

ComputeCanada

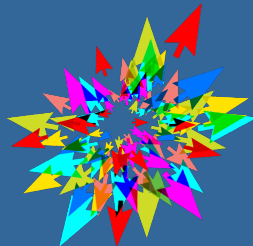
**Ontario Summer
School on HPC**

- LINUX/SHELL programming

Isaac Ye

HPTC @York University

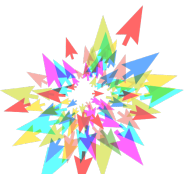
compute | calcul
canada | canada



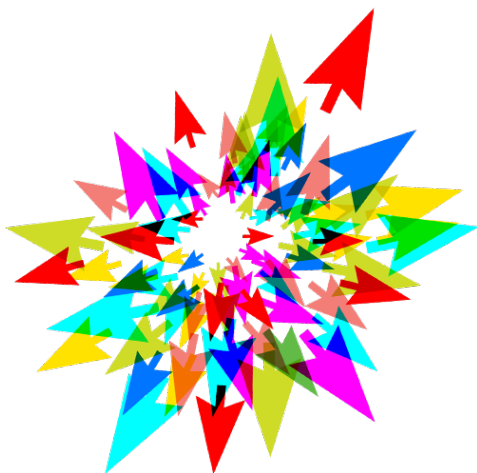
Overview

Session I (LINUX)

- 1) What/Why/Which LINUX ?
- 2) LINUX Basic
 - 1) User login/logout
 - 2) SHELL
 - 3) File system
 - 4) Process/Job
 - 5) Text editing
 - 6) Command



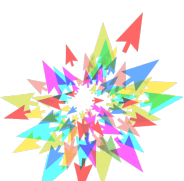
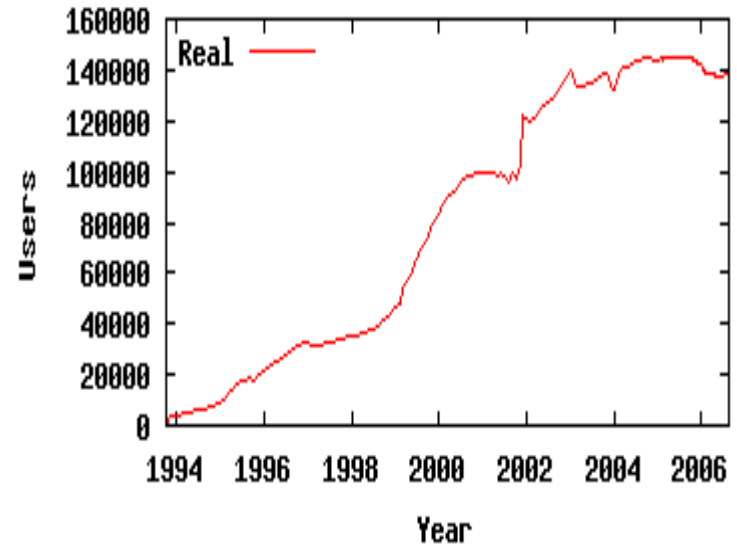
compute | **calcul**
canada | canada



What/Why/Which LINUX ?

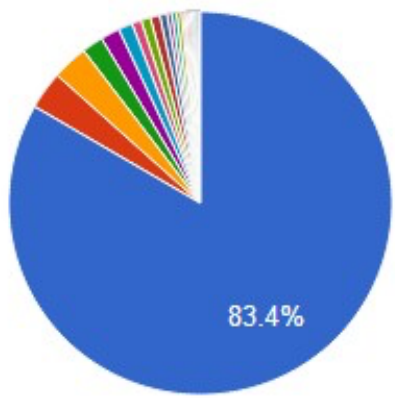
What is LINUX?

- History
 - A famous professor Andrew Tanenbaum developed **Minix**, a simplified version of UNIX that runs on PC
 - In Sept 1991, **Linus Torvalds** developed the preliminary kernel of Linux, known as Linux version 0.0.1
 - Recent (2015) estimates about 80M users in the world.
 - 95% of Top500 supercomputers running on Linux

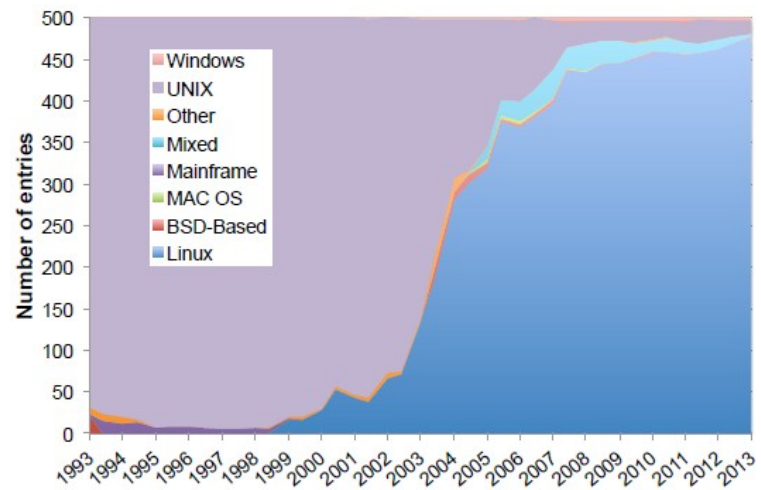
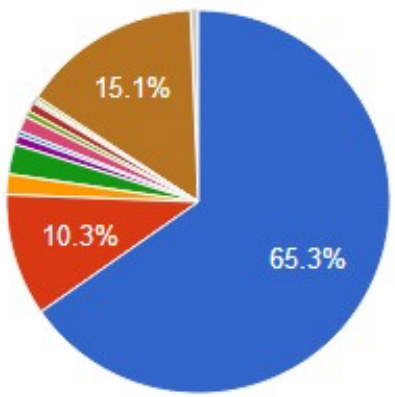


Operating System popularity

Operating System System Share



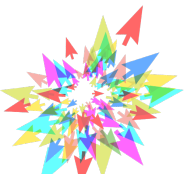
Operating System Performance Share



- Linux
- Cray Linux Environment
- AIX
- SUSE Linux Enterprise...
- CentOS
- SLES10 + SGI ProPac...
- bullx SuperComputer ...

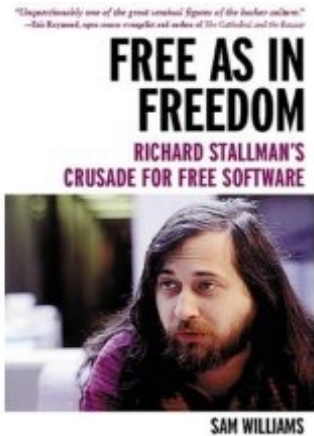


Linux



GNU project

- Established in 1984 by **Richard Stallman**, who believes that software should be free from restrictions against copying or modification in order to make better and efficient computer programs



GNU is a recursive acronym for “GNU's Not Unix”

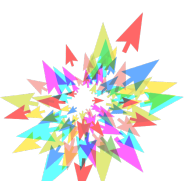
Aim at developing a complete Unix-like operating system which is free for copying and modification

Companies make their money by maintaining and distributing the software, e.g. optimally packaging the software with different tools (Redhat, Slackware, Mandrake, SuSE, etc)

Stallman built the first free GNU C Compiler in 1991.

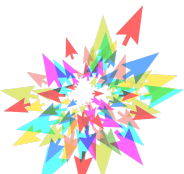
Why Linux?

- A fully-networked 32/64-Bit Unix-like OS
- Excellent system stability
- Unix tools and compilers
- Strong network tools and support
- Multi-user, Multitasking, Multiprocessor
- Has the network-based X Windows GUI
- Runs on multiple platforms(hardware)
- Plentiful software
- Includes the source code and documents
- FREE !!!



Which Linux

- Distributions
 - Red Hat Linux : One of the original Linux distribution.
 - The commercial, nonfree version: Red Hat Enterprise Linux, Free: Fedora Project.
 - Debian GNU/Linux : A free software distribution.
 - Popular for use on servers.
 - Hard for a beginner.
 - Ubuntu Linux: an immensely popular Debian-based distribution.
 - If you want to get up and running quickly and not fiddle around with the guts of the system as much, Ubuntu is better suited.
 - CentOS: an Enterprise-class Linux Distribution derived from sources freely provided to the public. (SHARCNET uses)
 - SuSE Linux : primarily available for pay because it contains many commercial programs, although there's a stripped-down free version that you can download.
 - Mandrake Linux : Mandrake is perhaps strongest on the desktop.
 - Gentoo Linux : Gentoo is a specialty distribution meant for programmers.

