



Girls Who Code en casa

Valientes, no perfectas
Depuración en Scratch

Resumen de la actividad

En esta actividad, trataremos con más detalle lo que significa ser **valientes, no perfectas** cuando programamos. Solo porque un programa no funcione de la forma prevista no quiere decir que sea un fracaso. Hay muchas habilidades que podemos aprender y aplicar para corregir errores y mejorar nuestros productos o proyectos.

La **depuración** es una estrategia que utilizan los expertos informáticos para encontrar y resolver problemas, o **bugs**, en los programas. En esta actividad, te ayudaremos a depurar, o resolver, ¡tres programas en Scratch! Antes de comenzar esta actividad, te recomendamos que leas el artículo destacado, Mujeres y tecnología: Ayanna Howard. El primer proyecto de Ayanna para la NASA fue crear un robot que reuniera información sobre el medio ambiente en Marte. Obtén más información de cómo Ayanna utiliza coraje y confianza para innovar.

Materiales

- [Scratch en línea](#) o [Scratch sin conexión](#)
- [Reto 1 de depuración](#)
- [Reto 2 de depuración](#)
- [Reto 3 de depuración](#)
- Opcional: Folleto del reto de depuración
- Soluciones del folleto del reto de depuración

Artículo destacado de «Mujeres en tecnología»: Ayanna Howard



Fuente de la imagen: [Black Sci-Fi](#)

Los modelos a seguir vienen en todas las formas y tamaños. De hecho, ¡el primer modelo a seguir de la Dra. Howard ni siquiera era humano! La mujer biónica, una superheroína robótica, inspiró en la Dra. Howard la pasión por la construcción de robots y la ingeniería desde temprana edad. Estudió ingeniería eléctrica y ciencias informáticas en la universidad y en la escuela de posgraduados y, con el paso del tiempo, obtuvo un título en negocios.

Como una forma de seguir cultivando su amor por el estudio, la robótica y la ingeniería eléctrica, la Dra. Howard actualmente se desempeña como profesora de bioingeniería y es cofundadora y directora de Tecnología de Zyrobotics, una compañía de robótica educativa. Además de eso, la Dra. Howard ha desarrollado con la NASA robots que están recolectando información para habitar Marte.

Mira este [video](#) para conocer más sobre el primer trabajo de la Dra. Howard en la NASA. Ella también analiza por qué es importante la diversidad en el lugar de trabajo, y cómo se enfrentó a una situación incómoda en el trabajo.

Reflexión

Ser una experta en informática significa mucho más que simplemente ser buena programando. Tómame unos minutos para reflexionar sobre cómo Ayanna y su trabajo se ven reflejados en los puntos fuertes que todo verdadero experto informático debe desarrollar: coraje, resistencia, creatividad y propósito.



La Dra. Howard mostró valentía cuando se hizo oír en una situación incómoda en el trabajo. ¿Qué te dirías a ti misma en ese momento?

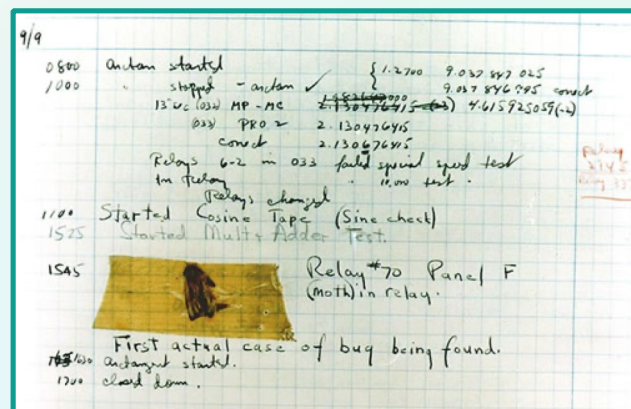
Comparte tus respuestas con un familiar o amigo. Anima a otros para que lean sobre Ayanna y se unan al debate.

