Lesson 4.1: Testing with a Test Plan

Objectives

In this lesson, students will:
- ❖ Be introduced to Quality Assurance and the basic phases of testing.
- ❖ Learn what a test plan is and how to write test cases
- ❖ Practice testing a program using test cases
- ❖ Practice debugging somebody else's program

Agenda

1. Testing Programs: Quality Assurance	10 mins
2. Activity 1 - My First Test Plan: In Search of Bugs!	10 mins
3. Activity 2:My Test Cases in Action: Bugs Can't Hide from Me	20 mins
4. Solution Review	10 mins
5. Wrap Up and Reflections	

Preparation

- ❑ Projector to review solutions
- ❑ Review the solutions to the 4 bugs in the game
- ❑ Print student activity worksheet one per student pair

Resources & Links

- ❑ Remix project: https://scratch.mit.edu/projects/256373906

1. Testing Programs: Quality Assurance

Engage students in an interactive discussion and instruction:

How would you like to have a job where you get rewarded for breaking things?

Well, there is such a job. It's called Quality Assurance or QA. In QA you make sure the quality of something that is built is good enough to go out to the public or to be sold. To do this, people have to test the product by using and trying it in all possible ways.

Every programmer, no matter how good they are, runs into errors in their code that need to be fixed. These errors are often called bugs. You have already tested lots of code you have written to find bugs. Everytime you write some code, you try it out to see if it works. That is called testing.

"QA Illustration" by Redwerk.

Many computer scientists are QA engineers that test other people's programs. Because it is easy to miss testing something in your own code, it's important for others to test your program, especially if it is a very large program.

Today we are going to be QA engineers so that we can find bugs that might be in somebody else's program. Experienced QA engineers become very good at breaking things and they find many bugs.

QA engineers follow these basic steps to test a program:

Test Plan	Write a list of test cases
Test Execution	Run each test case
Bug Reporting	Report any bugs found

- First, they write a Test Plan. A test plan is basically a list of what to test and how to perform the test. Each item to test is called a test case.
- Next , they run each test case in the test plan to check if there are bugs in the program.
- Finally, a QA engineer reports all the bugs they found and what they did to find it. The programmer reads the report and tries to fix the bugs that were found.

Note: There are additional phases and steps a QA engineer performs, but for the purposes of this exercise and grade level, these steps cover the essential tasks.

2. Student Activity: My First Test Plan - In Search of Bugs!

Activity Description

1. It is recommended that students work in pairs for this activity. Each team should have a copy of the Student Activity worksheet: My First Test Plan: In Search of Bugs!
2. Students will write 3 test cases to test the Maze game described in the activity worksheet. One test case is given as an example.
3. **Explain** to students that test cases should be specific and limited to small tasks. Each test case should say what will be tested, and how it will be tested. For example, “Make sure the ball moves” is not specific. It is better to write:

   Test What: Make sure arrow keys move the ball

   Test How: Click on each arrow key to make sure they move the ball in the direction of the arrow.

Additionally, explain that the specification of the game (which is similar to a design description) is needed to know what to test.

4. **Tell** students they have been given the task to test a brand new game developed by a company called ***Amazing Path***. They have been challenged to find at least 4 bugs in the game. Tell them to get started and make sure each team member comes up with at least one test case.

Solution:

This is a list of items that could be tested:

- Use each arrow key to make sure the ball moves in the direction of each arrow
- Move the ball into a maze wall to test if it bounces off of it
- Click on many other keys on the keyboard to make sure they don't do anything in the game
- Move the ball through the maze until reaching the red square target to test that the 'Game Over' banner shows up.
- When the game starts (click on green flag) make sure that everything in the maze is in the correct location. Run the game multiple times.

3. Student Activity: My Test Cases in Action - Bugs Can't Hide from Me

In this activity students will run the test cases they wrote. It is recommended that the same teams work together during this activity. Additionally, students are instructed to fix bugs in the code as they encounter problems. This is typically how a programmer would test their own code. When QA engineers test programs, they simply report the bugs and how they found them.

Note: Sometimes a test case cannot be run until a bug has been fixed. This is very common when a software product is being tested. When a bug has to be fixed before another test case can be run, it is said to be a ***blocking defect*** (bug).

Distribute the Activity 2 worksheet or optionally have students use their journal.

Instructions to give to students:

- ❏ Re-mix the project listed in the activity worksheet.
- ❏ Run each test case in your test plan. If you encounter a problem, write down the problem you found for each test case, or write 'worked' if you did not find a problem.
- ❏ Once you find a bug, try to fix the problem in the code.
- ❏ There are 4 bugs in the code, let's see if you can find them all and fix them !!

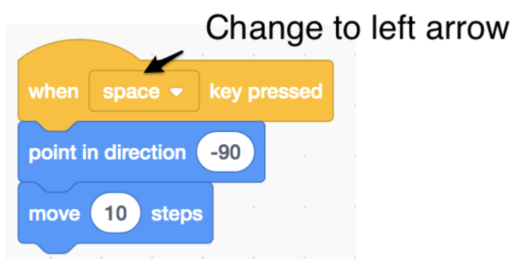
4. Solution Review

Follow up the activity by going through each bug and fix using the solution as a guide.

