Girls Who Code At Home

बहादुर, लेकिन परफ़ेक्ट नहीं
Scratch डीबगिंग

गतिविधि की समीक्षा

इस गतिविधि में हम अधिक गहराई से समझेंगे कि कोड करते समय **बहादुर, लेकिन परफ़ेक्ट नहीं** का क्या मतलब होता है! बस इस वजह से कि कोई प्रोग्राम अपेक्षा के अनुसार काम नहीं करता है उसका मतलब आपकी असफलता नहीं होता है। ऐसे कई कौशल हैं जिन्हें हम सीखकर अपनी ग़लतियों को सही करके अपने उत्पादों या परियोजनाओं को बेहतर बनाने के लिए उपयोग कर सकते हैं।

डीबगिंग एक रणनीति है जिसका उपयोग कम्प्यूटर वैज्ञानिक अपने प्रोग्राम में समस्याओं, या **बगों**, का पता लगाने और हल करने के लिए करते हैं। इस गतिविधि में आप Scratch में तीन प्रोग्रामों को डीबग, या ठीक करने में मदद करेंगी! इस गतिविधि में गोता लगाने से पहले, हम इस लेख में उल्लिखित वूमन इन टेक, एयन्ना हॉवर्ड के बारे में पढ़ने की अनुशंसा करते हैं। एयन्ना की नासा के साथ पहली परियोजना में एक रोबोट बनाना था जो मंगल के वातावरण के बारे में सीखता है। एयन्ना नवाचार करने के लिए साहस और आत्मविश्वास का कैसे उपयोग करती है इस बारे में और पढ़ें।

सामग्री

- ऑनलाइन Scratch या ऑफ़लाइन Scratch
- डीबगिंग चुनौती 1
- डीबगिंग चुनौती 2
- डीबगिंग चुनौती 3
- वैकल्पिक: डीबगिंग चुनौती का पर्चा
- डीबगिंग चुनौती के पर्चे के समाधान

विमेन इन टेक स्पॉटलाइट: एयन्ना हॉवर्ड

चित्र का स्रोत: ब्लैक साई-फाई

रोल मॉडल सभी आकृतियों और आकारों में उपलब्ध हैं। वास्तव में, डॉ. हॉवर्ड का पहला रोल मॉडल तो मनुष्य भी नहीं था! एक रोबोटिक सुपरहीरोइन, बायोनिक वूमन ने डॉ. हॉवर्ड के रोबोट बनाने और इंजीनियरिंग के लिए जुनून को छोटी उम्र में ही प्रेरित किया। उन्होंने कॉलेज और ग्रेजुएट स्कूल में इलेक्ट्रिकल इंजीनियरिंग और कम्प्यूटर विज्ञान की पढ़ाई की, और अंततः एक बिज़नेस डिग्री प्राप्त की।

सीखने, रोबोटिक्स, और इलेक्ट्रिकल इंजीनियरिंग के प्रति अपने प्रेम को जारी रखते हुए, डॉ. हॉवर्ड आजकल बायोइंजीनियरिंग की प्रोफ़ेसर और शैक्षणिक रोबोटिक्स कंपनी ज़ाइरोबोटिक्स की सहसंस्थापक और मुख्य टेक्नोलॉजी अधिकारी के रूप में काम करती हैं। यही नहीं, डॉ. हॉवर्ड ने नासा के साथ मिलकर ऐसे रोबोट विकसित किए हैं जो मंगल ग्रह में रहना सीख रहे हैं।

डॉ. हॉवर्ड की नासा में पहली नौकरी के बारे में अधिक जानने के लिए यह वीडियो देखें। वे इस बात पर भी चर्चा करती हैं कि कार्यस्थल में विविधता क्यों महत्वपूर्ण है, और कि उन्होंने कार्यस्थल में एक असहज परिस्थिति का सामना कैसे किया।

सोचें

एक कम्प्यूटर वैज्ञानिक होने का मतलब केवल कोड करने में माहिर होना ही नहीं है। इस बारे में सोचने में थोड़ा समय बिताएँ कि एयन्ना और उनका काम किस तरह से उन शक्तियों से संबंधित है जिन्हें विकसित करने पर महान कम्प्यूटर वैज्ञानिक ध्यान केंद्रित करते हैं - बहादुरी, लचीलापन, सृजनशीलता, और प्रयोजन।

बहादुरी	डॉ. हॉवर्ड ने बहादुरी दर्शाई जब उन्होंने कार्यस्थल में एक असहज परिस्थिति में आवाज़ उठाई। उस पल में आप अपने आप से क्या कहेंगी?

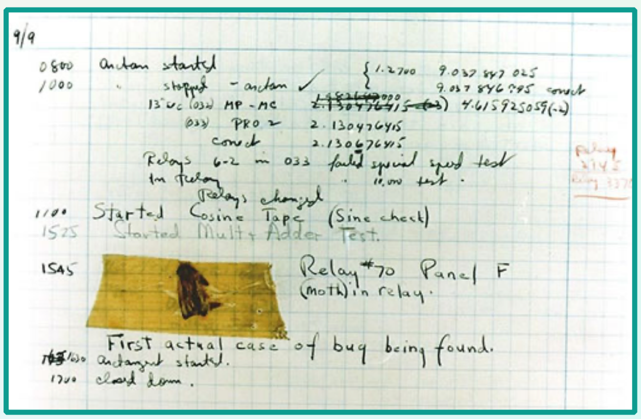
अपनी प्रतिक्रियाओं को परिवार के किसी सदस्य या मित्र के साथ साझा करें। दूसरों को एयन्ना के बारे में अधिक पढ़कर चर्चा में शामिल होने के लिए प्रोत्साहित करें!

