CS 536 (Park) **Assignment I** **Due: Sep. 9 (Thu.), 2004**

Submission instructions: Please type your answers and submit electronic copies using `turnin` *by 5pm on the due date. You may use any number of word processing software (e.g., Framemaker, Word, LaTeX), but the final output must be in pdf or ps format that uses standard fonts (a practical test is to check if the pdf/ps file prints on a CS Department printer without problem). For experiments and programming assignments that involve output to terminal, please use* `script` *to record the output and submit the output file. Use* `gnuplot` *to plot graphs. Use* `ps2gif` *to convert a eps/ps plot to gif format (e.g., for inclusion in Word).*

PROBLEM 1 (30 pts)

Read Chapter 1 from Peterson & Davie (henceforth P & D). Solve Problems 9, 17, 28, 29, and 38 (add `www.cs.purdue.edu` to the list in Problem 31). For Problem 38, repeat the experiment three times, ten minutes apart. At each experiment, run `ping` such that it sends 10 probes. Set the probing interval to be 1 second. At each experiment, probing is carried out for the three destinations at hand. Tabulate your results and describe your findings. Use `traceroute` to discover the routes taken to the three destinations. Do the hop counts (i.e., path length) correlate to with the delay, i.e., round-trip time (RTT) observed through `ping`? Discuss your results.

PROBLEM 2 (20 pts)

What are the key differences between a computer network, as understood by the lectures, and a highway road system? Distinguish between fundamental vs. cosmetic differences. Careful, logical thinking is needed to not miss key distinguishing features. Based on the fundamental traits, how would you define bandwidth, throughput, latency, delay, and "packet" loss for a highway system? Is routing circuit-switched or packet-switched? How is scheduling of automobiles done at intersections? Does it correspond to scheduling methods familiar in networking or CPU scheduling? Include ambulances and other emergency vehicles in your consideration.

PROBLEM 3 ($50 = 25 + 25$ pts)

(a) Give a 2-page summary overview of the following aspects of UNIX system programming: signals (`sigaction`, `signal`, `kill`, `pause`), interprocess communication (`pipe`, `mkfifo`, `semget`, `shmctl`), and process creation/execution (`fork`, `vfork`, `execve`). Explain the notions *atomicity*, *blocking/non-blocking*, *synchronous/asynchronous I/O*, *mutual exclusion and semaphores*, and *interrupts*, and in which system calls they play a role. The `man` pages should be the primary source of reference.

(b) You will find a client/server application under ~park/pub/cs536; the client program is `client.c` and the server program is `server.c`. Reverse-engineer the code and explain what the server and client are doing. Give a line-by-line explanation of the code—you may, when warranted, group several lines into one unit and give a modular description. The important criterion is that your answer makes clear that you understand all aspects of the code. If you find inefficiencies or code improvements (define what you would consider an "improvement"), indicate what and why, and suggest possible changes. Important: include in your answer a clear explanation of what technique is being used to direct the output of the server back to the client, why this is nontrivial, and how it works. Compile the code and get it to run. What do you observe? What happens when you replace `fortune` with `finger`?