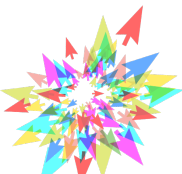


LINUX distribution popularity

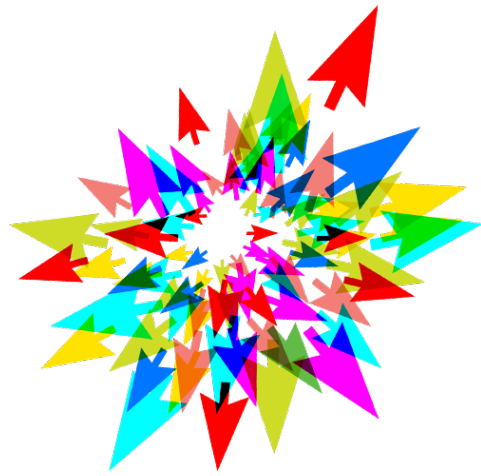
Linux Share Distributions



- Ubuntu 50.35%
- Fedora 5.76%
- Debian 2.24%
- CentOS 1.24%
- Mandriva 0.37%
- Unknown (combination) 34.83%
- SUSE 3.43%
- Red Hat 1.16%
- Mandrake 0.61%

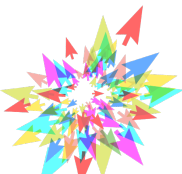
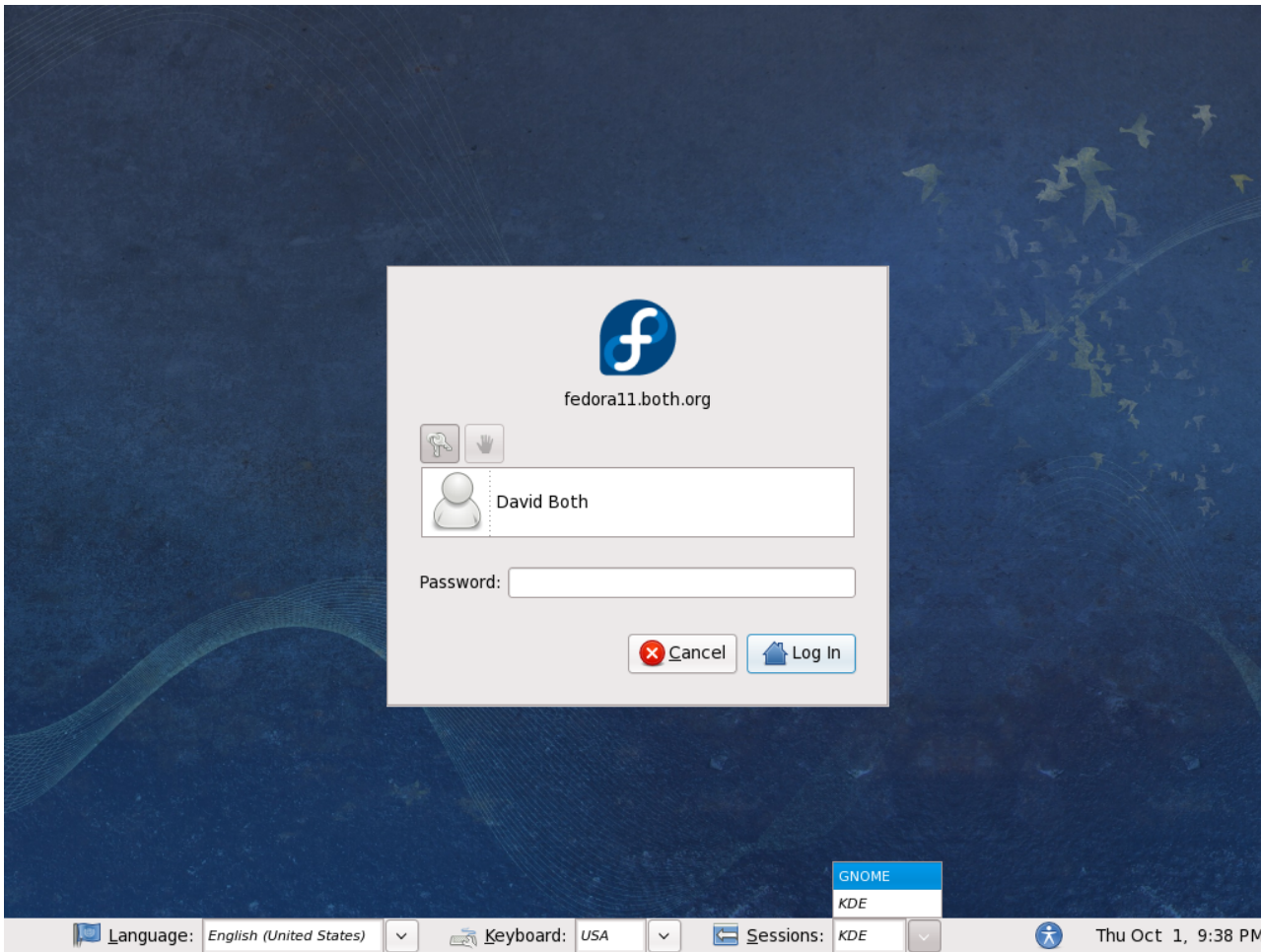


compute | **calcul**
canada | canada



LINUX Basic

Logging In/Out in Desktop

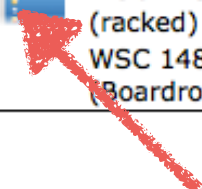


Linux Desktop Environment @SHARCNET

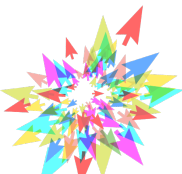
Visualization Systems

<https://www.sharcnet.ca/my/systems>

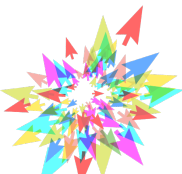
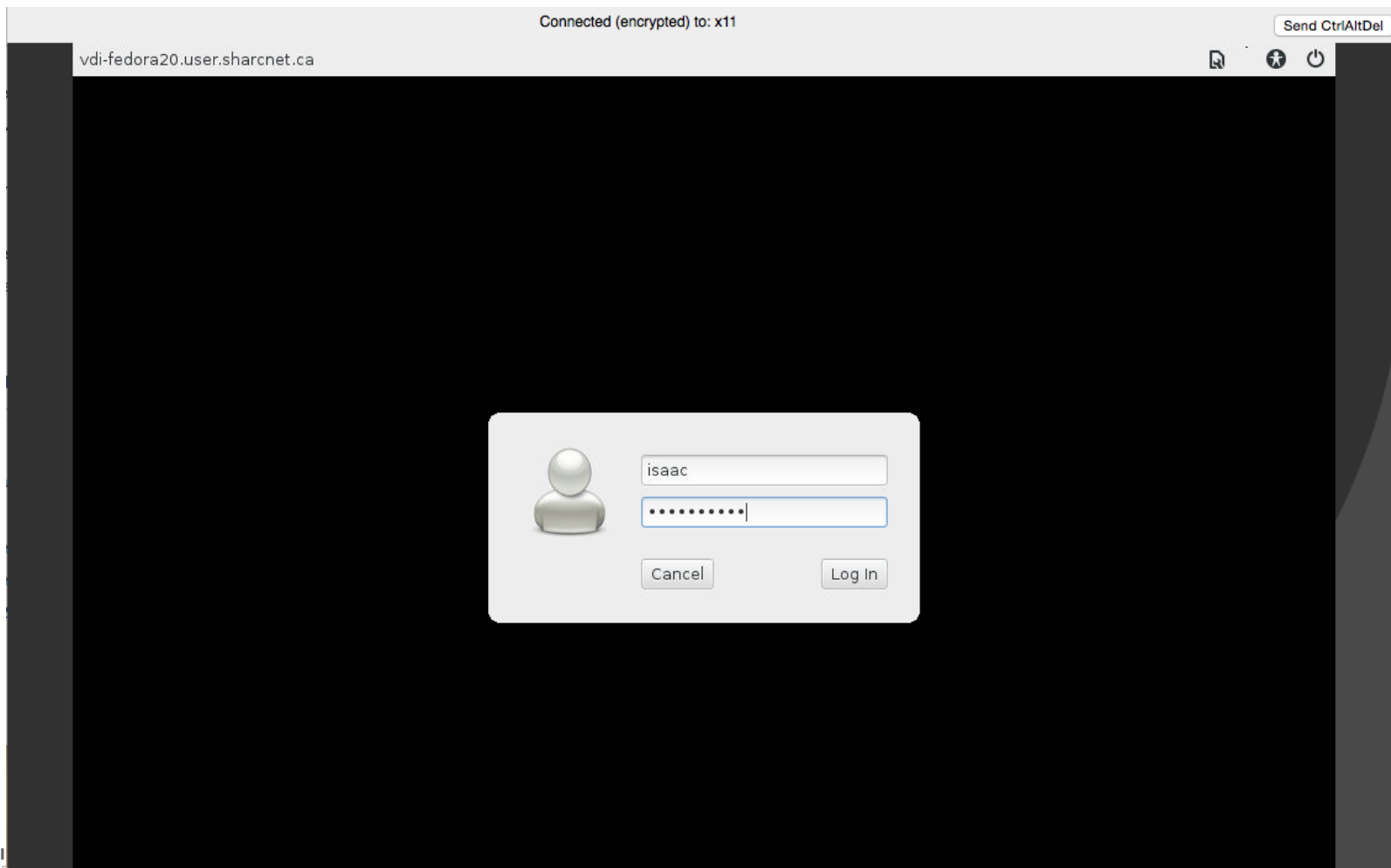
System	Room	State	Cores	Access	Memory	GPU	Notices
viz1-uwo	WSC 148 (Boardroom)	Conditions	12	Remote	48 GB	2 x GK104 GeForce GTX 670	02-Mar-2015
viz2-uwo	WSC 148 (Boardroom)	Conditions	12	Remote	48 GB	2 x GK104 GeForce GTX 670	02-Mar-2015
vdi-centos6	WSC 148 (racked)	Online	8	Remote	64 GB	2 x GRID K1	17-Apr-2015
viz6-uwo	MC 258	Online	8	Remote/Local	48 GB	2 x GT200GL Quadro FX 4800	
viz7-uwo	MC 258	Online	8	Remote/Local	48 GB	2 x ATI FirePro V9800	08-Apr-2015
viz8-uwo	MC 253 (racked)	Online	8	Remote	48 GB	2 x GT200GL Quadro FX 4800	13-Apr-2015
viz9-uwo	MC 253 (racked)	Online	8	Remote	48 GB	2 x ATI FirePro V9800	13-Apr-2015
viz10-uwo	MC 253 (racked)	Online	8	Remote	48 GB	2 x ATI FirePro V9800	13-Oct-2014
viz11-uwo	MC 253 (racked)	Online	8	Remote	48 GB	2 x ATI FirePro V9800	13-Oct-2014
vdi-fedora20	WSC 148 (racked)	Online	8	Remote	64 GB	2 x GRID K1	17-Apr-2015
viz3-uwo	WSC 148 (Boardroom)	Offline	8	Remote	48 GB	GT200GL Quadro FX 5800	17-Apr-2015



