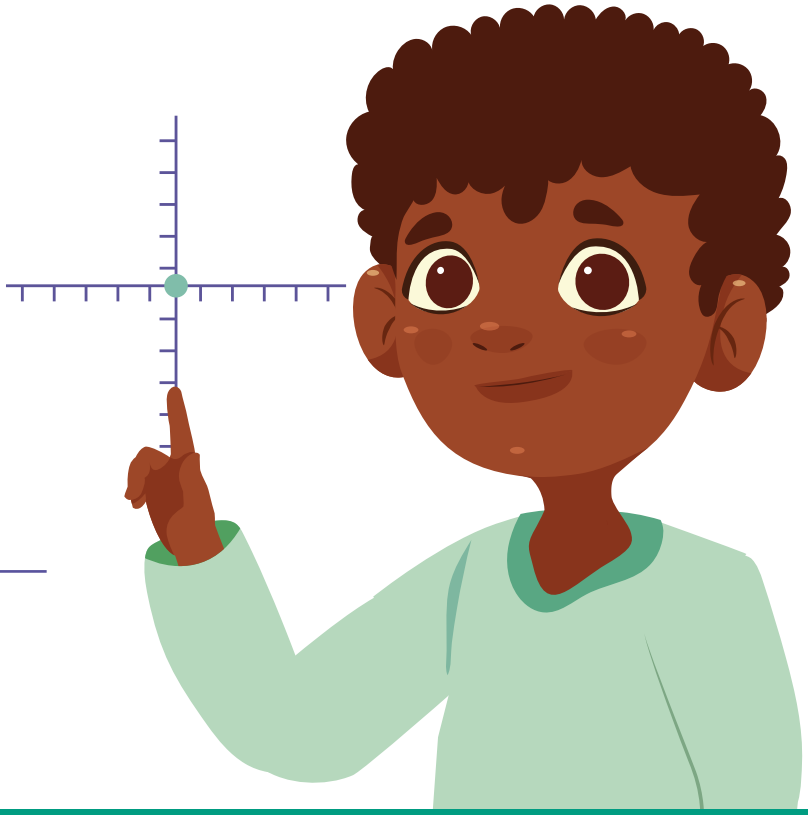
- Establecer rangos a partir de secuencias numéricas usando condicionales anidados.
- Ubicar rangos en el plano cartesiano para programar objetos en *Scratch*.
- Aplicar la programación de condicionales anidados en *Scratch* para discriminar datos provenientes de una única variable.

Material para la clase

- ○ Acceso a *Scratch*.

Duración sugerida





Lo que sabemos, lo que debemos saber

Esta sección corresponde al 20% de avance de la sesión

La sesión anterior aprendiste a usar condicionales anidadas para ejecutar diferentes acciones dependientes de un conjunto de preguntas organizadas jerárquicamente. A lo largo de esta sesión nos centraremos en aplicar las condicionales anidadas para establecer rangos en secuencias numéricas y cómo programar objetos en *Scratch* de acuerdo con la discriminación de dichos rangos.

Para discriminar rangos numéricos usando condicionales anidadas, se pueden emplear estructuras de decisión que verifican si un número se encuentra dentro de ciertos límites. Un ejemplo común de esta aplicación es cuando te califican tus docentes, has notado que si obtienes una nota por debajo de un valor o umbral pierdes dicha calificación, sin embargo, si obtienes una nota mayor cabe la posibilidad que tu calificación sea Básico, Alto o Superior. Observa la tabla a continuación que muestra un ejemplo de rangos de calificación, ten en cuenta que este es un ejemplo y no es aplicable a todos los colegios.

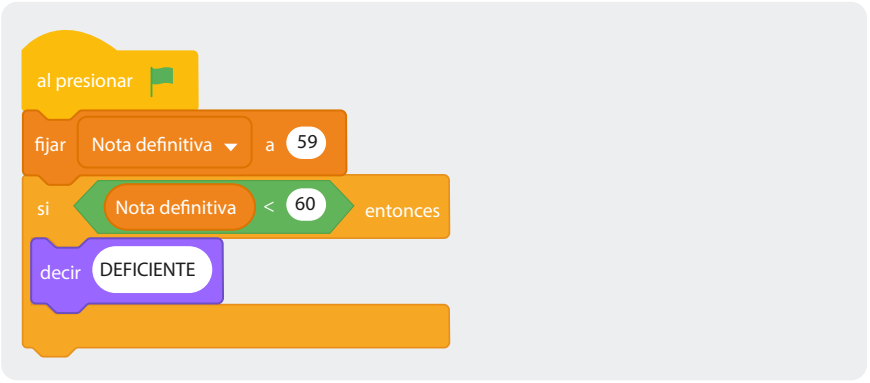
Nivel de calificación	Rango de nota
Excelente	Entre (86 a 100)
Alto	Entre (76 a 85)
Básico	Entre (60 a 75)
Deficiente	Por debajo de 60

¿Cómo es el rango de calificación en tu colegio?

Usemos este ejemplo para describirlo paso a paso y analicemos cómo podemos utilizar la condicionales anidadas para establecer estos rangos en programación.

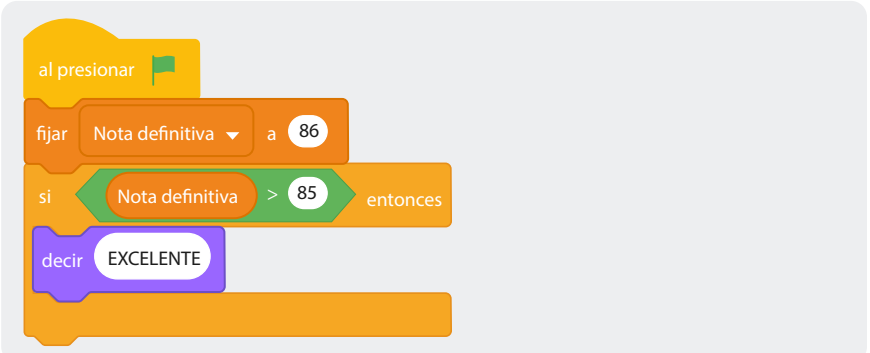
Supongamos que una estudiante llamada Gabriela ha recibido su nota definitiva de final de periodo y quiere hacer un programa que al ingresar su nota le indique de manera automática su nivel de calificación. Al tratarse de una única variable (nota definitiva) que vamos a ubicar en rango numérico debemos evaluar sus límites inferior y superior, por ejemplo, si Gabriela obtuvo una nota inferior a 60 es posible evaluar su nota con una condicional que utilice como pregunta el operador "menor que", de esta forma se podría clasificar en el rango Deficiente.

Figura 1. Clasificación de la nota definitiva en *Scratch*



Lo mismo sucede con el rango Excelente si utilizamos esta vez como pregunta condicional el operador "mayor que".

Figura 2. Clasificación en el rango "Excelente" en *Scratch*



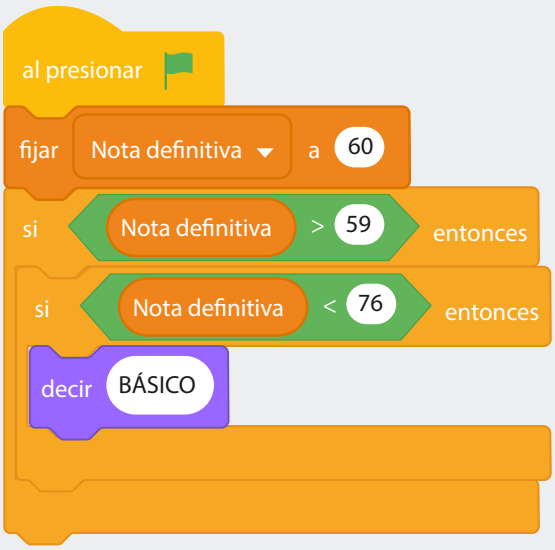
Sin embargo, es un poco más complicado con valores iguales o mayores a 60 y menores que 86, ya que en este caso se deben establecer los límites inferior y superior de cada uno de los rangos "Básico" y "Alto". Para esto utilizaremos las condicionales anidadas.

Imaginemos que Gabriela obtuvo una nota de 60, en la tabla claramente se ve que su calificación es Básico, sin embargo, al hacer un programa debemos ser muy específicos con lo que el programa debe hacer y el rango de Básico contiene las notas entre 60 a 75, por otro lado, en los operadores de *Scratch*, para este caso solamente contamos con los operadores "menor que" y "mayor que".

Analicemos los valores de este rango, si tenemos en cuenta que el número 60 está como límite inferior y 75 como límite superior podríamos decir que las notas que son mayores que 59 y menores que 76 son las que componen Básico, podemos usar el límite inferior como la pregunta condicional de mayor jerarquía y anidar la pregunta condicional referente al límite superior así:

Si la nota es mayor a 59 entonces:
    Si la nota es menor a 76 entonces:
        Decir (Básico)

Figura 3. Condicional anidada para clasificar la nota en *Scratch*



Prueba esta condicional anidada con notas entre 60 a 75.

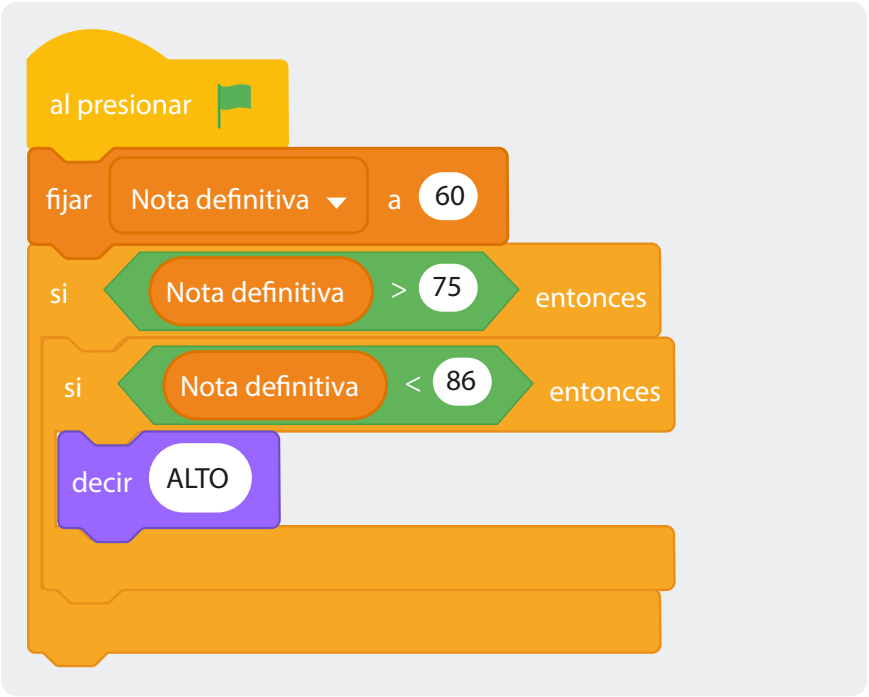
Lo mismo podríamos hacer para el caso del rango “Alto”.

