SHELL SCRIPT

February 2, 2023

Directions to VPN off-campus

- https://www.uvm.edu/it/kb/article/sslvpn2/
- Remember, you will not be able to use VACC-OOD off-campus without it!

Common Linux Commands

command	description	command	description	command	description
cd	change directories	find	find something	pwd	present working directory (where am I)
chmod	change read, write, execute (rwx) permissions	grep	find something	rm	remove
chown	change owner	history	history	scp	secure copy
clear	clear	man	manual	ssh	secure shell (remote login)
cp	copy	mkdir	make directory	su	swith users
crontab	cronological listing of jobs	mv	move/ rename	sudo	super user do (override)
df	disk free	netstat	check ports	tar	compress/extract directory
du	disk usage	nslookup	lookup ip address or hostname	touch	Create a file
exit	exit				

Shell script

- These are text files that contain commands we know we want to run.
- End with the extension **.sh** (bash shell script)
- These shell scripts can be created on a text editor like Vim or Nano, or a GUI text editor

First, we will create a simple shell script

- Using this script will answer the question what is the difference between using redirect (<) versus append(<<)

difference.sh

This shell script will do two things:

1. Create a file called redirect.txt
2. Give us the word count for redirect.txt

echo command

- Used to output status text
- Display a line of text/string on standard output or a file

Jupyter Notebook version: 04240a1

This app will launch a Jupyter Notebook server on one or more nodes.

Partition

bluemoon

- To request a GPU specify a partition such as dggpu or bdgpu

Number of hours (min-1, max-48)

1

Number of nodes (min-1, max-4)

1

Number of cores per node (min-1, max-32)

1

Number of GPUs per node (min-0, max-2)

0

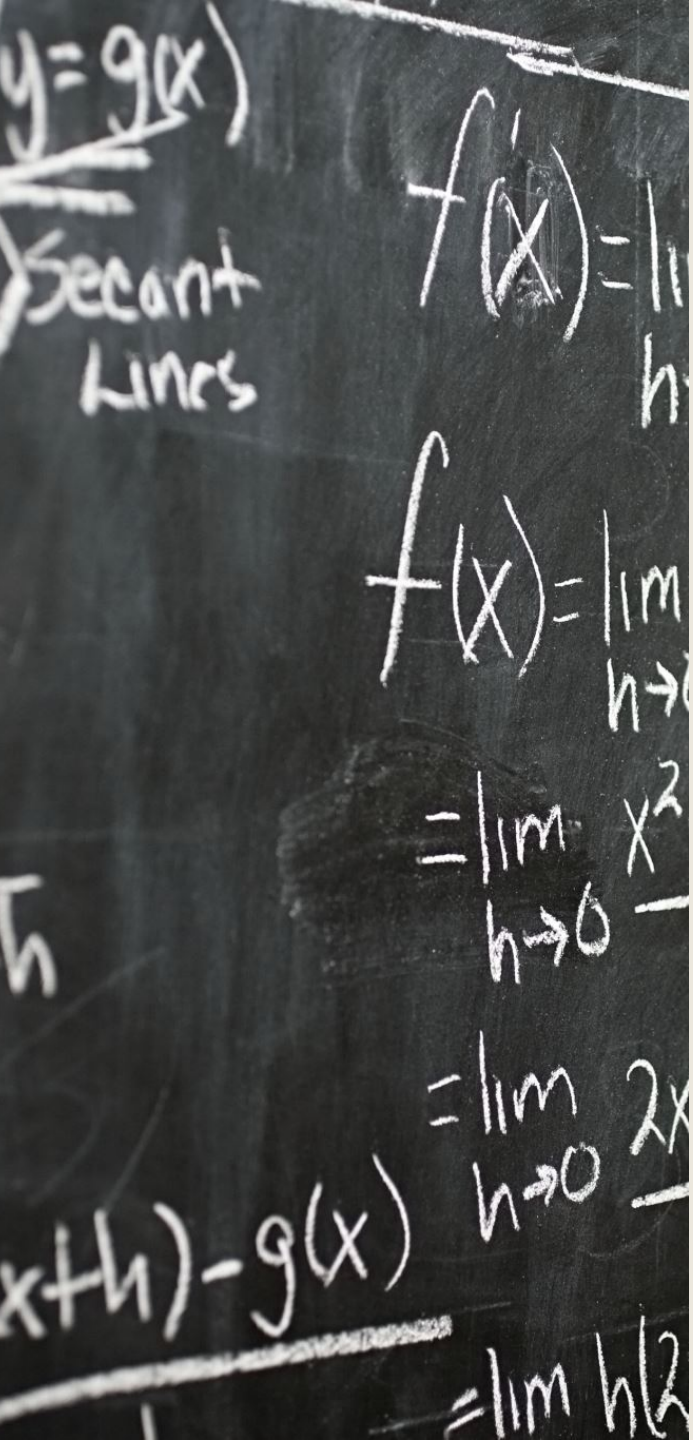
- If requesting GPU nodes, you must enter a GPU-enabled Partition above or the job will fail.

Extra Modules

- Specify additional environment modules here (space delimited)

BASH VARIABLE

Is one of the most important concepts in computer programming



One way to think about variables

In general, variables in computer programming work like variables in algebra

x=7

y=x+5

We can conclude that y=12

How to create a variable

nameofvarible=valueofvarible

To recall the contents of the variable

```
echo $nameofvariable
```

BASH Variables are not physical files

- When you create files you can use `ls` to list contents and see if the file exists.
- A variable will not be listed once created

Use variables as input to commands

Utility of variables

- Variables can be used to store information that can be used later in the script (once or many times over)

```
num=25
file=Mov10_oe_1.subset.fq
```

```
num=25
file=Mov10_oe_1.subset.fq

samplename=`basename`
```

Basename

- **basename** strips directory information and suffixes from **file names** *i.e.* it prints the file name **NAME** with any leading directory components removed.
- For example, in the pathname ~/unix_lesson/raw_fastq/Mov10_oe_1.subset.fq ,

the basename would be "**Mov10_oe_1.subset.fq**"

Basename usage

The command can be easily used by simply writing basename followed by the file name or the file path

$basename NAME

$basename FILE PATH

Application of basename

- It is frequently used in shell scripts for convenience – for example we can **trim off** the file extension or any part of the NAME not wanted
- Can also be used as a **Variable**

Let's put it all together & create an advanced script called **directory_info.sh:**

1. Assign the path of the directory to a variable

2. Create a variable that stores only the directory name (and no path information)

3. Move from the current location in the filesystem into the directory we selected in 1.

4. List the contents of the directory

5. List the total number of files in the directory

Hashtag

- We will be adding comments to this script using the hashtag symbol - #
- Lines in the script that begin with # will not be interpreted as code
- Very important to get into the habit of good documentation