चरण 1: डीबगिंग क्या है? (5 मिनट)

ऐसे किसी समय के बारे में सोचें जब आपने किसी समस्या को हल करने का प्रयास किया था, वह स्कूल में दिया गया कोई मुश्किल काम, या किसी खोई हुई चीज को खोजने का प्रयास हो सकता है। क्या आप उसे तत्काल हल कर सकी थीं? अक्सर जब हम किसी समस्या को हल करने का प्रयास करते हैं, तब सफल होने से पहले हमें कई विफलताओं को झेलना पड़ सकता है। प्रोग्रामर अपने अधिकांश दिन अपने कोड की समस्याओं को खोजने और हल करने में बिताते हैं! विफलता कम्प्यूटर विज्ञान का एक बड़ा हिस्सा है 😄 ।

कम्प्यूटर प्रोग्राम या हार्डवेयर में होने वाली त्रुटि को **बग** कहते हैं। कम्प्यूटर हार्डवेयर या सॉफ्टवेयर में त्रुटियों या बगों की पहचान करने और उन्हें हटाने की प्रक्रिया को **डीबगिंग** कहते हैं। इन शब्दों के आविष्कार का श्रेय ग्रेस हॉपर को जाता है, जो कम्प्यूटर विज्ञान के आरंभिक पथ-प्रदर्शकों में से एक थीं। पहले कम्प्यूटरों में से एक पर काम करते समय, ग्रेस हॉपर की टीम ने कम्प्यूटर के भीतर एक कीट, यानी एक वास्तविक भौतिक बग पाया जिसके कारण एक त्रुटि हुई थी! ग्रेस द्वारा टेप पर लगे कीट की पत्रिका में प्रविष्टि को अब इतिहास में पहला रिकार्ड किया गया कप्यूटर बग कहा जाता है। यह वाशिंगटन डी.सी. में स्मिथसोनियन म्यूज़ियम ऑफ अमेरिकन हिस्ट्री में प्रदर्शित है।

पहला रिकार्ड किया गया कम्प्यूटर बग



चित्र का स्रोत: एटलस ऑबस्क्योरा

इस बारे में सोचने के लिए थोड़ा समय निकालें कि आपके लिए **बहादुर, लेकिन परफ़ेक्ट नहीं** का क्या मतलब है। कोई नई चीज आज़माते समय **परफ़ेक्ट** होने की कोशिश करने की बजाय **बहादुर** होना क्यों महत्वपूर्ण है? कई बार हमें कोई नई चीज सीखने की कोशिश करते समय चुनौतियों का सामना करना पड़ता है। **लचीला** होना और यह कोशिश करना महत्वपूर्ण है कि हम अपनी ग़लतियों और चुनौतियों को सीखने के अवसरों के रूप में देखें। डीबगिंग अपनी ग़लतियों से सीखने का एक अवसर है। ये रणनीतियाँ अक्सर भविष्य में बड़ी और अधिक पेचीदा चुनौतियों का सामना करने में आपकी मदद करती हैं।

चरण 2: Scratch में साइन इन करें और इंटरफेस में नेविगेट करें (5-10 मिनट)

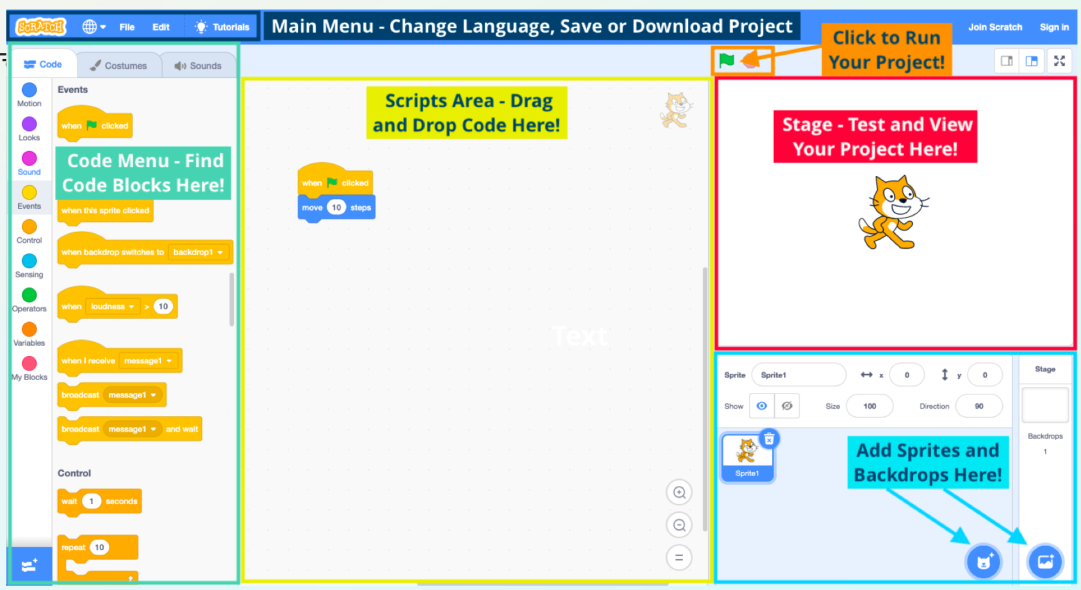
Scratch MIT के द्वारा विकसित किया गया एक मुफ्त प्रोग्रामिंग मंच और ब्लॉक पर आधारित प्रोग्रामिंग भाषा है जो आपको इंटरैक्टिव कहानियाँ, गेम, ऐनिमेशन प्रोग्राम करने में सक्षम करती है।