Paso 1: ¿Qué es la depuración? (5 minutos)

Piensa en una ocasión en la que intentaste resolver un problema; podría ser una tarea difícil de la escuela o intentar encontrar algo que perdiste. ¿Lo pudiste resolver al instante? Con frecuencia, cuando intentamos resolver un problema, fracasamos varias veces antes de tener éxito. Los programadores se pasan la mayor parte de sus días intentando encontrar y resolver problemas con el código que desarrollan. El fracaso es una parte importante de las ciencias de la informática 😊.

A un error en un programa informático o en una pieza de hardware se le llama **bug** («bicho», en inglés). Al proceso de identificar y eliminar los errores o «bugs» del hardware o el programa informático se le llama **depurar** («debugging», en inglés). Los orígenes del término provienen de Grace Hopper, una de las pioneras en el mundo de la informática. Mientras trabajaba en uno de los primeros ordenadores, el equipo de Grace Hopper encontró una polilla dentro de la computadora que estaba causando un error... ¡un bicho de verdad! La entrada del diario de Grace donde aparece la polilla pegada con papel de celo, se reconoce a día de hoy como la primera mención a un «bug» informático en la historia. Está exhibido en el Museo Smithsonian de Historia Americana en Washington D.C.

El primer «bug» informático de la historia



Fuente de la imagen: [Atlas Obscura](#)

Dedica un momento a pensar en lo que significa para ti la frase, **Valientes, no perfectas**. ¿Por qué es importante ser **valiente** cuando intentas hacer algo nuevo, en vez de intentar ser **perfecta**? A menudo, enfrentamos retos cuando intentamos hacer algo nuevo. Es importante ser **resilientes** y tratar de ver nuestros errores y retos como oportunidades de aprendizaje. La depuración es una oportunidad de aprender de nuestros errores. Generalmente, estas estrategias te ayudan a superar retos más grandes y complicados en el futuro.



Paso 2: Inicia sesión en Scratch y navega por la interfaz (5 a 10 minutos)

Scratch es una plataforma de programación gratuita con un lenguaje de programación basado en bloques desarrollado por el MIT que te permite animaciones, juegos y relatos interactivos.

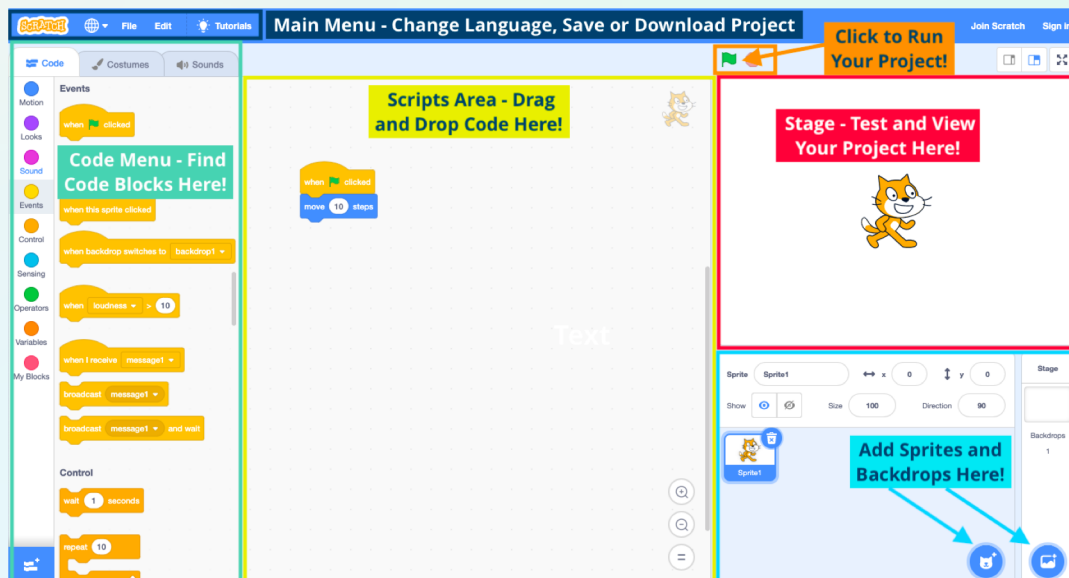
1. Inscríbete o inicia sesión en [Scratch](#).

