Week 1

Goal: Learn the fundamentals of commonly used software tools and environments especially open-source tools you may need in CS classes.

SEAS lab ethics:

- No food/drink
- Logout from machine at the end
- No phone calls/music
- Lab instructor f

Grading:

- Assignment 10 is a research and development project. Need to submit a report and make a presentation
- Assignments due at 23:55 Friday
- 50% Homework
- 50% Final, open book and notes

Need to get:

- Beagle bone green single board
    - Seed, Amazon, Digi-Key, Mouser Electronics, Verical

Operating System

- Manages computer hardware and provides common services for software programs to run
- Process Management
- Memory Management
- Security
- I/O management

Examples

- Find MacOS kernel name: uname ->output is Darwin
- Windows, DOS, Unix, Linux, Android

Timesharing OS

- Multiple users
- Segmented Memory
- Multiple processes

Linux OS

- Unix is a commercial OS developed in 1969
    - Used for big mainframe computers

- For academic used, Minix developed in 1987. Can't read or write, just to show how OS works
- Linus Torvalds starts linux project based on Minix in 1991

GNU

- GNU Not Unix
- Free and useful open source tool
- Not OS

Kernel: Layer that acts directly with hardware
Around that tools and utilities
Around that users

CLI (command line interface)	GUI (graphical user interface)
● Steep learning curve ● Pure control (eg scripting) ● Cumbersome multitasking ● Speed: Hack away at keys ● Convenient remote access ● Eg: grep foo file.txt|wc -l	● Intuitive ● Limited control ● Easy multitasking ● Limited by pointing ● Bulky remote access

Shell

- Interface bw user and UNIX
- Types
    - Sh
    - Bash
    - c-shell (csh)
    - Etc

After you ssh into the linux server, you are now in a secure server
In the SEASnet server

- uname
    - Linux
- uname -a
    - Linux lnxsrv07.seas.ucla.edu 3.10.0-693.2.2.el7.x86_64 #1 SMP Sat Sep 9 03:55:24 EDT 2017 x86_64 x86_64 x86_64 GNU/Linux
- uname -r
    - 3.10.0-693.2.2.el7.x86_64
- who

    - classrel pts/0 2017-10-02 10:35 (wifi-natpool-131-179-59-47.host.ucla.edu)
    - wangmi pts/1 2017-10-02 09:31 (cardiff.seas.ucla.edu)
    - mallett pts/2 2017-10-02 09:14 (s-169-232-245-113.resnet.ucla.edu)
    - gasing pts/3 2017-10-02 09:48 (s-169-232-93-43.resnet.ucla.edu)
    - wangmi pts/4 2017-10-02 09:56 (cardiff.seas.ucla.edu)
    - akshara pts/5 2017-10-02 10:58 (wifi-natpool-131-179-58-135.host.ucla.edu)
    - jair pts/6 2017-10-02 10:01 (wifi-131-179-50-16.host.ucla.edu)
    - cstrain pts/7 2017-10-02 11:00 (cs-211-174.cs.ucla.edu)
    - ganley pts/8 2017-10-02 10:57 (wifi-131-179-27-168.host.ucla.edu)
- whoami
    - Akshara
- This shows multiuser support environment by OS since all connected to same SEASnet server
- man: Gives a manual
    - an interface to the on-line reference manuals
    - [akshara@lnxsrv07 ~]$ man
    - What manual page do you want?
    - [akshara@lnxsrv07 ~]$ man whoami
    - man -k
        - **man -k** printf
        - Search the short descriptions and manual page names for the keyword
        - printf as regular expression. Print out any matches. Equivalent
        - to **apropos -r** printf**.**
- clear
    - Clears screen
- echo
    - [akshara@lnxsrv07 ~]$ echo "Hello"
    - Hello
    - [akshara@lnxsrv07 ~]$ lab_release -a
    - -bash: lab_release: command not found
    - [akshara@lnxsrv07 ~]$ echo $SHELL
    - /bin/bash
- pwd: present working directory
    - [akshara@lnxsrv09 ~]$ pwd
    - /u/cs/ugrad/akshara
- Find linux distribution
    - [akshara@lnxsrv07 ~]$ lsb_release -a

Unix File System Layout

- Everything is a file

- Tree structured hierarchy
- Root: /
    - bin
    - boot
    - home (user files/directories)
- Navigate using cd command
    - [akshara@lnxsrv09 ~]$ cd Desktop/
    - [akshara@lnxsrv09 ~/Desktop]$ pwd
    - /u/cs/ugrad/akshara/Desktop
- Show what's in current directory using ls
    - [akshara@lnxsrv09 ~/Desktop]$ ls
    - openmplab sthttpd-2.27.0
    - [akshara@lnxsrv09 ~/Desktop]$ ls -l
    - total 8
    - drwxr-xr-x 2 akshara csugrad 4096 Oct 2 10:57 openmplab
    - drwxr-xr-x 7 akshara csugrad 4096 Oct 2 10:57 sthttpd-2.27.0
    - The date and time is the time it was last accessed
    - So, akshara is the owner of file and csugrad is the group the file belongs to
    - 4096 is the size in bytes of the file
    - For drwxr-xr-x 7
        - d = directory, - in its place = file
        - After d are 3 groups of rwx, they indicate permissions for different kinds of users (read/write/execute)
        - G1: owner permission, G2:Other users added to group, G3: Everyone's permissions
        - - indicates that the permission isn't given.
        - Here owner can read/write/execute, other users in group can read/execute and everyone can read/execute
    - Using pipe(|)
        - [akshara@lnxsrv09 ~/Desktop/openmplab]$ ls -l|more
        - total 4
        - -rw-r--r-- 1 akshara csugrad 1005 May 22 2015 main.c
        - So everything before | is the output for comment after |
    - To display hidden files: ls -a or ls -la
- Go to the home directory: cd ~
- . is the current working directory
- .. is the parent directory

Dealing with Files

- Move a file(no undos): mv
- Copy a file: cp
- Remove a file: rm
- Make a directory: mkdir
- Remove a directory: rmdir

- List contents of directory: ls
    - Only directories: -d
    - All files (including hidden ones): -a
    - Show long listing including permission info: -l
    - Show size of each file in blocks: -s
- Update last modified timestamp or create a new file: touch

Sample commands

```
[akshara@lnxsrv09 ~]$ cd sophomore
[akshara@lnxsrv09 ~/sophomore]$ ls
[akshara@lnxsrv09 ~/sophomore]$ ls -l
total 0
[akshara@lnxsrv09 ~/sophomore]$ touch file1.txt
[akshara@lnxsrv09 ~/sophomore]$ ls
file1.txt
[akshara@lnxsrv09 ~/sophomore]$ cat file.txt
cat: file.txt: No such file or directory
[akshara@lnxsrv09 ~/sophomore]$ nano file.txt
[akshara@lnxsrv09 ~/sophomore]$ ls
file1.txt  file.txt
[akshara@lnxsrv09 ~/sophomore]$ ls -l
total 0
-rw-r--r-- 1 akshara csugrad 0 Oct  2 11:47 file1.txt
-rw-r--r-- 1 akshara csugrad 6 Oct  2 11:48 file.txt
[akshara@lnxsrv09 ~/sophomore]$ echo file.txt
file.txt
[akshara@lnxsrv09 ~/sophomore]$ echo "file.txt"
file.txt
[akshara@lnxsrv09 ~/sophomore]$ rm file1.txt
[akshara@lnxsrv09 ~/sophomore]$ ls
file.txt
[akshara@lnxsrv09 ~/sophomore]$ cp file.txt file1.txt
[akshara@lnxsrv09 ~/sophomore]$ ls
file1.txt  file.txt
[akshara@lnxsrv09 ~/sophomore]$ cat file1.txt
hello
```