1. **Scratch में साइन अप या लॉगिन करें।**
   Scratch के ऑनलाइन मंच पर अपने काम को सहेजने के लिए आपको, यदि वह आपके पास पहले से नहीं है, तो एक खाता बनाना होगा। खाता बनाने के लिए साइन अप पर दिए गए निर्देशों का पालन करें। यदि आपकी उम्र 13 से कम हैं तो आपको साइन अप करने के लिए आपको अपने माता/पिता के ईमेल पते की ज़रूरत पड़ेगी। यदि आप खाता नहीं बनाना चाहती हैं तो आप Scratch 3.0 के ऑफ़लाइन संस्करण को डाउनलोड करके भी उसका उपयोग कर सकती हैं।

2. **Scratch के इंटरफेस का अन्वेषण करें।**

   यदि आपके लिए Scratch नई चीज है तो एक नई परियोजना **बनाने** में कुछ मिनट बिताएँ ताकि आप Scratch इंटरफेस का अन्वेषण कर सकें। आप Scratch के इस आरंभ करना ट्यूटोरियल को भी देख सकती हैं!

   नी



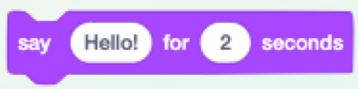
चरण 3: Scratch डीबगिंग रणनीतियों की समीक्षा करें (5 मिनट)

डीबगिंग शुरू करने से पहले, कुछ सुझाई गई रणनीतियों की समीक्षा करने में कुछ मिनट बिताएँ जिनका उपयोग आप अपने प्रोग्राम में किसी बग की खोज और मरम्मत करने के लिए कर सकती हैं।

1. **त्रुटि (या बग) का वर्णन करें।** आपको पहली बार कैसे पता चला कि प्रोग्राम में बग है? इस बारे में सोचें कि आपको क्या होने की अपेक्षा थी और वास्तव में क्या हुआ था।

2. **अपने कोड को पूरा पढ़ें और संभावित त्रुटियों के बारे में सोचें।** कोड की प्रत्येक पंक्ति को एक-एक करके पढ़ें। आप कोड की प्रत्येक पंक्ति को ज़ोर से पढ़कर भी सुना सकती हैं! जब आप अपने कोड को वापस पूरा पढ़ रही हों, तब सोचें कि कोड की कौन सी पंक्ति(याँ) (Scratch के ब्लॉक) त्रुटि या अनपेक्षित परिणामों का स्रोत हो सकती हैं।

3. **अपने सारे कोड में चेक प्वाइंटों के रूप में कोड मार्करों को रखें।** यदि आपका कोड लंबा है तो आपके कोड को अनेकों अनुभागों में विभाजित करने से मदद मिल सकती है। Scratch में आप एक **कहें** ब्लॉक का उपयोग कर सकती हैं ताकि आप जान सकें कि आपके कोड की पंक्ति में आपका प्रोग्राम कहाँ है। यह रणनीति बग के संभावित स्थानों की संख्या को कम करने में मदद कर सकती है।



   उदाहरण के लिए, यदि आप जानते हैं कि आपका पहला मार्कर कब हिट हुआ है (यानी आपके द्वारा मार्कर के रूप में प्रयुक्त कहें ब्लॉक में स्प्राइट कहता है टेक्स्ट) और प्रोग्राम अब भी अपेक्षा के अनुसार काम कर रहा है, तो आपको पता लग जाता है कि संभव है कि बग पहले मार्कर के पहले मौजूद नहीं है। यदि आपको दूसरे मार्कर को हिट करने से पहले अपने प्रोग्राम में बग नज़र आता है, तो आपको पता लग जाता है कि बग के पहले और दूसरे मार्कर के बीच के कोड में स्थित होने की संभावना है।

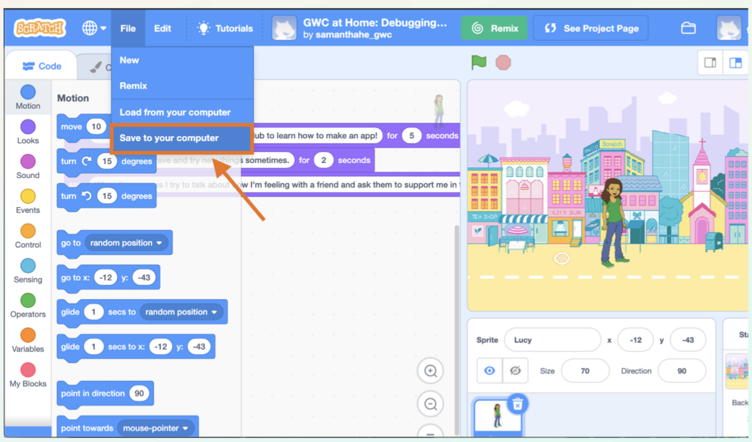
4. **आपको बग का पता लग जाने के बाद, यह समझने की कोशिश करें कि वह कोड बग क्यों है।** विचार करें कि कोड की पंक्ति या कोड ब्लॉक आपके प्रोग्राम में त्रुटि क्यों पैदा करता है। कुछ आम त्रुटियों में वर्तनी की त्रुटि, कोड ब्लॉकों को ग़लत क्रम में रखना, या ग़लत ब्लॉक का उपयोग करना शामिल है। कुछ कोड एडिटर उन त्रुटि संदेशों को शामिल करते हैं जो वर्णन करते हैं कि कोई विशिष्ट कोड की पंक्ति बग क्यों है। Scratch में ये त्रुटि संदेश नहीं होते हैं, इसलिए आपको अधिक सोचने की ज़रूरत पड़ेगी!