Prueba esta condicional anidada con notas entre 76 a 85, como lo puedes ver en la *Figura 4*.

Ahora es tu turno de programar este ejemplo completo con lo aprendido en las sesiones anteriores.

Unifica los ejemplos en un solo programa.

Figura 4. Programa unificado de clasificación de notas en *Scratch*



¿Cómo podrías hacer un programa que le permita a través de una pregunta al usuario ingresar la nota definitiva y que le diga su calificación (Deficiente, Básico, Alto o Excelente)?

Enlace

Acceso a la plantilla

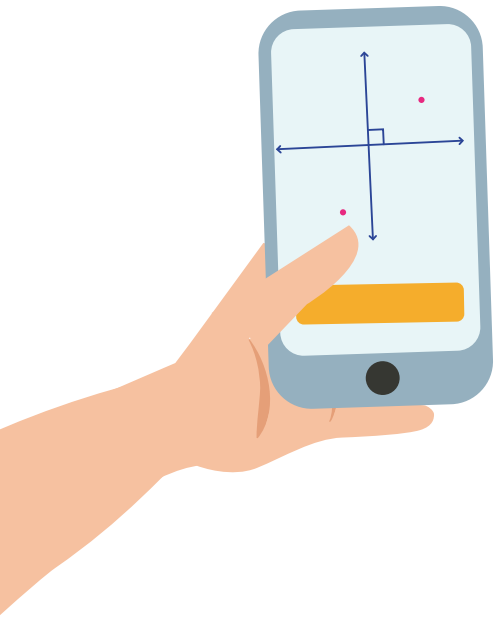
Manos a la obra

Conectadas

Esta sección corresponde al 60% de avance de la sesión

Organízate en grupo siguiendo las recomendaciones de tu docente. En esta sección aprenderemos a utilizar los rangos en el plano cartesiano para ubicarnos espacialmente en *Scratch*. Obtener rangos en el plano cartesiano puede ser útil para una variedad de aplicaciones en matemáticas, ciencia, ingeniería, programación y otras áreas, por ejemplo, es muy útil para identificar regiones en funciones matemáticas, analizar el movimiento de los objetos o simular objetos de acuerdo con propiedades físicas.

En este caso vamos a usarlo en programación para identificar los cuadrantes del plano cartesiano usando *Scratch*.



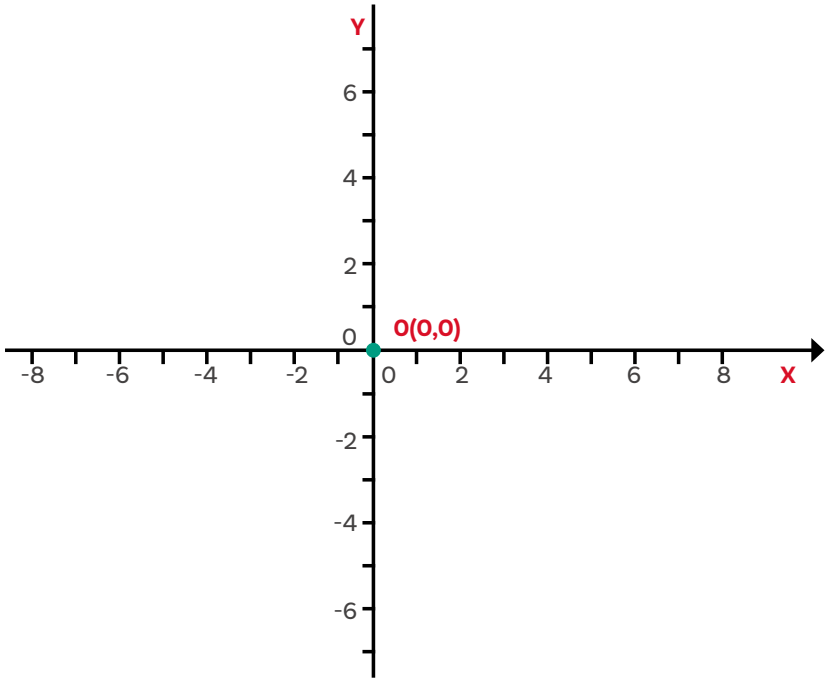
Seguir los pasos:

1. Escanear el código QR o escribir el enlace de la plantilla sobre la que van a programar.

En ella encontrarán el objeto llamado “Ball” centrado en un fondo que contiene el plano cartesiano.

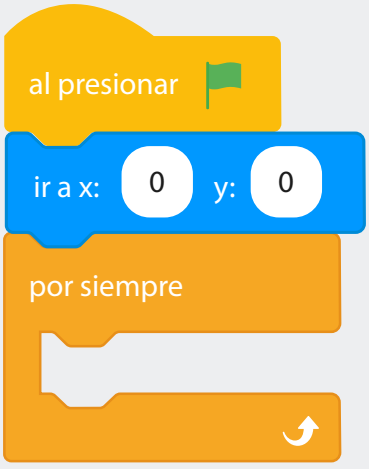
Observa que el plano cartesiano se divide en cuatro sectores o rangos de dos dimensiones (X, Y). Podemos catalogarlos como sector superior izquierdo, superior derecho, inferior izquierdo e inferior derecho. El sector superior izquierdo es aquel en el que los valores de Y son positivos y los de X son negativos (X=-;Y=+), el sector superior derecho es aquel en el que los valores de Y son positivos y los de X también son positivos (X=+;Y=+). Por otra parte, el sector inferior izquierdo es aquel en el que los valores de Y son negativos y los de X también son negativos (X=-;Y=-); finalmente, el sector inferior derecho es aquel en el que los valores de Y son negativos y los de X son positivos (X=+;Y=-). El centro del plano cartesiano tiene los valores (X=0; Y=0), ni positivos, ni negativos. En la *Figura 5* se puede ver el punto de origen o centro del plano cartesiano (X=0; Y=0).

Figura 5. Plano cartesiano



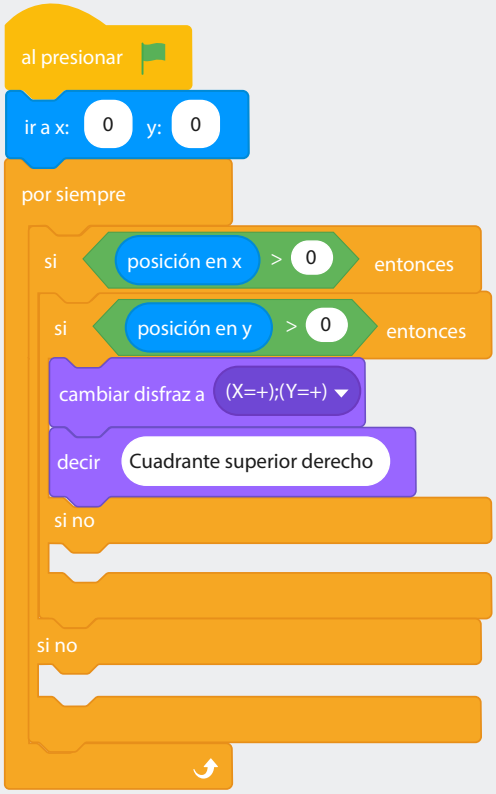
2 Vamos a programar a Pelota "Ball" para que al moverla nos indique en qué sector del plano cartesiano se encuentra. Iniciemos asegurándonos que al oprimir la bandera verde Pelota "Ball" esté en la posición (X=0; Y=0).

Figura 6. Posicionamiento inicial de Pelota en el plano cartesiano en *Scratch*



3 Ingresar una condicional doble dentro del ciclo cuya pregunta sea: ¿X es mayor que 0? y dentro de su parte afirmativa otra condicional doble cuya pregunta sea: ¿Y es mayor que 0? Si la respuesta a ambas preguntas es afirmativa quiere decir que Pelota "Ball" está en el cuadrante superior derecho donde los valores de Y son positivos y los de X también son positivos (X=+; Y=+), agrega el disfraz correspondiente.

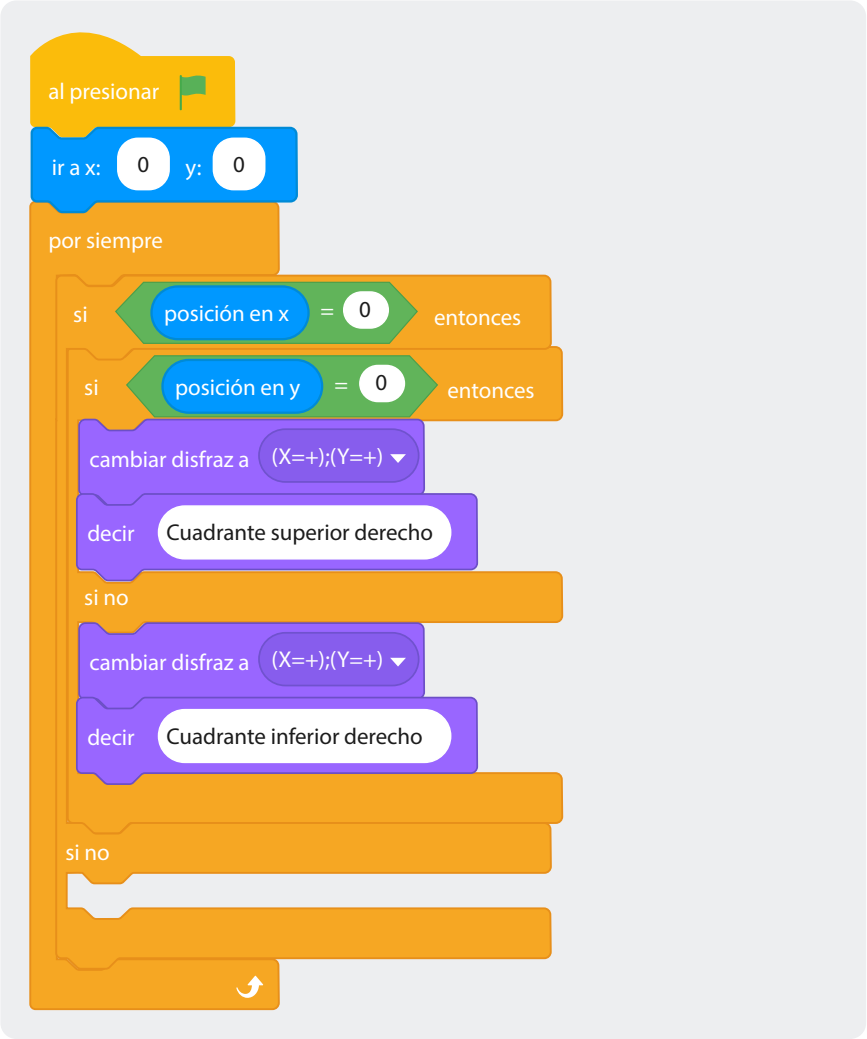
Figura 7. Detección del cuadrante superior derecho en *Scratch*



