Lecture 2: Essentials UNIX Libraries

- Let's put the `sumtwo` function in a _____ library:
  
  ```
  $ g++ -c mymath.cpp    # generate mymath.o object file
  $ ar rv libmymath.a mymath.o   # add obj file to lib
  $ ranlib libmymath.a       # not required under Linux
  $ g++ -o second.second.cpp -L. -lmymath
  ```

- `-L. option (required) tells g++ to look in the current directory for libraries. -lmypath tells g++ to open the library name `libmymath.a`

- `gcc` automatically looks in standard directories for libraries: `/usr/lib`, `/usr/local/lib`

---

Lecture 2: Essentials UNIX Libraries

- The `ar` utility is used to create, extract, list, and add modules to libraries:
  
  ```
  $ ar t /usr/lib/libm.a    # List modules in math lib
  $ ar x libmymath.a sumtwo.o   # Extract obj file
  ```

- The `nm` utility can be used to list all of the _____ in a library, object file, or executable.
  
  ```
  $ nm -C libmymath.a
  $ nm -C test
  ```

---

Lecture 2: Essentials UNIX Libraries

- Linux supports static and shared libraries.

- The shared versions of standard libraries are used by default. If you want to use only static libraries use the `-static` option with `g++`

- Code from shared libraries is loaded at ______ and may be shared by many applications.

- Shared libs have a `.so` (shared object) extension.

- Use “gcc -print-search-directories” to list static library locations and “ld -verbose” for shared library locations.

---

Lecture 2: Essentials UNIX Libraries

- The `ldd` utility will list the shared libraries required by a program:
  
  ```
  $ ldd /bin/ls
  libtermcap.so.2 => /lib/libtermcap.so.2
  libc.so.6 => /lib/tls/libc.so.6
  /lib/ld-linux.so.2 => /lib/ld-linux.so.2
  $ ldd test
  libstdc++.so.5 => /usr/lib/libstdc++.so.5
  libm.so.6 => /lib/tls/libm.so.6
  ```

- ________ is used by the administrator to make shared libraries available to others on the system.

---

Lecture 2: Essentials UNIX Philosophy

- Here are a few notes on UNIX programming philosophy:
  
  - Write small programs that do one thing well.
  - Read data from standard input (stdin, cin) and write to standard output (printf, cout). Such programs are known as filters. (Very complex applications can be created using sources, ________, sinks and pipes.)
  - Move core routines into well-documented libraries so that they can be reused.
Lecture 2: Essentials

Essential Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ls</td>
<td>Displays directory listing</td>
</tr>
<tr>
<td>cd</td>
<td>Change the current directory (working directory)</td>
</tr>
<tr>
<td>cd</td>
<td>Return to HOME directory</td>
</tr>
<tr>
<td>pwd</td>
<td>Display the current directory</td>
</tr>
<tr>
<td>_____</td>
<td>Create a directory</td>
</tr>
<tr>
<td>filename</td>
<td>Display file type</td>
</tr>
<tr>
<td>cat</td>
<td>Display file contents</td>
</tr>
<tr>
<td>less</td>
<td>Page file contents to display (also more)</td>
</tr>
<tr>
<td>exit or ^D</td>
<td>Leave this session</td>
</tr>
<tr>
<td>passwd</td>
<td>Change your password</td>
</tr>
<tr>
<td>man</td>
<td>Display man pages on command</td>
</tr>
<tr>
<td>info</td>
<td>Display info pages on command</td>
</tr>
</tbody>
</table>