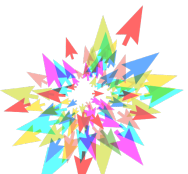
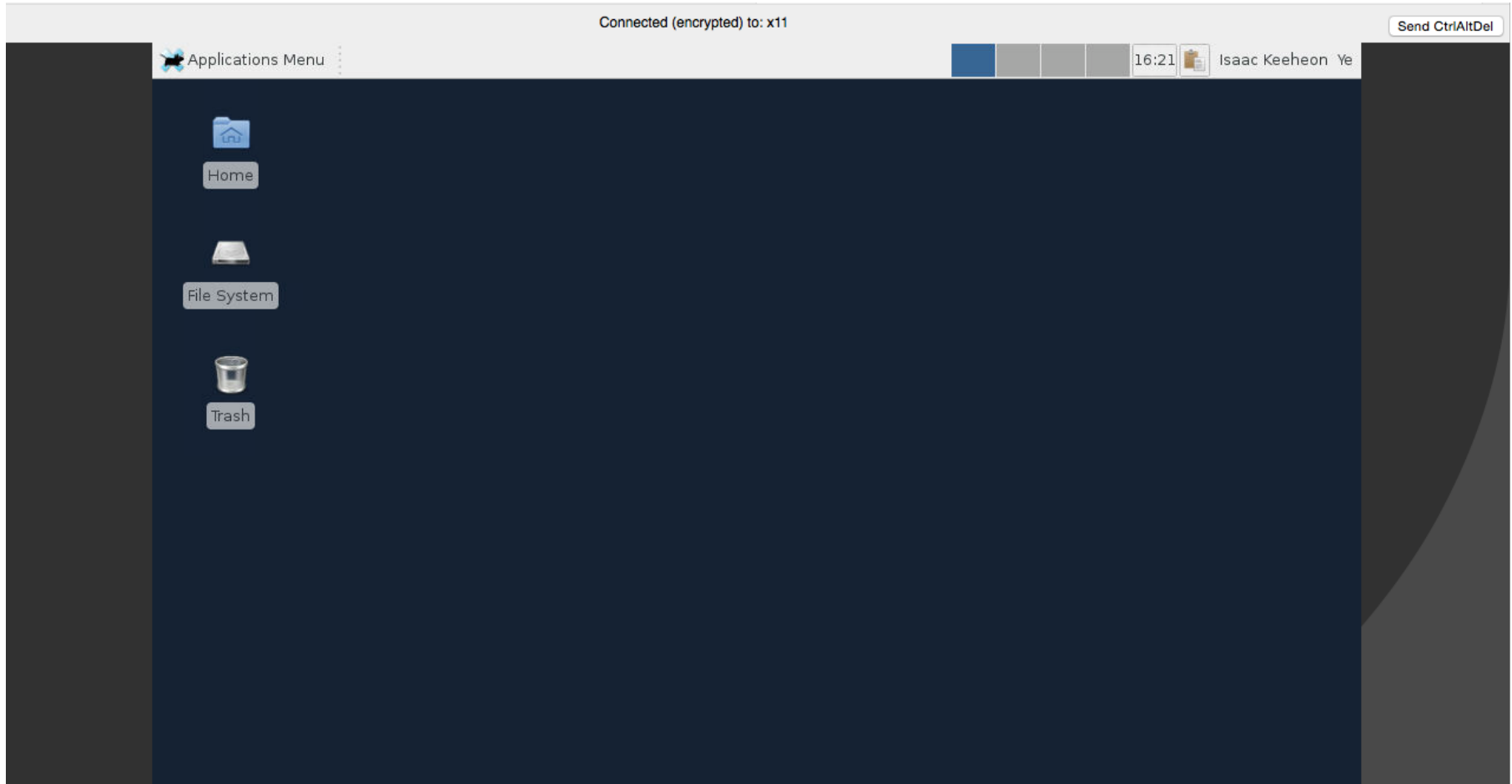
Click Here!



