# Software Interrupt – Signaling

Note: Please use text highlight color to highlight your answer.

## Signaling Basics

Signal is a software interrupt, which can be used for inter-process communication. Please read the handout and answer the below questions.

1. What is the purpose of *killall cat*?
2. What is the difference between kill and killall?
3. What is the Control-c from the viewpoint of signal?
4. What is the Control-z from the viewpoint of signal?
5. Signals actually are a list of values. These name and values are defined in which header file?
6. Write a source code file, loop.c, with int main() {while(1); sleep(10);}.
7. Compile loop.c and run it.
8. Then use another terminal to send the below signal: killall -SIGSEGV loop. What do you observe when you run jobs in the first terminal? Paste the screenshot.
9. In the same way as the last question, send the below signals and describe what you have observed? SIGUSR1, SIGUSR2, SIGFPE, SIGILL, SIGKILL, SIGQUIT.
10. Run loop in a daemon in the first terminal (loop &) and send the signal, SIGSTOP, in the second terminal to loop. Use jobs in the first terminal to view what has happened? Paste the screenshot.
11. Send SIGCONT to loop and use jobs to observe what has happened in the first terminal?

## Signaling Handler

1. Try to compile the below code snippet, hello\_signal.c, and explain the purpose of each line in the main function.

#include <stdlib.h>

#include <stdio.h>

#include <signal.h> /\*for signal() and raise()\*/

void hello(int signum){

printf("Hello World!\n");

}

int main(){

signal(SIGUSR1, hello); //execute hello() when receiving signal SIGUSR1

raise(SIGUSR1); //send SIGUSR1 to the calling process

}

1. Compile the code, hello\_signal\_loop.c, and run it. Type control-c and observe what have happened? Can you really terminate it using Control-c? If not, try control-\ to send SIGQUIT signal to the program. Observed what have happened?
2. Run the program and try to send SIGKILL signal to the program from another terminal using killall. Describe what you have observed. Paste the screenshot.
3. Download and compile the code, ipc\_signal.c, and run it. Look at the source code. What is the purpose of kill()?

## Basic Signaling Procedure for Multiple Processing Programming

1. Modify ipc\_signal.c to design a program, named my\_ipc\_signal.c. In the program, the parent creates two child processes, #1 and #2. Each child process is running a loop to print out the process’s # and sleep 2 seconds. Each child process should register SIGUSR1 signal to its own handler function. Once the parent process sends the SIGUSR1 to child processes (Note: the parent sends the signal and should NOT receive the signal). The child processes should receive the signal through the handler function and print out “good bye” to exit the program. This procedure indicates a commonly used basic design framework in many commercial software based on Linux.

Please show your code here and paste the output (screenshot is fine).