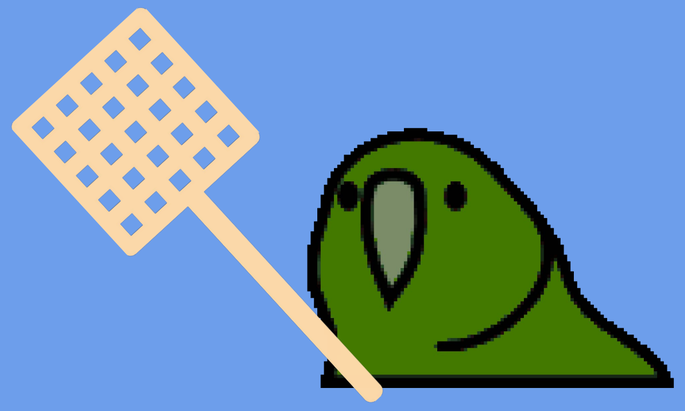
Debugging

Even if you're an experienced developer, you will almost always encounter problems (or *bugs*) with your software. Let's have a look at a process you can use to help deal with bugs in your source code. We'll also see a few different kinds of bugs that we will often encounter.

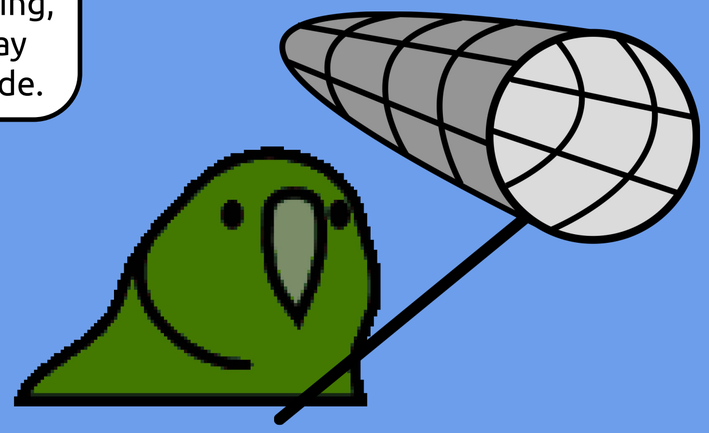


1. Identify Bugs

Almost all software has bugs! You can find bugs with thorough testing, especially of *edge cases* that may need special handling in your code.

4. Fix and Prevent

With the cause identified, determine how you will fix the bug. Also consider whether similar bugs might exist elsewhere in your code, and how you will prevent similar bugs in the future.



Version Control

You can also isolate bugs by comparing your current code with an old version that did not have the bug. *Version control* tools like **Git** help with this by tracking all changes made to your code; they also make it easier for multiple developers to work on the same code.

2. Isolate the Bug

After observing a bug, you must find the relevant section of code to fix. Try simplifying your code to find the smallest section that will reproduce the bug.

3. Identify the Cause

Once you have found the relevant section of code, you must identify what is causing the bug. Carefully read the code, and check the values of variables and object attributes when the code is running.

Debuggers

Special tools called *debuggers* can help you step through your code line-by-line while it runs, and let you check variable values while the code is running.

Types of Bugs

Syntax Errors

If your code does not follow the language's rules for writing code, a syntax error will occur when you try to run it.

Runtime Errors

When your code attempts to do something that is not possible, an *exception* will crash your program. For example, in many languages an exception will occur if you attempt to divide a number by zero.

Logic Errors

When your program runs without crashing, but does not behave as you expect it to, there is a problem with the logic of your code. For example, your loop executes one time too many or too few (a so-called *off-by-one* error).