Basic File Redirection

- Save output into file
    - ls > save_result.txt
    - No output, new file called save_results
- cat save_result will give contents of save_results.txt
- Can use ls -l > results.txt
- This redirects result from screen to a file

- Something like ls ../ > result2.txt will put contents of parent directory into the file. It will overwrite a pre existing file sharing the same name.
- cat results2.txt | more will display first page of the file, then second page and so on.
- Until now we have redirected standard output.
- Redirecting standard input:
    - < file: use contents of a file in stdin
- Appending to standard output
    - >> file:

History

- !!: replace with previous command
- ^old^new: repeat last command, replace old to new
    - [akshara@lnxsrv09 ~/sophomore]$ touch foo.txt
    - [akshara@lnxsrv09 ~/sophomore]$ ^foo^bar^
    - touch bar.txt
    - [akshara@lnxsrv09 ~/sophomore]$ ls
    - bar.txt file1.txt file.txt foo.txt
    - By default uses previous command, can use !-n to get a replacement in earlier command
- ![-n]: replace with previous nth command

Useful Commands

- Cat: concatenates
- head: Print the first 10 lines of each FILE to standard output. With more than one FILE, precede each with a header giving the file name. With no FILE, or when FILE is -, read standard input.
- tail: Print last 10 lines
- du - estimate file space usage
    - [akshara@lnxsrv09 ~/sophomore]$ du
    - 8 .
    - [akshara@lnxsrv09 ~/sophomore]$ du -a
    - 0 ./file1.txt
    - 0 ./file.txt
    - 4 ./foo.txt
    - 0 ./bar.txt
    - 8 .
- ps - report a snapshot of the current processes. Reads virtual files in /proc
    - [akshara@lnxsrv09 ~/sophomore]$ ps
    - PID TTY TIME CMD
    - 32080 pts/28 00:00:00 bash
    - 39909 pts/28 00:00:00 ps
    - For all users: ps -a
    - More info: ps -aux

    - Executable file becomes a process when loaded in memory and has execution state and is currently running
- kill - terminate a process
- diff
- cmp: bytewise comparison
- wc: counts number of words/bytes/lines etc
        - [akshara@lnxsrv09 ~]$ ps | wc -l
        - 4
        - [akshara@lnxsrv09 ~]$ ps -aux | wc -l
        - 531
    - **-c**, **--bytes:** print the byte counts
    - **-m**, **--chars:** print the character counts
    - **-l**, **--lines:** print the newline counts
- sort: sorts lines of text files
- grep: Search a file, use | to search output of a command
    - **egrep or grep -E:**Run grep with extended regular expressions.
    - **-i:**Ignore case (ie uppercase, lowercase letters).
    - **-v:**Return all lines which don't match the pattern.
    - **-w:**Select only matches that form whole words.
    - **-c:**Print a count of matching lines.
        - Can be combined with the -v option to print a count of non matching lines.
    - **-l:**Print the name of each file which contains a match.
        - Normally used when grep is invoked with wildcards for the file argument.
    - **-n:**Print the line number before each line that matches.
    - **-r:**Recursive, read all files in given directory and subdirectories.

Changing permissions

- Change mod ie chmod
- Chmod u + x filename
    - Changing permission of owner/user and add permission executable
    - If previously -r--r--r--, now -r-xr--r--
-

Reference	Class	Description
u	user	Owner of file
g	group	Users who are members of file's group
o	others	Users not owners of file or members of group

a (ugo)	all	All three above, same as ugo

- The dashes correspond to binary:
  - 101 101 101
  - 5 5 5
- So 777 (111) means everyone can do everything
- + adds specified modes
- - removes specified modes from classes
- = the modes specified are to be made the exact modes for the specified classes
- -groupadd : create new group
- -Useradd: add new user to group
- -usermod: add existing user to group
- Another command: chown
- Root user or superuser has ultimate privilege in system, can use sudo
  - Never run sudo in SEASnet machine

Special Permissions

- Sticky bit:
  - (o + t) <file>
  - On shared directories, it locks files within the directory from being modified/deleted by users other than the file creator, owner of the directory or root. Even if others have write permissions to directory
  - Eg: /tmp
- Set user id/set group id: setuid, setgid
  - (u+s, g+s)
  - “Set user ID upon execution”
  - Run an executable with the permissions of the executable’s owner or group
  - So g+s means when anyone executes this program it will run with the same permission as the file’s group

The find command:

-

```
[akshara@lnxsrv09 ~]$ find . -name o
[akshara@lnxsrv09 ~]$ find . -name o*
./original.cpp
./freshman/original.cpp
[akshara@lnxsrv09 ~]$ find . -name *o*
find: paths must precede expression: Desktop
Usage: find [-H] [-L] [-P] [-Olevel] [-D help|tree|search|stat|rates|opt|exec]
[path...] [expression]
[akshara@lnxsrv09 ~]$ find . -name "*o*"
./.bash_history
./sophomore
./sophomore/foo.txt
```

```
./Desktop
./Desktop/sthttpd-2.27.0/docs
./Desktop/openmplab
./Documents
./Downloads
```

- So the . means current directory
    - Can work without . as well, it is the pwd by default
- Arguments
    - -type: type of file
    - -perm: permission of a file
    - -name: name of a file
    - -user: owner of a file
    - -maxdepth: how many levels to search
- Note that the search is recursive
    - -prune: give it name of child you don't want to search
- File Name Matching
    - ?: matches any single character in a filename
        - Eg: colou?r searches for color or colour
    - *: matches any one or more characters in a filename
    - []: matches any one of the characters between the brackets. Use - to separate a range of consecutive characters
        - Eg: te[sx]tt1.txt will give tesr1.txt, text1.txt but not tast.txt
    -

```
[akshara@lnxsrv09 ~/sophomore]$ touch cs1.txt
[akshara@lnxsrv09 ~/sophomore]$ touch cs2.txt
[akshara@lnxsrv09 ~/sophomore]$ touch ct1.txt
[akshara@lnxsrv09 ~/sophomore]$ touch mycs.txt
[akshara@lnxsrv09 ~/sophomore]$ find . -name cs*
find: paths must precede expression: cs2.txt
Usage: find [-H] [-L] [-P] [-Olevel] [-D
help|tree|search|stat|rates|opt|exec] [path...] [expression]
[akshara@lnxsrv09 ~/sophomore]$ ls
bar.txt  cs1.txt  cs2.txt  ct1.txt  file.txt  mycs.txt
[akshara@lnxsrv09 ~/sophomore]$ find -name "cs*"
./cs1.txt
./cs2.txt
[akshara@lnxsrv09 ~/sophomore]$ find -name "*cs*"
./cs1.txt
./cs2.txt
./mycs.txt
[akshara@lnxsrv09 ~/sophomore]$ find -name "c[st]1"
[akshara@lnxsrv09 ~/sophomore]$ find -name "c[st]1*"
./cs1.txt
./ct1.txt
```

