**Steps of Debugging**

**Discussion**

Debugging is one of the biggest steps of developing your programs, as you’ve probably found out by now. It’s very rare to write a program that works perfectly the first time. Sometimes you have to fix syntax errors, sometimes logic errors. What kinds of errors have you encountered? What did you have to do to fix those bugs?

**Exercise**

Create a step by step plan or flowchart of what you should do when you encounter a bug. Remember all of the resources you have, such as the DOCS, Help tab, slides, videos, examples, and problem description. Make sure to consider the various kinds of bugs, such as syntax errors and logic errors. Finally, your plan or flowchart should include what to do if there is an error message or if the program runs but doesn’t behave correctly.

**Further Discussion**

This activity would work well having students working in pairs or small groups. You may even want to divide the pieces amongst the groups. For instance, some groups can create a plan for when there is a syntax error and other groups can create a plan for when there is a logic error.

A general plan of action would be:

1. Reread the problem description. Do I really understand what problem I am trying to solve?
2. If there is an error message, does the error message suggest which line to look at? What other information does the error message provide?
3. Make a hypothesis about which line could contain the error.
   1. If there is an error message about syntax, look at the line the error message suggests and see if you can spot it. If not, look at the surrounding lines to see if another error has propagated from previous lines.
   2. If there is an error message about undefined code, check your spelling. Check the variables/methods have been defined at this point in the code
   3. If it’s a logic error, you should be able to articulate what’s going wrong. Then think about why that particular thing would be going wrong.
4. Repeat until the bug is solved

To extend this activity, have students also come up with a list of debugging strategies, such as

1. Use the debugger
2. Check the DOCS
3. Look at the slides/other examples
4. Check the variable values around the bug
5. Check the types of parameters or other variables
6. Check the return type of any methods used
7. Comment out code and slowly uncomment until the bug occurs. Now you know where the bug is triggered!
8. Add print statements to check variable values or progress through the program
9. Simplify the program. Instead of asking for 15 values from the user, just ask for 2. Take out some loops. Pull out code that you think might be having problems and put it in a sandbox program to isolate it.
10. Ask a friend or peer. Verbalize the intended action and the bug that is being seen.