Fíjense que la primera condicional en su lado afirmativo solamente considera los valores mayores de cero para X y la condicional anidada los valores positivos de Y en su parte afirmativa. Esto quiere decir que, si los valores de X van a ser positivos pero los de Y son negativos podemos deducir que el cuadrante inferior derecho se verá reflejado en el "si no" de la condicional anidada.

4 Cambiar el disfraz de Pelota "Ball" y haz que diga "Cuadrante inferior derecho" en el "si no" de la condicional anidada.

Figura 8. Identificación del cuadrante inferior derecho en *Scratch*



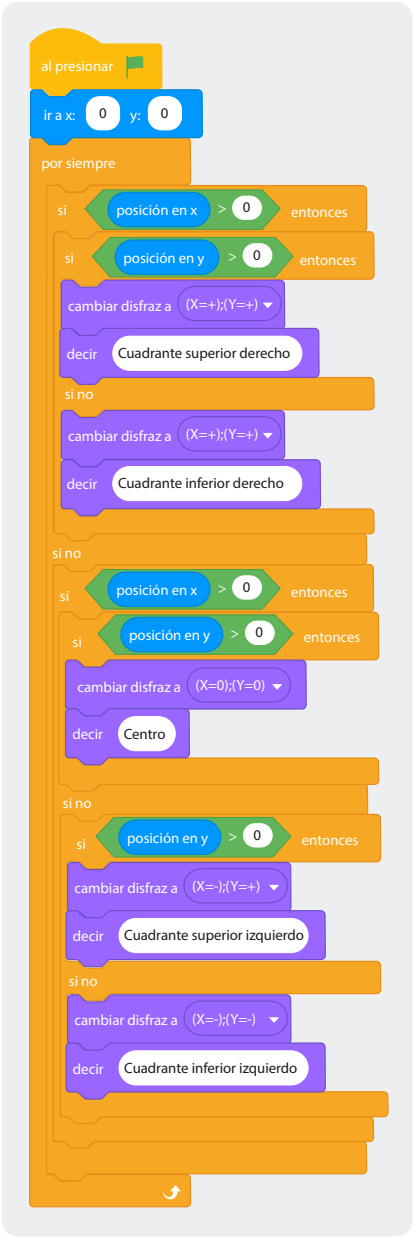
Observar que ahora nos queda la parte negativa o "si no" de la pregunta ¿X es mayor que 0?, lo cual nos da dos posibilidades. La primera es que X sea menor que 0 o que X sea igual a 0. Usemos otra condicional anidada para programar ambas opciones.

5 Ingresar otra condicional doble al "si no" de la condicional ¿X es mayor que 0? preguntando si ¿X es igual a 0? En caso de positivo, es posible que Pelota "Ball" se encuentre en el centro o el punto (X=0; Y=0) pero, para saberlo, debemos preguntar de manera anidada con una condicional simple si también se encuentra en Y=0. En tal caso, pondremos el disfraz (X=0;Y=0) y haremos que Pelota "Ball" diga "Centrado en X,Y". Ver *Figura 9*.

Figura 9. Detección del punto central en el plano cartesiano en *Scratch*

6 Ahora tenemos un “si no”, que pertenece a la parte negativa de la pregunta ¿X es igual a 0? Lo que se sabe, en este caso, es que X no puede ser 0 o mayor que 0 en esta parte de la condicional, pero Y sí puede ser positivo o negativo. Por esta razón, agregaremos otra pregunta condicional anidada: ¿Y es mayor que 0? para terminar de saber en qué rango o cuadrante de nuestro plano cartesiano se encuentra Pelota “Ball”.

Figura 10. Detección de cuadrantes para X negativa en *Scratch*

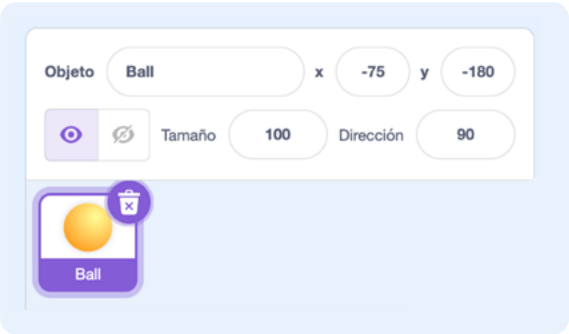


7 Para verificar la programación hacer clic sobre la bandera verde y, usando el puntero del ratón, arrastrar a Pelota "Ball" por los cuadrantes del plano cartesiano.

¿Funciona correctamente?

Aunque el programa funciona correctamente en la mayoría de los casos, ¿se podría afirmar que sucedería lo mismo en todos los casos?

Con la bandera verde activa se pueden modificar las coordenadas (X,Y). Tratar con las coordenadas (X=0,Y=180), (X=0,Y=-90), (X=90,Y=180) y (X=90,Y=-180).



En las coordenadas mencionadas anteriormente Pelota "Ball" sigue diciendo "Centro", aunque ya no se encuentra en el centro o coordenada (X=0; Y=0). El reto es agregar cuatro mensajes más a la programación.

1. "X = 0 y Y = +" cuando el valor de X sea 0 y el de Y sea mayor que 0.
2. "X = 0 y Y = -" cuando el valor de X sea 0 y el de Y sea menor que 0.
3. "X = + y Y = 0" cuando el valor de X sea mayor que 0 y el de Y sea 0.
4. "X = - y Y = 0" cuando el valor de X sea menor que 0 y el de Y sea igual a 0.

Usar condicionales anidadas para modificar la programación y conseguir superar este reto.

Antes de irnos

Esta sección corresponde al 100% de avance de la sesión

Revisa los aprendizajes esperados de forma individual respondiendo las preguntas de forma que mejor reflejen tu progreso:

1. ¿Puedes utilizar condicionales anidados para establecer rangos en secuencias numéricas?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ Aún no

2. ¿Puedes utilizar condicionales anidados para establecer rangos en el plano cartesiano?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ No

3. ¿Puedes crear programas en *Scratch* que contemplen la evaluación de una o más variables numéricas con el uso de condicionales anidados para ejecutar acciones de acuerdo con rangos?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ No

> **Si tus respuestas fueron "Parcialmente" o "Aún no", vuelve a las actividades propuestas en *Scratch*. Luego, discute con tus compañeras y compañeros de grupo lo que se hizo en cada momento de la actividad y el rol al que correspondía. Si todavía te quedan dudas, consúltale a tu docente.**

Te proponemos revisar la relación entre los aprendizajes logrados en esta sesión y el reto: ¿en qué te servirán?

En este punto te recomendamos realizar un gráfico o esquema con los aprendizajes logrados.

Grado 7º **Guía 1**

TIC

Sesión 4

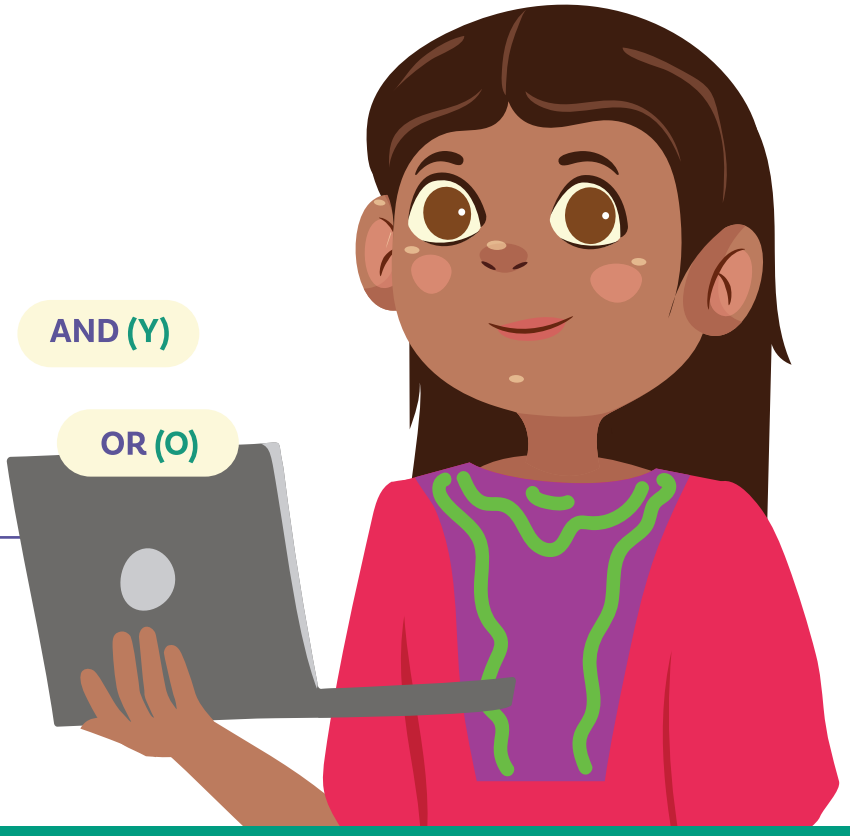
Aprendizajes esperados

Al final de esta sesión verifica que puedas:

- Reconocer la aplicación de los operadores lógicos AND y OR en situaciones cotidianas.
- Aplicar los operadores lógicos AND y OR para simplificar, optimizar o desanidar condicionales anidados.
- Programar condicionales con el uso de operadores lógicos en *Scratch*.