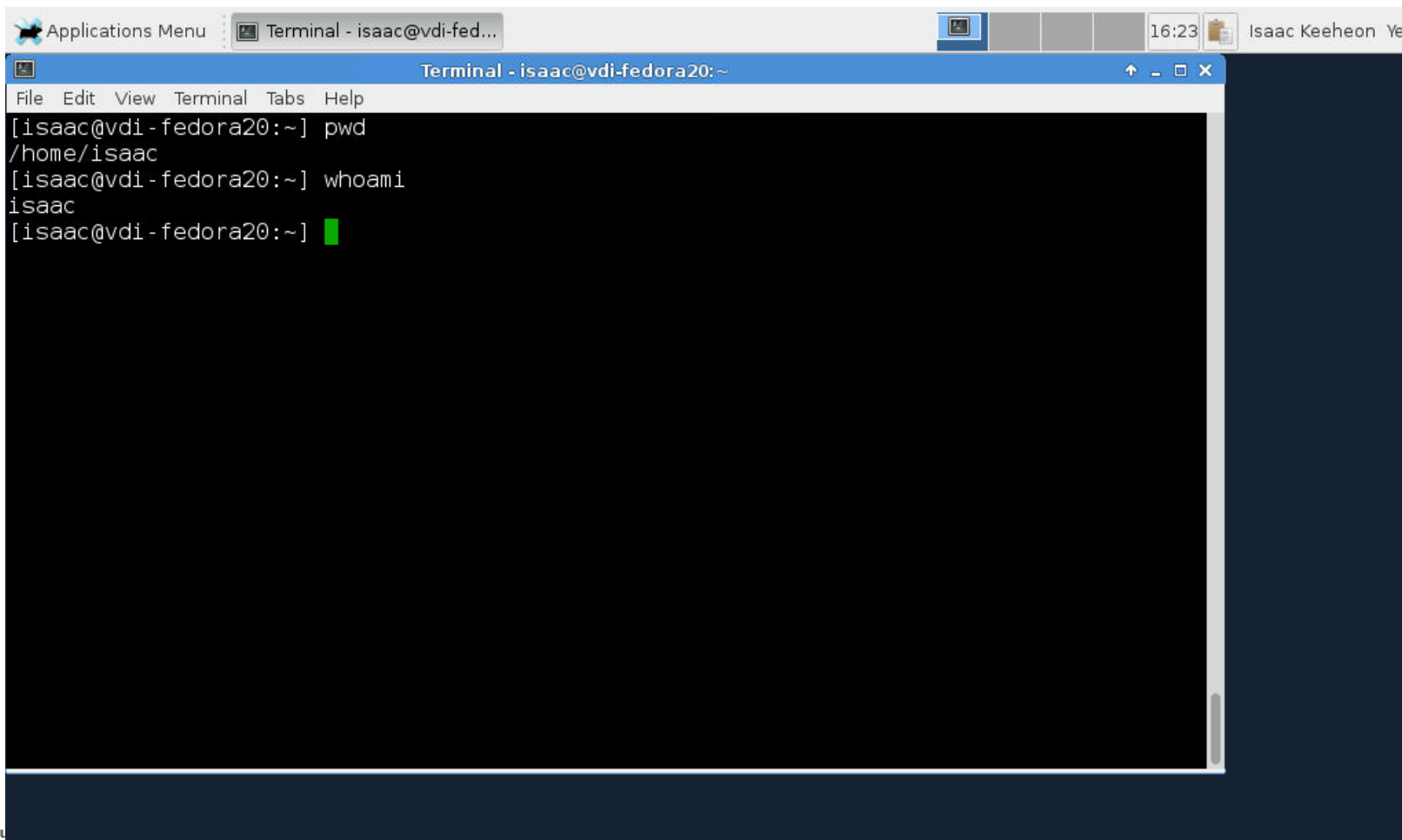
Login credentials



Desktop snapshot

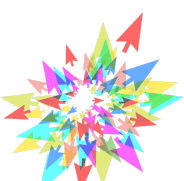


Open 'Terminal' for command line



The image shows a terminal window titled "Terminal - isaac@vdi-fedora20:~". The window has a menu bar with "File", "Edit", "View", "Terminal", "Tabs", and "Help". The terminal content shows the following commands and their outputs:

```
[isaac@vdi-fedora20:~] pwd
/home/isaac
[isaac@vdi-fedora20:~] whoami
isaac
[isaac@vdi-fedora20:~] █
```



Logging In/Out using SSH

- Connect to the server (SSH only in SHARCNET)

```
[isaac@cfdp8 isaac]$ ssh isaac@saw.sharcnet.ca  
isaac@saw.sharcnet.ca's password:
```

```
Last login: Tue May 25 11:36:11 2010 from bas9-toronto12-1128700169.dsl.bell.ca
```



Last login info

```
Welcome to Saw, a SHARCNET cluster.  
Please see the following URL for status of this and other clusters:  
https://www.sharcnet.ca/my/systems
```



Welcome message

```
****
```

```
ALL Sharcnet users must now also have a Compute Canada account. Please  
visit http://ccdb.sharcnet.ca for instructions.
```

```
****
```

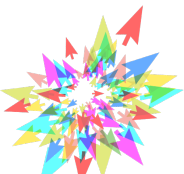
```
[isaac@saw377 ~]$
```



Command prompt

- Exit from the server (Don't forget !)

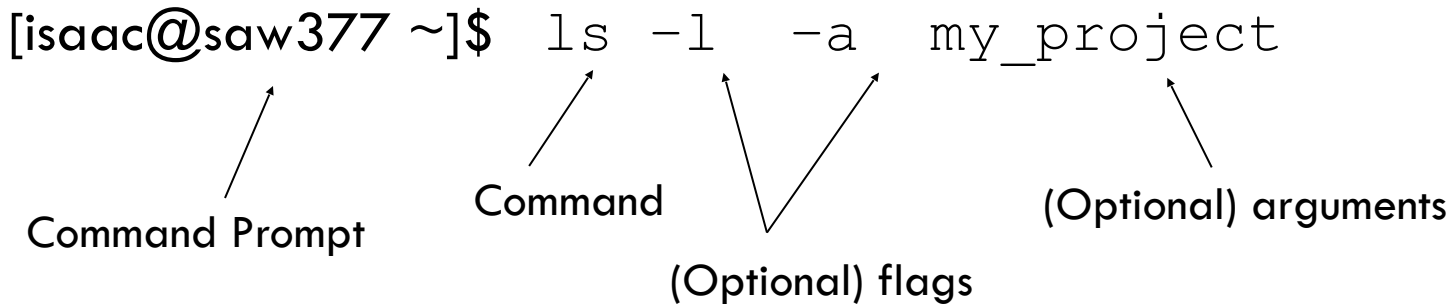
```
[isaac@saw377 ~]$ exit  
logout
```



The Command Prompt

- Commands are the way to “do things” in Unix
- A command consists of a command name and options called “flags”
- Commands are typed at the command prompt
- In Unix, everything (including commands) is case-sensitive

```
[prompt]$ <command> <flags> <args>
```



Note: In LINUX, you’re expected to know what you’re doing. Many commands will print a message only if something went wrong.



Two Basic Commands for Help

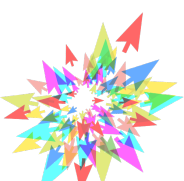
- The most useful commands you'll ever learn:
 - `man` (short for “*manual*”)
 - `info`
- They help you find information about other commands
 - `man <cmd>` or `info <cmd>` retrieves detailed information about `<cmd>`
 - `man -k <keyword>` searches the man page summaries (faster, and will probably give better results)
 - `man -K <keyword>` searches the full text of the man pages
- *command --help*

```
[isaac@saw377 ~]$ ls --help
```

```
Usage: ls [OPTION]... [FILE]...
```

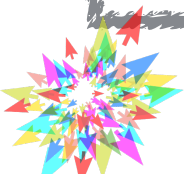
```
List information about the FILEs (the current directory by default).
```

```
Sort entries alphabetically if none of -cftuvSUX nor --sort...
```

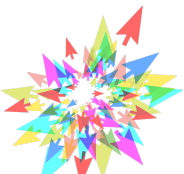
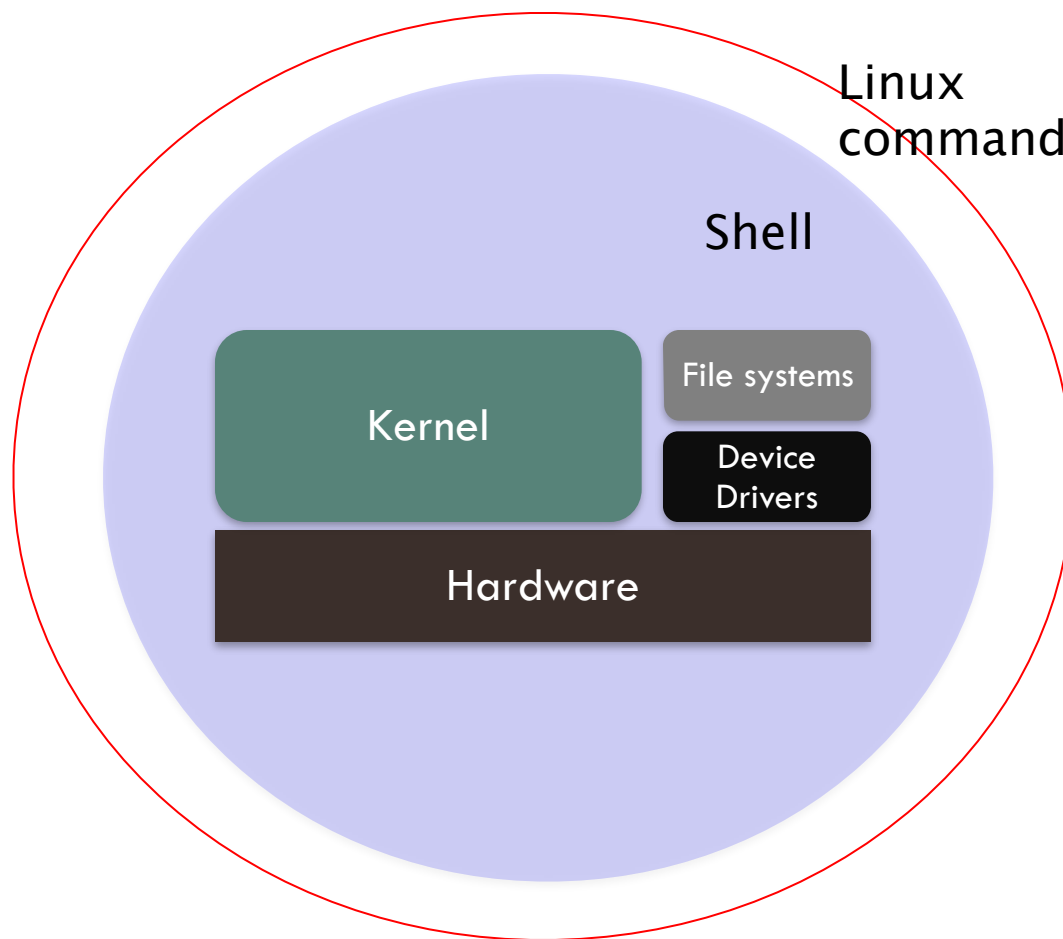


Exercise #1

1. Getting into LINUX system (Website/SSH)
2. Check your id 'whoami'
3. Check your files 'ls', 'ls -l', 'ls -lrt'
4. Get help on ls 'man ls', 'ls --help'
5. Find out who else is on the system 'w'
6. What is your current directory 'pwd'



Linux System

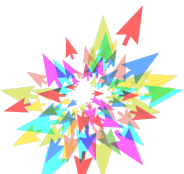


SHELL

- An interface between the Linux system and the user
- Used to call commands and programs
- An interpreter
- Powerful programming language
- Many available (bsh; ksh; csh; [bash](#); [tcsh](#))
 - How to check your shell ?

