Grading Rubric for Project 1

Team members: _____

Total possible points: 50

* Please do not include your names in the source files.

* No collaboration between different teams is allowed.

Project structure:

1. [/2] All files submitted:

   system_call.c
   context_switch.c
   makefile
   report.pdf

   * Make sure you test it on the C4 lab linux machines before you submit!!
2. [/2] Makefile has the following targets:
   a. *make* – compiles the source code
   b. *runsc* – runs the system calls test
   c. *runcs* – runs the context switch test
   d. *clean* – removes generated files

At a high-level, programs do what they are supposed to do:

3. [/2] source code compiles and runs in the C4 lab without errors
4. [/5] system_call.c – measures the time of a context switch
5. [/5] context_switch.c – measures the time of a context switch
6. [/2] tests are non-interactive (should not ask for user-input)
7. [/2] code is organized and commented

Source file system_call.c:

8. [/4] uses either gettimeofday, clock_gettime, rdtsc, or some other valid method to measure time
9. [/2] invokes a system call correctly
   * E.g. if you decide to measure the time of read(), you will also need to use open() and close().
10. [/4] calculates the average time of a system call using a sufficiently large number of samples

* Pay special attention to time units, data types, and removing unnecessary code between measurements.

Source file context_switch.c:

11. [/4] forces a context switch using pipes or some other valid method
12. [/2] sets the machine to use a single processor
13. [/4] calculates the average time of a context switch using a sufficiently large number of samples

The report.pdf:

1 [/1] introduces the problem

* Describe what you are trying to do.

2 [/1] motivates the reader

* Why is the problem important?

3 [/2] describes your approach

4 [/2] shows code output and explains results

5 [/3] mentions limitations and challenges

* Consider issues such as: accuracy, variability of context switch times, issues on multiprocessor architectures, etc.

6 [/1] concludes with summary and final thoughts

*One paragraph is enough.