

girls who
code

Girls Who Code At Home

Debug the Maze
Unplugged Activity

Activity Overview

A **computer scientist** works to write **code**, or instructions, for a computer to do a task. It may be surprising to know that most **programmers**, or people who write code, actually spend most of their time **debugging**, or finding and fixing problems, in their code! Debugging requires persistence and a willingness to keep looking for new ways to solve problems. In this activity you will get a chance to debug code that is supposed to move a character through a maze. You will also get a chance to write your own code and challenge a friend to debug it! Before you start learning more about debugging, check out the featured Woman in Tech Spotlight, Brittany Wenger. At the age of 15, Brittany created a tool that helps diagnose patients with breast cancer so they can receive early treatment.

Materials

- (Optional) Small Game Board Piece or Object
- Maze Debugging Worksheet Solutions

Women in Tech Spotlight: Brittany Wenger



When you have a question, how do you go about trying to find an answer? What do you do if your first source doesn't have an answer? Brittany Wenger, turned to code in her search for answers to important questions about breast cancer.

When Brittany was 15 years old, her cousin was diagnosed with breast cancer. Approximately 1 in 8 women in the United States will be affected with breast cancer in her lifetime. Early detection of breast cancer in a localized state has a 100% survival rate. Brittany decided to take action and developed an algorithm that gathers data from a patient and estimates if a breast mass is malignant or benign. Her algorithm [Cloud4Cancer](#) was able to reach 99.11% accuracy in classifying breast mass correctly.

Watch the [video](#) about Brittany Wenger to see how code can be used to find answers to big questions that can impact lives.

Reflect

Being a computer scientist is more than just being great at coding. Take some time to reflect on how Brittany and her work relates to the strengths that great computer scientists focus on building: bravery, resilience, creativity, and purpose.



BRAVERY

Brittany was able to use technology to help women around the world diagnose breast cancer early for immediate treatment.

Think about a time when you went above and beyond on a project or activity. What motivated you to work so hard?

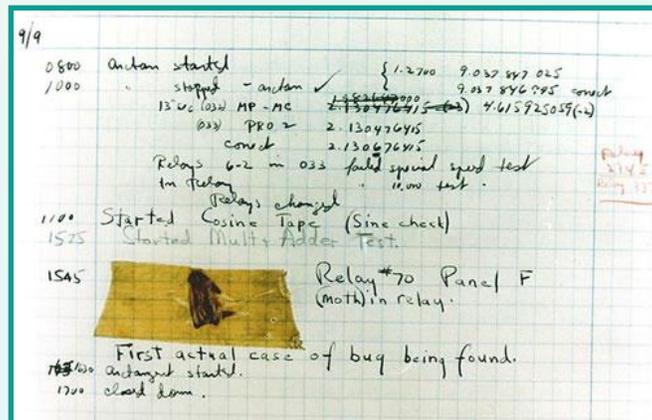
Share your responses with a family member or friend. Encourage others to read more about Brittany to join in the discussion!

Step 1: Understanding Debugging (2 mins)

Think of a time you tried to solve a problem, this could be from a difficult assignment at school, or trying to locate a missing item. Were you able to solve it right away? Many times when we try to solve a problem we might face a couple failures before we succeed. Programmers spend most of their days trying to find and solve problems in their code.

An error in a computer program or hardware is called a **bug**. The process of identifying and removing errors or bugs from computer hardware or software is called **debugging**. The origins of these terms stem back to Grace Hopper, one of the early pioneers of computer science. While working on one of the first computers, Grace Hopper's team found a moth inside the computer which caused an error, an actual physical bug! Grace's journal entry of the taped moth is now referred to as the first recorded computer bug in history. It is on display at the Smithsonian Museum of American History in Washington D.C.

In this activity you will get a chance to debug code that is supposed to move a character through a maze from a starting location to an ending location, while also trying to pick up a star. You will also get a chance to write your own code and challenge a friend to debug it!



The first recorded computer bug

Step 2: Rules of the Game (2 mins)

The goal of the game is to move a character from the starting point, pick up all the stars, and get to the end point by finding and fixing all the **bugs** (or problems) in the instructions.