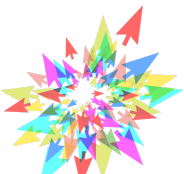
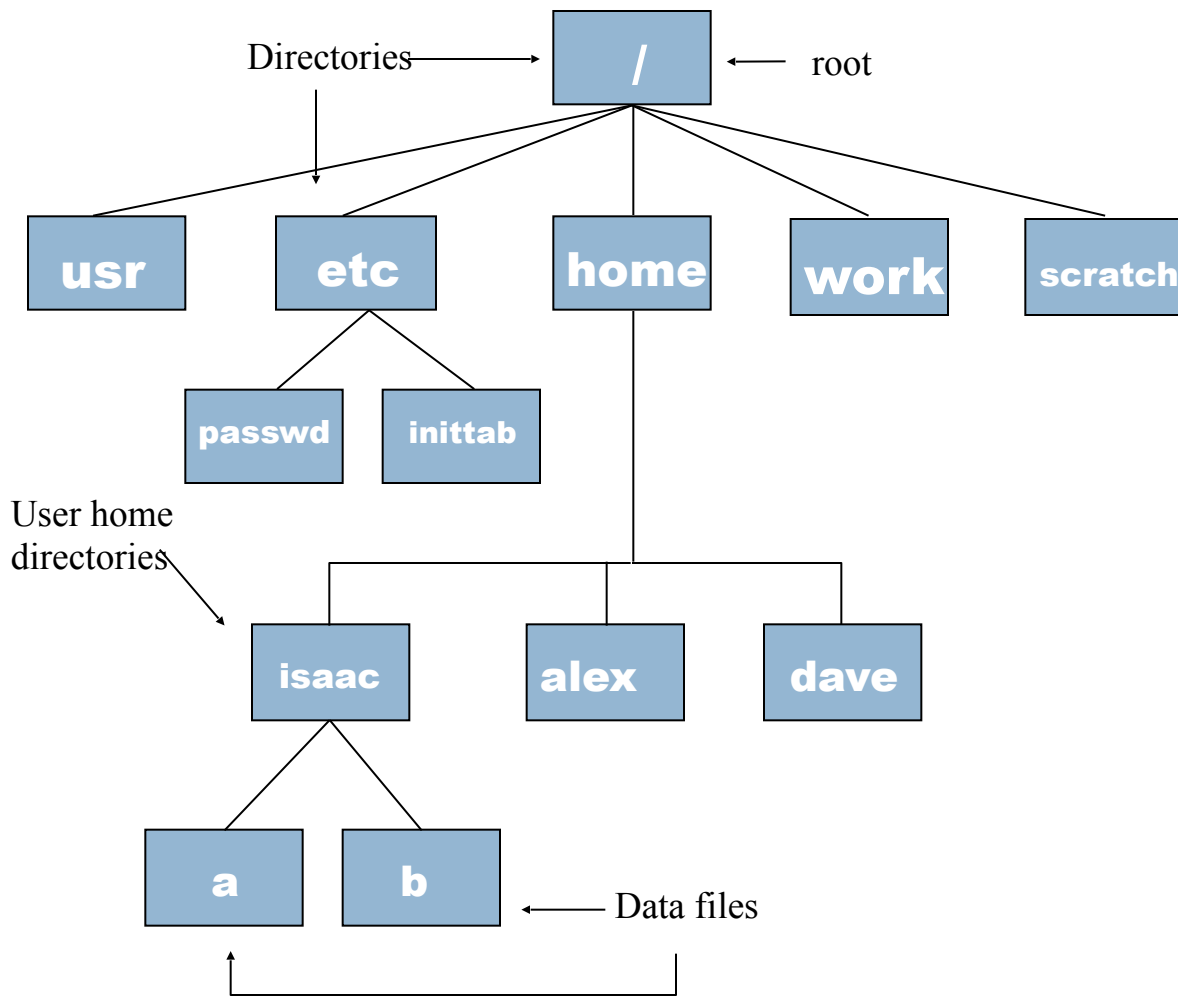
```
[isaac@saw377 ~]$ echo $SHELL  
/bin/bash
```

- ‘bash’ is in default on SHARCNET machines



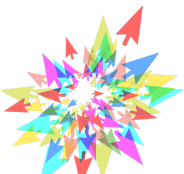
Linux File System Basics

- Linux files are stored in a single rooted, hierarchical file system
 - Data files are stored in directories (folders)
 - Directories may be nested as deep as needed



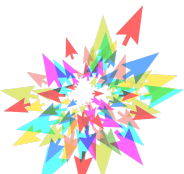
Some Special File Names

- Some file names are special:
 - / The root directory (not to be confused with the root user)
 - . The current directory
 - .. The parent (previous) directory
 - ~ My home directory
- Examples:
 - ./a same as a
 - ../isaac/x go up one level then look in directory
isaac for x



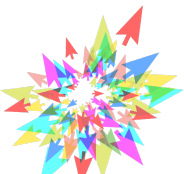
Command for Directories

- `ls`
 - **LiSts** the contents of the specified directories (or the current directory if no files are specified)
 - Syntax: `ls [<file> ...]`
 - Example: `ls backups`
- `pwd`
 - shows the present directory info
 - **Print Working Directory**
- `cd`
 - **Change Directory** (or your home directory if unspecified)
 - Syntax: `cd <directory>`
 - Examples:
 - `cd backups/unix-tutorial`
 - `cd ../class-notes`



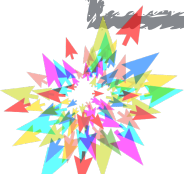
(cont'd)

- `mkdir`
 - **MaKe DIRectory**
 - **Syntax:** `mkdir <directories>`
 - **Example:** `mkdir backups class-notes`
- `rmdir`
 - **ReMove DIRectory**, which *must be empty*
 - **Syntax:** `rmdir <directories>`
 - **Example:** `rmdir backups class-notes`



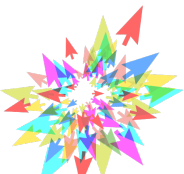
Exercise #2

1. Check your SHELL 'echo \$SHELL'
2. Change directory to /tmp 'cd /tmp'
3. Change directory back to 'cd \$USER'
4. Make a directory 'mkdir test1'
5. Change directory to test1 'cd test1'
6. Make a directory 'mkdir test1-1', 'mkdir test1-2'
7. Change directory to test1-2 'cd test1-2'
8. List files upper directory 'ls ..'
9. Change directory to test1-1 'ls ../test1-1', 'pwd'
10. Change directory to home directory 'cd ../../'
11. 'cd test1' and remove directories 'rmdir test1-1' 'rmdir test1-2'
12. Change directory to home 'cd ~'



Files

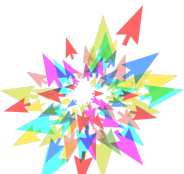
- Unlike Windows, in LINUX file types (e.g. “executable files, ” “data files,” “text files”) are *not* determined by file extension (e.g. “foo.exe”, “foo.dat”, “foo.txt”)
- Thus, the file-manipulation commands are few and simple
- Many commands only use 2 letters
- `rm`
 - **ReM**oves a file, *without a possibility of “undelete!”*
 - Syntax: `rm [options] <file(s)>`
 - Example: `rm tutorial.txt backups/old.txt`
 - `-r` option: recursive (delete directories)
 - `-f` option: force. Do no matter what



Files (cont'd)

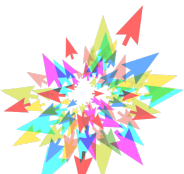
- cp
 - **CoPies** a file, preserving the original
 - **Syntax:** `cp [options] <sources> <destination>`
 - **Example:** `cp tutorial.txt tutorial.txt.bak`
 - **-r option:** recursive (copies directories)
- mv
 - **MoVes** (renames) a file or directory, destroying the original
 - **Syntax:** `mv [options] <sources> <destination>`
 - **Examples:**
 - `mv tutorial.txt tutorial.txt.bak`
 - `mv tutorial.txt tutorial-slides.ppt backups/`

Note: Both of these commands will over-write existing files without warning you!