Para guardar tu trabajo en la plataforma en línea de Scratch tendrás que crear una cuenta, si es que aún no tienes una. Sigue las instrucciones que aparecerán en el formulario de inscripción para crear una cuenta. Si eres menor de 13 años, necesitarás la dirección de correo electrónico de uno de tus padres para inscribirte. Si no deseas crear una cuenta, también tienes la opción de descargar y usar la [versión sin conexión de Scratch 3.0](#).

2. Explora la interfaz de Scratch.

Si Scratch es nuevo para ti, dedica unos minutos a **Crear**   un proyecto nuevo, así podrás explorar la interfaz de Scratch. También puedes ver el tutorial de Scratch [Getting Started \(Tus primeros pasos\)](#).

En la imagen a continuación, hay algunas secciones fundamentales de la plataforma Scratch con las que interactuarás.



Paso 3: Revisa las estrategias de depuración de Scratch (5 minutos)

Antes de comenzar a depurar, dedica algunos minutos a revisar algunas estrategias sugeridas que podrías utilizar cuando intentes encontrar y resolver un bug en uno de tus programas.

1. **Describe el error (o bug).** ¿Cómo te diste cuenta de que había un bug en el programa? Piensa en lo que esperabas que pasara y lo que en verdad ocurrió.
2. **Revisa tu código y piensa en los errores posibles.** Revisa línea por línea el código. ¡Incluso puedes leer en voz alta cada línea del código! Mientras vuelves a leer el código, piensa en cuál o cuáles líneas del código (o bloques para Scratch) pueden ser el origen del error o de los resultados inesperados.
3. **Coloca marcadores de código en tu código a modo de puntos de control.** Si el código es extenso, puede ser útil que lo dividas en varias secciones. En Scratch, puedes usar un bloque **say** (decir) para saber dónde una línea del código se encuentra en el programa. Esta estrategia puede ayudar a disminuir las ubicaciones posibles del bug.



Por ejemplo, si sabes que cuando se ejecuta tu primer marcador (es decir, el sprite dice el texto en el bloque determinado que estás usando como marcador) y el programa sigue funcionando según lo esperado, entonces sabes que es probable que el bug no se encuentre antes del primer marcador. Si observas el bug en tu programa antes de ejecutar el segundo marcador, sabes que es probable que el bug se encuentre en el código entre el primer y el segundo marcador.

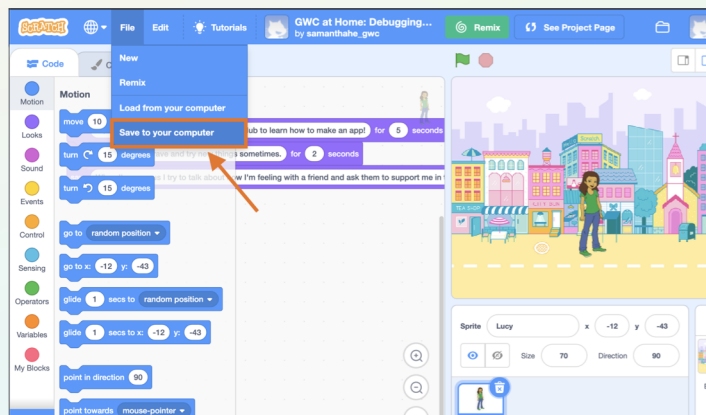
4. **Después de encontrar el bug, intenta comprender por qué el código es un bug.** Piensa en por qué la línea de código o bloque de código ocasiona un error en tu programa. Algunos errores comunes son errores de ortografía, colocar los bloques de código en un orden incorrecto, o usar un bloque equivocado. Algunos editores de código incluyen mensajes de error que describen por qué una línea de código es un bug. Scratch no tiene estos mensajes de error, así que tendrás que pensar más.
5. **Corrige y prueba tu código.** Intenta corregir la línea de código y luego ejecuta otra vez el programa. ¿Todavía existe el bug? Si es así, intenta otra solución y prueba de nuevo el programa. ¿Se produce el bug en distintas áreas? Entonces, es posible que tengas un segundo bug en el programa; repite los pasos nuevamente para encontrar el otro bug. Si no hay bugs en el programa, ¡quiere decir que corregiste el bug!