Step 2: Rules of the Game (continued)

Rules

1. Your character must start at the starting location .
2. Your character must end at the ending location . Once your character gets to the final location, the game ends and **no** instructions are followed afterwards.
3. You must pick up the star  **before** getting to the ending location.
4. The character must be in the same space as the star before picking it up.
5. You can only **edit** or **remove** instructions. Editing a step means replacing any **values** in the instructions with a different value or fixing a **spelling** error.
6. **ONLY** these instructions are allowed. Spelling and capitalization must be exactly the same as what is listed below.
 - a. Move Left
 - b. Move Right
 - c. Move Up
 - d. Move Down
 - e. Repeat [1 to 5] times:
 - f. Stop Repeat
 - g. Pick Up Star
7. You may **not** add instructions of your own

Step 3: Check for Valid Instructions (2 mins)

Just like learning a different speaking language (like Spanish, French, German, etc.) coding has its own set of rules. Of the following options, circle which lines of code are valid. **Remember that all the valid instructions are listed above (rule #6).** Recall that spelling and capitalization counts!

- | | | | |
|---------------------|---------------------|---------------------|--------------------|
| A. Turn Right | B. Move Forward | C. Pick Up Starzzzz | D. Repeat 3 times: |
| E. Repeat 10 times: | F. Move to the Left | G. Move Up | H. Stop Repeat |
| I. Pick Up Stars | J. Drop Star | K. Move to Goal | L. Move Down |

Be sure to check your answers with the solutions handout at the end of the document.

Step 4: Time to Start Debugging! (5 mins)

Below you will see the game board with the start flag at (A, 3), star at (B, 2), and finish line at (A, 1). When we run the code, or instructions, we expect the character to move from the start flag, pick up the star, and then end at the finish location at (A, 1). However we noticed the character's **actual** actions include not picking up the star and ending at (C, 1).

	A	B	C
1			
2			
3			

Expected Actions	Actual Actions
<ul style="list-style-type: none"> • Pick Up Star • Ends on Finish Location (A, 1) 	<ul style="list-style-type: none"> • Star is not picked up • Ends on Location (C, 1)

Using the rules that were listed before, help fix (or debug) the incorrect code and write the correct code under the label debugged code/instructions.

Tip: Use a small game board piece or object around the house to act as the character. You can move it around the maze as you execute each line of code to help you keep track of the character's location

Incorrect Code/Instructions	Debugged Code/Instructions
1. Move Up	
2. Move Right	
3. Pick Up Starz	
4. Move Up	
5. Move Right	

HINT: There are 2 bugs

You can check your answers with the solutions at the end of the document.

Step 5: Tackle the Next Challenge (5-10 mins)

Oh no! Another bug! Help debug the instructions below. Don't forget to check the rules.

	A	B	C
1			
2			
3			

Expected Actions	Actual Actions
<ul style="list-style-type: none"> Pick Up Star Ends on Finish Location (B, 2) 	<ul style="list-style-type: none"> Star is not picked up Ends on Location (B, 2)

Incorrect Code/Instructions	Debugged Code/Instructions
1. Move Up	
2. Move Right	
3. Move Up	
4. Move Right	
5. Pick Up Star	
6. Move Down	
7. Move Left	

HINT: There are 2 bugs

BONUS: The instructions propose one solution to pick up the star and move to the goal, but there are **multiple** paths to do this. Can you think of another set of code, or instructions, that has the same outcome?

You can check your answers with the solutions at the end of the document.

Step 6: Debugging Loops (5-10 mins)

Yet another bug! But wait, this map looks similar to the one in the last challenge. The correct code should follow the same path as the code in challenge #2 but uses the *repeat* command. In computer science we call these loops! **Loops** are used to *repeat* a set of instructions multiple times. We typically *indent* the instructions that we want to repeat to make sure the computer knows how many lines of code to repeat.

	A	B	C
1			
2			
3			

Expected Actions	Actual Actions
<ul style="list-style-type: none"> ● Pick Up Star ● Ends on Finish Location (B, 2) 	<ul style="list-style-type: none"> ● Star is not picked up ● Ends on Location (B, 2)

Incorrect Code/Instructions	Debugged Code/Instructions
1. Repeat <u>1</u> times:	
2. Move Up	
3. Stop Repeat	
4. Repeat <u>3</u> times:	
5. Move Right	
6. Stop Repeat	
7. Pick Up Star	
8. Go Left	
9. Move Down	

HINT: There are 3 bugs

You can check your answers with the solutions at the end of the document.

Step 7: Make Your Own! (10-15 mins)

This time it is your turn! Try to challenge a friend by writing your own set of code that is filled with **at the most 4 bugs**. Be sure to fill out the expected actions vs. actual actions table for your friend. Use this sheet as your **answer key** and once you are happy with your challenge, use the next sheet to **copy** the incorrect code, maze, and action table to challenge your friend!