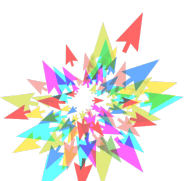
More Commands

- `diff` - attempts to determine the minimal set of changes needed to convert a file specified by the first argument into the file specified by the second argument
 - **Syntax:** `diff [options] <FILES>`
 - **Example:** `diff a.txt a1.txt`
- `find` - Searches a given file hierarchy specified by path, finding files that match the criteria given by expression
 - **Syntax:** `find [path...] [expression]`
 - **Example:** `find ./ -name "tes.h" -print`



More Commands

- `tar` - manipulates archives
 - An archive is a single file that contains the complete contents of a set of other files; an archive preserves the directory hierarchy that contained the original files. Similar to a VMARC file
 - Syntax: `tar [OPTION...] [FILE]...`
 - Archive files: `tar -cvf tarfile.tar ./isaac/*`
 - Archive & compress (gzip):
`tar -cvfz tarfile.tar.gz ./isaac/*`
 - Extract a tar file: `tar -xvf tarfile.tar`
 - Extract a tar-gzip file: `tar -xvfz tarfile.tar.gz`



File Permissions

- The long version of a file listing (`ls -l`) will display the file permissions:

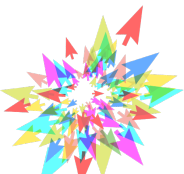
```

-rwxrwxr-x  1 rvdheij  rvdheij      5224 Dec 30 03:22 hello
-rw-rw-r--  1 rvdheij  rvdheij         221 Dec 30 03:59 hello.c
-rw-rw-r--  1 rvdheij  rvdheij         1514 Dec 30 03:59 hello.s
drwxrwxr-x  7 rvdheij  rvdheij         1024 Dec 31 14:52 posixuft
  
```

Permissions

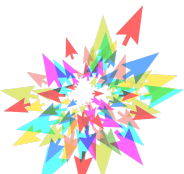
Group

Owner



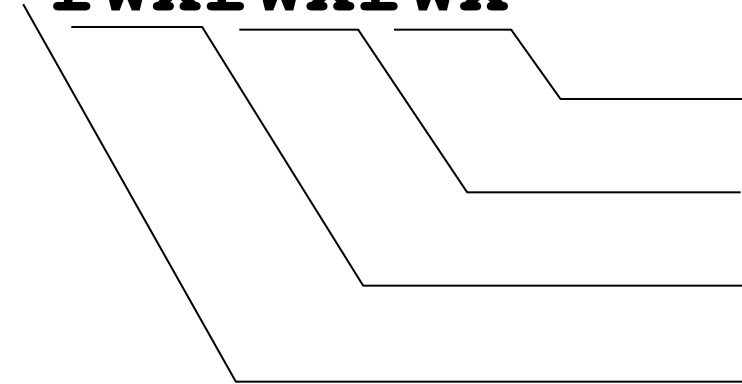
File Permissions

- Linux provides three kinds of permissions:
 - Read (r, 4) - users with read permission may read the file or list the directory
 - Write (w, 2) - users with write permission may write to the file or new files to the directory
 - Execute (x, 1) - users with execute permission may execute the file or lookup a specific file within a directory



Interpreting File Permissions

-rwxrwxrwx

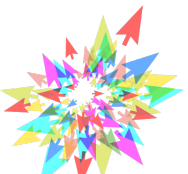


Other permissions

Group permissions

Owner permissions

Directory flag (- = file; d=directory; l=link)



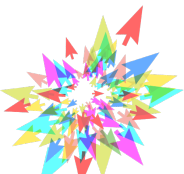
Changing File Permissions

- Use the `chmod` command to change file permissions
 - The permissions are encoded as an octal number

```

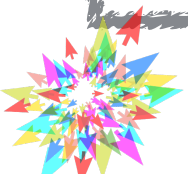
chmod 755 file # Owner=rwx Group=r-x Other=r-x
chmod 500 file2 # Owner=r-x Group=--- Other=---
chmod 644 file3 # Owner=rw- Group=r-- Other=r--

chmod +x file # Add execute permission to file for all
chmod o-r file # Remove read permission for others
chmod a+w file # Add write permission for everyone
  
```

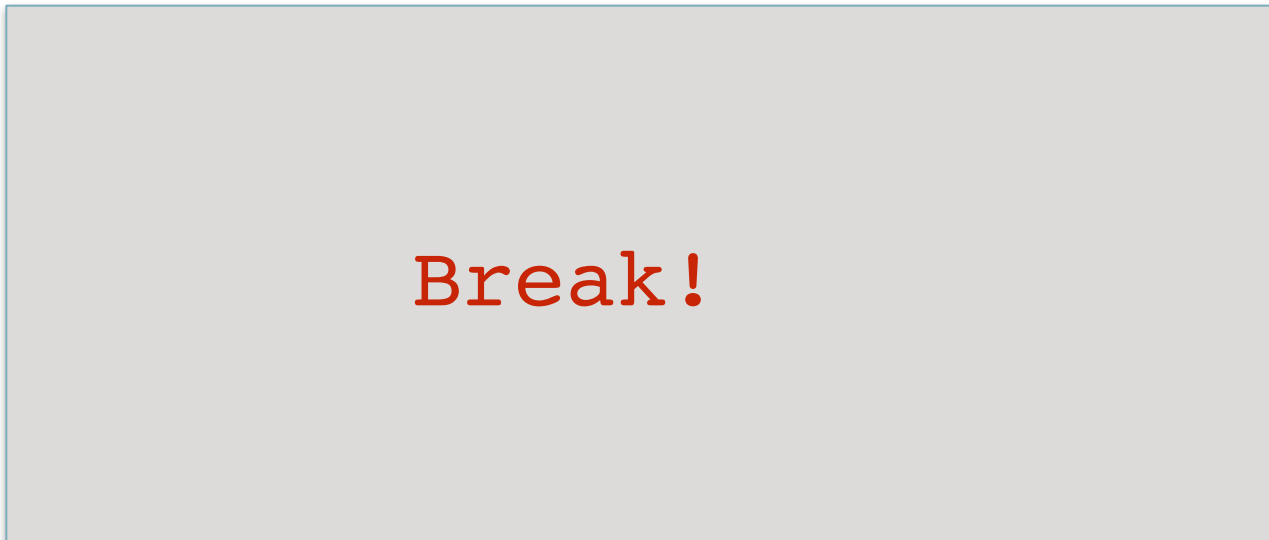
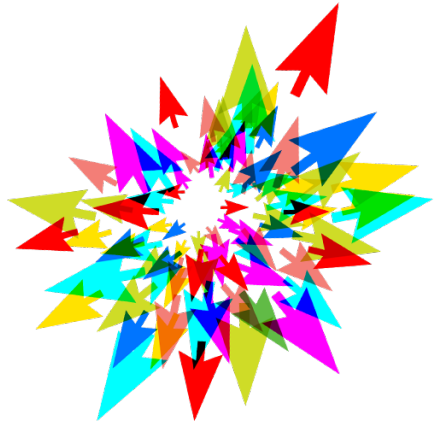


Exercise #3

1. Copy tutorial file 'cp /home/isaac/ss15_1.tar.gz ~'
2. Uncompress/untar the file 'gunzip ss15_1.tar.gz' and 'tar xvf ss15_1.tar'
or 'tar zxvf ss15_1.tar.gz'
3. Change directory to ss15_1 'cd ss15_1' and list files 'ls -lrt'
4. Find the file with name 'session1.pdf' using find command 'find ./ -name
'session1.pdf' -print'
5. Make a directory test1 in ss15_1 and copy session1.pdf to test1
6. Change directory to test1
7. Check the permission and make it accessible/readable to your group



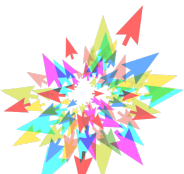
compute | **calcul**
canada | canada



Break!