Paso 4: Reto 1 de depuración (5 a 10 minutos)

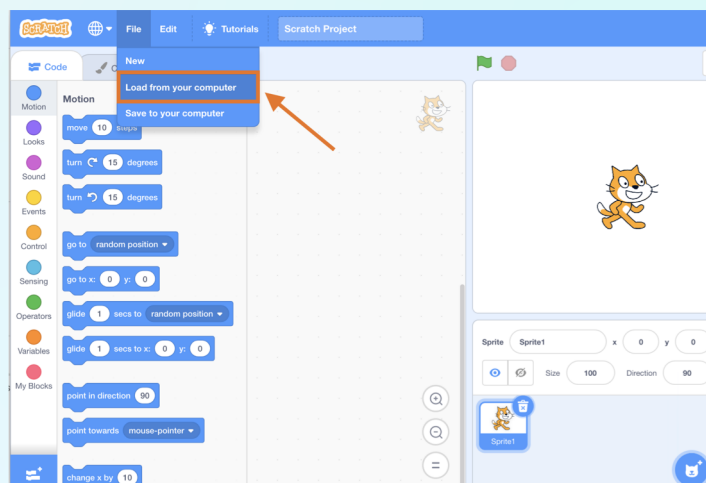
Conocimiento previo: Para resolver este reto, debes familiarizarte con los bloques de [Event](#) (Evento) y [Looks](#) (Aspecto). Procura ver este tutorial [Getting Started](#) (Primeros pasos) para familiarizarte con estos bloques.




1. **Adapta el [código inicial](#).** Haz clic en el botón **Remix** en la esquina superior derecha para hacer una copia del proyecto.
 - Si estás usando el editor fuera de línea de Scratch, tendrás que guardar una copia del proyecto en tu computadora. Haz clic en el botón **See inside** para ver el código del proyecto.
 - Haz clic en **File** (Archivo) en la barra de navegación superior y elige la opción **Save to your computer** (Guardar en tu computadora) en el menú desplegable.



- Abre el editor fuera de línea de Scratch en tu computadora. Haz clic en **File** (Archivo) en la barra de navegación superior y elige la opción **Load to your computer** (Cargar en tu computadora) en el menú desplegable. Busca el archivo guardado del proyecto en tu computadora y haz clic en **Open** (Abrir).






Paso 4: Reto 1 de depuración (continuación)

1. **Prueba el programa e identifica el bug.** En este proyecto, Lucy, nuestro sprite (u objeto) principal, se presenta al usuario. Sin embargo, al hacer clic en la bandera verde, ¡nada sucede! Oprime la bandera verde  para ejecutar el programa. Revisa el código y ve si puedes identificar el bug. Utiliza el **folleto del reto** (páginas 12 y 13) como ayuda para resolver y reflexionar sobre el bug.
2. **Corrige el bug y comprueba la solución [aquí](#).** Consulta la página 14 para ver un análisis más profundo de este bug.
3. **Reflexiona sobre el proceso de depuración.** Piensa en lo que era el bug y cómo afectó el programa. Cuando aprenden a programar, la mayoría de los programadores lleva una hoja de repaso con los errores comunes que les ayuda a aprender de sus errores cuando hacen programas para otros proyectos después.

