Homework/Lab 1
Introduction to (or Review of) C and Unix and the Bullpen Cluster
due Monday, February 10, 2003, 11:59 PM

There are several files to turn in for this assignment. They should all be included in a file named `hw01.tar` that you submit using the `turnin` utility. *Please use the filenames specified* and be sure to include your name in each file. Please ask if you have any questions about creating the tar file or using the turnin utility.

1. Send me mail at *terescoj@cs.williams.edu* with a brief (a couple sentences) description of your level of experience with the Unix operating system and its variants. Also include list programming languages you have used and your proficiency in each, any experience you've had with parallel programming, and anything else you'd like me to know about your background. (0 points)
2. Log into and familiarize yourself with your CSLab Unix account. Forward your CSLab electronic mail to an address you read regularly, as I will often use your *@cs.williams.edu* address. Try FreeBSD systems in the lab (*epirus, hinterwald, toto, pester, zebu, faeroes, dulong, bearnaise, pineywoods, basuto, baggerbont, watusi*) and the Solaris cluster (*bullpen*). Create a directory in your account for work from this class. Change the permissions on the directory so only you have read or write access to it. (0 points)
3. Read the web page for the Bullpen cluster at http://bullpen.cs.williams.edu. Don't worry if some of the terminology is not yet familiar. (0 points)
4. Copy the C program on the online version of this page that computes the late penalties for this course to your CSLab Unix account. Compile and run it, redirecting your output to a file `late.txt`. Include this file in the tar file that you submit when you are finished. (1 point)
5. In a plain text file `hw01.txt`, name an application you use that you think could benefit from parallel processing. Describe how you think it could be parallelized and what benefit parallelization might provide. (1 point)
6. Copy the file on the online version of this page to your CSLab Unix account, either from this link, or from `/home/faculty/terescoj/shared/cs338/hw01/make-example.tar`. It is a "tar file" of a small C program that demonstrates the use of multiple source files and `Makefiles`. Extract the files (`tar xvf make-example.tar`) and compile the program with `make`. Breifly describe in your plain text file `hw01.txt` how `make` uses the rules in the `Makefile` to produce the executable `main`. (1 point)
7. The Bullpen web page includes a simple parallel program called `mpihello.c`.
   - Copy `mpihello.c` to your account.

- Create a `Makefile` that builds an executable called `mpihello` when you type `make` or `gmake`. Be sure to create a 64-bit MPICH executable.
- Run the program with two processes interactively on *bullpen* and redirect the output to a file `mpihello.out`.
- Run the program three ways through the PBS batch system:
  - Create and run a PBS script `twoonone.pbs` that runs the program with two processes on any one node and sends the output to a file `twoonone.out`.
  - Create and run a PBS script `fouronfour.pbs` that runs the program that runs the program with four processes on four different nodes, one process per node, and sends the output to a file `fouronfour.out`.
  - Create and run a PBS script `all24.pbs` that runs the program that runs the program with 24 processes on all available nodes, one process per processor, and sends the output to a file `all24.out`. Hint: the cluster has two 4-processor nodes (ppn=4), six 2-processor nodes (ppn=2), and four uniprocessor nodes.
- Include the files `mpihello.c`, `Makefile`, `mpihello.out`, `twoonone.pbs`, `twoonone.out`, `fouronfour.pbs`, `fouronfour.out`, `all24.pbs`, and `all24.out` in your submitted tar file. Do not include any ".o" files or the `mpihello` executable. (2 points)