Processes

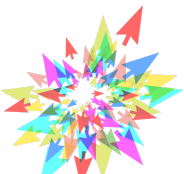
- Foreground
 - When a command is executed from the prompt and runs to completion at which time the prompt returns is said to run in the foreground
- Background
 - When a command is executed from the prompt with the token “&” at the end of the command line, the prompt immediately returns while the command continues is said to run in the background
- Check the process
 - Command: ps, top, kill



Top

```
top - 00:34:10 up 258 days, 14:53, 3 users, load average: 2.35, 2.46, 2.45
Tasks: 733 total, 3 running, 729 sleeping, 0 stopped, 1 zombie
Cpu(s): 8.1%us, 0.3%sy, 0.0%ni, 91.7%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 33011060k total, 32360816k used, 650244k free, 512k buffers
Swap: 31999988k total, 31999988k used, 0k free, 1992976k cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
15100	pliang	20	0	143m	15m	1280	R	87.9	0.0	827:27.72	perl
27937	pliang	20	0	159m	31m	1168	R	86.6	0.1	2756:46	perl
19100	pliang	20	0	33504	16m	676	S	3.3	0.1	0:00.10	samtools
19101	pliang	20	0	0	0	0	Z	3.0	0.0	0:00.09	samtools <defunct>
10357	nobody	20	0	384m	133m	756	S	0.7	0.4	46:22.81	gmond
18964	isaac	20	0	16464	1720	892	R	0.7	0.0	0:00.48	top
99	root	20	0	0	0	0	S	0.3	0.0	71:16.49	events/0
103	root	20	0	0	0	0	S	0.3	0.0	12:22.73	events/4
1301	root	20	0	0	0	0	S	0.3	0.0	3:27.49	edac-poller
1872	root	20	0	10912	592	400	S	0.3	0.0	76:22.21	irqbalance
3880	alikey	20	0	15816	660	488	S	0.3	0.0	2:17.05	top
1	root	20	0	21436	1068	876	S	0.0	0.0	0:13.59	init
2	root	20	0	0	0	0	S	0.0	0.0	0:03.25	kthreadd
3	root	RT	0	0	0	0	S	0.0	0.0	2:56.19	migration/0
4	root	20	0	0	0	0	S	0.0	0.0	3:08.42	ksoftirqd/0
5	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
6	root	RT	0	0	0	0	S	0.0	0.0	0:26.57	watchdog/0
7	root	RT	0	0	0	0	S	0.0	0.0	3:00.91	migration/1
8	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	migration/1
9	root	20	0	0	0	0	S	0.0	0.0	2:35.75	ksoftirqd/1



Processes

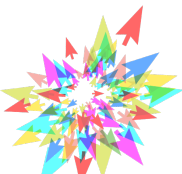
& causes process to be run in "background"

```
[root@penguinvm log]# sleep 10h &
[1] 6718
[root@penguinvm log]# ps -ef
UID          PID    PPID  C  STIME TTY          TIME CMD
root         6718   6692  0  14:49 ttyp0        00:00:00 sleep 10h
```

Job Number

Process ID (ID)

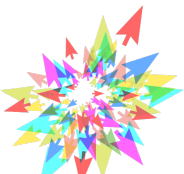
Parent Process ID



Grep

- grep - Searches files for one or more pattern arguments. It does plain string, basic regular expression, and extended regular expression searching

```
ps -ef |grep -i "isaac"
```

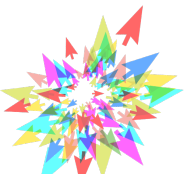


Command for Processes

- kill - sends a signal to a process or process group
- You can only kill your own processes unless you are root

```

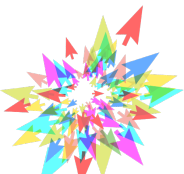
UID          PID    PPID    C  STIME TTY          TIME CMD
root         6715   6692    2  14:34 ttyp0        00:00:00 sleep 10h
root         6716   6692    0  14:34 ttyp0        00:00:00 ps -ef
[root@penguinvm log]# kill 6715
[1]+  Terminated                  sleep 10h
  
```



Environment Variables

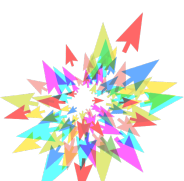
- Environment variables are global settings that control the function of the shell and other Linux programs. They are sometimes referred to global shell variables.
- Check your environment

```
[isaac@saw377 ~]$ env
MKLRROOT=/opt/sharcnet/intel/11.0.083/ifc/mkl
MODULE_VERSION_STACK=3.2.6
MANPATH=/opt/sharcnet/octave/current/share/man:/opt/sharcnet/netcdf/current/man:
FOAM_SOLVERS=/work/isaac/OpenFOAM/OpenFOAM-1.6/applications/solvers
FOAM_APPBIN=/work/isaac/OpenFOAM/OpenFOAM-1.6/applications/bin/linux64GccDPOpt
FOAM_TUTORIALS=/work/isaac/OpenFOAM/OpenFOAM-1.6/tutorials
FOAM_JOB_DIR=/work/isaac/OpenFOAM/jobControl
HOSTNAME=saw377
snrestart=--nosrun /opt/sharcnet/blcr/current/bin/sn_restart.sh
IPPROOT=/opt/sharcnet/intel/11.0.083/icc/ipp/em64t
INTEL_LICENSE_FILE=/opt/sharcnet/intel/11.0.083/ifc/licensesADFBIN=/opt/sharcnet/adf/current/bin
```



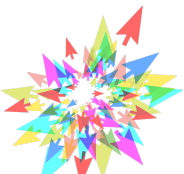
Environment Variables

- Using Environment Variables:
 - `echo $VAR`
 - `cd $VAR`
 - `cd $HOME`
- Displaying - use the following commands:
 - `set` (displays local & env. Vars)
 - `export`
- Vars can be retrieved by a script or a program



Some Important Environment Variables

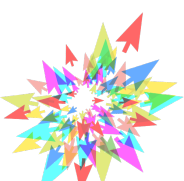
- HOME
 - Your home directory (often be abbreviated as “~”)
- TERM
 - The type of terminal you are running (for example vt100, xterm, and ansi)
- PWD
 - Current working directory
- PATH
 - List of directories to search for commands



PATH Environment Variable

- Controls where commands are found
 - PATH is a list of directory pathnames separated by colons. For example:


```
PATH=/bin:/usr/bin:/usr/X11R6/bin:/usr/local/bin:/home/alex/bin
```
 - If a command does not contain a slash, the shell tries finding the command in each directory in PATH. The first match is the command that will run
 - Set in `/etc/profile`, `~/.profile`, `~/.bashrc`



Editing Text

- Which text editor is “the best” is a holy war. Pick one and get comfortable with it.
- Three text editors you should be aware of:
 - nano – An improved ‘pico’ editor
 - To quit: Ctrl-x
 - emacs/xemacs – A heavily-featured editor commonly used in programming
 - To quit: Ctrl-x Ctrl-c
 - vim/vi – Another editor, also used in programming
 - To quit: <Esc> : q <Enter> (or QQ -- capitals matter)

Knowing the basics of emacs and vim will help with the rest of Unix; many programs have similar key sequences.

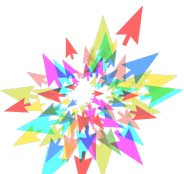


Alias

- An alias is nothing but shortcut to commands
- Use alias command to display list of all defined aliases
- Add aliases to ~/.bashrc file

```
alias name='command arg1 arg2'
```

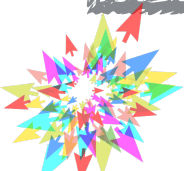
```
alias rm='rm -i'  
alias cp='cp -i'  
alias mv='mv -i'  
alias grep="grep -n"
```



Exercise #4

1. Execute 'top' to see what processes are on (quit : 'q')
2. Do background job 'sleep 10m &' and check 'ps -ef |grep \$USER'
3. Kill the job 'kill %1' or 'kill PID'
4. Goto ss15_1/run
5. Execute 'a.out' using two different ways
 1. './a.out'
 2. 'a.out'
6. Check your PATH 'echo \$PATH'
7. Add /home/\$USER/ss15/run to the existing path and re-execute 'a.out'
8. Add one alias into ~/.bashrc using text editor

```
alias grep="grep -n"
```



Pipe and Redirection

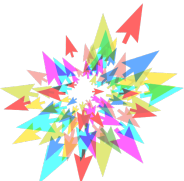
- Redirection (< or >)

```
$ ls -l > lsoutput.txt (save output to lsoutput.txt)
$ ps >> lsoutput.txt (append to lsoutput.txt)
$ more < killout.txt (use killout.txt as parameter to more)
```

- Pipe (|)

- Process are executed concurrently

```
$ ps | sort | more
$ ps -xo comm | sort | uniq | grep -v sh | more
$ cat mydata.txt | sort | uniq | > mydata.txt (generates an empty file !)
```



Quota

- Quota

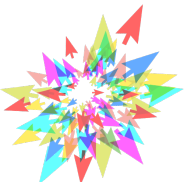
```
[isaac@orc-login1:~] quota
```

Filesystem	Limit	Used	File Count	Checked
davy:/home	10 GB	1.4 GB (13%)	17,545	24h ago
cove:/work	1 TB	212.5 GB (20%)	951,454	19h ago

- df : report the space left on the file system
- du : output the number of kilobytes used by each dir.