Paso 5: Reto 2 de depuración (5 a 10 minutos)

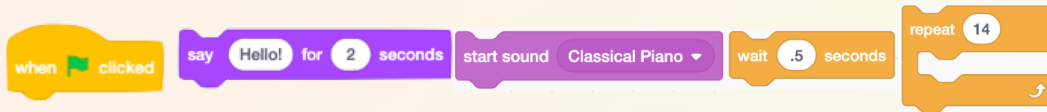
Conocimiento previo: Para resolver este reto, debes familiarizarte con los bloques de [Event](#) (Evento), [Looks](#) (Aspecto), [Motion](#) (Movimiento), [Sensing](#) (Sentidos) y [Loops](#) (Bucles). Procura ver este tutorial [Code a Cartoon](#) (Programar una caricatura) para familiarizarte con estos bloques.






1. **Adapta el [código inicial](#).** Haz clic en el botón  en la esquina superior derecha para hacer una copia del proyecto.
 - Si estás usando el editor fuera de línea de Scratch, tendrás que guardar una copia del proyecto en tu computadora. Haz clic en el botón  para ver el código del proyecto.
 - Haz clic en **File** (Archivo) en la barra de navegación superior y elige la opción **Save to your computer** (Guardar en tu computadora) en el menú desplegable.
 - Abre el editor fuera de línea de Scratch en tu computadora. Haz clic en **File** (Archivo) en la barra de navegación superior y elige la opción **Load to your computer** (Cargar en tu computadora) en el menú desplegable. Busca el archivo guardado del proyecto en tu computadora y haz clic en **Open** (Abrir).
2. **Prueba el programa e identifica el bug.** En este programa, Avery debería comenzar en el centro y presentarse. Después, camina hacia la derecha hasta que llega al borde para ir al Club de programación. Funciona la primera vez que oprimimos la bandera verde  pero, cuando oprimimos nuevamente la bandera verde, Avery está todavía en el lado derecho en vez de aparecer en el centro de la pantalla. Oprime la bandera verde para ejecutar el programa. Revisa el código y ve si puedes identificar el bug. Utiliza el **folleto del reto** (páginas 12 y 13) como ayuda para resolver y reflexionar sobre el bug.
3. **Corrige los bugs y comprueba la solución [aquí](#).** Observa que puedes haber programado a Avery para comenzar en una posición distinta a la enumerada en las soluciones. Consulta la página 15 para ver una análisis más profundo de este bug.
4. **Reflexiona sobre el proceso de depuración.** Piensa en lo que era el bug y cómo afectó el programa. Cuando aprenden a programar, la mayoría de los programadores lleva una hoja de repaso con los errores comunes que les ayuda a aprender de sus errores cuando hacen programas para otros proyectos después.

Paso 6: Reto 3 de depuración (5 a 15 minutos)

Conocimiento previo: Para resolver este reto, debes conocer los bloques de [Event](#) (Evento), [Looks](#) (Aspecto), [Sounds](#) (Sonidos), [Wait](#) (Espera) y [Loops](#) (Bucles). Procura ver este tutorial [Code a Cartoon](#) (Programar una caricatura) para familiarizarte con estos bloques.