5. **अपने कोड की मरम्मत और जाँच करें।** कोड की पंक्ति की मरम्मत करने और फिर अपने प्रोग्राम को दोबारा चलाने की कोशिश करें। क्या बग अब भी मौजूद है? यदि ऐसा है, तो किसी अलग समाधान को आज़माएँ और दोबारा जाँच करें! क्या बग किसी अलग क्षेत्र में है? तब हो सकता है कि आपके प्रोग्राम में कोई दूसरा बग है, अपने नए बग को ढूँढने के लिए चरणों को फिर से दोहराएँ। यदि आपके प्रोग्राम में कोई बग नहीं हैं तो उसका मतलब है कि आपने अपने बग की मरम्मत कर ली है!

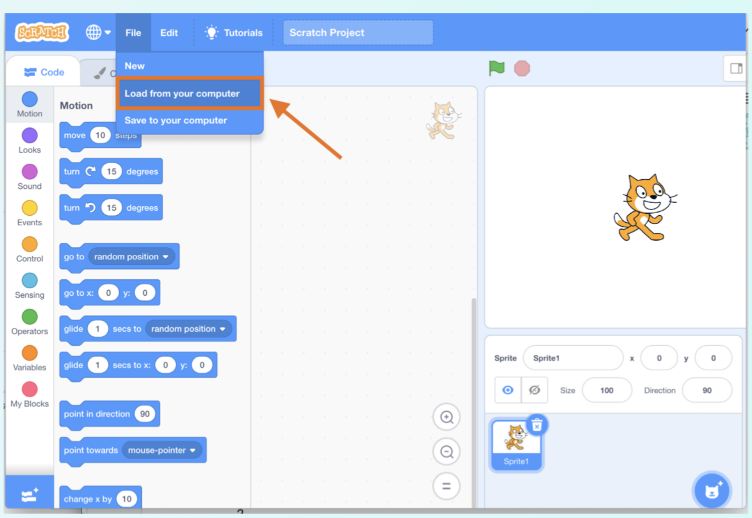
चरण 4: डीबगिंग चुनौती 1 को सुलझाएँ (5-10 मिनट)

पूर्व ज्ञान: *इस चुनौती को हल करने के लिए आपको इवेंट और लुक्स ब्लॉकों से परिचित होना चाहिए। इन ब्लॉकों से परिचित होने के लिए इस ट्युटोरियल शुरू करना को देखने की कोशिश करें।*

1. **स्टार्टर कोड को रीमिक्स करें।** प्रॉजेक्ट की एक Remix कॉपी बनाने के लिए ऊपर दायीं ओर बटन पर क्लिक करें।
   - यदि आप Scratch ऑफ़लाइन एडिटर का उपयोग कर रहे हैं तो आपको अपने कम्प्यूटर में प्रॉजेक्ट की एक कॉपी सहेजनी होगी। प्रॉजेक्ट कोड को देखने के लिए See inside बटन पर क्लिक करें।
   - ऊपरी नेविगेशन बार पर फाइल **को क्लिक करें और ड्रॉप-डाउन मेन्यू में** अपने कम्प्यूटर में सहेजें विकल्प चुनें।



   - अपने कम्प्यूटर पर Scratch ऑफ़लाइन एडिटर खोलें। ऊपरी नेविगेशन बार पर **फाइल** पर क्लिक करें और ड्रॉप-डाउन मेन्यू में **अपने कम्प्यूटर में लोड करें** विकल्प चुनें। अपने कम्प्यूटर पर अपनी सहेजी हुई प्रॉजेक्ट फाइल ढूँढें और **खोलें** पर क्लिक करें।



चरण 4: डीबगिंग चुनौती 1 को सुलझाएँ (जारी)

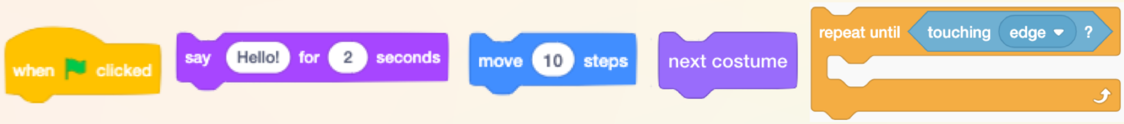
2. **प्रोग्राम की जाँच करें और बग की पहचान करें।** इस प्रॉजेक्ट में लूसी, जो हमारा मुख्य स्प्राइट (या वस्तु) है, प्रयोक्ता को अपना परिचय देती है। हालाँकि, जब हरी झंडी पर क्लिक किया जाता है तो कुछ भी नहीं होता है! हरी झंडी को दबाकर प्रोग्राम को चलाएँ। कोड की समीक्षा करें और देखें कि क्या आप बग को पहचान सकती हैं। बग को हल करने और उसके बारे में सोचने के लिए पृष्ठ 12-13 पर **चुनौती पर्चे** का उपयोग करें।

3. **बग की मरम्मत करें और अपने समाधान की [यहाँ](#) जाँच करें।** इस बग पर अधिक गहरी चर्चा के लिए पृष्ठ 14 देखें!