Lecture 2: Essentials

Useful Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apropos string</td>
<td>Search the whatis (manual page) database for string (same as “man -k string”)</td>
</tr>
<tr>
<td>whatis command</td>
<td>Display brief description from whatis database (same as “man -f command”)</td>
</tr>
<tr>
<td>command --help</td>
<td>Display brief help on command</td>
</tr>
<tr>
<td>ps aux</td>
<td>Display all processes</td>
</tr>
<tr>
<td>kill</td>
<td>Kill a process (requires ownership)</td>
</tr>
<tr>
<td>du foo</td>
<td>Display disk usage by file or dir foo</td>
</tr>
<tr>
<td>df</td>
<td>Display free disk space</td>
</tr>
<tr>
<td>_____</td>
<td>Search for character strings</td>
</tr>
<tr>
<td>find</td>
<td>Find files meeting criteria</td>
</tr>
<tr>
<td>tar</td>
<td>Standard UNIX archive tool</td>
</tr>
</tbody>
</table>

Lecture 2: Essentials

File Information

- Use "ls -l" to display complete file info:
  
  $ ls -l
drwxrwxr-x 13 ar63 ar63 4096 Jul 25 10:43 qt
-rw-r--r--  1 ar63 ar63 6455 Aug 19 14:53 readme.txt
lrwxrwxrwx  1 ar63 ar63 23 Aug 8 12:59 rt -> a.txt
prw-rw-r--  1 ar63 ar63 0 Aug 27 15:18 mypipe

- Shown: file type, permissions, # of hard link, owner, group, size, creation time, and name.

- File type is indicated by first character:
  - Regular file d
  - Directory l
  - Soft link p
  - Named pipe c
  - Char device b
  - Block device s

Lecture 2: Essentials

Soft (Symbolic) Links

- Use "ln -s" to create soft or symbolic links. A soft link is a pointer to a "real name" for a file. These are similar to Windows shortcuts but are supported by both the CLI and the GUI.
  
  $ ln -s oldfile newname

- The file may be on another filesystem.
- You can create soft links to directories.
- Deleting the link has no effect on the file.
- Deleting the file can leave a soft link to nothing.

Lecture 2: Essentials

Hard Links

- Use ln to create hard links. Creating a hard link creates another "real name" for a file.
  
  $ ln newname oldfile

- Data is not deleted until all hard links are deleted
- Data must be on the same filesystem as the link
- System call to delete a file is actually "unlink"
- Only the administrator can create hard links to directories (to prevent filesystem corruption due to directory "loops" by unwitting users)

Lecture 2: Essentials

File Links

- File with two hard links
- File with one hard link and one soft link
Lecture 2: Essentials
Special File Names

- A filename whose first character is a _____ will not show up in normal directory listing.
  - Use 'ls -a' to list hidden files
  - These are usually configuration files.
  - This has NOTHING to do with security.
- Every directory automatically includes . and .. which are links to itself and its parent directory.

$ cp /etc/passwd .
$ cd ../../../datadir

Lecture 2: Essentials
Device Files

- Device files provide access to hardware. There are two types: character and ________.
  - They reside in the /dev directory.
  - They can be created with mknod by the administrator.
  - Permissions determine access to corresponding hardware.
  - Character devices provide sequential access (serial ports) while block devices are used for random access (drives).
- See man pages for sd, random, etc.

Lecture 2: Essentials
A Few Standard Device Files

/dev/sda  First detected disk drive
/dev/sdb  Second detected disk drive
/dev/sdc
/dev/sr0  CD/DVD drive
/dev/cdrom Link to actual CDROM device (/dev/sr0)
/dev/st0  SCSI tape drive
/dev/sda1  First primary partition on sda drive (C: drive in Windows)
/dev/sda2  Second primary partition on sda drive
/dev/sda5  First logical partition
/dev/fd0   First _______ drive (autodetect format)
/dev/snd* ALSA sound system devices

Lecture 2: Essentials
A Few Standard Device Files

/dev/tty   Current terminal device
/dev/tty1  First console device
/dev/ps/1  First pseudo-terminal (xterm) device
/dev/fd/0 File descriptor 0 or standard input (1 is stdout, 2 is stderr)
/dev/stdin Alias for /dev/fd/0 (also /dev/stdout, /dev/stderr)
/dev/ttyS0 First serial port (COM1)
/dev/p0    First _______ port
/dev/audio Sound card i/o (obsolete)
/dev/null  Null device (bit bucket)
/dev/zero  Zero device
/dev/random Random number generator (better)
/dev/random Random number generator (faster)
/dev/mem   Physical memory
/dev/psaux PS/2 mouse port

