Lesson #2.9: Even More Debugging Challenges

Overview:

In this lesson, students will exercise the practice of testing and debugging. They will discover problems in a program, then develop strategies for fixing the problem. This activity comes from the Creative Computing Curriculum Guide and includes additional teacher tips and suggestions for implementation.

Student Agency:

Students who enjoy solving puzzles will like this activity. Students can solve the challenges in any order they wish and work in pairs to solve the problem. There are multiple solutions to these challenges. Reinforce that there is no single right answer, but there are ways to make each program more efficient.

Pathway: Coding/Computational Thinking

Duration: 40 minutes

Essential Question: What steps do you take to test and debug?

Objectives:

- Students will be able to investigate and find solution to “buggy” programs.
- Students will be able to explore a range of concepts (including events and parallelism) through the practice of testing and debugging.
- Students will be able to explain the steps they took to test and debug through code comments.
- Students will be able to develop a list of debugging strategies.

Competencies & Practices	Student Artifacts
Collaboration	Developing a class set of debugging

Lesson #2.9: Even More Debugging Challenges

	strategies
Testing and Debugging	Debugged programs

Teacher Preparation:

- Check out the Scratch Wiki website's article on testing and debugging: https://en.scratch-wiki.info/wiki/Debugging_Scripts and make note of items to share with students.
- Review the "buggy" programs in the Debug-It 3 Studio and make note of possible solutions.

Materials for Students:

- Debug-It 3 Studio: https://scratch.mit.edu/studios/475554/
- Debug-It Studio - created by each student.

Students Prior Knowledge:

Students should have experience with creating simple programs. The "bugs" in the programs in this activity are similar to those they may have encountered when completing the storytelling projects in previous lessons. Prior knowledge of parallelism, sequences, loops, and events will be explored through this activity.

Concepts:

Testing and Debugging
Event
Sequence
Parallelism
Loop
Comments

Lesson #2.9: Even More Debugging Challenges

Habits of Mind:

Thinking flexibly
Striving for accuracy
Thinking about your thinking

Lesson Sequence:

- **Anticipatory Set (5 minutes):**
  Lead a brief discussion about previous experiences with testing and debugging. Then, remind students that there are ways to prevent errors when programming. Share with them these tips:
  - The best way to prevent silly mistakes is to have neat, organized code. Order scripts by their functions or order of evaluation will help find mistakes and prevent them.
  - Leave comments wherever a lot of time was spent on a script; this way, if it breaks again, they can be referred back to for a glimpse of how the script worked in the past. Anyone with enough experience should be able to understand the script at first glance.
  - Do not blindly copy scripts from other projects, unless that was the intention of the project. Other than moral copytheft issues, chances are the script will not work in the context of the program. To learn how to make easy, integratable tools, see Advanced Clone Usage.

  These tips came from the Scratch Wiki website: https://en.scratch-wiki.info/wiki/Debugging_Scripts.

  Tell students that today they will be fixing programs that have bugs in them. Remind them that a software bug is an error, flaw, failure or fault in a computer program or system that causes it to produce an incorrect or unexpected result, or to behave in unintended ways. The “bugs” they will encounter may be similar to those they ran into when they were learning about conversations, making a block, and scenes.

Lesson #2.9: Even More Debugging Challenges

- **Engaging Activities:**
  - **Introduce the Debug It 3 Challenge (5 minutes):** Have students open Scratch and Sign In. Then have them go to the Debug It 3 Studio: https://scratch.mit.edu/studios/475554/. They will notice that there are 5 programs in the studio. Explain that each of the programs have bugs in them. Students should remember completing previous debugging challenges from the last unit.

    Teacher Tip: Have students right click on the program and select "Open link in new tab." This will save them from having to press the back button on their browser over and over to get back to the studio. Additionally, once students have the project open, encourage them not to look at the comments for the solutions. It is more rewarding when they figure it out on their own.
  - Work through the first debugging challenge together. Tell students their job is to read the description of what the program should do under the instructions. Then, they should click on **Remix** so they have their own copy of the program. This will also take them inside the program. Instruct them that the first step in Testing and Debugging is Testing. They should run the program once or twice to see first hand the problem in the program. Then, have them look at the code and try to modify it to fix the problem. Finally, test the program again to be sure their modification fixed the problem. If it did not, try something else.
  - Once they have fixed the problem, review the concept of using comments in a code to clarify what the code does or to explain steps taken to solve the problem. Explain that comments do not affect how the program runs. Demonstrate how to make comments in their code (right click on the code, then select Add Comment). Instruct students to write in the comment field what they did to solve the problem, or explain how their modification fixed the bug in the program.
  - **Test and Debug (20 minutes)**: Allow time for the students to complete each of the debugging challenges. Remind them to comment in their code. Encourage them to share their solutions with classmates around them. How did you identify the problem?

Lesson #2.9: Even More Debugging Challenges

- How did you fix the problem?
- Did others have alternative approaches to fixing the problem?

Teacher Tip: When students complete each of the challenges, have them click **Share** and add them to their own Debugging Studio that they created for the first set of debugging challenges. As they progress through each of the Scratch Units, they can place debugging challenges into this studio for future reference.

Wrap Up (10 minutes): Ask students to share solutions to each of the problems.

Questions to Ask Students:

- How did you identify the problem?
- How did you fix the problem?
- Did others have alternative approaches to fixing the problem?

Explain to students that they just practiced testing and debugging. There may be more than one solution to a problem and there may be more than one way to approach the problem.

- As a class, let's list some strategies or approaches to testing and debugging on the board.

Assessment Questions	**Yes**	**No**
Were students able to solve all 5 debugging challenges?		
Were the students able to employ different testing and debugging strategies?		
Did students use the comment feature to explain the code and/or their solution?		

Activities for Relearning:

If students were not able to solve the challenges, how might you clarify the concepts expressed in the unsolved programs? Present students with specific challenges to clarify the missed concepts. For instance, if students did not solve the third challenge, ask them to create a program using the pen tool to draw a picture with multiple parts (like a house with windows).

Lesson #2.9: Even More Debugging Challenges

Activities for Enrichment:

Invite students to create their own debugging challenges for their classmates. Have them think back to a problem they encountered in a program they created in the past. Recreate that problem in a debugging challenge and have their classmates solve the challenge. Optionally, create a studio of student created debugging challenges to use in later classes.

Resources for Teachers:

- Creative Computing Guide
- Debug-It 3 Studio: https://scratch.mit.edu/studios/475554/
- Scratch Wiki: https://en.scratch-wiki.info/wiki/Debugging_Scripts