Process: ps and kill

- Process
    - An instance of a computer program in execution

- ps
    - List currently running processes
- kill
    - Terminate a certain proccess
    - Usage
        - Kill PID

diff

- File comparison utility that outputs the differences between two files
- Shows the changes between one version of a
    -

```
[akshara@lnxsrv09 ~/sophomore]$ diff file.txt bar.txt
1c1
< hello
---
> hello hi hi
```

- Line by line comparison

cmp

-

```
[akshara@lnxsrv09 ~/sophomore]$ cmp file.txt bar.txt
file.txt bar.txt differ: byte 6, line 1
```

- Compares bytewise

wget

- Downloads a file from a URL (a web server)

wh… Commands

- whatis <command>:
- whereis <command>: gives where the executable and where the manual file is
- which <command>: gives only where the executable file is

Emacs	Vim

• Eggert maintains this lmao	• vi(m)
	• Modes
	  ○ Normal: enter commands
	  ○ Insert: insert text (type i)
	  ○ Replace mode: Escape and then r. Works like the insert

	key on windows laptops. ● Esc :wq - save and exist ● Esc :w - save ● Esc :q - quit

Week 2
Shell scripting

A locale
- Set of parameters that define a user's cultural preferences
    - Language
    - Country
    - Other area-specific things

locale command
Prints info about current locale environment to std output

LC_*Environment Variables
- Locale gets data from the LC_*environment variables
- Eg:
    - LC_TIME: date and time format
    - LC_NUMERIC: non-monetary numeric format

Environment Variables
- Variables that can be accessed from any child process
- Eg:
    - HOME: path to user's home directory
    - PATH: list of directories to search in for command to execute
        - PATH will contain /usr/bin for find during hw
    - USER: the current username
    - SHELL: name of current shell
- Print value using echo: echo $VARIABLE
- Change value: export VARIABLE=...

Locale Settings
- Default order for sort depends on:
    - LC_COLLATE='C':sorting is in ASCII order (Unix collate)
    - LC_COLLATE='en_US':sorting is case insensitive except when 2 strings are otherwise equal and one has a capital letter earlier than the other

- Other locales have other sort orders

sort

- Sorts lines of text files
- sort [option] [file]
- Depends on locale
- C: sorts by ASCII value
- -u removes repeated values and gives unique lines

```
[akshara@lnxsrv07 ~]$ cat eg1
Hello World
I am writing my first file
Hello World
This is done using emacs editor
We are working on assignment 2
[akshara@lnxsrv07 ~]$ sort eg1
Hello World
Hello World
I am writing my first file
This is done using emacs editor
We are working on assignment 2
[akshara@lnxsrv07 ~]$ sort eg1 -u    //sort -u eg1
Hello World
I am writing my first file
This is done using emacs editor
We are working on assignment 2
```

comm:

- Compare 2 sorted files line by line prints to STDOUT what common lnes and unique lines to each file
- Comm [option] file1 file2
- Depends on locale

tr

- Translate or delete characters
- tr [option] SET1 [SET2]
- Character to character translation
- Eg:

```
[akshara@lnxsrv07 ~]$ tr ab cd
ucla
uclc
haaaahaaaab
Hcccchccccd
```

- Can translate using ranges too
-

```
[akshara@lnxsrv07 ~]$ tr a-z A-Z
```

```
ahHHHaaaabbB
AHHHHAAAABBB
```

- To put translated output in a file and translate a preexisting file
-

```
[akshara@lnxsrv07 ~]$ tr a-z A-Z < example.txt >tr_output.txt
```

- You can delete characters too
    -

    ```
    tr -d o < example.txt
    ```

    - Can also delete a range of characters (a-h)
    - Can't delete a string of characters (ab)

The Shell and OS

- The shell is a user interface to the OS
- Accept commands as text, interprets them, uses OS API to carry out what the user wants - open files, start programs
- Common shells: bash, csh, ash

Scripting

- When a shell script is executed a new child "shell" process is spawned to run it
- The first line is used to state which child shell to use
- #! : shebang
- #! /bin/sh
- #! /bin/bash
- 2 ways to run a script
    - bash script1.sh
    - Make script an executable
- Eg:
    -

    ```
    #! /bin/sh
    echo "Hello World"
    ```

    -

    ```
    #! /bin/sh
    echo "Hello World"
    read -p"Enter your name" name //-p allows you t
    echo "Welcome, $name"
    exit 0
    //in terminal
    [akshara@lnxsrv07 ~/sophomore]$ bash script1.sh
    Hello World
    Enter your nameAks
    Welcome, Aks
    ```

- Highly encouraged to have shell location after shebang

- Comments in scripting start with #

Compiled vs Interpreted

Compiled	Interpreted
• Programs are translated from the original source code into machine code that is executed by the hardware • Efficient and fast • Require recompiling • Work at low level, dealing with bytes, inters, float etc. • Eg: C/C++	• Reads commands, carries out actions commanded as it goes • Much slower • More portable • High level, easier to learn • Eg: PHP, Ruby, bash

Debugging tip: Execution tracing

- Shell prints out each line as it is executed
    - set -x:turn it on
    - set +x: turn it off
-

```
#! /bin/sh
set -x
echo "Hello World"
read -p "Enter your name" name
echo "Welcome, $name"
exit 0
[akshara@lnxsrv07 ~/sophomore]$ bash script1.sh
+ echo 'Hello World'
Hello World
+ read -p 'Enter your name' name
Enter your nameAKsha
+ echo 'Welcome, AKsha'
Welcome, AKsha
+ exit 0
```

Output using echo/printf

- echo:
    - Writes arguments to stdout, can't output escape char (without -e)
- printf
    - http://www.informit.com/articles/article.aspx?p=169489&seqNum=2

