Computer Systems and Networks

ECPE 170 – Dr. Pallipuram– University of the Pacific

Basic BASH Scripting

Slides are courtesy of Dr. Shafer

Lab Schedule

Activities

- **Labs**
  - **Lab 7 – Memory Hierarchy**

Assignments Due

- **Lab 7**
  - **Due by OCT 21st 5 PM**
- ** **Video Presentation #1** **
  - **DUE this THURSDAY**

BASH Scripting

Bash Scripting Exercise

Every bash script usually begins with a **Shebang (#!)** – It is used to specify the absolute path of the bash interpreter

```
#!/bin/bash
```

1. Create a folder inside your home folder called `bash_lab`
2. `cd` to `bash_lab`
3. Gedit a file: `mybash1.sh`
4. Add the above shebang to your new file and save
5. Change the mode of `mybash1.sh` to an executable. (Recall our Linux exercise)

Bash Scripting Exercise: For Loops v1

Add this code to `mybash1.sh`

```
#!/bin/bash
for i in 1 2 3 4 5
do
    echo "Welcome number: $i"
done
```

Execute the script

```
$ ./mybash1.sh
```

Bash Scripting Exercise: For Loops v2

Create a new file called `mybash2.sh` with this improved loop:

```
#!/bin/bash
for ((i=0;i<12;i++))
do
    echo "Welcome number: $i"
done
```

$ replaces the variable with its value

Execute the script

```
$ ./mybash2.sh
```

Can you modify the above code to create folders lab2 to lab12?

Bash Scripting Exercise: Conditionals

Conditional statements in Bash follow this format:

```
if ((<some C-like conditional>))
then
<commands>
else
<other commands>
fi
```

Create a new file called `mybash3.sh` based on `mybase2.sh`.
Modify the code to create folders: lab**0**2, lab**0**3,..,lab**0**9,.. lab12

Bash Arrays

Arrays in Bash follow this format:

```
declare -a arrayname=(element1 element2 element3);
```

Example:

```
declare -a Unix=('Debian' 'Red hat' 'Suse' 'Fedora');
```

Length of an array: `${#ArrayName[@]}`

Accessing an element at i^{th} position: `${ArrayName[i]}`

Bash Arrays

I want to run the amplification program on Lenna_org_1024.pgm with gaussian width = 11 and for sigma values: 0.3, 0.4, …1.1 (totaling 9 executions). Automate these lines:

```
./amplify Lenna_org_1024.pgm 11 0.3 2
./amplify Lenna_org_1024.pgm 11 0.4 2
./amplify Lenna_org_1024.pgm 11 0.5 2
...
./amplify Lenna_org_1024.pgm 11 1.1 2
```

Tip: To turn on “debug mode” in Bash to see variables and commands as they happen, add the line: **`set -x`**

```
#!/bin/bash
function happy_birthday(){
        get cake
        light candles
        open gifts
        i=0
    while cake_count>i

Output=''
for i in {1..4}
do output=$output"Happy Birthday"
 if [ $i -eq 3 ]
 then
  output=$output"DearAaron"

else
 output=$output
     "to you"

echo -e $output
 3
```

```
if [$date +%d%b]-eq '22 Oct'
  then
     happy_birthday

 echo "Happy Birthday Aaron!"

  else
    y=$(date --date'22 oct'+%j)
    x=$(date +%j)
    ((z=${y}-${x}))

      echo  "$(z) days
        until Aaron's
        next birthday!"

        fi
        done
```