Bash Shell Scripts

Disclaimer: The opinions expressed in this presentation are the views of the author and do not reflect the official policy or position of any agency of the U.S. government.

email: mronell@alumni.upenn.edu

Newton Free Library

October 14, 2015

http://www.gnu.org/copyleft/copyleft.en.html

Programming with the BASH shell

Original UNIX shell interpreter (sh) written by Dr. Stephen R. Bourne

- Bourne spent nine years at Bell Labs
- Worked with the Seventh Edition Unix team.
- Ph.D. in mathematics from Trinity College, Cambridge

The Bourne shell (sh) is the foundation for the standard command line interfaces to UNIX.

What is a shell?

A UNIX shell is both a:

- command interpreter
- programming language

Allows GNU utilities to be combined to create functionality

- Files consisting of commands can become commands themselves.
- Allows automation of tasks.

Programming with the BASH shell

Comments start with a # Oglethorpe character

- *# This would be a comment in BASH*
- Scripts often begin with the symbol #!
- Followed by the path to the script interpreter:
  - #!/bin/bash
  - #!/usr/bin/perl
- Determines in which shell the script is executed.

echo: Writing output from a BASH script.

Use *echo* to write output to the terminal

- echo “This is a sample output line.”

echo takes some arguments, like -n which suppresses printing carriage return.

- echo -n “This text “
- echo “ can be continued by the next string.”

printf can also be used to write output to stdout.

Sample Script

```
#!/bin/bash
echo "This program computes the cost to cut a lawn."
echo "Please enter the rectangular lawn LENGTH and
WIDTH in feet: "
read LENGTH WIDTH
echo -n "The total area for a lawn $LENGTH feet in length
by $WIDTH feet wide is $(($LENGTH * $WIDTH))"
echo " square feet."
echo
echo "How much do you wish to charge your customer in
dollars/square foot?" read COSTPERFOOT
echo "At that rate, the customer should be charged: \$
$(($LENGTH * $WIDTH * $COSTPERFOOT)) "
echo "Thank you for using our lawn maintenance cost price
estimator."
```

Sample Program Output example

This program computes the cost to cut a lawn.
Please enter the rectangular lawn LENGTH and WIDTH in feet:
10 15
The total area for a lawn 10 feet in length by 15 feet wide is 150 square feet.

How much do you wish to charge your customer in dollars/square foot?
2
At that rate, the customer should be charged: $ 300
Thank you for using our lawn maintenance cost price estimator.

Sample Program Output example

Arithmetic can be computed by more than one method

- Arithmetic Expansion
  - Use an expression of the form $((*expr*))
  - where expr is the arithmetic expression to be evaluated
  - $((5 * 8 + 4)) would evaluate to 44.
  - At a bash shell type *echo $((5 * 8 + 4))*

BASH shell: Loop Constructs

3 types of loops in Bash

- until
  - *until* test-commands; *do* consequent-commands; *done*
  - execute loop until test-commands are false.
- while
  - *while* test-commands; *do* consequent-commands; *done*
  - execute loop until test-commands are false
- for
  - *for* name [[in [words ...]] ;] *do* commands; *done*
  - for each member in the list, execute the commands

Conditional Statement

Conditional Contructs (if statement)

- The if statement
  if test-commands; *then*
  consequent-commands;
  [*elif* more-test-commands; *then* more-consequents;]
  [*else* alternate-consequents;]
  fi
- The case statement
  case word in [[(] pattern [| pattern]. . .)
  command-list ;;]... *esac*

BASH shell Loop Example: compute factorial

```
echo "This program computes the factorial of a number entered which is <=
10"
echo "Please enter the number:"
read fact
if [ $fact -gt 10 ] || [ $fact -le 0 ]
then
   echo "The number must be greater than 0 and less than or equal to 10."
   echo "Exiting program. Restart program to try again."
   exit
fi
echo "The factorial of $fact is:"
factorial=1
counter=0
while [ $counter -lt $fact ]; do
   let counter=counter+1
   let factorial=$(($factorial * $counter))
done
echo $factorial
```

Programming with the BASH shell

Example Code run that works:

```
This program computes the factorial of a number entered which is <= 10
Please enter the number:
8
The factorial of 8 is:
40320
```

Example Code run that fails:

```
This program computes the factorial of a number entered which is <= 10
Please enter the number:
13
The number must be greater than 0 and less than or equal to 10.
Exiting program. Restart program to try again.
```

BASH shell: String comparison

Table of String Comparison Operators

expression	resulting value
string	True if the length of string is non-zero.
string1 == string2	True if the strings are equal.
string1 = string2	True if the strings are equal.
string1 != string2	True if the strings are not equal.
string1 < string2	True if string1 sorts before string2
string1 > string2	True if string1 sorts after string2

BASH shell: Number/value comparison

Table of Value comparison operators

expression	resulting value
arg1 -eq arg2	True if arg1 equals arg2
arg1 -ne arg2	True if arg1 does not equal arg2
arg1 -lt arg2	True if arg1 less than arg2
arg1 -le arg2	True if arg1 less than or equal to arg2
arg1 -gt arg2	True if arg1 greater than arg2
arg1 -ge arg2	True if arg1 greater than or equal to arg2

BASH shell logic: AND operator &&

The **AND** Operator

expr1	expr2	expr1 && expr2
True	True	True
True	False	False
False	True	False
False	False	False

BASH shell logic: OR operator ||

The **OR** Operator

expr1	expr2	expr1 || expr2
True	True	True
True	False	True
False	True	True
False	False	False