CSc33200: Operating Systems, CS-CCNY, Fall 2003 Jinzhong Niu October 4, 2003

Nachos Project Mechanics[1]

1 Group formation

The project is to be done in teams of two or three. The assignments will be the same for either size team. Groups of three will have less coding to do per person but will also have to manage the interactions between more people. Every member must actively take part in the team work.

When you form a team, please send email to `jniu at ccny.cuny.edu` to register yourselves. Every group will be given a *unique group email address* in the form of

```
csc33200_groupX@yahoo.com, X is a digit
```

where you should submit your work before the deadlines. Any question should go to my personal email box or our Yahoo group email address instead of this one.

2 Code and documentation

You must surround every piece of code you modify with:

```
#ifdef CHANGED
   put your changes here
#endif
```

so that I can recognize what changes you have made.

In addition to your code, I also expect you to turn in a report containing:

- a written outline of what you did for the project, explaining and motivating your design choices. Keep this short and to the point! Of course, you should also document your code if you want the reader to understand what you did (a requirement for getting a good grade!). Some assignments have parts that ask you to explain something about your design; this explanation should also go in this report file.

[1] This file is originally from Prof. Tom Anderson of CS at UC-Berkeley. Here I did some minor alterations to fit our needs.

- an explanation of test cases you used. I expect you to run test cases to demonstrate that your code works (or that it doesn't!); the results of these test cases should be included (along with an explanation of what each test case is supposed to show). I am not going to tell you what test cases to run – it is up to you to convince me that your code works. You should pick the minimal set of test cases that do this.

3 Submission

To submit your project work, you should first run

```
make clean
```

to delete any temporary files and object code in your Nachos directory tree, and then make a package from all the source files, Makefiles, as well as the report file at the root directory of the package. The package should be named `project_X.tar.gz`, where `X=1, 2, ...` and once uncompressed is supposed to be ready for me to build on UNIX lab's Solaris platform by using `make`. Finally the package should be sent to the email address assigned to your group.

4 Grading

If I think it necessary, an interview may be held with your group, where you will be asked to explain and defend the design you chose for implementing each assignment. Normally I will meet you at UNIX lab or PC lab in NAC so that you may show me your work right there.

The intent of the grading for the project is NOT to differentiate among those students who do a careful design and implementation of the assignments – in other words, don't implement some complex feature just because someone else in the class is implementing it. My expectation is that most of you will get close to full credit on each assignment. Rather, the grading is to help me identify those students who (i) don't do the assignments or (ii) claim unable to do the project but fail to ask any reasonable question or (iii) don't think carefully about the design, and therefore end up with a messy and over-complicated solution. To me what really matters is your attitude.

The grading for the projects will be as follows: 50% design, 50% implementation. I have structured the grading in this way to encourage you to think through your solution before you start coding. Even if your implementation doesn't completely work, you may also have a high score if your design is logical and easy to understand. The implementation portion of the grade considers whether you implemented your design, ran reasonable test cases, and provided documentation that the reader could understand. Part of being a good engineer is coming up with simple designs and easy to understand code; a solution which works isn't necessarily the best that you can do. Thus, part of the design and implementation grade will be based on whether your solution is elegant, simple, and easy to understand.

The grading will be mainly based on your report since it is to tell me what you have done. I will also check your package to confirm what you claim in the report.