Lecture 2: Essentials
Cool Device Tricks

$ cat /dev/fd0 > floppy.img  # Create floppy image
$ cat floppy.img > /dev/fd0  # Duplicate a floppy
$ dd if=/dev/cdrom of=/cdrom.iso # Create ISO Image of CD
$ cat /dev/audio > sound.au  # Record sound from microphone (obs)
$ cat sound.au > /dev/audio  # Play sound (obs – see pulseaudio docs)
$ program 2 > /dev/null    # Redirect program error messages to bit bucket
$ program /dev/stdin      # Pass stdin as filename argument
$ cd -A n -N & & u2 < /dev/random # Generate 4 16-bit unsigned random #s

- You can create, format, and mount disk _______!
$ dd if=/dev/zero of=f.img bs=1k count=1440  # Create 1.44MB zero file
$ mkfs -t msdos f.img    # Create a file system
$ mount -o loop f.img a  # Mount the file system

Lecture 2: Essentials
Cool Device Tricks

// Print 10 random 32-bit integers
#include <iostream>
#include <cstdlib>

using namespace std;

int main(void)
{
  int randnum;

  ifstream randstr("/dev/random");

  for(int count=0; count<10; count++) {
    randstr.read((char *) &randnum, sizeof(randnum));
    count << count << " " << randnum << endl;
  }
  randstr.close();
  return 0;
}
Lecture 2: Essentials

File Permissions

- Regular file permissions are specified for user (owner), group and other (anyone else).
  - Permissions are r (read), w (write) and x (execute)
  - Write permission allows you to alter a file's contents. It does not give you permission to ______ the file.
  - Permissions also apply to named pipes and devices.
- Here are some examples:
  - rw-r--r-- User can read and write, group other just read
  - rw-r-xr-x All can read and execute, user can write

Directory Permissions

- Directory perms apply to user, group and other.
  - Read implies permission to get a directory listing.
  - Write indicates permission to create and delete files in the directory. (Write permission on the file is not required!)
  - Execute means permission to _____ files in the directory.
  - This makes more sense if you think of a directory as a file that contains filenames. (Actually, that's what it is.)
- Here are some examples:
  - drwxr-xr-x User has full access, others can use ls, cp
drwxrwx--- User and group have full access, others none

Changing Permissions

- Use the **chmod** command
- You must be the owner (or root) to change permissions.
- ______ mode examples
  - $ chmod u=rw,og=r readme.txt
  - $ chmod ugo+rx datadir
- Numeric mode (r=4, w=2, x=1)
  - $ chmod 644 readme.txt
  - $ chmod 755 datadir

Additional Permission Info

- Special permissions: setuid, setgid, sticky. See 'info chmod' and 'info "File Permissions"
  - rwxr-xr-x setuid (run as owner), chmod u+s
  - rwxr-xr-x setgid (run as group), chmod g+s
  - drwxrwxr-x sticky (only allow owner to delete), o+t
- Look at perms on /usr/bin/chsh (setuid) and /tmp (sticky)
- Linux also supports _____ (access control lists) for fine-grained control, i.e. to give individual users special access to directories and files.

File Ownership

- Use **chown** to change file ownership
  - $ chown richardson readme.txt
  - You must be administrator
  - You become the owner of copied files
- Use ______ to change file group
  - $ chgrp richardson readme.txt
  - You must belong to both groups

File Timestamps

- There are three times associated with a file: last access, modification, and change times.
  - Modification time is when file data was altered. Change time is when file, owner, group, permission, or hard link count was altered.
  - “ls -l” shows modification time by default.
  - Use ______ to see all three times (or use “ls -l --time=atime” or “ls -l --time=cctime”)
- You can update access and modification times using touch.