Variables

- Eg: var="hello" #NO SPACES
- Referenced using $

- Built in ones:
    - # - number of args
    - 0 - name of the script itself
    - 1 - first arg
    - ? - exit status of previous command
    - IFS - internal field separator, between words, usually set to space, tab and newline
- Eg:
-

```
#! /bin/sh
set -x
echo "Hello World"
read -p "Enter your name" name
echo "Welcome, $name"
echo "Number of args = $#"
echo "First $0"
echo "Second $1"
exit 0
[akshara@lnxsrv07 ~/sophomore]$ bash script1.sh John
+ echo 'Hello World'
Hello World
+ read -p 'Enter your name' name
Enter your nameabc
+ echo 'Welcome, abc'
Welcome, abc
+ echo 'Number of args = 1'
Number of args = 1
+ echo 'First script1.sh'
First script1.sh
+ echo 'Second #1'
Second #1
+ exit 0
[akshara@lnxsrv07 ~/sophomore]$ bash script1.sh
+ echo 'Hello World'
Hello World
+ read -p 'Enter your name' name
Enter your nameabc
+ echo 'Welcome, abc'
Welcome, abc
+ echo 'Number of args = 0'
Number of args = 0
+ echo 'First script1.sh'
First script1.sh
+ echo 'Second #1'
Second #1
+ exit 0
```

-

```
[akshara@lnxsrv07 ~/sophomore]$ cp script1.sh aaa.s
[akshara@lnxsrv07 ~/sophomore]$ echo $?
0
[akshara@lnxsrv07 ~/sophomore]$ cp aaaaaaaaa aaaaaa
cp: cannot stat 'aaaaaaaaa': No such file or directory
[akshara@lnxsrv07 ~/sophomore]$ echo $?
1
```

-
- You can have conditions in scripts
    - Use the test command or []
    - “man test” to see the expressions that can be done
    -

```
#! /bin/bash
if [5 -gt 1]
then
echo "5 greater than 1"
else
echo "not possible"
fi
```

    - test -f <stuff>
        - Checks if file exists

Quotes in scripting

- 3 kinds
    - Single quotes ‘’
        - Don’t expand at all, literal meaning
            - temp=’$hello$hello’;echo $temp
    - Double quotes “”
        - Like single quotes but expand backticks and $
    - Backticks ` ` or $()
        - Expand shell commands
            - temp=`ls`; echo $temp
-

```
[akshara@lnxsrv07 ~/sophomore]$ temp='$hello$hello';echo $temp
''
[akshara@lnxsrv07 ~/sophomore]$ temp=`ls`; echo $temp
aaa.s bar.txt bar.txt~ file.txt #script1# script1.sh script1.sh~ script2.sh
vimfile1.txt
```

-

```
#! /bin/bash
echo 'Using single quotes'
echo "Using double quotes 'ls'"

[akshara@lnxsrv07 ~/sophomore]$ chmod +x script3.sh
[akshara@lnxsrv07 ~/sophomore]$ emacs script3.sh
[akshara@lnxsrv07 ~/sophomore]$ ./script3.sh
Using single quotes
Using double quotes 'ls'
```

Loops

- While
    - while

```
#!/bin/sh
```

```
COUNT=6
while[$COUNT -gt 0]
do
echo "Value of count is $COUNT"
let COUNT=COUNT-1
done
```

- “let" command is used to do arithmatic
- F refers to each word in ls output
-

```
#!/bin/sh
temp='ls'
for f in $temp
do
echo $f
```

- For
  - f will refer to each word in ls output
  -

```
#! /bin/bash
temp=`ls`
for f in $temp
do
echo $f
done
```

  - Result of script

```
[akshara@lnxsrv07 ~/sophomore]$ chmod +x script4.sh
[akshara@lnxsrv07 ~/sophomore]$ ./script4.sh
aaa.s
bar.txt
bar.txt~
file.txt
#script1#
script1.sh
script1.sh~
script2.sh
script3.sh
script3.sh~
script4.sh
vimfile1.txt
```

Regular Expressions

- Lets you search for text with a particular pattern
- Two versions
  - Basic regular expressions
    - Default in sed and grep
  - Extended regular expressions
    - Use -r or -e to enable ERE
- regexpal.com can be used to test your regex
-

```
[akshara@lnxsrv07 ~/sophomore]$ grep Hello eg.txt
hello Hello world
[akshara@lnxsrv07 ~/sophomore]$ grep m eg.txt
I am writing my first file
Working on assignment 2
[akshara@lnxsrv07 ~/sophomore]$ grep [sk]i eg.txt
Working on assignment 2
[akshara@lnxsrv07 ~/sophomore]$ emacs eg2.txt
[akshara@lnxsrv07 ~/sophomore]$ grep c eg2.txt
cat
cbt
cot
cut
[akshara@lnxsrv07 ~/sophomore]$ grep c[ao] eg2.txt
cat
cot
[akshara@lnxsrv07 ~/sophomore]$ grep c[abou]t eg2.txt
cat
cbt
cot
cut
[akshara@lnxsrv07 ~/sophomore]$ grep c.t eg2.txt
cat
cbt
cot
cut
[akshara@lnxsrv07 ~/sophomore]$ grep c*t eg2.txt
cat
cbt
cot
cut
[akshara@lnxsrv07 ~/sophomore]$ grep -E c[a-z]t eg2.txt
cat
cbt
cot
cut
[akshara@lnxsrv07 ~/sophomore]$ grep -E c[a-c]t eg2.txt
cat
cbt
```

-

```
[akshara@lnxsrv07 ~/sophomore]$ emacs ex.txt
[akshara@lnxsrv07 ~/sophomore]$ grep c..t ex.txt
cut caot
[akshara@lnxsrv07 ~/sophomore]$ grep c*t ex.txt
I Love my cat
cbt
cat is naughty
cot
I have two cats who live with me
cut caot
[akshara@lnxsrv07 ~/sophomore]$ grep .+ca.+ ex.txt //more than one char before and
//after ca
```

- $ means the pattern has to be at the end of the string
- ^ means the pattern has to be in the beginning of the string
- . means match any single character except NULL

- \{n,m\} is an interval expression that matches a range of occurrences of a single character that immediately precedes it
- grep b[^u]t means that the letter in between can be anything except u
- Similarly, b[^ua]t means that the letter in between can be anything except u or a

who

- man -a grep
    - all greps, shows first one first, when you exit it shows the next man page
- man -e 5 grep
    - man grep from manual 5

sed

- Stream editor for filtering and transforming text
- Reads and transforms file line by line
- sed [options] commands [file-to-edit]
- 3 major commands:
    - p: print
        -

```
[akshara@lnxsrv07 ~/sophomore]$ sed 'p' e.txt
I have two cats. I love my cat.
I have two cats. I love my cat.
We are trying a new command.
We are trying a new command.
I live in 1310 street name.
I live in 1310 street name.
I am learning about a new command.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ sed -n 'p' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in 1310 street name.
I am learning about a new command.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ sed -n '1p' e.txt
I have two cats. I love my cat.
[akshara@lnxsrv07 ~/sophomore]$ sed -n '3p' e.txt
I live in 1310 street name.
[akshara@lnxsrv07 ~/sophomore]$ sed -n '1,3p' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in 1310 street name.
[akshara@lnxsrv07 ~/sophomore]$ sed -n '2-4p' e.txt
sed: -e expression #1, char 2: unknown command: `-'
[akshara@lnxsrv07 ~/sophomore]$ sed -n '2,4p' e.txt
We are trying a new command.
I live in 1310 street name.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ sed -n '1~2,4p' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in 1310 street name.
I am learning about a new command.
```

```
[akshara@lnxsrv07 ~/sophomore]$ sed -n '1~2p' e.txt
I have two cats. I love my cat.
I live in 1310 street name.
[akshara@lnxsrv07 ~/sophomore]$ sed -n '1~2p' e.txt | sed '1,2p'
I have two cats. I love my cat.
I have two cats. I love my cat.
I live in 1310 street name.
I live in 1310 street name.
[akshara@lnxsrv07 ~/sophomore]$ sed -n '1~2p' e.txt | sed -n '1,2p'
I have two cats. I love my cat.
I live in 1310 street name.
```