```
[isaac@orc-login1:/work/isaac/ss15_2] du -h .
```

```
133K ./Ex4
261K ./Ex1
6.5K ./Ex5/BCK
139K ./Ex5
1.4M ./MK
261K ./Ex3
133K ./Ex2
2.3M .
```

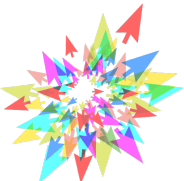


File

- File classifies the named files according to the type of data they contain

```
[[isaac@orc-login1:/work/isaac/channelflow] file *
```

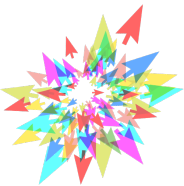
bin:	directory
branches:	directory
channelflow-1.4.2:	directory
couette.args:	ASCII text
data:	directory
frames:	directory
include:	directory
lib:	directory
movieframes.args:	ASCII text
randomfield.args:	ASCII text
trunk:	directory
u0.ff:	data



History

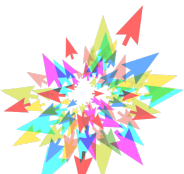
- Shell keeps an ordered list of all the commands that you have entered.

```
$ history (show all your command history)
$ !!      (recall last command)
$ !-3     (recall third most recent command)
$ !5      (recall 5th command in list)
$ !grep   (recall last command starting with grep)
$ set history = 1000
```



Modules

- What is a module system?
 - A user interface to provide for the dynamic modification of a user's environment via module files.



Modules (Example: loading WRF)

- Module list – list up the presently loaded modules

```
[isaac@hnd50:~] module list
Currently Loaded Modulefiles:
  1) moab/5.4.2           7) r/2.10.0           13) gromacs/4.0.5
  2) sq-tm/2.4           8) namd/2.7b3         14) vmd/1.8.7
  3) intel/11.0.083     9) ansys/12.1.1       15) util/2.0
  4) openmpi/intel/1.4.2 10) lsdyna/ls971dR5.0 16) user-environment/1.0.0
  5) compile/1.3        11) fftw/intel/2.1.5
  6) octave/3.2.4       12) lammps/10.08.2010
```

- Module avail – list up all available modules

```
[isaac@hnd50:~] module avail
----- /opt/sharcnet/modules -----
acml/gfortran/3.6.0      acml/pgi-int64/4.2.0      lammps/10.08.2010
acml/gfortran/4.2.0     openmpi/intel-debug/1.4.3  wrf/3.2
```



Modules (Cont'd)

- Module show [module] – load the module into the env

```
[isaac@hnd50:~] module show wrf/3.2
```

```
-----  
/opt/sharcnet/modules/wrf/3.2:
```

```
module-whatis  Provide WRF/WPS 3.2 built using intel 11.0.083 and openmpi 1.4.2 on centos.
```

```
conflict      wrf
```

```
prereq        intel/11.0.083
```

```
prereq        openmpi/intel/1.4.2
```

```
module        load gmp/4.3.2
```

```
module        load mpfr/2.4.2
```

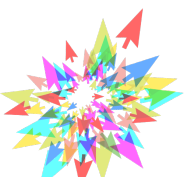
```
module        load netcdf/intel/4.1.2
```

```
prepend-path  PATH /opt/sharcnet/wrf/3.2/wrfv3/main:/opt/sharcnet/wrf/3.2/wrfv3/run:/opt/  
sharcnet/wrf/3.2/wrfv3/tools:/opt/sharcnet/wrf/3.2/wps:/opt/sharcnet/wrf/3.2/wps/util
```

```
prepend-path  LD_RUN_PATH /opt/sharcnet/wrf/3.2/wps_libs/lib
```

```
prepend-path  --delim  LDFLAGS -L/opt/sharcnet/wrf/3.2/wps_libs/lib -L/opt/sharcnet/wrf/3.2/  
wrfv3/main
```

```
prepend-path  --delim  CPPFLAGS -I/opt/sharcnet/wrf/3.2/wps_libs/include -I/opt/sharcnet/wrf/  
3.2/wrfv3/inc
```



Modules (Cont'd)

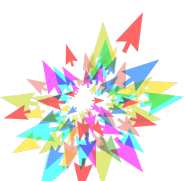
- Module load [module] – load the module into the env

```
[isaac@hnd50:~] module load wrf/3.2
```

```
[isaac@hnd50:~] module list
```

Currently Loaded Modulefiles:

- | | | | |
|------------------------|-----------------------|------------------------|-------------------------------|
| 1) moab/5.4.2 | 6) octave/3.2.4 | 11) fftw/intel/2.1.5 | 16) user-environment/1.0.0 |
| 2) sq-tm/2.4 | 7) r/2.10.0 | 12) lammmps/10.08.2010 | 17) gmp/4.3.2 |
| 3) intel/11.0.083 | 8) namd/2.7b3 | 13) gromacs/4.0.5 | 18) mpfr/2.4.2 |
| 4) openmpi/intel/1.4.2 | 9) ansys/12.1.1 | 14) vmd/1.8.7 | 19) netcdf/intel/4.1.2 |
| 5) compile/1.3 | 10) lsdyna/ls971dR5.0 | 15) util/2.0 | 20) wrf/3.2 |



Modules (Cont'd)

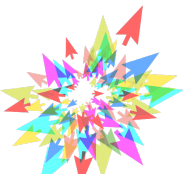
- Module unload [module] – unload the module from the env

```
[isaac@hnd50:~] module unload wrf
```

```
[isaac@hnd50:~] module list
```

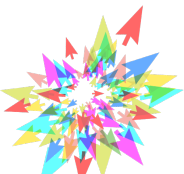
Currently Loaded Modulefiles:

1) moab/5.4.2	5) compile/1.3	9) ansys/12.1.1	13) gromacs/4.0.5
2) sq-tm/2.4	6) octave/3.2.4	10) lsdyna/ls971dR5.0	14) vmd/1.8.7
3) intel/11.0.083	7) r/2.10.0	11) fftw/intel/2.1.5	15) util/2.0
4) openmpi/intel/1.4.2	8) namd/2.7b3	12) lammps/10.08.2010	16) user-
environment/1.0.0			



Job scheduler

- Job must be submitted through a job scheduler
 - LSF/MOAB/PBS/TORQUE/MAUI
- SHARCNET provides a unified job scheduling script
 - sqjobs – list the status of submitted jobs
 - sqkill – stop/dequeue a runn



Job scheduler (Cont'd)

- **Submitting serial jobs**

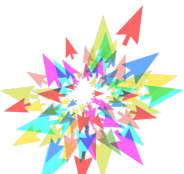
- `sqsub -r1h --test ./sim`

- **Submitting parallel jobs**

- `sqsub -r1h -q mpi --test -n 24 -o sim.out ./sim`

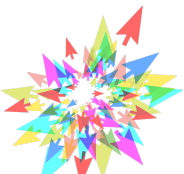
- **Submitting jobs with a memory request**

- `sqsub -r1h --mpp=2G -o sim.out ./sim`



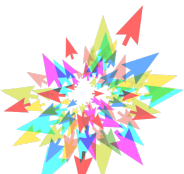
Programming languages

- **Languages**
 - Fortran, C/C++, Java, MATLAB, etc.
- **Compilers**
 - SHARCNET unified compilation environment
 - *cc, c++, f77/90, mpicc, mpic++, mpif77, mpif90*
- **Key Parallel Development Support**
 - MPI, POSIX threads API, OpenMP



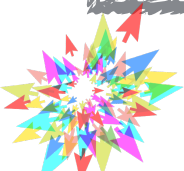
X11

- X-Windows is the most common graphical interface for Unix
- It allows graphics to be sent over the network (Windows Remote Desktop is similar to this)
- If you login via the ssh-x shortcuts, you will start and “X-server” on your machine and you will be able to get graphics from your unix commands
- If you log into a linux box, you will automatically have X-windows setup in that login.

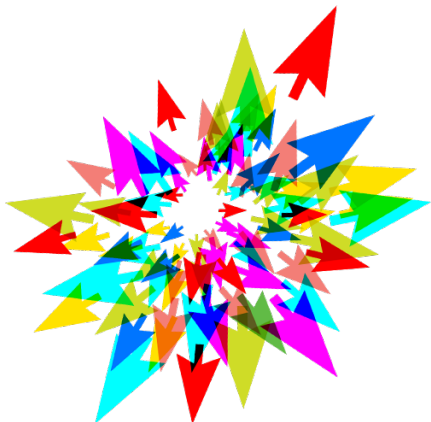


Exercise #5

1. Practice slide 49 for pipe
2. Make a 'sqjobs.log' file using 'sqjobs -n' and pipe direction
3. Move into your home directory and add list files into sqjobs.log
4. Check your quota and make sure size of each directory at your home
5. Practice 'file' command at your home
6. Check your module and module load samtools/1.1
7. Check your history and practice slide 52.



compute | **calcul**
canada | canada



Thank you !

For further questions,