4. **अपनी डीबगिंग प्रक्रिया पर विचार करें।** इस बारे में सोचें कि बग क्या था और उसने प्रोग्राम को कैसे प्रभावित किया। कोड करने का तरीक़ा सीखते समय, कई प्रोग्रामर आम त्रुटियों वाला काग़ज़ का एक टुकड़ा पास में रखते हैं ताकि बाद में अन्य प्रॉजेक्ट्स को कोड करते समय अपनी ग़लतियों से सीख सकें।

चरण 5: डीबगिंग चुनौती 2 को सुलझाएँ (5-10 मिनट)

पूर्व ज्ञान: *इस चुनौती को हल करने के लिए आपको इवेंट, लुक्स, गति, सेंसिंग और लूपों ब्लॉकों से परिचित होना चाहिए। इन ब्लॉकों से परिचित होने के लिए इस कोड ए कार्टून ट्युटोरियल को देखने की कोशिश करें।*



1. **स्टार्टर कोड को रीमिक्स करें।** प्रॉजेक्ट की एक कॉपी बनाने के लिए ऊपर दायीं ओर बटन Remix पर क्लिक करें।
   - यदि आप Scratch ऑफ़लाइन एडिटर का उपयोग कर रहे हैं तो आपको अपने कम्प्यूटर में प्रॉजेक्ट की एक कॉपी सहेजनी होगी। प्रॉजेक्ट कोड को देखने के लिए See inside बटन पर क्लिक करें।
   - ऊपरी नेविगेशन बार पर **फाइल** को क्लिक करें और ड्रॉप-डाउन मेन्यू में **अपने कम्प्यूटर में सहेंजें** विकल्प चुनें।
   - अपने कम्प्यूटर पर Scratch ऑफ़लाइन एडिटर खोलें। ऊपरी नेविगेशन बार पर **फाइल** पर क्लिक करें और ड्रॉप-डाउन मेन्यू में **अपने कम्प्यूटर में लोड करें** विकल्प चुनें। अपने कम्प्यूटर पर अपनी सहेजी हुई प्रॉजेक्ट फाइल ढूँढें और **खोलें** पर क्लिक करें।

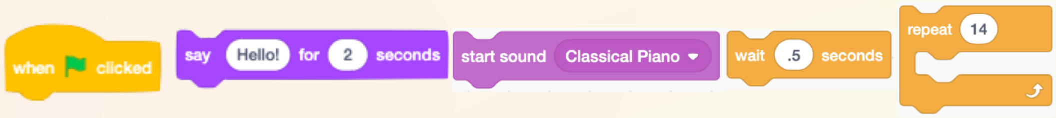
2. **प्रोग्राम की जाँच करें और बग की पहचान करें।** इस प्रोग्राम में एवरी को मध्य में शुरू करना चाहिए और ख़ुद का परिचय देना चाहिए, फिर वह दायीं ओर चलती है जब तक कि वह कोडिंग क्लब जाने के लिए सिरे तक नहीं पहुँच जाती है। जब हम हरी झंडी को पहली बार दबाते हैं तब यह काम करता है लेकिन जब हम हरी झंडी को फिर से दबाते हैं तो एवरी स्क्रीन के मध्य में शुरू करने की बजाय अब भी दायीं ओर ही रहती है। हरी झंडी को दबाकर प्रोग्राम को चलाएँ। कोड की समीक्षा करें और देखें कि क्या आप बग को पहचान सकती हैं। बग को हल करने और उसके बारे में सोचने के लिए पृष्ठ 12-13 पर **चुनौती पर्चे** का उपयोग करें।

3. **बगों की मरम्मत करें और अपने हल की यहाँ जाँच करें।** ध्यान रखें, आपने एवरी को समाधानों में सूचीबद्ध स्थिति से किसी अलग स्थिति से शुरू करने के लिए कोड किया हो सकता है। इस बग पर अधिक गहरी चर्चा के लिए पृष्ठ 15 देखें!

4. **अपनी डीबगिंग प्रक्रिया पर विचार करें।** इस बारे में सोचें कि बग क्या था और उसने प्रोग्राम को कैसे प्रभावित किया। कोड करने का तरीक़ा सीखते समय, कई प्रोग्रामर आम त्रुटियों वाला काग़ज़ का एक टुकड़ा पास में रखते हैं ताकि बाद में अन्य प्रॉजेक्ट्स को कोड करते समय अपनी ग़लतियों से सीख सकें।

चरण 6: डीबगिंग चुनौती 3 को सुलझाएँ (5-15 मिनट)

पूर्व ज्ञान: *इस चुनौती को हल करने के लिए आपको इवेंट, लुक्स, ध्वनियों, प्रतीक्षा, और लूपों ब्लॉकों का ज्ञान होना चाहिए। इन ब्लॉकों से परिचित होने के लिए इस कोड ए कार्टून ट्युटोरियल को देखने की कोशिश करें।*