- Prints twice by default because it prints input line then output line
- -n removes the repetition
- Use commas to print a range of lines
- Use | so you can print multiple lines not together in a range.
- Print between two patterns
  - sed -n '/start,/end/p' filename.txt

- d: delete
  -

```
[akshara@lnxsrv07 ~/sophomore]$ sed '2d' e.txt
I have two cats. I love my cat.
I live in 1310 street name.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ sed '1~2d' e.txt
We are trying a new command.
I am learning about a new command.
```

  - You can specify line numbers by preceding d with a number
  - d deletes the entire line.
  - To delete everything between two patterns
    - sed '/start/,/end/d' filename.txt
- s: substitute
  - sed /s/oldword/newword/
  - Example:

```
[akshara@lnxsrv07 ~/sophomore]$ echo 'www.ucla.edu' | sed
's/ucla/usc/'
www.usc.edu
[akshara@lnxsrv07 ~/sophomore]$ sed 's/cat/dog/' e.txt
I have two dogs. I love my cat.
We are trying a new command.
I live in 1310 street name.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ sed 's/cat/dog/g' e.txt
I have two dogs. I love my dog.
We are trying a new command.
I live in 1310 street name.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ cp e.txt f.txt
[akshara@lnxsrv07 ~/sophomore]$ sed -i 's/cat/dog/' f.txt
[akshara@lnxsrv07 ~/sophomore]$ cat f.txt
I have two dogs. I love my cat.
We are trying a new command.
```

```
I live in 1310 street name.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ sed -i 's/cat/dog/g' f.txt
[akshara@lnxsrv07 ~/sophomore]$ cat f.txt
I have two dogs. I love my dog.
We are trying a new command.
I live in 1310 street name.
I am learning about a new command.
[akshara@lnxsrv07 ~/sophomore]$ sed 's/cat[. ]/dog/g' e.txt
I have two cats. I love my dog
We are trying a new command.
I live in 1310 street name.
I am learning about a new command.
```

- Only replaced the first occurrence of old word on each line by default.
- Need to add ‘s/old/new/g’ or global to replace each occurrence
- sed -i means in place transformation. Transform input file in place.
- add [.] ensures that cats isn’t replaced because it will replace either ‘cat ‘ or ‘cat.’
- The period after dog in the case with cat [.] disappears, since the entirety of ‘cat.’ becomes the expression replaced.
-

```
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[0-9]/NA/g' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in NANANANA street name, Los Angeles.
CSNANAL is a lot of work.
final line, bye.
I have NANA apples.
Yesterday, I ate NA bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[0-9]*/NA/g' e.txt
NAINA NAhNAaNAvNAeNA NAtNAwNAoNA NAcNAaNAtNAsNA.NA NAINA
NAlNAoNAvNAeNA NAmNAyNA NAcNAaNAtNA.NA
NAWNAeNA NAaNArNAeNA NAtNArNAyNAiNAnNAgNA NAaNA NAnNAeNAwNA
NAcNAoNAmNAmNAaNAnNAdNA.NA
NAINA NAlNAiNAvNAeNA NAiNAnNA NA NAsNAtNArNAeNAeNAtNA
NAnNAaNAmNAeNA,NA NALNAoNAsNA NAANAnNAgNAeNAlNAeNAsNA.NA
NACNASNALNA NAiNAsNA NAaNA NAlNAoNAtNA NAoNAfNA NAwNAoNArNAkNA.NA
NAfNAiNAnNAaNAlNA NAlNAiNAnNAeNA,NA NAbNAyNAeNA.NA
NAINA NAhNAaNAvNAeNA NA NAaNApNApNAlNAeNAsNA.NA
NAYNAeNAsNAtNAeNArNAdNAaNAyNA,NA NAINA NAaNAtNAeNA NA
NAbNAaNAnNAaNAnNAaNAsNA.NA
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[0-9][0-9]*/NA/g' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in NA street name, Los Angeles.
CSNAL is a lot of work.
final line, bye.
I have NA apples.
Yesterday, I ate NA bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[^A-Za-z][0-9][0-9]*/NA/g'
e.txt
I have two cats. I love my cat.
We are trying a new command.
I live inNA street name, Los Angeles.
CSNAL is a lot of work.
final line, bye.
```

```
I haveNA apples.
Yesterday, I ateNA bananas.
```

- [0-9]* will replace every other letter. At least one digit, then rest. Repeat.
- [0-9][0-9]*
- Replace all numbers surrounded by spaces that aren't course names
    - ([0-9][0-9]*)
    - ([0-9][0-9]*[ ])

```
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[0-9][0-9]*/NA/g' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in NA street name, Los Angeles.
CSNAL is a lot of work.
final line, bye.
I have NA apples.
Yesterday, I ate NA bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[^A-Za-z][0-9][0-9]*/NA/g'
e.txt
I have two cats. I love my cat.
We are trying a new command.
I live inNA street name, Los Angeles.
CSNAL is a lot of work.
final line, bye.
I haveNA apples.
Yesterday, I ateNA bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[^s][0-9][0-9]*/NA/g' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live inNA street name, Los Angeles.
CNAL is a lot of work.
final line, bye.
I haveNA apples.
Yesterday, I ateNA bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed 's/[^s][^A-Z][0-9][0-9]*/NA/g'
e.txt
I have two cats. I love my cat.
We are trying a new command.
I live iNA street name, Los Angeles.
CNAL is a lot of work.
final line, bye.
I havNA apples.
Yesterday, I atNA bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed -E 's/[^s][^A-Z][0-9][0-9]*/NA/g'
e.txt
I have two cats. I love my cat.
We are trying a new command.
I live iNA street name, Los Angeles.
CNAL is a lot of work.
final line, bye.
I havNA apples.
Yesterday, I atNA bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed -E 's/([0-9][0-9]* )/NA /g' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in NA street name, Los Angeles.
CS35L is a lot of work.
final line, bye.
I have NA apples.
```

```
Yesterday, I ate NA bananas.
```

- To include what you have extracted as part of the substitute use &:
-

```
[akshara@lnxsrv07 ~/sophomore]$ sed -E 's/([0-9][0-9]* )/(&) /g' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in (1310 ) street name, Los Angeles.
CS35L is a lot of work.
final line, bye.
I have (33 ) apples.
Yesterday, I ate (4 ) bananas.
[akshara@lnxsrv07 ~/sophomore]$ sed -E 's/([0-9][0-9]* )/(&)/g' e.txt
I have two cats. I love my cat.
We are trying a new command.
I live in (1310 )street name, Los Angeles.
CS35L is a lot of work.
final line, bye.
I have (33 )apples.
Yesterday, I ate (4 )bananas.
```

