More bash & SLURM

C. Ryan Campbell

Duke University

c.ryan.campbell@duke.edu

19 Sept 2017

Overview

Today's Goals

- Log into the cluster

Today's Goals

- Log into the cluster
- Learn some basic commands

Today's Goals

- Log into the cluster
- Learn some basic commands
  - sort, uniq, cut, nano

Today's Goals

- Log into the cluster
- Learn some basic commands
  - sort, uniq, cut, nano
- Learn some advanced commands

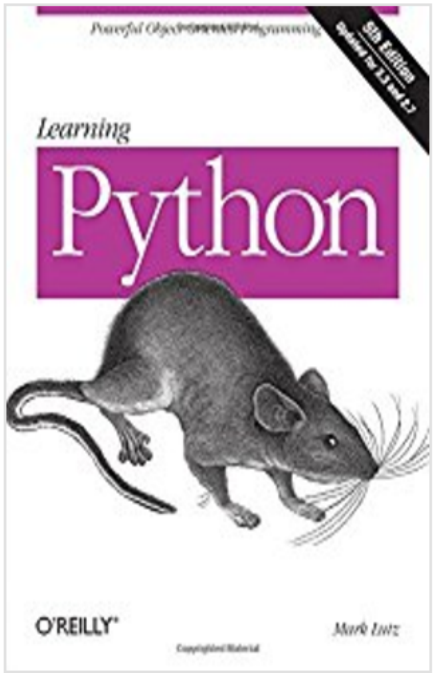
Today's Goals

- Log into the cluster
- Learn some basic commands
  - sort, uniq, cut, nano
- Learn some advanced commands
  - grep, sed, awk

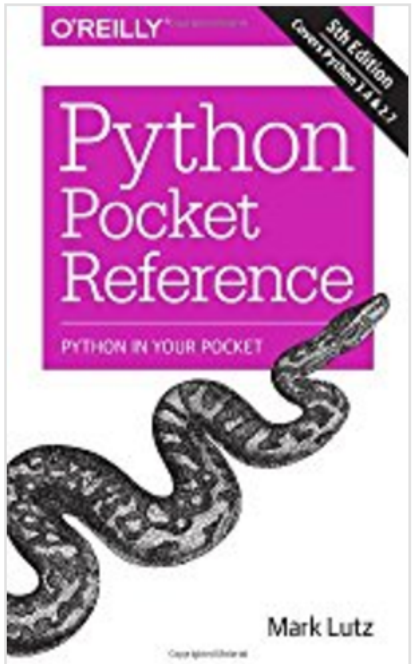
Today's Goals

- Log into the cluster
- Learn some basic commands
  - sort, uniq, cut, nano
- Learn some advanced commands
  - grep, sed, awk
- Answer some Emmy trivia questions

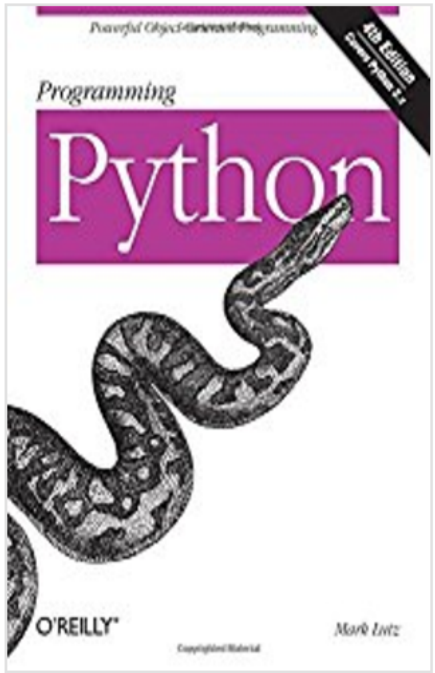
More Twitter



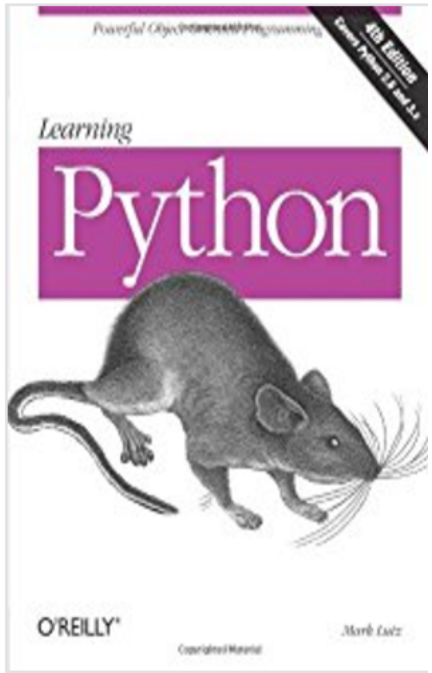
$39.58
Paperback



$10.76
Paperback



$43.46
Paperback



$6.74
Paperback

by Mark Lutz

More Twitter

Logging On

- Use command “ssh”
- Logs on to one of two compute nodes:

```
ssh <netid>@dscr-slogin-01.oit.duke.edu
```

- Your terminal window is now on the computer “dscr-slogin-01.oit.duke.edu”
- You may have set up an alias:

```
alias DSCR="ssh <your netid>@dscr-slogin-01.oit.duke.edu"
```

File Generation

- nano - text-based text editor
- Adding a filename "`nano <file>`" opens that file

- And use `scp` to move it up to the cluster

File Generation

- nano - text-based text editor
- Adding a filename "`nano <file>`" opens that file

- Or make a file on your local computer
- And use `scp` to move it up to the cluster

Tips and Help

- Spaces and order matter
- General grammar rules:

  `<command> <flags> <input>`
- If you're unsure what the flags are "-h" or "–help" often works
- Using "tab" auto-completes filenames and commands
- Prevents typos

You Ran a Cluster Script!

