CS35L Software Construction Laboratory

Lab 5: Sneha Shankar

Week 6; Lecture 2

Some important points

```
ch = getchar()

putchar(ch)

int numRead = read(STDIN_FILENO, ch, size)

int numWritten =  write(STDOUT_FILENO, ch, size)
```

Lab Assignment

- Write `tr2b` and `tr2u` programs in 'C' that transliterates bytes. They take two arguments 'from' and 'to'. The programs will transliterate every byte in 'from' to corresponding byte in 'to'
- `./tr2b 'abcd' 'wxyz' < bigfile.txt`
  - Replace 'a' with 'w', 'b' with 'x', etc
- `./tr2b 'mno' 'pqr' < bigfile.txt`
- `tr2b` uses **getchar** and **putchar** to read from STDIN and write to STDOUT.
- `tr2u` uses **read** and **write** to read and write each byte, instead of using getchar and

  putchar. The nbyte argument should be 1 so it reads/writes a single byte at a time.
- Test it on a big file with 5,000,000 bytes
- `$ head --bytes=# /dev/urandom > output.txt`

tr2b.c

- Write a main function which accepts arguments
    - main(int argc, const char* argv[])
- Check for the length of arguments
- Retrieve first argument in char * from, second argument in char * to
- Compare the lengths of from and to; If not same, throw an error and exit
- You can use strlen to get lengths
- To throw an error, write to stderr
- To exit, write exit(1)
- Check if ‘from’ has duplicates

tr2b.c

- In a loop, take input from stdin (till you reach eof of stdin) using getchar()
- Check if the character you just retrieved is a part of from; if yes then put the corresponding character in stdout with putchar()

tr2u.c (read and write)

- Repeat the same procedure as in tr2b.c except replace:
  - getchar() with read
  - putchar() with write

time and strace

- **time [***options***]** *command* **[***arguments...***]**
- Output:

–real 0m4.866s: elapsed time as read from a wall clock

–user 0m0.001s: the CPU time used by your process

–sys 0m0.021s: the CPU time used by the system on behalf of your process

- **strace**: intercepts and prints out system calls.

–$ strace –o strace_output ./tr2b ‘AB’ ‘XY’ < input.txt

–$ strace –o strace_output2 ./tr2u ‘AB’ ‘XY’ < input.txt

Pointers on system calls

www.cs.uregina.ca/Links/class-info/330/SystemCall_IO/SystemCall_IO.html

courses.engr.illinois.edu/cs241/sp2009/Lectures/04-syscalls.pdf

www.bottomupcs.com/system_calls.xhtml

Homework 5

- Rewrite sfrob using system calls (sfrobu)

- sfrobu should behave like sfrob except:
  - If stdin is a regular file, it should initially allocate enough memory to hold all data in the file all at once
  - It outputs a line with the number of comparisons performed

- Functions you'll need: read, write, and fstat (read the man pages)

- Measure differences in performance between sfrob and sfrobu using the time command

- Estimate the number of comparisons as a function of the number of input lines provided to sfrobu

Homework 5

- Write a shell script “sfrobs” that uses tr and the sort utility to perform the same overall operation as sfrob
- Use pipelines (do not create temporary files)
- Encrypted input -> tr (decrypt) -> sort (sort decrypted text) -> tr (encrypt) -> encrypted output

Homework 5 (sfrob.txt)

- Measure any differences in performance between sfrob and sfrobu using the time command.
- Run your program on inputs of varying numbers of input lines, and estimate the number of comparisons as a function of the number of input lines
- Use the time command to compare the overall performance of sfrob, sfrobu, sfrobs, sfrobu –f and sfrobs -f

Read and Write system calls

#include <unistd.h>

- ssize_t read(int fildes, void *buf, size_t nbyte) – – –
  - fildes: file descriptor
  - buf: buffer to write to
  - nbyte: number of bytes to read
- ssize_t write(int fildes, const void *buf, size_t nbyte);
  - fildes: file descriptor
  - buf: buffer to write from
  - nbyte: number of bytes to write
- int open(const char *pathname, int flags, mode_t mode);
- int close(int fd);
- File descriptors
  - 0 stdin]
  - 1 stdout
  - 2 stderr

fstat system call

- int fstat(int filedes, struct stat *buf)
    - Returns information about the file with the descriptor filedes into buf