# COSC 350 System Software Lab #7

How to submit

- For each program, you need write detailed comments for each statement.
- Submit each program by email to cosc350@gmail.com.

**Task 1: <u>Alarm using sigaction</u>**

The BLP version of alarm.c uses the old-fashioned signal methodology. In this task, you will modify alarm.c to use sigaction.

A. Copy alarm.c from BLP 4th edition page 485, (page 468 in BLP 3rd edition).
B. Modify alarm.c so it uses sigaction.

**Task 2: <u>Catch ^C with sigaction</u>**

A. Copy ctrlc1.c from BLP 4th edition, page 483( page 466 in 3rd edition). Run it to see how it behaves.
B. Copy ctrlc2.c from BLP 4th edition, page 488 ( pages 470 in BLP 3rd edition). Run it to see how it behaves. Note that it does not behave exactly like ctrlc1.
C. BLP advises using SIGQUIT (^\) to quit from ctrlc2. Try it to see how it behaves.
D. As an alternative way to quit from it, put it in the background, then issue a kill. Write down a brief description of exactly what you did.
E. Modify your ctrlc2.c so it behaves exactly the same as ctrl1.c. Hint: read the sigaction Flags section of BLP, Chapter 11.
F. Print your modified ctrlc2.c to hand in.
G. While we're at it, take a look at what is printed by kill -l. It's an easy way to get a list of all the signal numbers and names.

**Task #3: <u>process communication witout sending signal</u>**

Write a complete C program in which a child process write a message "Hi, Mom" to a file named foo. The parent process reads the message and prints it to standard output “My son said Hi Mom”. The part of message “Hi Mom” get from the file foo. Assume that all system calls succeed (no need to error check). Use only low-level file operations (fork, wait, open, close, read, write, lseek). You must make sure a child process terminate first.

**Task #4: <u>using signal and kill system calls</u>**

Write a complete C program in which two chidren processes send signal to the parent. The first child send message SIGUSR1 to the parent and the parent process reponse by writing message “Hi Honey!

Anything wrong?”. The second child send message SIGUSR2 to the parent and the parent response by writing message “Do you make trouble again? ”.

**Task 5: <u>sigprocmask</u>**

Write a complete C program that demonstrates how you can block and unblock signals. Your program will have two loops, each of which simply prints the integers from 1 to 5 at 1 second intervals. During the first loop, SIGINT and SIGQUIT are blocked. During the second loop, only SIGINT is blocked. While the program is running, you can try ^C and/or ^\ to see if they are blocked as expected.

- The signal SIGINT is usually bound to ^C by the terminal driver. The SIGINT signal terminates the program by default.
- The signal SIGQUIT is usually bound to ^\ by the terminal driver. The signal SIGQUIT by default causes the program to terminate and creates a core file