Duración sugerida

20% 40% 40%



Material para la clase

- ○ Acceso a *Scratch*.

Lo que sabemos, lo que debemos saber

Esta sección corresponde al 20% de avance de la sesión

A lo largo de las sesiones 2 y 3 utilizamos condicionales anidadas con dos propósitos. El primero, conseguir ejecutar más de dos acciones al evaluar el valor de una variable, por ejemplo sumar un segundo o tercer número de acuerdo con una o más preguntas condicionales como el caso del juego “Adivina el número secreto”. El segundo propósito de su uso fue programar rangos usando las condicionales anidadas para establecer los límites inferior y superior de dichos rangos, como el caso de clasificar la nota de una estudiante o ubicar los cuadrantes del plano cartesiano. En esta sesión aprenderemos a simplificar los programas de *Scratch* que tengan condicionales anidadas con el uso de los operadores lógicos AND y OR.

El operador lógico AND es utilizado en programación para combinar dos o más condiciones, de manera que la expresión completa se evalúe como verdadera solo si todas las condiciones individuales son verdaderas. En términos simples, AND requiere que ambas (o todas) las condiciones sean verdaderas para que el resultado sea verdadero o afirmativo.

Imagina que estás planeando salir a caminar y tienes dos condiciones para decidir si saldrás o no:

1. La condición climática: ¿No está lloviendo?
2. La disponibilidad de tiempo: ¿Tengo tiempo libre?

Para decidir salir a caminar, ambas condiciones deben ser afirmativas.

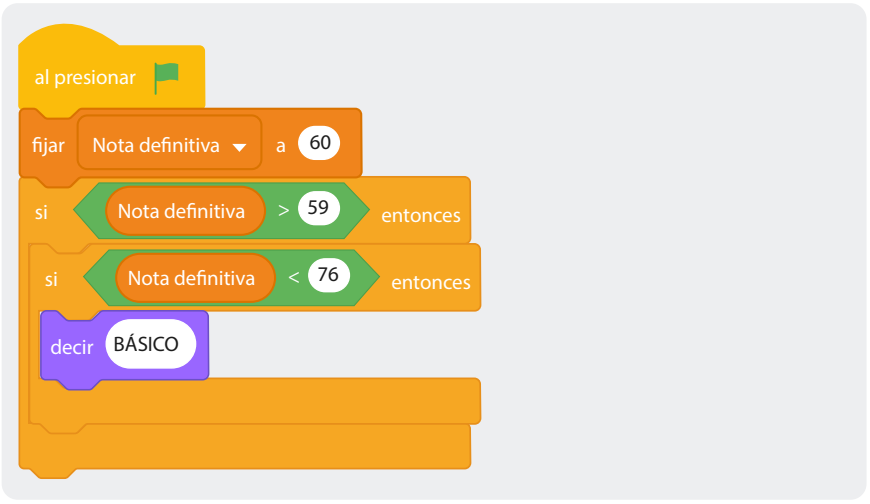
La manera en la que se puede usar el operador lógico AND (Y) en este caso es uniendo ambas preguntas condicionales en una sola.

Si ¿no está lloviendo? (Y) ¿tengo tiempo libre? entonces salir a caminar si no quedarse en casa.

En este caso, si no está lloviendo (Y) tienes tiempo libre, entonces decides salir a caminar. Si cualquiera de las dos condiciones no se cumple (por ejemplo, está lloviendo o no tienes tiempo libre), entonces decides quedarte en casa.

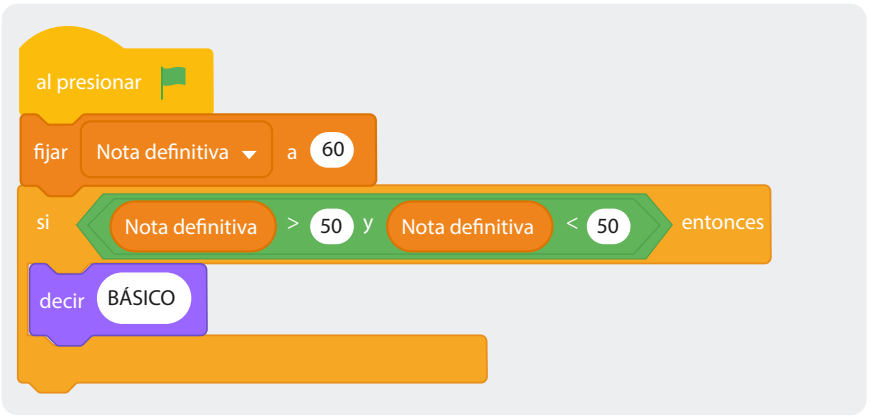
Retomemos una de las condicionales anidadas para clasificar las notas de una estudiante que analizamos la sesión pasada.

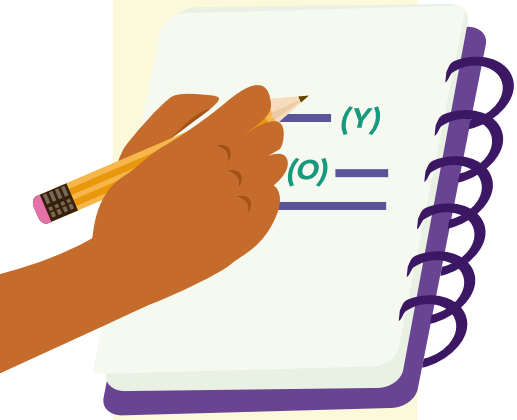
Figura 1. Clasificación de nota en el rango "Básico" en *Scratch*



En este caso podemos hacer la simplificación de esta condicional anidada con el uso del operador AND que en español es (Y) con el fin de eliminar la condicional anidada o desanidarla.

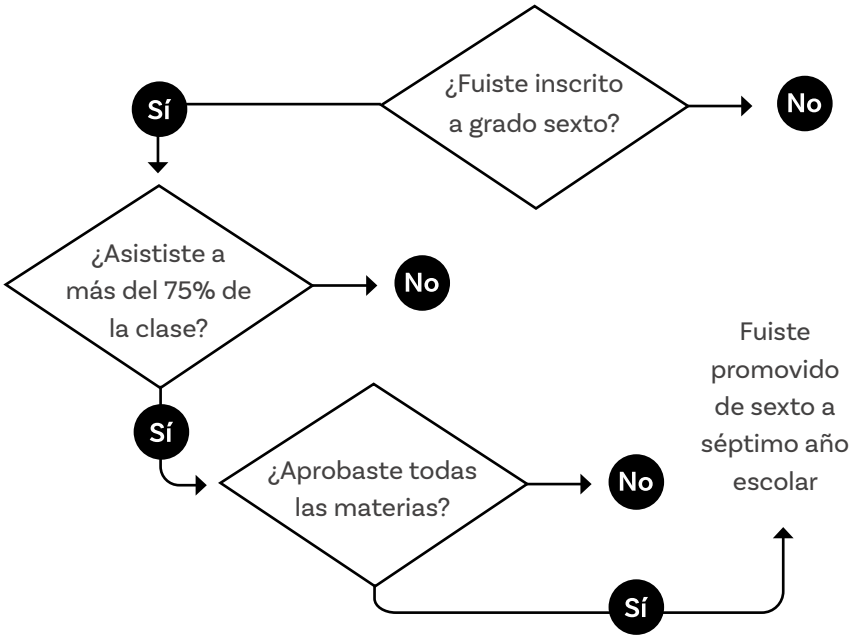
Figura 2. Simplificación de la condicional con el operador AND en *Scratch*.





Reflexiona en lo siguiente:

Fíjate que el operador lógico AND (Y) lo utilizamos continuamente en nuestro lenguaje cotidiano, piensa en situaciones de tu colegio que dependen de dos o más factores para cumplirse, por ejemplo, el caso tratado en la primera sesión de esta guía.



Si aplicamos el operador lógico AND (Y) a esta situación podemos reescribirla de tal manera que no será necesario anidar las condicionales dejando solamente una única condicional.

Si fuiste inscrito a grado sexto (Y) asististe más del 75% de las clases (Y) aprobaste todas las asignaturas entonces fuiste promovido a séptimo año escolar.

Con el operador AND (Y) podemos ejecutar una acción en el lado afirmativo de una condicional siempre y cuando las preguntas condicionales que incluyamos sean de respuesta afirmativa.

Otro operador lógico de uso común en programación para desanidar condicionales es el operador OR (O), con él que podemos ejecutar acciones en el lado afirmativo de una condicional con la condición de que sea afirmativa alguna de las respuestas de las preguntas que incluyamos.

Analicemos este ejemplo que clarifica las diferencias entre ambos operadores AND (Y) y OR (O):

Imagina que quieres ir a una reunión con tus amigas(os) el siguiente fin de semana y en tu casa te ponen dos condiciones para que puedas ir. La primera es que tiendas tu cama todos los días y la segunda es que obtengas una calificación mayor a 80 en el examen de matemáticas que tienes justo esa misma semana.

Si tendiste la cama todos los días (Y) sacaste más de 80 en matemáticas entonces vas a la reunión.

Redactada así la situación es lógico pensar que si sacas una nota menor o igual a 80 no irás a la reunión, lo mismo ocurriría si a pesar de haber sacado una nota mayor a 80 en el examen no tendiste tu cama todos los días.

Por otra parte, si redactamos este ejemplo de esta forma:

Si tendiste la cama todos los días (O) sacaste más de 80 en matemáticas entonces vas a la reunión.

Es lógico pensar que a pesar de haber perdido el examen de matemáticas irás a la reunión si tendiste tu cama todos los días o al revés, irás a la reunión, aunque no hayas tendido tu cama ningún día, pero sacaste más de 80 en matemáticas.

¿Puedes describir más acciones cotidianas así?, acciones donde es común usar los operadores AND (Y) y OR (O) en la vida cotidiana.

