CSCI 631 **Computer Assignment 1** **Spring 2018**
(30 points)

Daytime Client–Server Model
<Client Side>

The source code of a TCP daytime client program is given in your textbook. After compiling the source files of your program and linking the resulting object files with the system library routines, execute your program as a daytime client.

For error checking, use the I/O routines such as fprintf ( ) or perror ( ), and remember that all error messages are supposed to be printed out on stderr. For error handling, use wrapper functions for all system calls that you use in your program, as explained in your textbook.

Guard the statements in your header files using the following format. This is necessary because you don't want the statements in a header file are processed more than once.

```
#ifndef A–CONST–VALUE  // not defined any other place
#define A–CONST–VALUE // same const value as for ifndef directive

// put all statements for your header file here

#endif
```

Insert the following statement: #include "/home/cs631/common/config.h" and all system header files that you need in your program at the top of your header files. For example, to use the socket ( ) function in your program, insert the statement: #include <sys/socket.h> in your header files. (You can find the information for the socket ( ) function on its man pages.) At the top of each your source file, insert the corresponding header files.

Name the source file of your client as prog1cli.c and the source file of your wrapper functions as wrap1cli.c, and name your header files as prog1cli.h and wrap1cli.h.

To compile your source files and link its object files with the system library routines, execute the command: Make N=1 M=1. If the compilation and linking is successful, test your client by executing the command: Make execute N=1 M=1. This executes your client by calling the script P1–1 in directory: ~cs631/bin/18s and gets the current daytime information from two different systems (one is local, and one is remote).

The output will be displayed on your terminal screen and stored on the file prog1–1.out. You can find the correct output in file: ~cs631/progs/18s/p1/prog1–1.out.

After you are done with your program, you don't need its object and executable files any more. To delete them, execute: Make clean.

You need to include proper documentation in all files you submit for grading. In the source file of your wrapper functions, only include the ones that you use in the current

program, and you need wrappers for the following system calls: close ( ), connect ( ), fputs ( ), inet_pton ( ), read ( ), and socket ( ).

When the inet_pton ( ) function flags an error by returning an integer value <= 0, the return value –1 sets errno but not the return value 0, so for the return value 0, do not use the function perror ( ) to print out the error message. Instead, use the I/O function fprintf ( ) on stderr without using the function strerror ( ).

When you execute your client program for an Internet address and if you don't receive a response in a few minutes, terminate the execution of your program by the control character <ctrl>–C.

Since a socket is a file in the UNIX system, do not forget to close it at the end of your program, and remember that the close ( ) function is a system call and it needs a wrapper.

When your program is ready, submit its source and header files to your instructor by executing mail_prog prog1cli.c prog1cli.h wrap1cli.c wrap1cli.h. When the script prompts, enter 1 for the program number.