1. **स्टार्टर कोड को रीमिक्स करें।** प्रॉजेक्ट की एक कॉपी बनाने के लिए ऊपर दायीं ओर बटन Remix पर क्लिक करें।
    - यदि आप Scratch ऑफ़लाइन एडिटर का उपयोग कर रहे हैं तो आपको अपने कम्प्यूटर में प्रॉजेक्ट की एक कॉपी सहेजनी होगी। प्रॉजेक्ट कोड को देखने के लिए See inside बटन पर क्लिक करें।
    - ऊपरी नेविगेशन बार पर **फाइल** को क्लिक करें और ड्रॉप-डाउन मेन्यू में **अपने कम्प्यूटर में सहेंजें** विकल्प चुनें।
    - अपने कम्प्यूटर पर Scratch ऑफ़लाइन एडिटर खोलें। ऊपरी नेविगेशन बार पर **फाइल** पर क्लिक करें और ड्रॉप-डाउन मेन्यू में **अपने कम्प्यूटर में लोड करें** विकल्प चुनें। अपने कम्प्यूटर पर अपनी सहेजी हुई प्रॉजेक्ट फाइल ढूँढें और **खोलें** पर क्लिक करें।

2. **प्रोग्राम की जाँच करें और बग की पहचान करें।** जब आप हरी झंडी दबाती हैं, एडा को आपको अपने बारे में बताना चाहिए और फिर संगीत की धुन पर नाचना चाहिए। जब एडा नाचना शुरू करती है तब संगीत के साथ कुछ गड़बड़ होती है! वह एडा जब भी हिलती है तब बार-बार शुरू से बजने लगता है। हम कोड की कैसे मरम्मत करें कि एडा के नाचने के दौरान संगीत उसके साथ-साथ बजता रहे? हरी झंडी को दबाकर प्रोग्राम को चलाएँ। कोड की समीक्षा करें और देखें कि क्या आप बग को पहचान सकती हैं। बग को हल करने और उसके बारे में सोचने के लिए पृष्ठ 12-13 पर **चुनौती पर्चे** का उपयोग करें।

3. **बगों की मरम्मत करें और अपने हल की यहाँ जाँच करें।** ध्यान रखें, आपने एवरी को समाधानों में सूचीबद्ध स्थिति से किसी अलग स्थिति से शुरू करने के लिए कोड किया हो सकता है। इस बग पर अधिक गहरी चर्चा के लिए पृष्ठ 16 देखें!

4. **अपनी डीबगिंग प्रक्रिया पर विचार करें।** इस बारे में सोचें कि बग क्या था और उसने प्रोग्राम को कैसे प्रभावित किया। कोड करने का तरीक़ा सीखते समय, कई प्रोग्रामर आम त्रुटियों वाला काग़ज़ का एक टुकड़ा पास में रखते हैं ताकि बाद में अन्य प्रॉजेक्ट्स को कोड करते समय अपनी ग़लतियों से सीख सकें।

चरण 7: अपनी रचना को साझा करें (5 मिनट)

1. **अपने प्रॉजेक्ट को Scratch पर साझा करें।**
   चुनौती को हल कर लेने के बाद Scratch पर साझा करें बटन दबाएँ। निर्देश अनुभाग में लिखें कि आपने बग को कैसे हल किया!

2. **साझा करें कि आप Girls Who Code At Home के साथ चुनौतियों को कैसे हल कर रही हैं!**
   सोशल मीडिया पर अपने प्रोजेक्ट्स साझा करना न भूलें। @girlswhocode #codefromhome टैग जोड़ दें, और हो सकता है कि हम हमारे अकाउंट पर आपकी रचना दिखाएं!

डीबगिंग चुनौती का पर्चा

निर्देश: जब आप प्रत्येक चुनौती से गुजरती हैं तब निम्नलिखित प्रश्नों के बारे में सोचें।

- बग क्या है? क्या होना चाहिए था? क्या वहाँ एकाधिक बग हैं?
- बग ने प्रोग्राम को कैसे प्रभावित किया और क्यों?
- आपने बग की मरम्मत कैसे की?

डीबगिंग चनौती 1 (5-10 मिनट)

स्टार्टर कोड: https://scratch.mit.edu/projects/387548698/

बग:

इसने प्रोग्राम को कैसे प्रभावित किया और क्यों:

आपने बग(गों) की मरम्मत कैसे की:

डीबगिंग चुनौती का पर्चा

डीबगिंग चुनौती 2 (5- 10 mins)

स्टार्टर कोड: https://scratch.mit.edu/projects/387549618/

बग:

इसने प्रोग्राम को कैसे प्रभावित किया और क्यों:

आपने बग(गों) की मरम्मत कैसे की:

डीबगिंग चुनौती 3 (5-15 मिनट)

स्टार्टर कोड: https://scratch.mit.edu/projects/387851932/

बग:

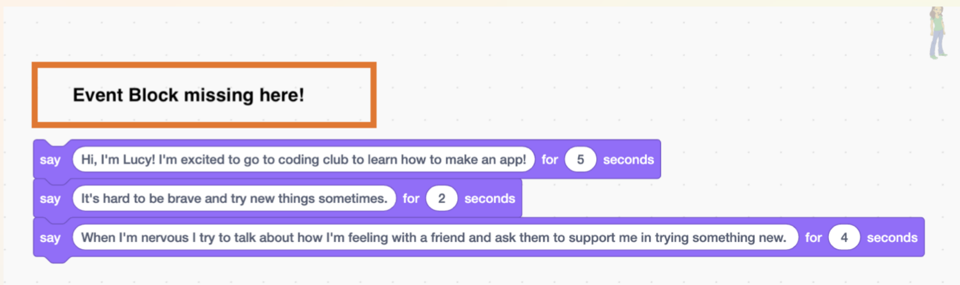
इसने प्रोग्राम को कैसे प्रभावित किया और क्यों:

आपने बग(गों) की मरम्मत कैसे की:

डीबगिंग चुनौती 1 का समाधान

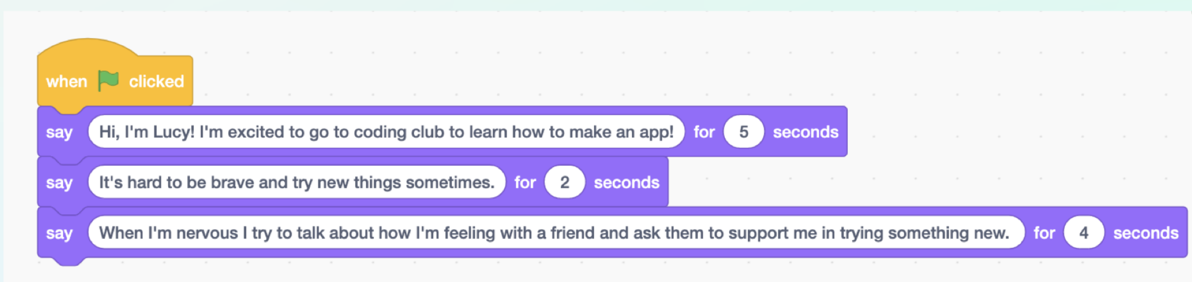
बग वाला स्टार्टर कोड: https://scratch.mit.edu/projects/387548698/
नमूना समाधान कोड: https://scratch.mit.edu/projects/388218006/

बग: हरी झंडी पर क्लिक करते समय लूसी को अपना परिचय देना चाहिए। इसकी बजाय, कुछ नहीं होता है! ऐसा इसलिए हुआ क्योंकि कोड में कोई इवेंट ब्लॉक मौजूद नहीं था।



इसने प्रोग्राम को कैसे प्रभावित किया और क्यों: इवेंट ब्लॉक के बिना कम्प्यूटर को पता नहीं होता है कि हमारे द्वारा लिखा गया कोड कब चलाना है। इसलिए जब हरी झंडी को दबाया गया तो कुछ नहीं हुआ क्योंकि हमने कम्प्यूटर को कुछ भी करने के लिए नहीं कहा था।

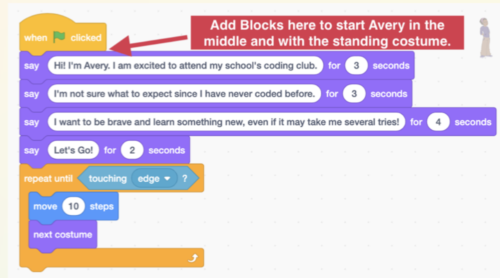
आपने बग(गों) की मरम्मत कैसे की: हमें कम्प्यूटर को बताना चाहिए कि **जब हरी झंडी दबाई जाए तब** हमारे द्वारा लिखे गए कोड को करे। हमने ब्लॉक का उपयोग किया when clicked और इसे अन्य कोड ब्लॉकों के सामने संलग्न कर दिया।.



डीबगिंग चुनौती 2 का समाधान

बग वाला स्टार्टर कोड: https://scratch.mit.edu/projects/387549618/
नमूना समाधान कोड: https://scratch.mit.edu/projects/388333942/

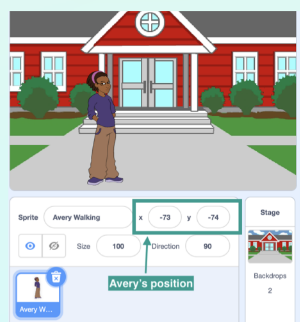
बग: इस प्रोग्राम में **दो** बग हैं। हरी झंडी को दूसरी बार क्लिक करने के बाद, एवरी अब स्क्रीन के मध्य में उसी आरंभ बिंदु से शुरू नहीं करती है। इसकी बजाय वह स्क्रीन की दायीं ओर उस स्थान से जहाँ उसने छोड़ा था, और चलने की वेशभूषा में शुरू करती है। पहला बग उसकी स्थिति है और दूसरा बग उसकी वेशभूषा (या लुक) है।

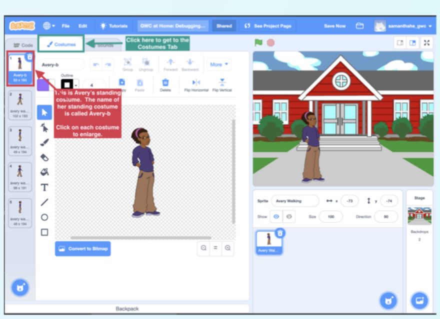


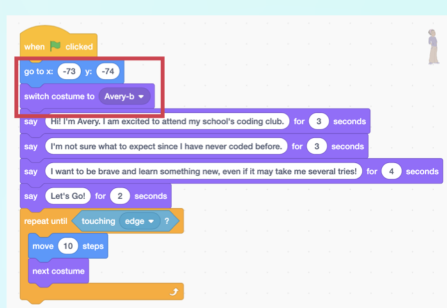
इसने प्रोग्राम को कैसे प्रभावित किया और क्यों: इस प्रोग्राम में एवरी को मध्य में शुरू करना चाहिए और ख़ुद का परिचय देना चाहिए, फिर वह दायीं ओर चलती है जब तक कि वह कोडिंग क्लब जाने के लिए सिरे तक नहीं पहुँच जाती है। क्योंकि हमारे पास वह कोड नहीं है जो हरी झंडी पर क्लिक करने पर हर बार एवरी को हमेशा एक ही स्थिति से शुरू करता है, एवरी अपने कोड के अनुक्रम को वहाँ से शुरू करती है जहाँ पर उसने पिछली बार छोड़ा था। हमारे मामले में, दायीं ओर स्क्रीन के सिरे पर।