Week 3

How to Install Software

- Windows
  - Installshield
  - Microsoft/Windows Installer
  - .exe file
- Mac
  - .DMG
  - Linux way
- Linux
  - rpm (Redhat Package Management)
    - RedHat Linux (.rpm)
  - apt-get (Advanced Package Tool)
    - Debian Linux, Ubuntu Linux (.deb)
- All systems have files with binary formats for installing. Except Linux
- Build process:
  - Might do this to download from source if it is not downloading/installing correctly with the file
  - Best way to get latest version of software
  - configure, make, make install

Decompressing Files

- Linux software in tarbell format (.tgz) or (.gz)
- Decompress file in current directory
  - tar -xzvf filename.tar.gz

        - x: --extract
        - z: --gzip
        - v: --verbose
        - f: --file
- Git created by the dude who made Linux
    - Used if you want to download a large number of files/software that is the latest version
    - SVN is similar to Git
- Steps to execute software
    - Download source code
    - Compile
        - Want to recompile only files that have been modified to save time

Command line Compilation

- give g++ the names of files for compilation for C++
- gcc for C
- .h - header file
- .c - definition file
- shop.c
    - #includes shoppingList.h and item.h
- shoppingList.c
    - #includes shoppingList.h
- item.c
    - #includes item.h
- To compile: g++ shop.c shoppingList.c item.c -o shop
    - By default output file called a.out
    - #include ensures that the header is added automatically to the beginning since it is a preprocessor directive
- If you make only a small change to item.c, don't recompile everything
    - Solution: produce a separate object code file for each source file
        - gcc -c item.c
        - gcc -c shoppingList.c
        - gcc -c shop.c
        - gcc item.o shoppingList.o shop.o -o shop (combine)
        - Less work for compiler, but more commands
    - -c: only compile, doesn't link
    - Issues with this solution:
        - Hard to keep track when large number of input files
- If you change one of the headers or source files, rerun command to generate a new executable if large change

Make

- utility for managing large software projects

- compiles files and keeps them up to date
- efficient compilation

Makefile Example

```
#Makefile - A Basic Example
all: shop #usually first, all, item.o, shoppingList.o etc are labels
item.o: item.c item.h                          #rule
        gcc -c item.c                          #rule continued
shoppingList.o: shoppingList.c item.h shoppingList.h
                        gcc -c shoppingList.c
shop.o: shop.c item.h shoppingList.h
        gcc -c shop.c
shop: item.o shoppingList.o shop.o #executable being made
clean:
     rm -f item.o shoppingList.o shop.o shop
```

Makefile (recommended)

```
#comment
rulename: dependencies
tob gcc -c item.c -o item.o (commands)
shoppingList.o: item.o shoppingList.c shoppingList.h
            gcc -c shoppingList.c item.o -o shoppingList.o
-------
-------
```

- Runs hello first by default

```
hello:
        echo "Hello World"
bye:
        echo "bye bye"
#these are dummy rules
```

-

```
hello:
        bye #run bye first
        echo "Hello World"
bye:
        echo "bye bye"
#these are dummy rules
```

-

```
item.o: item.c item.h
            gcc -c item.c -o item
bye:
         echo "bye bye"
```

- Order doesn't matter, make makes first one by default
- Dependency order matters

Lab 3

- Coreutils 7.6 has a problem
    - Different users see different date formats

    - ls -l /bin/bash
        - -rwxr-xr-x 1 root root 729040 2009-03-02 06:22 /bin/bash
        - -rwxr-xr-x 1 root root 729040 Mar 2 2009 /bin/bash
    - Go into coreutils-7.6 directory
    - Read the INSTALL file on how to configure the project, especially with the --prefix flag
        - Run the configure script using the prefix flag so that when everything is done, coreutils will be installed in the directory ~/coreutilsInstall
    - Compile it: make
    - Install it:
- Reproduce Bug
- Patching
    - A patch is a piece of software designed to fix problems with or update a computer program
    - diff -u blah blah > something.patch to get the patch
- diff Unified Format
    - diff -u file1 file2
    - --- path/to/original/file
    - +++ path/to/modified/file
    - @@-l,s+l,s @@
        - @@: beginning of a chunk
        - l: beginning line number
        - s: number of lines the change chunk applies to for each file
        - A line with a:
            - - sign was deleted from the original
            - + sign was added to original
            - stayed the same
    -

```
[akshara@lnxsrv07 ~/sophomore]$ emacs hello1.txt
[akshara@lnxsrv07 ~/sophomore]$ cp hello1.txt  hello2.txt
[akshara@lnxsrv07 ~/sophomore]$ emacs hello2.txt
[akshara@lnxsrv07 ~/sophomore]$ diff hello1.txt  hello2.txt
[akshara@lnxsrv07 ~/sophomore]$ wmacs hello2.txt
-bash: wmacs: command not found
[akshara@lnxsrv07 ~/sophomore]$ emacs hello2.txt
[akshara@lnxsrv07 ~/sophomore]$ diff -u hello1.txt hello2.txt > my_patch
[akshara@lnxsrv07 ~/sophomore]$ emacs my_patch
[akshara@lnxsrv07 ~/sophomore]$ emacs hello2.txt
[akshara@lnxsrv07 ~/sophomore]$ patch hello2.txt < my_patch
patching file hello2.txt
Reversed (or previously applied) patch detected!  Assume -R? [n] y
[akshara@lnxsrv07 ~/sophomore]$ vim hello2.txt
[akshara@lnxsrv07 ~/sophomore]$ diff -u hello1.txt  hello2.txt
```

Random: ./configure prefix = add --prefix to configure, ~/coreutilsInstall, make, make install, tar -xzvf, after bug bash -p(number here)
Build Process

- configure
    - script that checks details about the machine before installation
        - dependency between packages

What is Python?

