CSE306 Software Quality in Practice

callgrind overview
POST
expectations

Piazza post @367
Valgrind
“val-grinned” - the gate to Valhalla

A suite of tools (see http://valgrind.org/info/tools.html)

Memcheck “detects memory-management problems”

Cacheegrind “is a cache profiler”

Callgrind “is an extension to Cacheegrind. It provides all the information that Cacheegrind does, plus extra information about callgraphs.”

Massif “is a heap profiler”

Helgrind “is a thread debugger which finds data races in multithreaded programs”

DRD “is a tool for detecting errors in multithreaded C and C++ programs”
callgrind

valgrind --tool=callgrind [opts] prog [opts]

[opts] callgrind options

[opts] program options
callgrind_annotate

Helps to make sense of callgrind data.

callgrind_annotate --inclusive=yes -- tree=both --auto=yes callgrind.out.<PID>

https://web.stanford.edu/class/archive/cs/cs107/cs107.1196/resources/callgrind
To redirect output to a file

You can redirect the output of a command to a file using `>`. For example, to redirect the output of `ls` to a file named `abc`:

```
ls > abc
```

Note: `>` creates the file if it does not already exist, and overwrites it if it does (without warning).
EXERCISE

Work through the gprof exercise from last Monday (April 8) but now using callgrind instead (or in addition to).

https://classroom.github.com/a/1D7Ht9-V

NOTE

During this in-class exercise several students and I did not get annotated source code included in the callgrind_annotate output, though it should have been present. I am as yet uncertain as to why. However, the code for LEX22 does reliably produce the annotated source code, so there is some hope that LEX22 will work as intended.