

## Lesson 10: Let's explore bugs and debugging

Do you want to know a secret about working with computers and writing code? Here it is:

**SOMETHING ALWAYS GOES WRONG!**

Okay, so that isn't actually a secret, but it is really important. A major part of computational thinking and working with computers is problem-solving.



### Don't forget

**Computational thinking** means thinking about a problem or task in a way that is similar to how a computer thinks. It is a way of planning, problem-solving and analysing information the same way a computer does.

When things don't work the way you want, remember that is okay! It just means it is time to problem-solve. One of the main types of problem-solving you do in computer science is finding and fixing **bugs**.

**Debugging** is a major part of coding and using robots. People who work as computer scientists, computer programmers or roboticists professionally spend a lot of time debugging. Probably even more time than they do writing their code!



### Jargon buster

When something isn't working in a computer program, it is called a **bug**.

Finding and fixing bugs in a computer program is called **debugging**.



### Why is that?

Calling something that's going wrong a 'bug' might seem a bit strange, but that's really what these errors are called.

Why are computer problems called **bugs**? A woman named Grace Murray Hopper, who is one of the inventors of modern computer programming, came up with this term. She once discovered that the issue causing her computer to malfunction was an actual moth that had gotten into the hardware! She fixed the problem by literally **debugging** her computer.

The name stuck and now problems with software or hardware in computers are called **bugs** and fixing them is called **debugging**!

## Debugging in EdScratch

The bug box in the EdScratch programming environment is a special feature to help you find and fix any bugs in your code.



### Don't forget

The **bug box** is the area below the block pallet and programming area in EdScratch. If there is a bug or if it seems like something isn't quite right in a program, a warning message will show up in the bug box.

The warning messages that appear in the bug box are there to give you information about any problems or potential problems in your code. These messages are EdScratch's way of saying that there is an error in your code, or, that you might find an error when you run your program in Edison. In coding, there are two main types of errors: **syntax errors** and **logical errors**.



### Jargon buster

**Syntax** is the rules of how a programming language works. All languages have rules. For languages people speak, like English or Chinese, there are rules about spelling and grammar and how to write the letters or characters in that language. Syntax is the same thing but for a computer language.

**Syntax errors** are caused by problems in how you wrote your code which break the rules of the language. These errors are sort of like typos or spelling mistakes in writing.