Helpful Tip: Try writing the correct instructions first and picking 4 lines or less to change or add additional bugs to your correct code.

	A	B	C
1			
2			
3			

Expected Actions	Actual Actions
<ul style="list-style-type: none"> ● Pick Up Star ● Ends on Finish Location (,) 	<ul style="list-style-type: none"> ● Star is ● Ends on Location (,)

Incorrect Code/Instructions	Debugged Code/Instructions
1.	
2.	
3.	
4.	
5.	
6.	
7.	
8.	
9.	
10.	

HINT: There are bugs

CREATOR:

	A	B	C
1			
2			
3			

CHALLENGER:

Expected Actions	Actual Actions
<ul style="list-style-type: none">• Pick Up Star• Ends on Finish Location (,)	<ul style="list-style-type: none">• Star is• Ends on Location (,)

Incorrect Code/Instructions	Debugged Code/Instructions
1.	
2.	
3.	
4.	
5.	
6.	
7.	
8.	
9.	
10.	

HINT: There are __ bugs

Step 8: Share Your Creation! (5 mins)

1. Take a picture or scan your maze and challenge a friend to debug your instructions!
2. Don't forget to share your maze on social media. Tag @girlswhocode #codefromhome and we might even feature you on our account!

Debug the Maze Solutions

Step 3: Check for Valid Instructions

A. Turn Right Not a valid command	B. Move Forward Not a valid command	C. Pick Up Starzzzz Spelling Error	D. Repeat 3 times:
E. Repeat 10 times: Not a valid command. Repeat can be done 1-5 times.	F. Move to the Left Not a valid command	G. Move Up	H. Stop Repeat
I. Pick Up Stars Spelling Error	J. Drop Star Not a valid command	K. Move to Goal Not a valid command	L. Move Down

Step 4: Time to Start Debugging Solutions

	A	B	C
1			
2			
3			

Expected Actions	Actual Actions
<ul style="list-style-type: none"> ● Pick Up Star ● Ends on Finish Location (A, 1) 	<ul style="list-style-type: none"> ● Star is not picked up ● Ends on Location (C, 1)

Incorrect Code/Instructions	Debugged Code/Instructions
1. Move Up	1. Move Up
2. Move Right	2. Move Right
3. Pick Up Starz	3. Pick Up Star
4. Move Up	4. Move Up
5. Move Right	5. Move Left

Step 5: Tackle the Next Challenge Solutions

	A	B	C
1			
2			
3			

Expected Actions	Actual Actions
<ul style="list-style-type: none"> Pick Up Star Ends on Finish Location (B, 2) 	<ul style="list-style-type: none"> Star is not picked up Ends on Location (B, 2)

Incorrect Code/Instructions	Debugged Code/Instructions
1. Move Up	1. Move Up
2. Move Right	2. Move Up
3. Move Up	3. Move Right
4. Move Right	4. Move Right
5. Pick Up Star	5. Pick Up Star
6. Move Down	6. Move Down
7. Move Left	7. Move Left

BONUS: The instructions propose one solution to pick up the star and move to the goal, but there are **multiple** paths to do this. Can you think of another set of code, or instructions, that has the same outcome?

Another proposed set of instructions is Move Right, Move Right, Move Up, Move Up, Pick Up Star, Move Down, and Move Left. The last two steps can also be interchanged and still be a valid solution. Challenge #4 will also explore yet another type of solution using the repeat loop.

Step 6: Debugging Loops Solutions

	A	B	C
1			
2			
3			

Expected Actions	Actual Actions
<ul style="list-style-type: none"> ● Pick Up Star ● Ends on Finish Location (B, 2) 	<ul style="list-style-type: none"> ● Star is not picked up ● Ends on Location (B, 2)

Incorrect Code/Instructions	Debugged Code/Instructions
1. Repeat <u>1</u> times:	1. Repeat <u>2</u> times:
2. Move Up	2. Move Up
3. Stop Repeat	3. Stop Repeat
4. Repeat <u>3</u> times:	4. Repeat <u>2</u> times:
5. Move Right	5. Move Right
6. Stop Repeat	6. Stop Repeat
7. Pick Up Star	7. Pick Up Star
8. Go Left	8. Move Left
9. Move Down	9. Move Down

Note that this solution is actually longer than the previous, so WHY should we use loops? Loops come in handy when there are commands or steps we want to do a lot of times (more than twice) or have multiple steps that we want to be repeated. In this particular case we do not see less steps, but there are other cases where it does help our code to be shorter and more readable to anyone else who may be reading our code.