Toma tu cuaderno y escribe por lo menos 5 ejemplos para cada operador lógico.

Manos a la obra

Conectadas

Esta sección corresponde al 60% de avance de la sesión

Organízate en grupo siguiendo las indicaciones de tu docente.

Retomemos el uso de condicionales para discriminar rangos de secuencias numéricas, vamos a usar *Scratch* para crear un programa que, por la edad de una persona, nos diga si se trata de niña, niño, adolescente, persona joven, persona adulta promedio o persona adulta mayor de acuerdo con la *Tabla 1*:

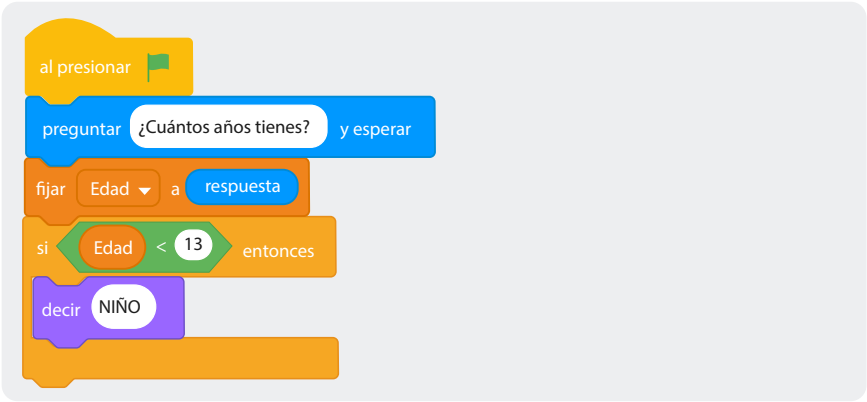
Tabla 1. Categorías por rangos de edad

Categoría	Rango de edad (años)
Niña o niño	Menos de 13 años
Persona adolescente	13 a 17 años
Persona joven	18 a 25 años
Persona adulta promedio	26 a 60 años
Persona adulta mayor	Más de 60 años

Seguir los pasos en la plataforma *Scratch*.

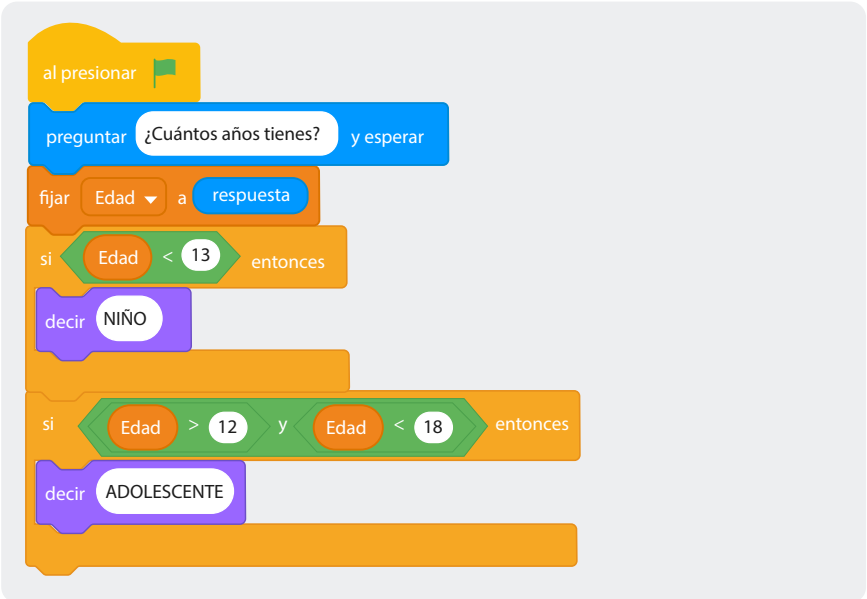
1. Crear una variable "Edad" y asignar la respuesta a la pregunta ¿cuántos años tienes? Ubicar debajo de la asignación la condicional para evaluar si se trata de una niña o un niño.

Figura 3. Evaluación de edad para clasificar niña o niño en *Scratch*



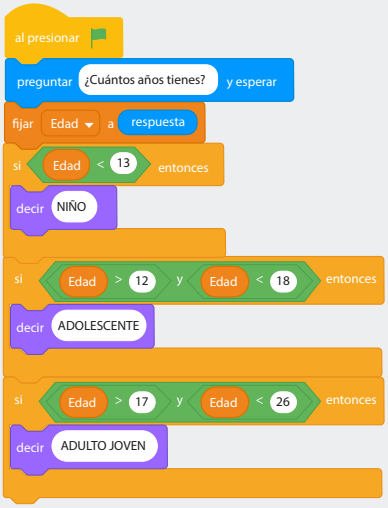
2. Usando el operador lógico AND (Y) crear una segunda condicional que delimite la edad en el rango entre 13 y 17 años para evaluar si se trata de una persona adolescente. Ubicar la segunda condicional justo debajo de la primera.

Figura 4. Clasificación de adolescencia con el operador AND en *Scratch*



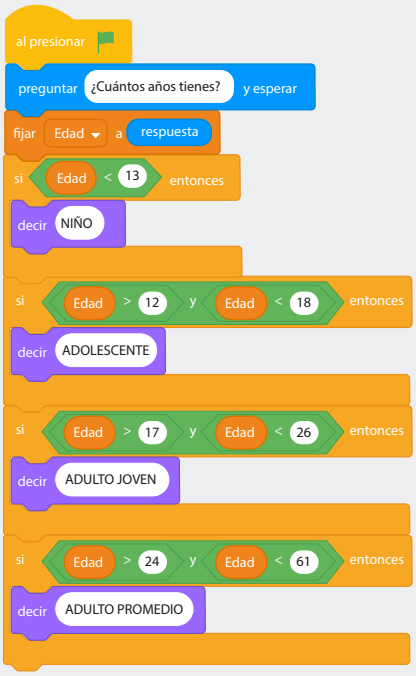
3 Siguiendo la misma lógica de programación, crear una tercera condicional para el rango entre 18 a 25 años para determinar si se trata de una persona adulta joven.

Figura 5. Clasificación de juventud con el operador AND en *Scratch*



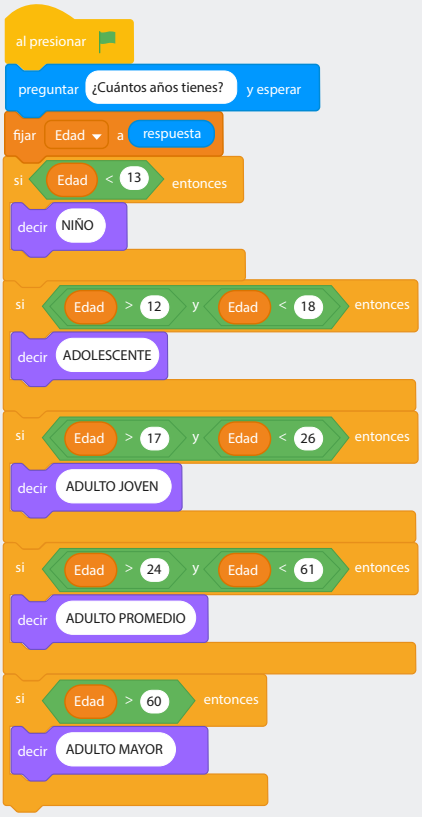
4 Crear una cuarta condicional para el rango entre 26 a 60 años para determinar si se trata de una persona adulta promedio.

Figura 6. Clasificación de adulto promedio con el operador AND en *Scratch*



5 Por último, agregar otra condicional simple para evaluar si se trata de una persona adulta mayor.

Figura 7. Clasificación de adulto mayor en *Scratch*



Reflexionar en lo siguiente:

¿?

¿Se puede hacer este mismo programa usando solamente condicionales anidadas?

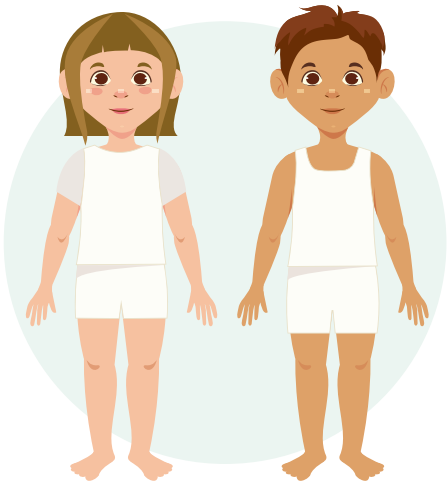
¿Cómo te resultó más sencillo programar rangos en Scratch?, ¿con condicionales anidadas o con operadores lógicos?

¿Cuántas veces se puede usar este programa antes de tener que oprimir nuevamente la bandera verde?

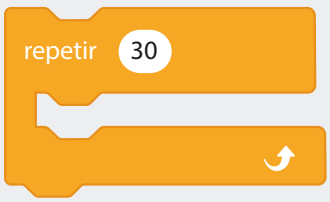
¿Por qué en Scratch no colocamos exactamente los valores de la tabla de rangos en los operadores "mayor que" y "menor que"?

Para ir más lejos

Este programa para clasificar a las personas por su edad puede mejorar al incluir aspectos para su personalización como condicionales que pregunten si la persona que responde es un hombre o una mujer y cambiar los disfraces de acuerdo al rango de edad. Por ejemplo, si la respuesta a Escribe tu sexo es femenino y al colocar una edad menor que 13 años puede cambiar el disfraz a “Harper-b” o, en caso de ser masculino, a “Harper-a”. Explora los disfraces para que cambien en función de las respuestas de sexo y edad.



También se puede personalizar la cantidad de veces que el programa puede ejecutarse antes de tener que oprimir nuevamente la bandera verde. Se puede usar un ciclo “Repetir ()” y agregar el número de compañeras(os) en el salón, hacer que el programa solicite el nombre de quien responde y, de esta forma, jugar con el programa con tus amigas(os).



Antes de irnos

Esta sección corresponde al 100% de avance de la sesión

Revisa los aprendizajes esperados de forma individual respondiendo las preguntas de forma que mejor reflejen tu progreso:

1. ¿Puedes reconocer situaciones cotidianas donde se encuentra implícito el uso de operadores lógicos?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ Aún no

2. ¿Puedes desanidar condicionales anidados con el uso del operador lógico AND (Y)?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ Aún no

3. ¿Puedes optimizar la programación de rangos en *Scratch* reemplazando el uso de condicionales anidados por el operador lógico AND (Y)?