1. **Adapta el [código inicial](#).** Haz clic en el botón  en la esquina superior derecha para hacer una copia del proyecto.
 - Si estás usando el editor fuera de línea de Scratch, tendrás que guardar una copia del proyecto en tu computadora. Haz clic en el botón  para ver el código del proyecto.
 - Haz clic en **File** (Archivo) en la barra de navegación superior y elige la opción **Save to your computer** (Guardar en tu computadora) en el menú desplegable.
 - Abre el editor fuera de línea de Scratch en tu computadora. Haz clic en **File** (Archivo) en la barra de navegación superior y elige la opción **Load to your computer** (Cargar en tu computadora) en el menú desplegable. Busca el archivo guardado del proyecto en tu computadora y haz clic en **Open** (Abrir).
2. **Prueba el programa e identifica el bug.** Cuando oprimes la bandera verde , Ada debería hablarte de sí misma y después bailar con la música. Cuando Ada empieza a bailar, algo falla con la música. Se reinicia una y otras vez cada vez que Ada se mueve. ¿Cómo podemos corregir el código para que la música se reproduzca durante todo el tiempo que baila Ada? Oprime la bandera verde para ejecutar el programa. Revisa el código y ve si puedes identificar el bug. Utiliza el **folleto del reto** (páginas 12 y 13) como ayuda para resolver y reflexionar sobre el bug.
3. **Corrige los bugs y comprueba la solución [aquí](#).** Observa que puedes haber programado a Avery para comenzar en una posición distinta a la enumerada en las soluciones. Consulta la página 16 para ver una análisis más profundo de este bug.
4. **Reflexiona sobre el proceso de depuración.** Piensa en lo que era el bug y cómo afectó el programa. Cuando aprenden a programar, la mayoría de los programadores lleva una hoja de repaso con los errores comunes que les ayuda a aprender de sus errores cuando hacen programas para otros proyectos después.

Paso 7: Comparte tu creación (5 minutos)

1. **Comparte tu proyecto en Scratch.**

Después de que resuelvas el reto, oprime el botón Share (Compartir) en Scratch. En las instrucciones, incluye cómo resolviste el bug.

2. **¡Comparte cómo abor das los desafíos con Girls Who Code en casa!**

No te olvides de compartir tus proyectos en las redes sociales. Etiqueta @girlswhocode #codefromhome, ¡y puede que te etiquetemos en nuestra cuenta!

Folleto del reto de depuración

Instrucciones: A medida que avanzas por cada reto, piensa en las siguientes preguntas.

- ¿Qué es un bug? ¿Qué se suponía que pasara? ¿Hay varios bugs?
- ¿Cómo afectó el bug al programa y por qué?
- ¿Cómo corregiste el bug?

Reto 1 de depuración (5 a 10 minutos)

Código inicial: <https://scratch.mit.edu/projects/387548698/>

Bug(s):

Cómo afectó al programa y por qué:

Cómo corregiste el o los bugs:

Folleto del reto de depuración

Reto 2 de depuración (5 a 10 minutos)

Código inicial: <https://scratch.mit.edu/projects/387549618/>

Bug(s):

Cómo afectó al programa y por qué:

Cómo corregiste el o los bugs:

Reto 3 de depuración (5 a 15 minutos)

Código inicial: <https://scratch.mit.edu/projects/387851932/>

Bug(s):


Cómo afectó al programa y por qué:

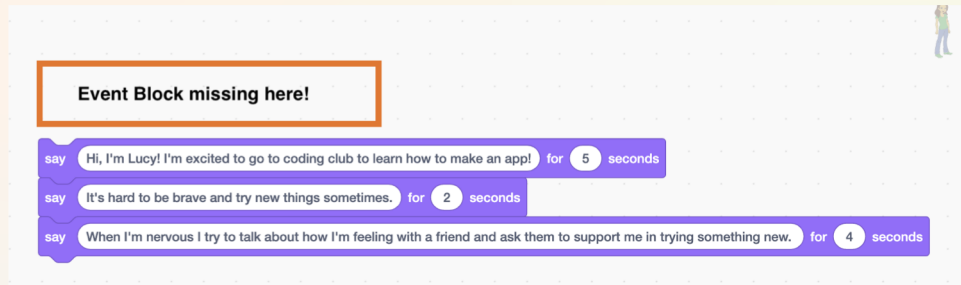
Cómo corregiste el o los bugs:

Solución del reto 1 de depuración


Código inicial con bugs: <https://scratch.mit.edu/projects/387548698/>

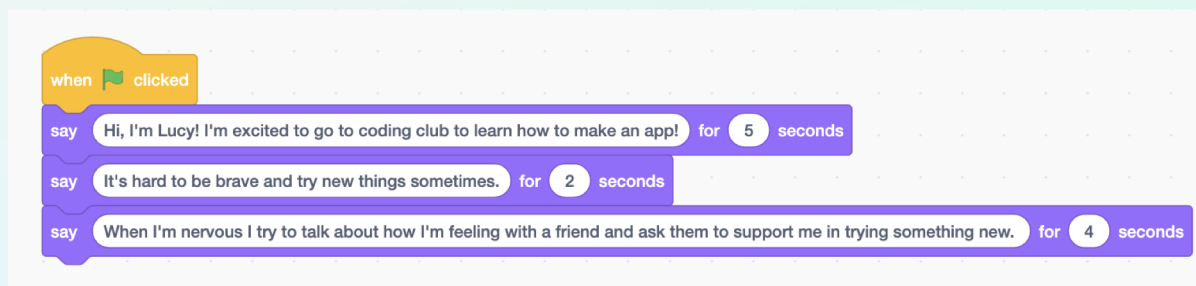
Ejemplo del código solución: <https://scratch.mit.edu/projects/388218006/>

Bug(s): Al hacer clic en la bandera verde  , Lucy debería presentarse. En lugar de eso, ¡nada sucede! Esto se debía a que no había ningún bloque de evento en el código.



Cómo afectó al programa y por qué: Sin un bloque de evento, la computadora no sabe cuándo ejecutar el código que escribimos. Por lo tanto, no pasa nada cuando se hace clic en la bandera verde porque no le dijimos a la computadora lo que debía hacer.

Cómo corregiste el o los bugs: Debemos decirle a la computadora que ejecute el código que escribimos **cuando se haga clic en la bandera verde**. Utilizamos el bloque  y lo pusimos en la parte delantera de los otros bloques de códigos.

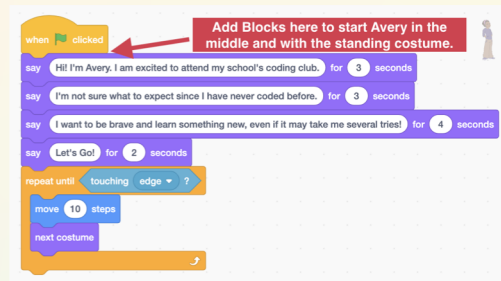


Solución del reto 2 de depuración

Código inicial con bugs: <https://scratch.mit.edu/projects/387548698/>

Ejemplo del código solución: <https://scratch.mit.edu/projects/388218006/>

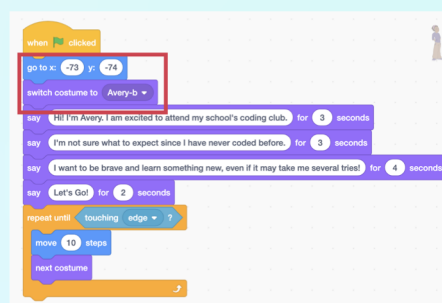
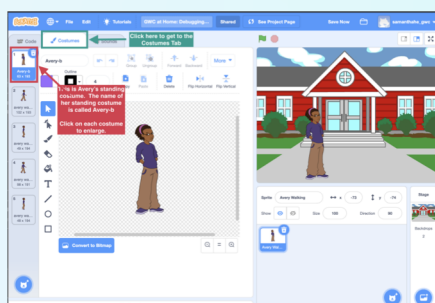
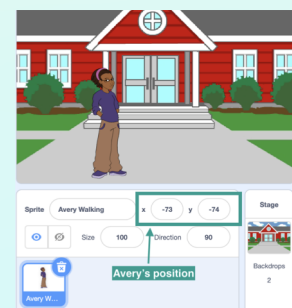
Bug(s): Este programa tiene **dos** bugs. Después de hacer clic en la bandera verde por segunda vez, Avery ya no comienza en el mismo punto de partida, en el centro de la pantalla. En lugar de eso, comienza donde quedó la última vez, a la derecha de la pantalla y en un traje de paseo. El primer bug es su posición y el segundo bug es el traje (o aspecto).



