

## 9.6: DEBUGGING



Contributed by [Chuck Severance](#)  
Clinical Associate Professor (School of Information) at [University of Michigan](#)

As you work with bigger datasets it can become unwieldy to debug by printing and checking data by hand. Here are some suggestions for debugging large datasets:

### Scale down the input

If possible, reduce the size of the dataset. For example if the program reads a text file, start with just the first 10 lines, or with the smallest example you can find. You can either edit the files themselves, or (better) modify the program so it reads only the first `n` lines.

If there is an error, you can reduce `n` to the smallest value that manifests the error, and then increase it gradually as you find and correct errors.

### Check summaries and types

Instead of printing and checking the entire dataset, consider printing summaries of the data: for example, the number of items in a dictionary or the total of a list of numbers.

A common cause of runtime errors is a value that is not the right type. For debugging this kind of error, it is often enough to print the type of a value.

### Write self-checks

Sometimes you can write code to check for errors automatically. For example, if you are computing the average of a list of numbers, you could check that the result is not greater than the largest element in the list or less than the smallest. This is called a "sanity check" because it detects results that are "completely illogical".

Another kind of check compares the results of two different computations to see if they are consistent. This is called a "consistency check".

### Pretty print the output

Formatting debugging output can make it easier to spot an error.

Again, time you spend building scaffolding can reduce the time you spend debugging.