   - ◯ Sí
   - ◯ Parcialmente
   - ◯ Aún no

> **Si tus respuestas fueron "Parcialmente" o "Aún no", vuelve a las actividades propuestas en *Scratch*. Luego, discute con tus compañeras y compañeros de grupo lo que se hizo en cada momento de la actividad y el rol al que correspondía. Si todavía te quedan dudas, consúltale a tu docente.**

En este punto te recomendamos realizar un gráfico o esquema con los aprendizajes logrados. Esto te ayudará a aprender más.

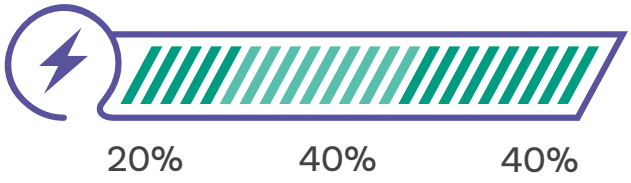
Sesión 5

Aprendizajes esperados

Al final de esta sesión se verifica que puedas:

- Utilizar los aprendizajes logrados para resolver el reto planteado.
- Utilizar el operador OR (O) y AND (Y) para establecer rangos bidimensionales que dependen de dos secuencias numéricas.
- Hacer programas simultáneos entre Objetos y Fondos en *Scratch* a través del envío de mensajes.
- Programar un juego de clasificación de objetos de acuerdo con sus atributos a través de rangos bidimensionales.

Duración sugerida

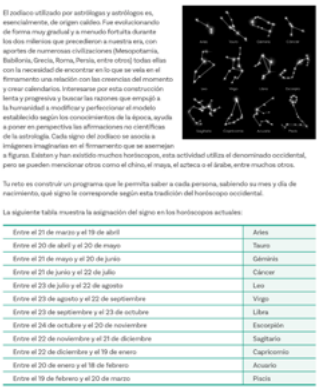


Material para la clase

- ○ Anexo 1.1.
- ○ Acceso a *Scratch*.

Anexo

Anexo 1.1



El zodiaco utilizado por astrólogos es, esencialmente, de origen caldeo. Fue evolucionando de forma muy gradual y a menudo fortuita durante los dos milenios que precedieron a nuestra era, con aportes de numerosas civilizaciones (Mesopotamia, Babilonia, Grecia, Roma, Persia, entre otros) todas ellas con la necesidad de encontrar en lo que se veía en el firmamento una relación con las creencias del momento y crear calendarios. Interesarse por esta construcción lenta y progresiva y buscar las razones que empujó a la humanidad a modificar y perfeccionar el modelo establecido según los conocimientos de la época, ayuda a poner en perspectiva las afirmaciones no científicas de la astrología. Cada signo del zodiaco se asocia a imágenes imaginarias en el firmamento que se asemejan a figuras. Existen y han existido muchos horóscopos, esta actividad utiliza el denominado occidental, pero se pueden mencionar otros como el chino, el maya, el azteca o el árabe, entre muchos otros.

Tu reto es construir un programa que le permita saber a cada persona, sabiendo su mes y día de nacimiento, qué signo le corresponde según esta tradición del horóscopo occidental.

La siguiente tabla muestra la asignación del signo en los horóscopos actuales:

Entre el 21 de marzo y el 19 de abril	Aries
Entre el 20 de abril y el 20 de mayo	Tauro
Entre el 21 de mayo y el 20 de junio	Géminis
Entre el 21 de junio y el 22 de julio	Cáncer
Entre el 23 de julio y el 22 de agosto	Leo
Entre el 23 de agosto y el 22 de septiembre	Virgo
Entre el 23 de septiembre y el 23 de octubre	Libra
Entre el 24 de octubre y el 20 de noviembre	Escorpión
Entre el 22 de noviembre y el 21 de diciembre	Sagitario
Entre el 22 de diciembre y el 19 de enero	Capricornio
Entre el 20 de enero y el 18 de febrero	Acuario
Entre el 19 de febrero y el 20 de marzo	Piscis

Lo que sabemos, lo que debemos saber

Esta sección corresponde al 20% de avance de la sesión

Es el momento de revisar el reto que se encuentra en el *Anexo 1.1*, que ya conoces, y resolverlo.

Imaginemos que estamos explorando la interesante idea de la clasificación mediante atributos numéricos, utilizando rangos bidimensionales, es decir, dos secuencias de números. Aunque este concepto puede parecer abstracto, tiene aplicaciones prácticas en la vida cotidiana. Un ejemplo común es la clasificación de personas en signos zodiacales según su día y mes de nacimiento, justamente el reto planteado. El día y mes de nacimiento son representados numéricamente, donde enero es 1 y diciembre es 12.

Hasta ahora, hemos aprendido a usar condicionales simples, dobles y anidadas en *Scratch* y también hemos explorado cómo aplicar operadores lógicos AND y OR en situaciones comunes de programación como la "desanidación" de condicionales para establecer rangos en secuencias numéricas.

Una forma de describir en programación este caso de clasificación por rangos bidimensionales puede ser la siguiente:

si (mes = 3 y día > 20) o (mes = 4 y día < 20) entonces "Aries"
si (mes = 4 y día > 19) o (mes = 5 y día < 21) entonces "Tauro"
si (mes = 5 y día > 20) o (mes = 6 y día < 21) entonces "Géminis"
si (mes = 6 y día > 20) o (mes = 7 y día < 23) entonces "Cáncer"
si (mes = 7 y día > 22) o (mes = 8 y día < 23) entonces "Leo"
si (mes = 8 y día > 22) o (mes = 9 y día < 23) entonces "Virgo"
si (mes = 9 y día > 22) o (mes = 10 y día < 23) entonces "Libra"
si (mes = 10 y día > 23) o (mes = 11 y día < 21) entonces "Escorpio"
si (mes = 11 y día > 21) o (mes = 12 y día < 22) entonces "Sagitario"
si (mes = 12 y día > 21) o (mes = 1 y día < 20) entonces "Capricornio"
si (mes = 1 y día > 19) o (mes = 2 y día < 19) entonces "Acuario"
si (mes = 2 y día > 18) o (mes = 3 y día < 21) entonces "Piscis"

Vamos a programar este caso y a su vez aprenderemos a programar simultáneamente objetos y fondos o escenarios en *Scratch*.

Enlace

Acceso a *Scratch*

Manos a la obra

Conectadas

Esta sección corresponde al 60% de avance de la sesión

Organízate en grupos siguiendo las indicaciones del docente.

Vamos a hacer una calculadora que al recibir el día y mes de tu nacimiento te diga tu signo del zodiaco.

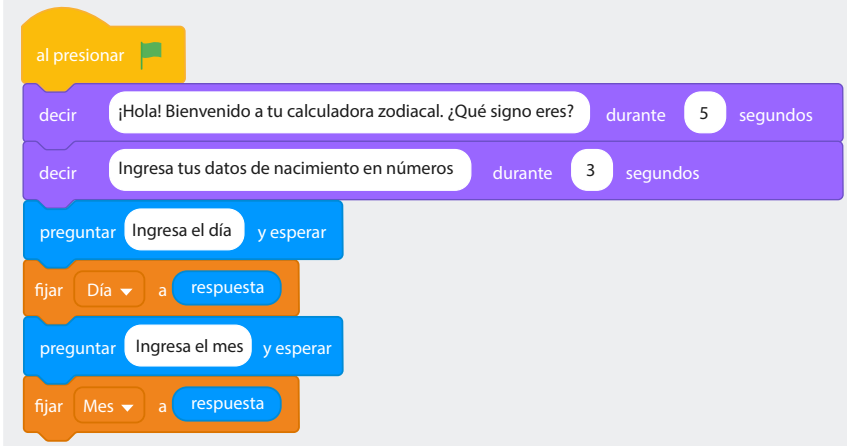
Sigue los pasos a continuación hasta completar la calculadora zodiacal.

1. Escanear el código QR o usa el enlace para acceder a *Scratch*.

En esta plantilla encontrarán 13 disfraces del fondo de *Scratch*, uno con la puerta del castillo sin mensaje y los otros 12 con los signos zodiacales pintados en la puerta del castillo. También encontrarán al objeto llamado “Mago” que estaremos programando simultáneamente junto con el fondo que contiene los 13 disfraces de la puerta del castillo.

2. Hacer clic sobre el “Mago” e iniciar ensamblando en el lienzo la bandera verde. Debajo, hacer que el mago de un mensaje de bienvenida y, posteriormente, solicite a la usuaria o usuario ingresar su día y mes de nacimiento en números. Almacenar cada valor en variables independientes (Día)(Mes).

Figura 1. Registro de día y mes de nacimiento en *Scratch*



3 Armar con los operadores “mayor que” e “igual que” los límites de un rango dentro de un operador “AND”.

Figura 2. Definición de rangos con operadores "mayor que" e "igual que" en *Scratch*

4 Ahora, armar el segundo límite del rango usando el operador “menor que” e “igual que” dentro de otro “AND”.

Figura 3. Configuración del segundo límite del rango con “menor que” e “igual que” en *Scratch*

5 Ensamblar ambos rangos en un operador “OR” y ensambla el conjunto en la pregunta de la condicional que marca la clasificación para Aries

Figura 4. Clasificación del signo Aries con operadores AND y OR en *Scratch*

Figura 5. Uso de "Enviar ()" en *Scratch*

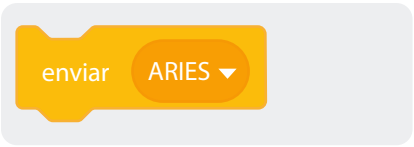
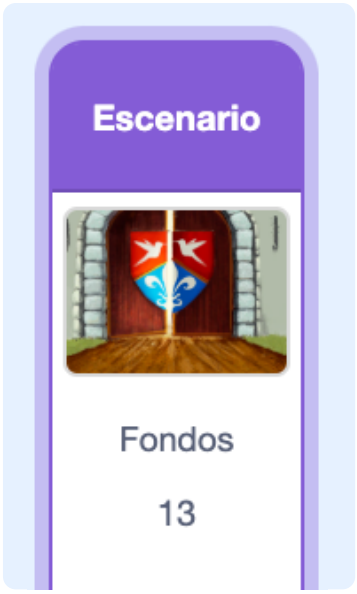


Figura 7. Inicio de la programación del fondo en *Scratch*



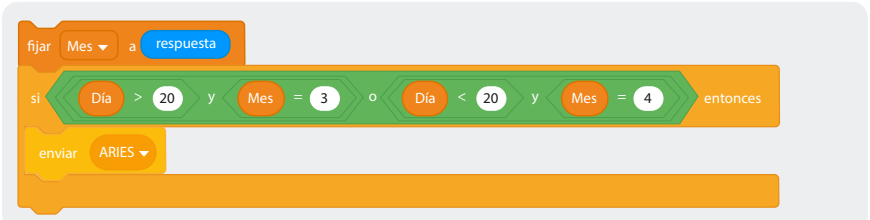
Fíjense que todas las personas nacidas entre el 21 de marzo y el 19 de abril están en esta clasificación.

