

Advanced Programming in the UNIX Environment

Week 05, Segment 8: Unix Development Tools: Debugging Your Code

**Department of Computer Science
Stevens Institute of Technology**

Jan Schaumann

`jschauma@stevens.edu`

<https://stevens.netmeister.org/631/>

```
ap here: 1
here: 2
here: 1
here: 1
here: 2
here: 1
here: 1
here: 3
here: 1
here: 2
here: 1
here: 1
here: 2
here: 1
here: 1
0
[apue$ fg
vim fib.c
[1] + Stopped vim fib.c
[apue$ cc fib.c
[apue$ ./a.out
usage: ./a.out num
[apue$ ./a.out 7
[2] Segmentation fault (core dumped) ./a.out 7
```

apue\$

Using a debugger

The process of “debugging” is not so much “finding bugs”. It’s observing what the program *does* versus what you thought it *should* be doing.

Popular methods:

- thinking really hard and reading the docs ← What “real programmers” claim they do.
 - guessing
 - printf debugging
 - using a debugger ← What “real programmers” *should* do.
- Diagram description: A large right-facing curly brace groups the items 'guessing' and 'printf debugging'. Three horizontal arrows point from the text on the right to the corresponding items in the list on the left. The top arrow points to 'thinking really hard and reading the docs'. The middle arrow points to the group of 'guessing' and 'printf debugging'. The bottom arrow points to 'using a debugger'.