- High level programming language
- Like pseudocode
- Readable
- Classes and member functions
- Python compiled to bytecode
    - Bytecode interpreted by python interpreter
- Not as fast as C, but easy to learn, read and write
- Lot of useful libraries

Python data types

- Python isn't a strongly typed language

```
x=3
print(type(x)) //<class int>
print(x)
print(x**2) //9
print(x,x+1,x+2) //3 4 5
x="Hello"
```

- Doesn't have a ++ operator (use +=1)
- Booleans operations: True, False: and, or, not, !=

```
>>> print(type(x))
<type 'int'>
>>> print(x*2)
4
>>> print(x**4)
16
>>> print(3!=4)
True
```

- Strings
    - len
    - string concatenation
    - sprintf style string formatting
    - Formatted output:

```
>>> s="Hey"
>>> print(s.capitalize())
Hey
>>> print(s.upper())
HEY
>>> print(s.rjust(4))
 Hey
>>> print(s.center(4))
Hey
>>> print(s.replace('e','(ell)'))
```

```
H(ell)y
>>> print('    world ' .strip())
world
>>> print(s.center(5))
 Hey
```

    - strip deletes leading and trailing whitespace
    - capitalize capitalizes to first letter in a string
    - rjust right justifies, padding with spaces
    - center centers a string, padding with spaces
  - Can use either “” or ‘’ for string literals
  - None is the python equivalent of null

Types of Python

- Python 2: 2.X
- Python 3: 3.0
-

```
>>> x=2
>>> print(x) //both python 2 and 3
2
>>> print x
2
>>> //print x only allowed in python 2
```

Building a Python Script

-

```
Aksharas-MacBook-Pro:inClass akshara$ emacs hello.py
x="Hello"
print(x)
Aksharas-MacBook-Pro:inClass akshara$ python hello.py
Hello
#change hello.py
Add #! /location/to/python in the beginning
Aksharas-MacBook-Pro:inClass akshara$ chmod +x hello.py
Aksharas-MacBook-Pro:inClass akshara$ ./hello.py
```

Python containers

- Built in container types: lists, dictionaries, sets, tuples
- List: python equivalent of an array
  - resizable
  - can contain elements of different types
  - access elements with List_name[index]
  - has negative indexing

```
>>> t=[123,3.0,"hello!"]
>>> print t[0]
123
>>> print t[1]
```

```
3.0
>>> print t[2]
hello!
>>> print t[-1]
hello!
```

- List Operations

```
>>> list1 = [1,2,3,4]
>>> list2=[5,6,7,8]
>>> list1.append(5)
>>> print list1
[1, 2, 3, 4, 5]
>>> merged=list1+list2
>>> print merged
[1, 2, 3, 4, 5, 5, 6, 7, 8]
>>> list1.append(list2)
>>> print list1
[1, 2, 3, 4, 5, [5, 6, 7, 8]]
```

- Can access sublists

```
l = ['a',' b',' c',' d',' e'] c_index = l.index("c") l2 = l[:c_index
```

For loops

- Make sure to be consistent with spacing/tabs

```
>>> list1
[1, 2, 3, 4, 5, [5, 6, 7, 8]]
>>> for item in list1:
...     print item
...     print "----"
...
1
----
2
----
3
----
4
----
5
----
[5, 6, 7, 8]
----
```

Indentation

- Needs to be consistent
- Part of syntax

List Comprehension

- Easy way to transform elements of a list

```
>>> nums=[0,1,2,3,4]
>>> squares=[x**2 for x in nums]
>>> print squares
[0, 1, 4, 9, 16]
>>> even_squares=[x**2 for x in nums if x%2==0]
>>> print even_squares
[0, 4, 16]
```

Dictionaries

- Stores (key,value) pairs
- Can use key as an index
-

```
>>> d={'cat':'cute','dog':'furry'}
>>> print d[dog]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'dog' is not defined
>>> print d['dog']
furry
```

Assignment 3

- Understand what comm.py does
    - 1: Where's the python interpreter
    - 2: Imports functionality
    - 3: OptionParser, parse command line arguments
        - Should be able to run with blah.py -n=3 file or blah.py file -n=3
        - Elements of sys.arg[1:]
    - def __init__ is a line defining the class's constructor
    - Open file, return a list of each line in the file
    - Returns a random element from the list
    - Note: doesn't have to be called main() in Python
    - Use OptionParser in main(),

Week 4

Debugger

- A program used to run and debug other target programs
- Advantages
    - step through source code line by line
    - interact with and inspect program at run time
    - if program crashes, it outputs where and why it crashed

Segmentation Fault

- Access part of memory that is not allowed
- Line number where it crashed and parameters to the function that caused the error

Logic Error

- Program will run and exit successfully

Using GDB

- Normally: gcc [flags] source -o output
- Debugging: gcc otherflags -g source -o output
    - enables built in debugging support
-

```
[akshara@lnxsrv07 ~/sophomore]$ emacs helloworld.cc
[akshara@lnxsrv07 ~/sophomore]$ g++ helloworld.cc -o hello1
[akshara@lnxsrv07 ~/sophomore]$ g++ -g helloworld.cc -o hello2
[akshara@lnxsrv07 ~/sophomore]$ ls -l hello1 hello2
-rwxr-xr-x 1 akshara csugrad  8968 Oct 23 10:26 hello1
-rwxr-xr-x 1 akshara csugrad 19552 Oct 23 10:26 hello2
[akshara@lnxsrv07 ~/sophomore]$ gdb hello2
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-100.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /w/home.14/cs/ugrad/akshara/sophomore/hello2...done.
(gdb) run
Starting program: /w/home.14/cs/ugrad/akshara/sophomore/hello2
Hello[Inferior 1 (process 45877) exited normally]
Missing separate debuginfos, use: debuginfo-install glibc-2.17-196.el7.x86_64
libgcc-4.8.5-16.el7.x86_64 libstdc++-4.8.5-16.el7.x86_64
```

- Use commands like help to find out more
    - Can also run stuff like help data for more on that

Breakpoints

- used to stop the running program at a specific point
- it will pause at the location and ask for another command
- (gdb) break file.c:6
- (gdb) break functionname
- (gdb) break [position] if expression
    - conditional break
- Use info breakpoints/break/b to find out where the breakpoints
- Deleting/Disabling/Ignoring breakpoints
    - delete bpnumber/range
    - disable bpnumber/range
    - enable bpnumber/range
        - restores disabled breakpoints