Ahora aprenderán a programar simultáneamente al "Mago" y al "Fondo" o escenario.

6. Hacer clic en la paleta "Control" y ubicar la opción "Enviar ()".

7. Ubicar esta opción en la parte afirmativa de la condicional correspondiente a Aries.

Figura 6. Envío de mensaje para Aries en *Scratch*

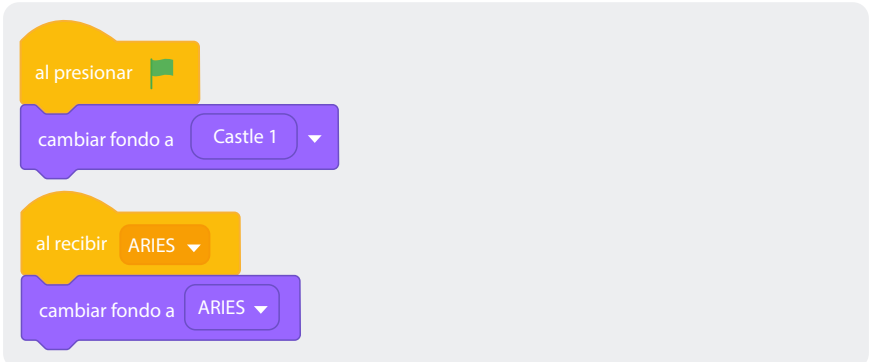


8. Ahora, dar clic sobre el fondo para iniciar con su programación. Ver *Figura 7.*

Una vez allí, seleccionar la pestaña "Código".

9. Programar con la bandera verde al fondo para que al oprimirse se inicie con el disfraz Castillo1 "Castle1". Luego, en la paleta de control encontrarán la opción "Al recibir ()", arrastrar al lienzo de programación y ensamblar en ella la opción "Cambiar fondo a Acuario".

Figura 8. Cambio de fondo al recibir mensaje en *Scratch*

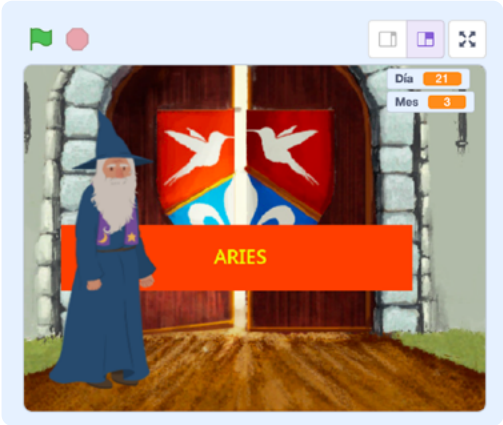


Ahora probar parcialmente el código. Usar este caso de prueba:

Día = 21, Mes = 3

Notar que el fondo ha cambiado y ahora la puerta dice “ARIES”.

Figura 9. Fondo con resultado del programa



10 Volver nuevamente a dar clic sobre el objeto “Mago” y continuar con el código. Ahora, crear la condicional para Tauro siguiendo la misma alternancia entre enviar el mensaje “TAURO” al fondo y, al recibir el mensaje “TAURO”, cambiar el escenario al disfraz correspondiente.

Código del “Mago”

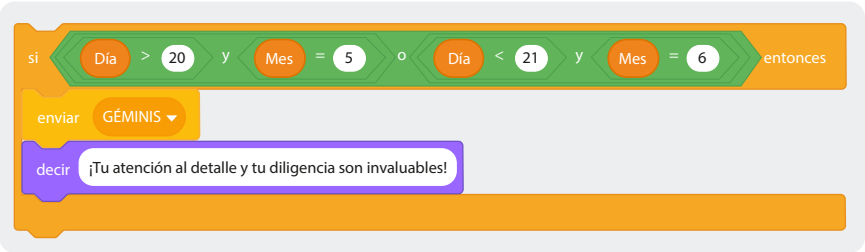
Figura 10. Clasificación del signo Tauro y cambio de fondo en *Scratch*

Código del “Fondo”

11 Completar el código con los 12 signos zodiacales programando simultáneamente el “Mago y el “”Fondo”.

12 Incluir textos a cada condicional del “Mago”.

Figura 11. Programación completa de los 12 signos zodiacales en *Scratch*



Probar el código completo con las fechas de nacimiento de tus compañeras(os) de salón y familiares.

Ahora que han resuelto el reto, podrán realizar la actividad de pensamiento crítico que se menciona en el reto.

Antes de irnos

Esta sección corresponde al 100% de avance de la sesión

De forma individual, regresa a revisar los aprendizajes esperados. Elije la opción de respuesta que mejor describa lo que alcanzaste.

1 ¿Puedes utilizar los aprendizajes logrados para resolver el reto planteado?

- ◯ Sí
- ◯ Parcialmente
- ◯ Aún no

2 ¿Puedes establecer rangos por más de una secuencia numérica con el uso de AND y OR?

- ◯ Sí
- ◯ Parcialmente
- ◯ Aún no

3 ¿Puedes programar simultáneamente Objetos y Fondos en *Scratch* al enviar mensajes entre ellos?

- ◯ Sí
- ◯ Parcialmente
- ◯ Aún no

4 ¿Puedes aplicar lo aprendido en la creación de juegos y aplicaciones?

- ◯ Sí
- ◯ Parcialmente
- ◯ Aún no

Si tus respuestas fueron "Parcialmente" o "Aún no", vuelve a las actividades propuestas en *Scratch*. Luego, discute con tus compañeras y compañeros de grupo lo que se hizo en cada momento de la actividad y el rol al que correspondía. Si todavía te quedan dudas, consúltale a tu docente.

Igualmente, te invitamos a elaborar un esquema de lo que aprendiste esta sesión. Recuerda que el esquema puede mostrar definiciones, conexiones entre conceptos o simplemente ejemplos.

Anexo 1.1 Reto

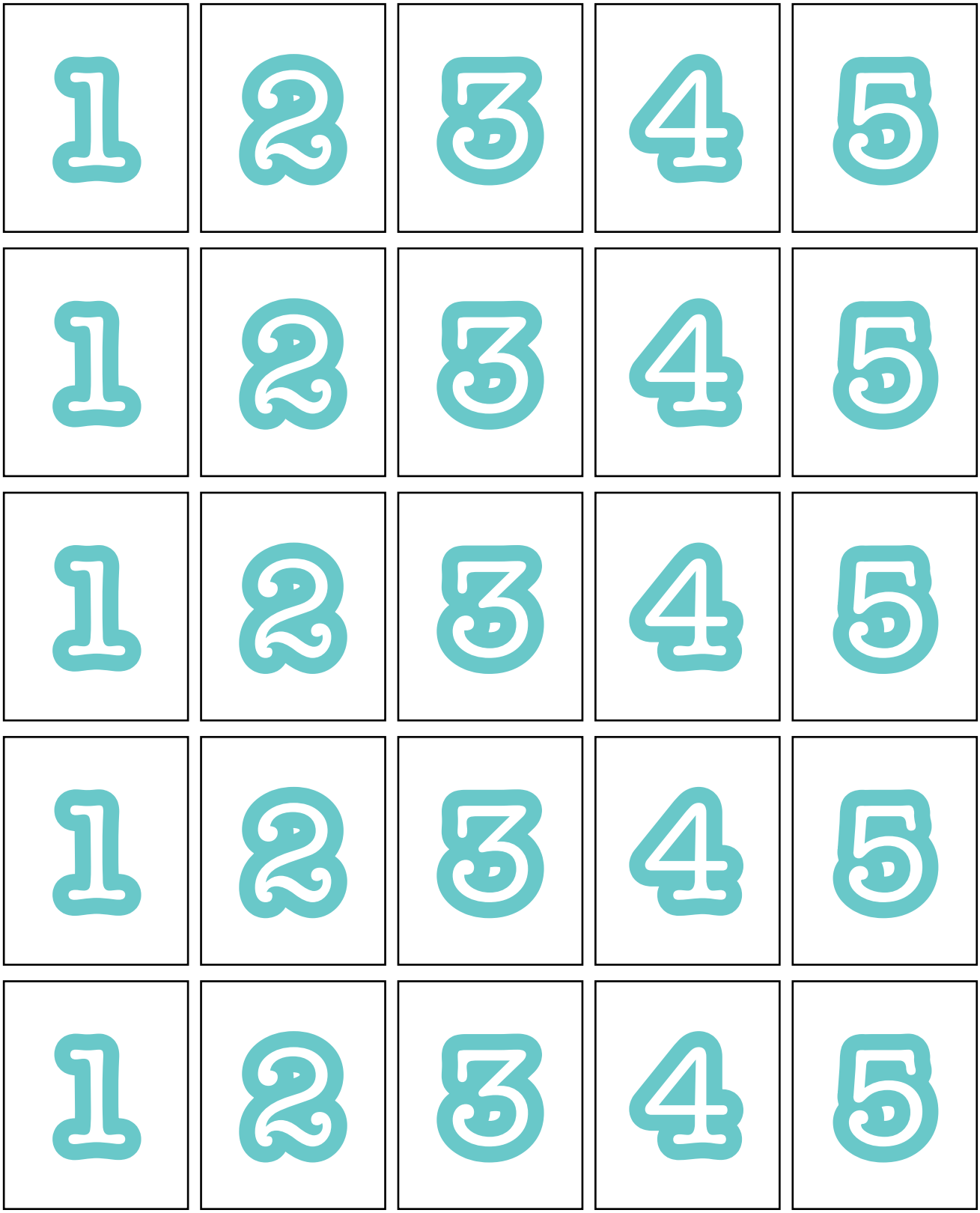
El zodíaco utilizado por astrólogas y astrólogos es, esencialmente, de origen caldeo. Fue evolucionando de forma muy gradual y a menudo fortuita durante los dos milenios que precedieron a nuestra era, con aportes de numerosas civilizaciones (Mesopotamia, Babilonia, Grecia, Roma, Persia, entre otros) todas ellas con la necesidad de encontrar en lo que se veía en el firmamento una relación con las creencias del momento y crear calendarios. Interesarse por esta construcción lenta y progresiva y buscar las razones que empujó a la humanidad a modificar y perfeccionar el modelo establecido según los conocimientos de la época, ayuda a poner en perspectiva las afirmaciones no científicas de la astrología. Cada signo del zodíaco se asocia a imágenes imaginarias en el firmamento que se asemejan a figuras. Existen y han existido muchos horóscopos, esta actividad utiliza el denominado occidental, pero se pueden mencionar otros como el chino, el maya, el azteca o el árabe, entre muchos otros.

