CS 6/73201 **Advanced OS**

Project #2

Due via email by 11:59pm on Sunday 2 May 1999

The Problem

In this assignment, you are to (i) use Nachos to implement **Garcia-Molina's Bully Algorithm for Elections** (described in Lecture 16), and (ii) come up with appropriate **test code or methods** to demonstrate that your algorithm works for at least 4 copies of Nachos. Take part (ii) seriously — think about how you can have these copies of Nachos run, "fail", notice that others have failed, etc., and then make sure that your implementation and test cases demonstrate this effectively.

Note that Nachos does not allow you to send messages to a Nachos process after it crashes — if you try, the sending copy of Nacho will fail with an assertion error. So, for simplicity in this project, just have the "failed" process send back an "I've crashed" message, instead of trying to work around this restriction or use timeouts.

This project will comprise 15% of your final course grade.

What to Turn In

(30 points) In a file called **p2.overview**, write a brief overview of your project. Describe what you implemented, any design decisions that you made, and why you made those decisions. Describe your test cases, and how they demonstrate that the project work. Since the TA will have a large number of different projects to grade, supply instructions to the TA as to how he should test your project to demonstrate that it works. Do a good job on this writeup — convince the TA and me that you implemented something interesting, that you learned something from doing it, and that you actually had some tough choices to make in your implementation. A superficial 10-line writeup will definitely not be worth 30 points!

(70 points) The implementation of your project. If it doesn't quite work as desired, clearly describe in the file **p2.overview** what you have done, what is not working, and how you would go about finishing the project if you had more time. The better you describe the status, the better your chances will be for getting partial credit for what you've done. Also, if you have something that works, and you can document that status with test cases, etc., you'll probably get more credit than if you have a lot of code written, but nothing that compiles (since in the latter situation it's harder for the TA to evaluate the status of your work).

Identifying Your Changes

So that the TA and I can easily identify which code you have changed or added, surround all changes and additions in your code by comments in the following form:

```
// PROJECT 2 CHANGES START HERE
```

<your changed code goes here>

```
// PROJECT 2 CHANGES END HERE
```

Use your own judgment about how much code to surround in a single comment.

Where to Get Help

Help is available from Prof. Walker and from Mr. Kun Qiu (who will act as the course TA for programming projects, but ***not*** for helping with homework assignments, exam preparation, etc.):

- For questions on what the assignment is asking, please contact Prof. Walker.
- For questions on Nachos, please contact either Prof. Walker or Mr. Qiu.
- For help with your code or debugging, please contact Mr. Qiu (***not*** Prof. Walker)

Our office hours are on the class web page, and may be extended if necessary as the project deadline approaches; see the class web page for any announcements of extended office hours.

Also, if there are corrections or amplifications to this project, or if someone asks a question and we feel the answer may be relevant to other people, that information will be posted on the class web page under the project assignment. Thus, you might want to check the class web page periodically until the project due date to avoid getting bogged down in some problem to which a solution has been announced.

Cooperation versus Cheating

See the class syllabus, and contact me if you have any questions. You are allowed to discuss the problems with your friends, and to study the Nachos internals with your friends, but you are not allowed to write pseudo-code to solve the problems with your friends, and you are certainly not allowed to copy anyone else's code.

Submitting Your Project

When you finish, submit all files that you modified to the TA, just as you did in Project 1.

Important warning — once you submit your files, **DON'T TOUCH THEM AGAIN** — if your email didn't reach the TA, or something happens, the TA may need to ask you to resubmit your files. However, before he lets you do so, he will ask you to log on in his presence, and he will check the modification dates on your files to make sure that they haven't been modified after the due date (if they have been, you will be assessed the appropriate late penalties).

The project is due at 11:59pm on Sunday 2 May 1999. For a discussion of my late policy, see the class syllabus.