- Ta-Da!! Now you can run a job on the cluster!

You Ran a Cluster Script!

- Ta-Da!! Now you can run a job on the cluster!
- You now have the minimum knowledge to get onto the cluster and use the resource to run software that is unsuitable for your laptop

You Ran a Cluster Script!

- Ta-Da!! Now you can run a job on the cluster!
- You now have the minimum knowledge to get onto the cluster and use the resource to run software that is unsuitable for your laptop
- Now, there are a lot more things that the cluster and bash can do, mostly involving data manipulation

You Ran a Cluster Script!

- Ta-Da!! Now you can run a job on the cluster!
- You now have the minimum knowledge to get onto the cluster and use the resource to run software that is unsuitable for your laptop
- Now, there are a lot more things that the cluster and bash can do, mostly involving data manipulation
- Let's delve into some of those!

Datafile

- Emmy Winners 1949 to Present

  `/work/cc216/490S/cc216/emmy-awards-1949-2017.csv`
- Copy to your own folder, then pull out the nominees

```
grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list
```

grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list

Tools

- Input and output from commandline
  - Using > and |
- `cat, less, head, tail` - inspecting files and output
- `sort` - sorting
- `uniq` - sort and eliminate duplicates
- `cut` - split a file into columns

Input and Output

- Unless otherwise noted:
  - Input comes after the command
  - Output “prints to screen”
- Using > and | changes that
- > takes output and (over)writes to a file

```
ls > list.txt
```

- If you use double >> it appends to a file

```
ls -lt >> list.txt
```

Input and Output

- Using | changes input
- | takes output and passes it to the next command

```
ls | head -n1
```

- You can string together many |'s to perform complicated actions

Inspecting Files and Output

- `cat, less, head, tail`
- `cat` - prints the whole file to screen

```
cat <name of file>
```

- `less - opens the file so you can scroll through it (q to quit)`
- `head, tail` - return the first or last 10 lines of a file

```
head <name of file>; tail <name of file>
```

- Common flags:
  - -n number of lines (overrides 10)

sort

- `sort` takes input and sorts it! (simple, right?)

```
sort <name of file>
cat <name of file> | sort
```

- Common flags:
  - -n sorts numerically
  - -u sorts and only presents unique hits
  - -r sorts reverse
  - -t, and -k sort the nth field (k), separated by ,

uniq

- `uniq` takes input and removes adjacent duplicates! (still simple, right?)

```
uniq <name of file>
cat <name of file> | uniq
```

- Common flags:
  - -c counts each unique line
  - -d reverses the meaning (prints only duplicates)

cut

- `cut` takes input and divides it into “columns”

```
cut -d<what to divide by> -f<which columns
you want> <name of file>
```

- Common flags:
  - -d what to divide by (“,” “ ” “tab”)
  - -c take n characters (-c1-10 takes first 10 characters in each line)
  - -f which columns you want:
    - -f1, first only
    - -f1-5 one through five
    - -f1,5 first and fifth only

Advanced Tools

- The following tools are more advanced and complicated
- You will often see them in online forums
- You don't have to be a wizard
- It is good to be familiar with them

(Once you are, there isn't a dataset you won't be able to manage)

- `grep` - regular expression search
- `sed` - search and replace patterns
- `awk` - counting as well as search

grep

- `grep` searches, line by line, for a pattern or "regular expression"
- Globally search a Regular Expression and Print

```
grep <pattern to find> <name of file>
```

- Common flags:
  - -i case-independent
  - -v reverse-search (all lines WITHOUT the pattern)
  - -c returns a count instead of the lines
  - -w surround the pattern with whitespace
  - -A or -B return pattern and n-lines after (A) or before (B)

sed

- `sed` searches and replaces a pattern
- Prints result to screen

```
sed ‘s,<old pattern>,<new pattern>,g’ <name of file>
```

- Common flags:
  - -i in-place, replaces the pattern in the file and overwrites
  - -i.bak in-place, same as above but saves the initial as originalfilename.bak

awk

- `awk` is very powerful and mysterious
- If I'm using it, it is likely from a google search
- To find the average of the second column:

```
cat <name of file> | awk ' sum += $2  END  if
(NR > 0) print sum / NR '
```

- "Add up column 2, if more than 1 row then print the sum divided by the rows"
- Great for advanced math (bash can, generally, only handle integers)

Revisiting the Datafile

- Emmy Winners 1949 to Present
- What did these commands do?

```
grep ^20.., emmy-awards-1949-2017.csv | sed 's/,
Nonfiction,//g' | cut -d, -f4 > nominees.list
```

grep -

| -

sed -

cut -

> -

Revisiting the Datafile

- `grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list`

Revisiting the Datafile

- `grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list`
- grep -

Revisiting the Datafile

- `grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list`
- grep -
- | -

Revisiting the Datafile

- `grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list`
- grep -
- | -
- sed -

Revisiting the Datafile

- `grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list`
- grep -
- | -
- sed -
- cut -

Revisiting the Datafile

- `grep ^20.., emmy-awards-1949-2017.csv | sed 's/, Nonfiction,//g' | cut -d, -f4 > nominees.list`
- grep -
- | -
- sed -
- cut -
- > -

Revisiting the Datafile

Answer the following:

How many Emmys was Netflix nominated for in 2015?

How many different shows?

Which show had the most? (House of Cards -

How many Emmys has GoT been nominated for?

The End