आपने बग(गों) की मरम्मत कैसे की: एवरी हमेशा एक ही स्थान से शुरू करे इसके लिए हमें एवरी के शुरू करने की स्थिति चुनने का सबसे [go to x: -73 y: -74] आसान तरीक़ा है अपने माउस का उपयोग करके एवरी को वहाँ ले जाना जहाँ से आप चाहते हैं कि वह शुरू करे। आपको दिखना चाहिए कि x: और y: के बाद की संख्याएँ एवरी की स्थिति के अनुसार बदलती हैं। आपको Scratch स्टेज के नीचे स्प्राइट के वर्णन में भी यह दिखना चाहिए। हम ब्लॉक पर जाएँ का उपयोग अपने इवेंट ब्लॉक के बाद के सबसे पहले ब्लॉक के रूप में करते हैं [when clicked] ।

अब, हम चाहते हैं कि एवरी उस वेशभूषा में शुरू करे जो उसे खड़ी अवस्था में दिखाता है। इसके लिए हमें उसके शुरू करने की स्थिति को सेट करने के ठीक बाद [switch costume to Avery-b] का उपयोग करने की जरूरत पड़ेगी ताकि सुनिश्चित हो सके कि वह हमेशा खड़े रहने की वेशभूषा में शुरू करे। यदि आप निश्चित नहीं हैं कि उसके खड़े होने की स्थिति की वेशभूषा कौन सी है, तो **वेशभूषाएँ** टैग में देखें और उस वेशभूषा का नाम रिकार्ड करें जिसमें आप चाहते हैं कि वह शुरू करे। हम इन दो नए ब्लॉकों को पहले कोड करते हैं क्योंकि हम अपने स्प्राइट को उसी स्थान में सेट करना चाहते हैं दायें जब हरी झंडी को एवरी के बोलना और चलना शुरू करने से पहले क्लिक किया जाता है!



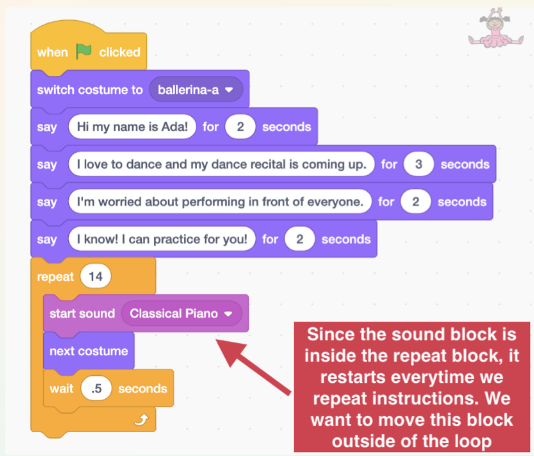


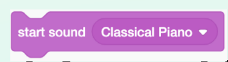


डीबगिंग चुनौती 3 का समाधान

बग वाला स्टार्टर कोड: https://scratch.mit.edu/projects/387851932/
नमूना समाधान कोड: https://scratch.mit.edu/projects/388346112/

बग: जब आप हरी झंडी दबाते हैं, एडा को आपको अपने बारे में बताना चाहिए और फिर संगीत की धुन पर नाचना चाहिए। जब एडा नाचना शुरू करती है तब संगीत के साथ कुछ गड़बड़ होती है! वह एडा जब भी हिलती है तब बार-बार शुरू से बजने लगता है। बग का संबंध अनुक्रम, या क्रम से है जिसमें ब्लॉक कोड किए गए हैं।



इसने प्रोग्राम को कैसे प्रभावित किया और क्यों: ध्यान दें कि हमारे ध्वनि ब्लॉक को 

दोहराने वाले लूप के भीतर कोड किया गया है। इसका मतलब है कि हर बार जब हम कोड के माध्यम से भीतर लूप करते हैं, क्लासिकल पियानो दोबारा शुरू हो जाता है!

आपने बग(गों) की मरम्मत कैसे की: हमें अपने ध्वनि ब्लॉक को लूप के बाहर खींचने की ज़रूरत है, लेकिन कहाँ? हमें वापस सोचने की ज़रूरत है कि हम क्या होने देना चाहते हैं। ध्वनि ब्लॉक को दोहराने वाले ब्लॉक के पहले और बाद ले जाकर प्रयोग करने की कोशिश करें। यदि हम ध्वनि ब्लॉक को दोहराने वाले लूप के बाद ले जाते हैं तो एडा के नाचने के बाद संगीत बजता है, हम ऐसा नहीं करना चाहते हैं। यदि हम ध्वनि ब्लॉक को दोहराने वाले लूप के पहले ले जाते हैं, तो पहले संगीत बजता है और फिर एडा नाचती है। हम यही करना चाहते हैं! आप फ़ैसला कर सकती हैं कि क्या आप चाहती हैं कि ध्वनि उसके नाचने पर या फिर आरंभ में जब वह अपना परिचय देती है तब शुरू हो।