Gprof profiler

- One of the injection-style profilers, counts how often each function/method is called and how much time is spent in it
- Also tracks how often each function/method is called by each other function/method, and breakdown of time there
- Requires compilation with special flags to insert profiling code, on our system these flags are:

    -pg -no-pie -fno-builtin

  (not position independent, don’t replace my funcs with inlines)

Running the program

- First, run the recompiled program normally, e.g.

  ./myprogx whatever args it takes

- The injected code will have caused the creation of a file named gmon.out, that contains the profiling data
- Then we run the gprof profiler on the executable, and it analyzes the data from gmon.out

  gprof ./myprogx

Output from gprof

- Gprof produces a lot of output, but there are two tables of particular interest
- first table lists all the functions, and for each one shows:
  - what % of the total run time was spent in that function
  - total seconds spent in the function + functions it called
  - total seconds spent in just that function
  - number of times function was called
  - average time per call to the function (again, showing both time spent in just the function and also in calls to others)

gprof output cont.

- The second table repeats this information, but breaks it down further
- For each function, it gives the timing breakdown based on which functions called it, and the timing breakdown based on which functions it called
- If a function is called from many different other functions, and timing behaviour differs based on this, gives the developer a chance to evaluate this

Use with C++ and STL

- Note that gprof analysis includes methods, including all the many many methods from the STL
- We often want to filter that out, and just look at the analysis of the code we've written
- Can be done at the shell level, using grep -v and keywords to filter out the lines we're not interested in, e.g.

```
gprof ./myprogx | grep -v std | grep -v static | grep -v cxx
```