NOTE: Remember to compile with -g for it to work

Using GDB

-

```
[akshara@lnxsrv07 ~/sophomore]$ g++ testgdb.cc -o prog1
[akshara@lnxsrv07 ~/sophomore]$ g++ -g testgdb.cc -o prog2
[akshara@lnxsrv07 ~/sophomore]$ ./prog1
Enter x
5
y=120
[akshara@lnxsrv07 ~/sophomore]$ gdb prog1
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-100.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /w/home.14/cs/ugrad/akshara/sophomore/prog1...(no debugging
symbols found)...done.
(gdb) r
Starting program: /w/home.14/cs/ugrad/akshara/sophomore/prog1
Enter x
2
y=2
[Inferior 1 (process 47025) exited normally]
Missing separate debuginfos, use: debuginfo-install glibc-2.17-196.el7.x86_64
libgcc-4.8.5-16.el7.x86_64 libstdc++-4.8.5-16.el7.x86_64
(gdb) list
No symbol table is loaded.  Use the "file" command.
(gdb) q
[akshara@lnxsrv07 ~/sophomore]$ gdb prog2
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-100.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /w/home.14/cs/ugrad/akshara/sophomore/prog2...done.
(gdb) r
Starting program: /w/home.14/cs/ugrad/akshara/sophomore/prog2
Enter x
2
y=2
[Inferior 1 (process 47054) exited normally]
Missing separate debuginfos, use: debuginfo-install glibc-2.17-196.el7.x86_64
libgcc-4.8.5-16.el7.x86_64 libstdc++-4.8.5-16.el7.x86_64
(gdb) q
[akshara@lnxsrv07 ~/sophomore]$ gdb prog2
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-100.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
```

```
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /w/home.14/cs/ugrad/akshara/sophomore/prog2...done.
(gdb) break main
Breakpoint 1 at 0x400986: file testgdb.cc, line 11.
(gdb) r
Starting program: /w/home.14/cs/ugrad/akshara/sophomore/prog2

Breakpoint 1, main () at testgdb.cc:11
11      cout<<"Enter x"<<endl;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-196.el7.x86_64
libgcc-4.8.5-16.el7.x86_64 libstdc++-4.8.5-16.el7.x86_64
(gdb) list
6       return result;
7       }
8       int main()
9       {
10      int x;
11      cout<<"Enter x"<<endl;
12      cin>>x;
13      int y=myfunc(x);
14      cout<<"y="<<y<<endl;
15      return 0;
(gdb) print x
$1 = 0
(gdb) print $rip
$2 = (void (*)(void)) 0x400986 <main()+8>
$2 = (void (*)(void)) 0x400986 <main()+8>
(gdb) step
Enter x
12      cin>>x;
(gdb) s
5
13      int y=myfunc(x);
(gdb) print x
$3 = 5
(gdb) continue
Continuing.
y=120
[Inferior 1 (process 47184) exited normally]
(gdb) q
```

-

```
[akshara@lnxsrv07 ~/sophomore]$ gdb prog2
GNU gdb (GDB) Red Hat Enterprise Linux 7.6.1-100.el7
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /w/home.14/cs/ugrad/akshara/sophomore/prog2...done.
(gdb) break main
Breakpoint 1 at 0x400986: file testgdb.cc, line 11.
```

```
(gdb) break myfunc
Breakpoint 2 at 0x400958: file testgdb.cc, line 4.
(gdb) info b
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400986 in main() at testgdb.cc:11
2       breakpoint     keep y   0x0000000000400958 in myfunc(int) at testgdb.cc:4
(gdb) ignore 2 2
Will ignore next 2 crossings of breakpoint 2.
(gdb) r
Starting program: /w/home.14/cs/ugrad/akshara/sophomore/prog2

Breakpoint 1, main () at testgdb.cc:11
11      cout<<"Enter x"<<endl;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-196.el7.x86_64
libgcc-4.8.5-16.el7.x86_64 libstdc++-4.8.5-16.el7.x86_64
(gdb) s
Enter x
12      cin>>x;
(gdb) s
4
13      int y=myfunc(x);
(gdb) print x
$1 = 4
(gdb) continue
Continuing.

Breakpoint 2, myfunc (x=2) at testgdb.cc:4
4       if (x==0) return 1;
(gdb) read
Undefined command: "read".  Try "help".
(gdb) print x
$2 = 2
(gdb) bt
#0  myfunc (x=2) at testgdb.cc:4
#1  0x0000000000400972 in myfunc (x=3) at testgdb.cc:5
#2  0x0000000000400972 in myfunc (x=4) at testgdb.cc:5
#3  0x00000000004009bd in main () at testgdb.cc:13
```

- Resuming execution after a break
    - c or continue
    - n or next
        - next instruction
    - s or step
        - next line
    - f or finish
        - resume execution until the current function returns

Watchpoints

- Watch changes to variables
- (gdb) watch my_var
    - debugger stops program when value of my_var changes

    - old and new values will be printed

Process Memory Layout

Arg and env vars

Stack: stores local vars, grows down as a result of a newly called function
  |
  V

Unused memory
^ | Heap: grows up when dynamic memory is requested by C's malloc or C++'s new
Uninitialized Data Segment (bss): uninitialized global vars
Initialized Data Segment
Text Segment

Stack

- a program is made of one or more functions which interact by calling each other
- for each function call, an area of memory is set aside, called a stack frame
- it holds
    - storage space for all the local vars
    - memory address to return to when function is complete

Analyze the Stack

- Use backtrace
    - Shows the call stack
    - Without function calls:
        - #0 main() at program.c:10
        - one frame on stack, numbered 0, line 10
- Use info locals to find values of local vars

Useful function

- info functions
    - list of all functions in the program
- list
    - lines around %rip

```
(gdb) b main
Breakpoint 1 at 0x400986: file testgdb.cc, line 11.
(gdb) b myfunc
Breakpoint 2 at 0x400958: file testgdb.cc, line 4.
(gdb) info b
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400986 in main() at testgdb.cc:11
2       breakpoint     keep y   0x0000000000400958 in myfunc(int) at testgdb.cc:4
(gdb) list
2       using namespace std;
3       int myfunc(int x){
4       if (x==0) return 1;
5       int result = x* myfunc(x-1);
6       return result;
7       }
8       int main()
9       {
10      int x;
11      cout<<"Enter x"<<endl;
(gdb) bt
No stack.
(gdb) r
Starting program: /w/home.14/cs/ugrad/akshara/sophomore/prog2

Breakpoint 1, main () at testgdb.cc:11
11      cout<<"Enter x"<<endl;
Missing separate debuginfos, use: debuginfo-install glibc-2.17-196.el7.x86_64
libgcc-4.8.5-16.el7.x86_64 libstdc++-4.8.5-16.el7.x86_64
(gdb) bt
#0  main () at testgdb.cc:11
(gdb) list
6       return result;
7       }
8       int main()
9       {
10      int x;
11      cout<<"Enter x"<<endl;
12      cin>>x;
13      int y=myfunc(x);
14      cout<<"y="<<y<<endl;
15      return 0;
```

Week 6

Multithreaded Performance

Multiprocessing

- Multiple tasks at the same time over different ports on the same machine
- A CPU can have more than one core

Parallelism

- Higher performance
    - Multitasking
        - Different process run on different CPU cores

    - Multithreading
        - One task has to achieve different things at the same time, f
        - Eg: A computer game has to render the pixels, play music and correspond to the inputs given

Thread

- A flow of instructions. path of execution within a process
- Smallest unit of processing scheduled by OS
- A process consists of at least one thread
    - Uniprocessor(time sharing)
        - Processor switches between different threads
        - Virtual parallelism ie concurrency
        - Only one thread runs at a given time
    - Multiprocessor
        - True parallelism
        - Everything runs at the same time
        - Process and each thread has a different address space, but same memory space

Multitasking

- Running tr sfksj | sort -u | comm -23 flj
- All three get started at the same time and initializes a communication pipe between each process
- No extra work to share data

Shared Memory

- Makes everything more powerful and efficient
- Non-trivial, need to deal with race conditions

Race Conditions

- Use synchronization to make sure you get the correct result
- Result depends on order of execution and when CPU switches between threads
- Simultaneous read and write

Lab

- Evaluate performance of a multithreaded program
- Generate a file with random data
    - /dev/urandom
    - head -c 80 /dev/urandom > outfile
    - To format as floating point
        - |od -tf
    - head -c 80 /dev/urandom | od -tf
    - Parse this using tr and sed, convert spaces to newlines etc

- Redirecting output to /dev/null means that we don’t care about the output and the output will get deleted