

# Lesson 1.4: Testing and Debugging

## Objectives

In this lesson, students will:

- ❖ Learn what test cases are and practice writing test cases
- ❖ Practice testing a program using test cases
- ❖ Practice debugging somebody else's program

## Agenda

- |  |         |
|--|---------|
| 1. Debugging   | 5 mins  |
| 2. Let's Find Some Bugs !                                | 15 mins |
| 3. Student Activity: What's Wrong With This Talent Show? | 10 mins |
| 4. Student Activity: Let's Fix This Talent Show?         | 15 mins |
| 5. Wrap Up and Reflections                               | 5 mins  |

## Preparation

- Projector and speakers for demonstration
- Become familiar with the demonstration, the remix debug 1 and 2 projects.
- Student activity worksheet printouts one per student pair

## Resources & Links

- Solution to Peter's joke project: <https://scratch.mit.edu/projects/296241615>
- Peter's project with bugs: <https://scratch.mit.edu/projects/296246955>
- Talent Show remix for activity: <https://scratch.mit.edu/projects/296498608>

## 1. Debugging



It is very common for programs and code not to work the first time you try it. When there is a problem in the code, programmers refer to it as a bug. Can someone tell me why they are called bugs?

Program errors are called bugs because many years ago a very large computer called the Mark II malfunctioned because they found a moth stuck inside the computer. Since then programmers use the term debugging to remove bugs from programs.

Testing a program means running the program to check if there are any bugs in it. Programmers spend a great deal of time testing and debugging code. That is why it is important to develop good debugging strategies.


Today we are going to do some testing and debugging.

## 2. Let's Find Some Bugs !



**Tell students:**

Peter wrote a project in which Avery tells Devin a joke. Devin is listening and then answers the question. When the joke is finished he starts laughing.

**Open** the following Scratch project and run it in full screen so students cannot see the scripts: (click on )

<https://scratch.mit.edu/projects/296246955>

Hmmm.... The project is not quite working as described. Peter must have made a couple of mistakes. What is not working? Prompt students to call out what is actually happening that is not as described.

Answer: 1) Peter talks at the same time as Avery instead of waiting for the question  
2) He does not laugh at the end.

### Student Activity:



In this activity students will fix Peter's project. You may choose to do this as a class activity or assign it as a student activity. Instructions:

- Remix Peter's project: [296246955](https://scratch.mit.edu/projects/296246955)
- Test his project and debug it so it works as described.
- There are 2 bugs in the project that need to be fixed.

**Solution:** Review the 2 bugs with students

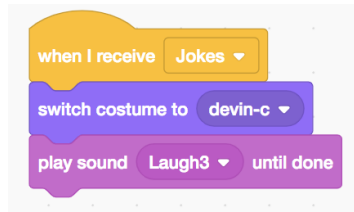
Bug 1:



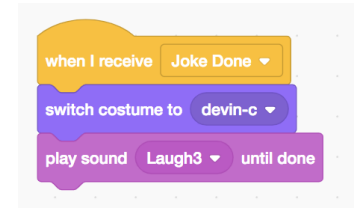
Devin does not wait long enough to talk, so he talks over Avery. Avery first talks for 2 seconds, so Devin should wait 2 seconds, not 1. Show the code for both and change Devin’s wait time to 2 seconds. **Test** the code by running it again.

Bug 2:

After Avery finishes the joke, Devin does not laugh. Devin’s is waiting to receive a message called “Joke’s”. But Avery broadcasts a message called “Joke Done”.



Change Devin’s code to wait for the “Joke Done” message.



Change the code and rerun the project to make sure It works.

### 3. Student Activity: What’s Wrong With This Talent Show?

**Student Activity:**



Instructions to give to students:

In this next activity, you have been given the task to test a new program. Your job is to read how the program is supposed to work. Then you will test the program and write down in your journal all the bugs you find.

**Distribute** the activity worksheet. Students may work in pairs or on their own. After the activity, **prompt** students for the bugs they found.

#### 4. Student Activity: Let's Fix that Talent Show



In this activity, students will fix the bugs they found in the previous activity. It is recommended that the same teams work together.

**Instruct** students what to do. Follow-up the activity by reviewing the fixes to the bugs in the program. There were a total of 3 bugs.

#### Solution:



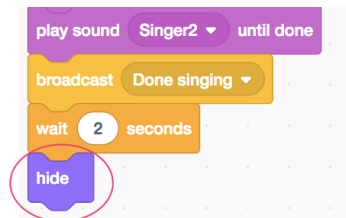
Review the 3 bugs with students.

#### Bug 1:



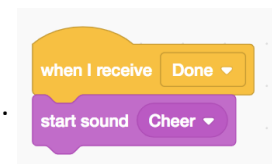
After Bella sings, she does not leave the stage. A hide command is missing

**Fix:** Add a **hide** command at the end of the script.

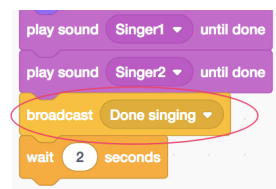


#### Bug 2:

After Bella sings, there is no cheering ! The problem is that in the **Welcome** sprite, the cheering code is listening to the **“Done”** message.



But when Bella finishes singing, she broadcasts the **“Done singing”** message:



**Fix:** After singing, the Bella sprite should broadcast the **“Done”** message.

#### Bug 3:

When the Contestant 3 button is clicked, contestant 1, Diego, shows up again to dance.

The problem is that when the button is clicked, it broadcasts the message **“Contestant1”**. This is the wrong contestant.

Fix: Change the broadcast message to **“Contestant3”** for the button sprite with the label “Contestant 3”.

## 5. Wrap Up and Reflections



### Reflection Points:

- What is one thing you learned about testing?
- What is a programming bug?
- What does it mean to debug a program?
- Why is it important to have a good debugging strategy?

## Activity Worksheet: What's Wrong with this Talent Show?

### Description of the Talent Show Program:

1. There are 3 contestants. When the green flag is clicked, the program introduces the show and each contestant by saying their name. You cannot see any of the contestants yet.



2. For each contestant there is a button. When the button is clicked, the contestant shows up on the stage and performs the activity they announce they will perform.



3. After the contestant finishes, you hear clapping and cheering. The contestant then leaves the stage.

### What to do:

- Remix the following project. This is the talent show program.

[296498608](https://www.ck12.org/idea/296498608/)

- Test the program and write down each bug you find in your design journal.