CSc 352
Shell Scripting

Benjamin Dicken

Announcements

- Student Course Survey
  - 1 PA grade dropped if response percentage gets to 80% or more
  - Give honest feedback
- PA 9 - Autograder not on gradescope yet

Shell Scripts

Many cases where it is useful to string together multiple bash commands to complete a task

This: NBA and Currency examples from last class!

Can write **bash scripts**.

Programs written in bash

Shell Scripts

- Shell scripts typically use the .sh extension, but ultimately, as with many file types, they're just text file behind-the-scenes
- The first line should always be of the form:

```
#! /path/to/interpreter
```

More specifically:

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

See: https://stackoverflow.com/questions/8967902/ and https://linux.die.net/man/2/execve

Shell Scripts

```
1 #! /bin/bash
2
3 wget https://www.nba.com/suns/roster -O /tmp/team.html 2> /dev/null
4
5 cat /tmp/team.html | sed -rn -e 's/.*roster__player__header__heading">([A-Za-z ]+)<\/h2>/\1/p' > /tmp/names.txt
6
7 echo "Suns player names sorted:"
8
9 cat /tmp/names.txt | sort
```

Shell Scripting Variables

Shell scripts support variables

```
name="Ben"
occupation=Lecturer
echo "${name} is a ${occupation}"
```

By default, the “type” of all variables are basically just strings

- There are attributes, but for now just expect that every variables is just a string
- https://stackoverflow.com/questions/29840525

Command Line Arguments

Special variables for the command line arguments:

```
#! /bin/bash
echo "Your name is: ${1}"
echo "Your occupation is: ${2}"
echo "The command line arguments: ${@}"
```

Modify the script

How could this script be modified to allow the user to specify the team to get the roster for as a command line argument?

```
1 #! /bin/bash
2
3 wget https://www.nba.com/suns/roster -O /tmp/team.html 2> /dev/null
4
5 cat /tmp/team.html | sed -rn -e 's/.*roster__player__header__heading">([A-Za-z ]+)<\/h2>/\1/p' > /tmp/names.txt
6
7 echo "Suns player names sorted:"
8
9 cat /tmp/names.txt | sort
```

Command Substitution

Storing the standard out that a command produces in a variable is useful when scripting with bash

Use **command substitution** with **`$(command)`**

```
temp_files=$(ls /tmp/)
username=$(whoami)
search_results=$(cat roster.txt | grep [A-Z])
```

Loops

Can loop through a sequence of tokens with a for loop

```
for VARIABLE in X Y Z;
do
  echo ${VARIABLE}
done
```

Activity

Rewrite SBT as a shell script (simplified)

Re-implement the SBT script as a shell script
The script should:

- Run make to build a program
- Iterate through test directories
- Check if output matches expected
- Run make clean at the end