Comp 4981 Computer Systems Technology January 2012

Data Communication Option

Assignment #1

Due: January 24, 0930 hrs. This is an individual assignment.

Objective: To use ***fork***, ***pipe*** and **signals** in a program that reads and processes characters entered on a terminal keyboard.

Your Mission:

Write and test a C program which reads characters from a terminal keyboard, echoes the characters to the terminal screen, process each line of characters and writes the processed line to the screen. The program must be composed of three cooperating processes, which perform the following functions:

- Function '**input**' reads each character entered at the keyboard. It sends the character to the function '**output**' to be echoed. It also stores the character in a buffer until a special ENTER character (like 'return' in normal UNIX processing) is received. When ENTER is received the data in the buffer is passed to function '**translate**' and the buffer is emptied in preparation for another line of characters.

- Function '**translate**' processes the characters passed to it. It performs the normal UNIX erase and line-kill processing and also translates all occurrences of the character "**a**" to "**z**". After completing the processing it sends the new line of characters to the function '**output**'.

- Function '**output**' simply receives characters from '**input**' or sets of characters from '**translate**' and displays them on the terminal screen.

Constraints:

- The program must disable all UNIX terminal processing.

- The three cooperating processes must use UNIX pipes to send data between processes.

- Instead of using UNIX default values for erase (backspace), line-kill (escape), etc., use 'X' for erase, 'K' for line-kill, 'control-k' for interrupt (i.e. abnormal termination of the program) and 'T' for normal termination of the program.

To Be Submitted:

- Hand in complete and well-documented design work and listings of your program.

- Provide a copy of all your **design work**, **testing documentation**, **code listings** and an **executable** on a disk.

<u>Evaluation:</u>

(1). Design Work:	5
(2). Code Quality:	5
(3). Testing (process & documentation):	10
(3). Functionality:	30
Total:	50