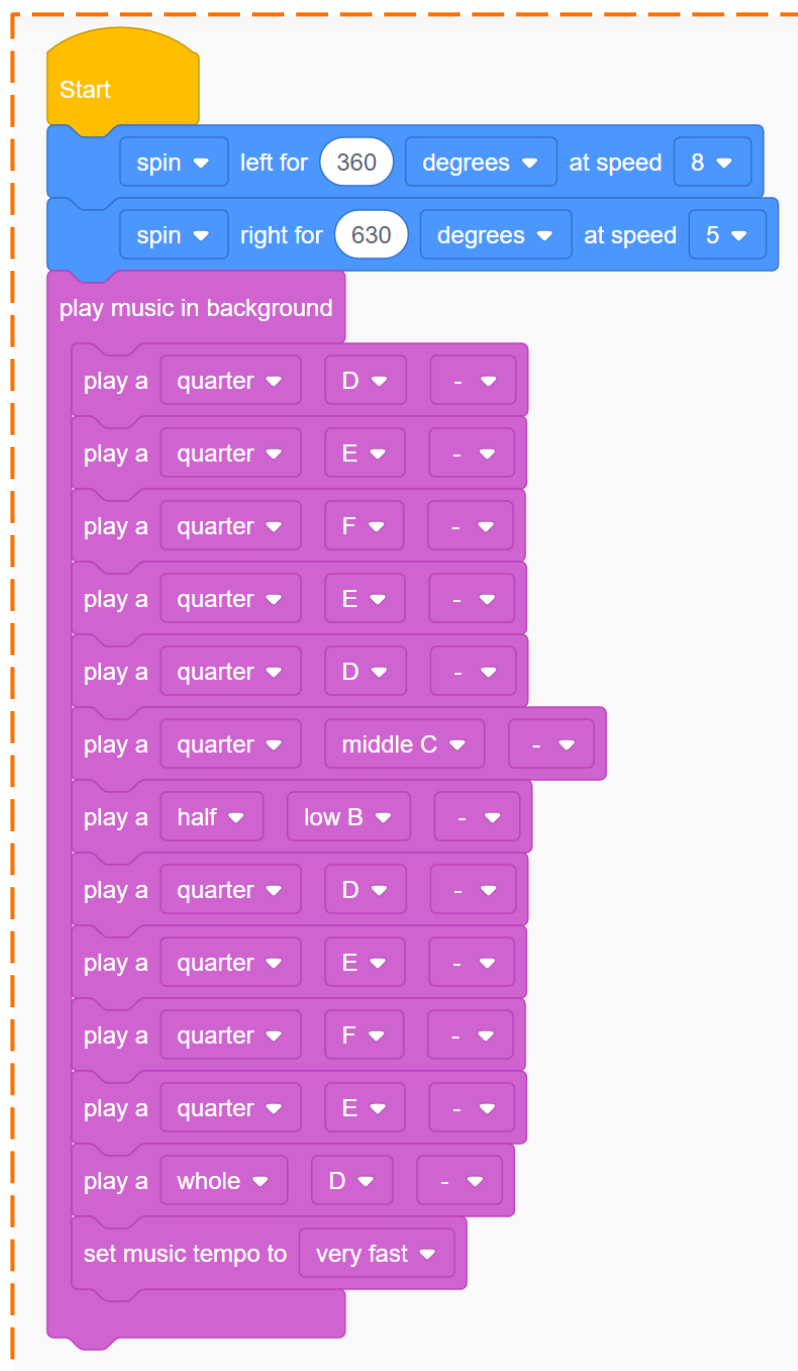
**Logic** is an organised way of thinking that makes sense to a computer. Logic determines the flow of a program, how you order things inside a program and what inputs you use to generate the outputs you want.

**Logical errors** are problems with the logic of a program. Logical errors are basically problems with the way of thinking in a program. Programs that do not make sense to the computer and programs that ask a computer to do something that it cannot do are examples of logical errors. If a program works differently than you expected it to work, there is a good chance that there's a logical error somewhere.

Understanding if a bug is the result of a syntax error or a logical error can help you fix the problem. If there is a syntax error in your EdScratch program, you will get a red warning message in the bug box, and you won't be able to download the program to Edison. If you can download your program and run it in Edison, but it doesn't do what you want, that probably means there is a logical error.

**Try it out!**

Look at the following program:



The image shows a Scratch script with the following blocks:

- Start block (yellow)
- spin left for 360 degrees at speed 8 (blue)
- spin right for 630 degrees at speed 5 (blue)
- play music in background (purple)
- play a quarter D (purple)
- play a quarter E (purple)
- play a quarter F (purple)
- play a quarter E (purple)
- play a quarter D (purple)
- play a quarter middle C (purple)
- play a half low B (purple)
- play a quarter D (purple)
- play a quarter E (purple)
- play a quarter F (purple)
- play a quarter E (purple)
- play a whole D (purple)
- set music tempo to very fast (purple)

This program is full of bugs. Your job is to fix it!

**Hint!**

One of the best ways to see what's going wrong – and right – with a program is to test it out using EdScratch and Edison!

1. The programmer who wrote this program made two errors with the **set music tempo** block. One error is a **syntax error**, and one error is a **logical error**. Write an explanation of each of these two errors to explain them to the programmer. *Hint:* the programmer wants the music to play at a very fast tempo.

**Syntax error:** \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

**Logical error:** \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

This is what the programmer who wrote this code said about what the program is meant to do:

"I want the Edison robot to play a tune very fast. While the music is playing, I want the robot to spin left for one full circle (360 degrees), then spin back right the same distance at the same speed."

2. The program does not work the way the programmer wants. You need to debug the program and get it working for the programmer. Test your program using Edison and keep debugging until the program works just the way the programmer explained. Describe the bugs you found and what you did to fix them.

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_