Cómo afectó al programa y por qué: En este programa, Avery debería comenzar en el centro y presentarse. Después, camina hacia la derecha hasta que llega al borde para ir al Club de programación. Debido a que no tenemos un código que inicie siempre a Avery en la misma posición cada vez que se hace clic en la bandera verde, Avery comienza su secuencia de código donde quedó la última vez. En nuestro caso, en el borde del lado derecho de la pantalla.

Cómo corregiste el o los bugs: Para que Avery se inicie siempre en el mismo lugar, debemos utilizar el `go to x: -73 y: -74`. La manera más fácil de elegir la posición inicial de Avery es utilizar el mouse y arrastrar a Avery al lugar donde quieres que ella se inicie. Observa que los números después de la x: y la y: cambian según la posición de Avery. Deberías ver lo mismo en la descripción del sprite, debajo del Stage (Escenario) de Scratch. Usamos el bloque Go to (Ir a) como el primer bloque después de nuestro bloque de evento `when green flag clicked`.


Siguiente, queremos que Avery comience en el traje que muestra cuando está de pie. Para esto, tenemos que usar el `switch costume to Avery-b`, exactamente después de establecer su posición inicial para asegurar que ella siempre comience con el traje con que aparece cuando está de pie. Si no estás segura de cuál es el traje de la posición de pie, mira la pestaña **Costume** (Trajes) y anota el nombre del traje con el que deseas que se inicie. Codificamos primero estos dos nuevos bloques porque queremos configurar nuestro sprite en la misma ubicación exactamente cuando se hace clic en la bandera verde antes de que Avery empiece a hablar y a caminar.

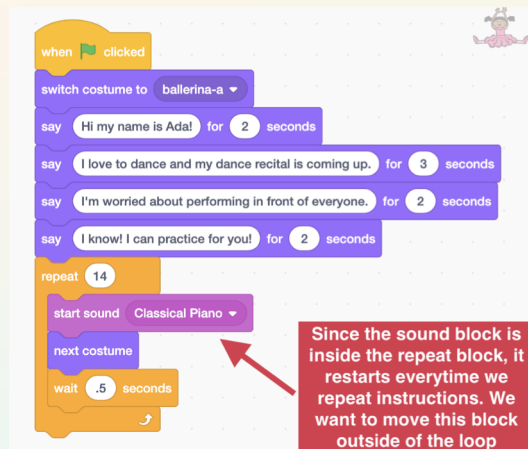


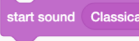
Solución del reto 3 de depuración

Código inicial con bugs: <https://scratch.mit.edu/projects/387548698/>

Ejemplo del código solución: <https://scratch.mit.edu/projects/388218006/>

Bug(s): Cuando oprimes la bandera verde , Ada debería hablarte de sí misma y después bailar con la música. Cuando Ada empieza a bailar, algo falla con la música. Se reinicia una y otras vez cada vez que Ada se mueve. El bug se relaciona con la secuencia, o el orden, en que se codificaron los bloques.



Cómo afectó al programa y por qué: Observa que nuestro bloque de sonido  está programado en el interior del bucle de repetición. Esto significa que cada vez que repetimos el código del interior, ¡comienza nuevamente la música de piano clásico!

Cómo corregiste el o los bugs: Tenemos que sacar nuestro bloque de sonido fuera del bucle pero para ponerlo ¿dónde? Tenemos que repasar lo que queremos que ocurra. Para experimentar, prueba mover el bloque de sonido antes y después del bloque de repetición. Si movemos el bloque de sonido después del bucle de repetición, la música se reproduce después de que baila Ada; no queremos esto. Si movemos el bloque de sonido antes del bucle de repetición, la música se reproduce primero y luego baila Ada. ¡Eso es lo que queremos! Puedes decidir si quieres que el sonido empiece cuando ella baila o exactamente en el momento en que ella comienza a presentarse.