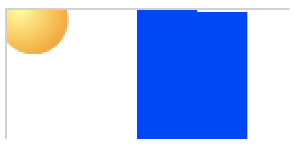
Solution:

The 4 bugs in the code are:

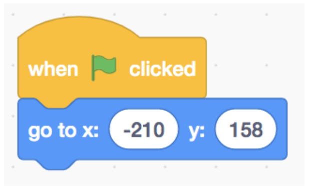
1. To move the ball to the left, the Space key is used in the code instead of the "left arrow"



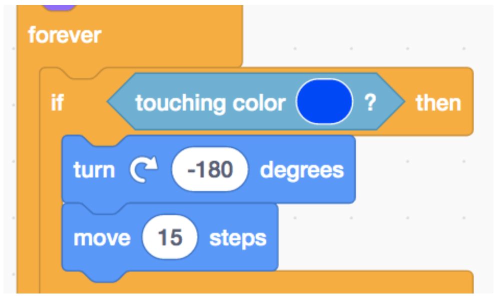
2. When the game starts, the yellow ball is slightly off the screen. The goto statement needs to be adjusted so that the ball is further down and to the right.



Change the Ball sprite initialization code to the ***go to*** values on the right:

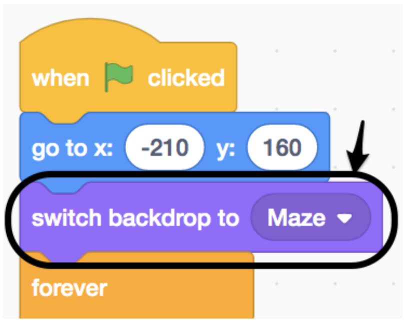


3. The code to keep the ball from going through the wall is incorrect. It should be:



```
forever
    if touching color ( ) ? then
        turn ↻ -180 degrees
        move 15 steps
```

4. If you play the game a second time, there is no code to set the initial backdrop to the Maze. It still shows the "Game Over" backdrop from the previous game. The initialization code is missing for the backdrop. The following statement is missing for the Ball sprite where we initialize the position of the ball.



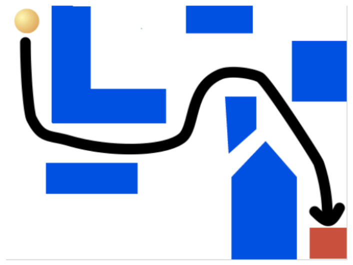
```
when 🏳 clicked
go to x: -210 y: 160
switch backdrop to Maze ▾
forever
```

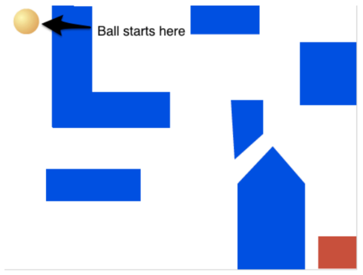
5. Wrap Up and Reflections

Reflection Points:
● What is a test case? ● What bugs did you find in the code ? ● How did you fix the bug? ● What are some benefits of knowing what to test before you start testing?

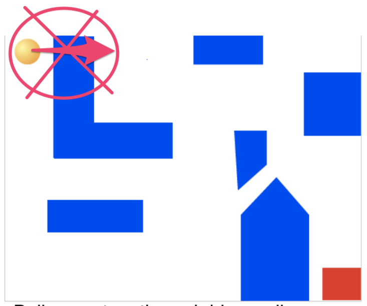
Activity Worksheet: My First Test Plan - In search of Bugs!

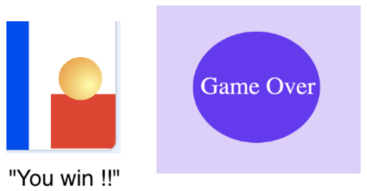
Game specification (description):


The object of the game is to move the ball to the red square



When game starts, it looks like this


Ball cannot go through blue walls



"You win !!"

When the ball reaches the red square, sprite says "You win" and the Game Over backdrop is shown

The arrow keys move the ball in the direction of the arrow. No other key moves the ball.

Test Cases: (use journal or back of handout for more test cases)

1) **Test What:** Make sure arrow keys move the ball
   Test How: Click on each arrow key to make sure they move the ball in the direction of the arrow.

2) **Test What**: ____
   Test How: ____

3) **Test What:** ____
   Test How: ____

4) **Test What**: ____
   Test How: ____

Activity: Bugs Can't Hide from Me

What to do:

- ❑ Remix the following project: 256373906
- ❑ Run each of your test cases. If a test case finds a bug, write it down below. Then proceed to fix the bug.
- ❑ You may not be able to run a test case because a bug has not been fixed yet. This is very common in QA

Problems found:

Test Case 1:

____

____

Test Case 2:

____

____

Test Case 3:

____

____

Test Case 4:

____

____