Tu reto es construir un programa que le permita saber a cada persona, sabiendo su mes y día de nacimiento, qué signo le corresponde según esta tradición del horóscopo occidental.

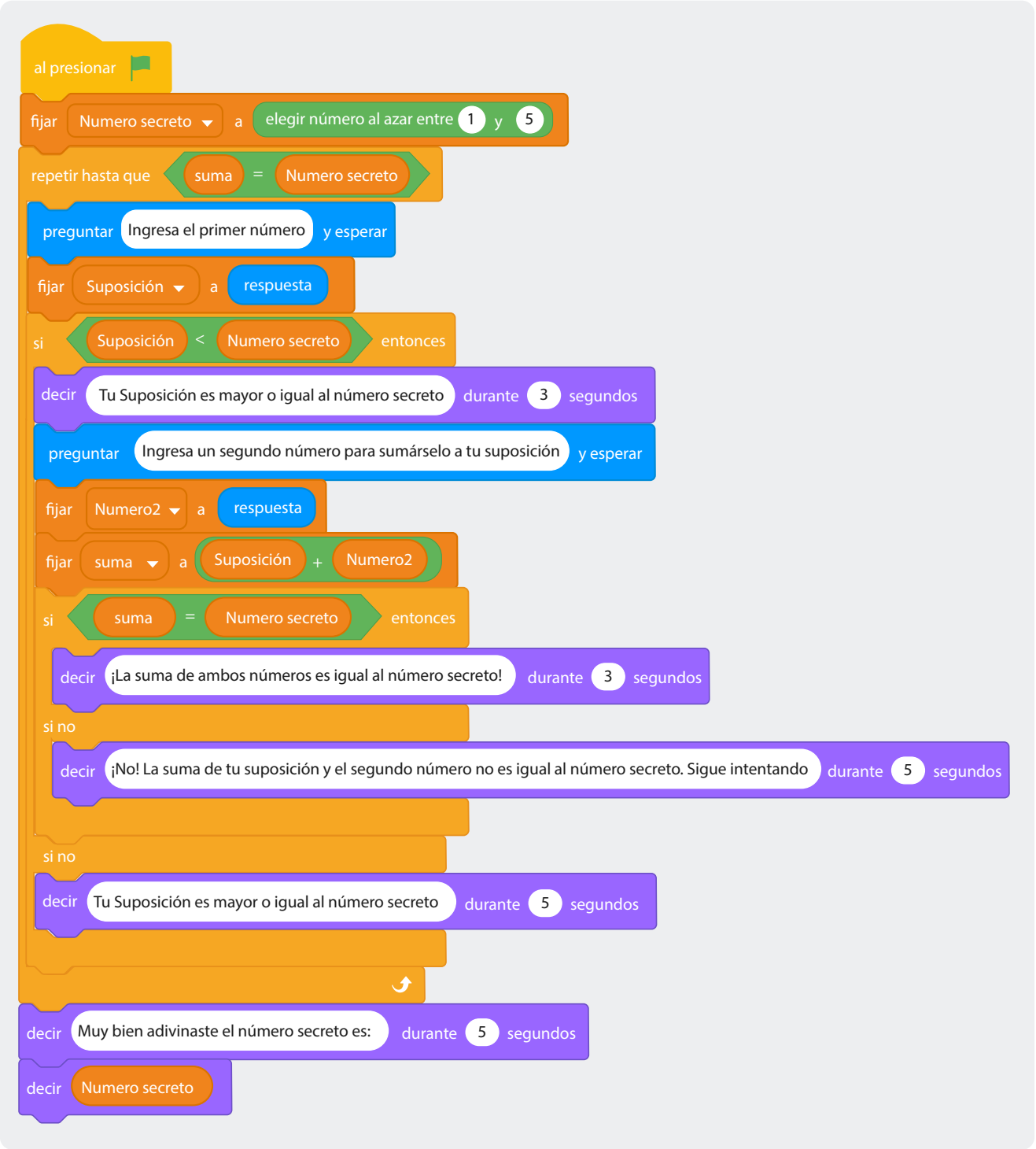
La siguiente tabla muestra la asignación del signo en los horóscopos actuales:

Entre el 21 de marzo y el 19 de abril	Aries
Entre el 20 de abril y el 20 de mayo	Tauro
Entre el 21 de mayo y el 20 de junio	Géminis
Entre el 21 de junio y el 22 de julio	Cáncer
Entre el 23 de julio y el 22 de agosto	Leo
Entre el 23 de agosto y el 22 de septiembre	Virgo
Entre el 23 de septiembre y el 23 de octubre	Libra
Entre el 24 de octubre y el 20 de noviembre	Escorpión
Entre el 22 de noviembre y el 21 de diciembre	Sagitario
Entre el 22 de diciembre y el 19 de enero	Capricornio
Entre el 20 de enero y el 18 de febrero	Acuario
Entre el 19 de febrero y el 20 de marzo	Piscis

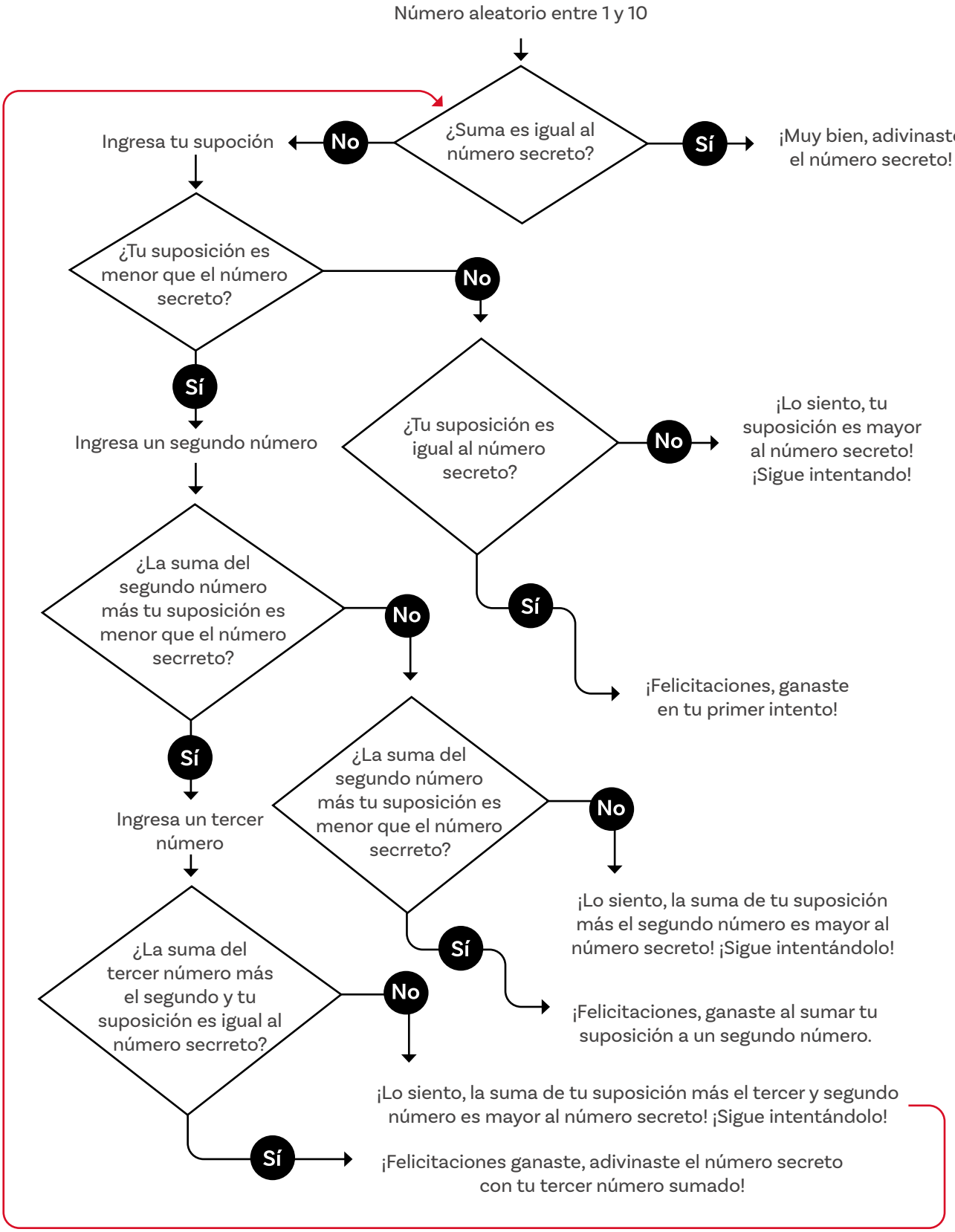
Anexo 2.1 Cartas “Adivina el número secreto”



Anexo 2.2 Código "Adivina el número secreto"



Anexo 2.3 Diagrama de flujo - Adivina el número secreto"



TIC

Apoya:

Educación

BRITISH COUNCIL

Colombia Programa
{EL CÓDIGO A TU FUTURO}