CSE 303
Concepts and Tools for Software Development

Magdalena Balazinska
Winter 2010
Lecture 4 – More shell scripts

Outline

- More shell scripting
  - Shell arithmetics
  - Loops... fancy loops
  - Arrays

Shell Variables (review)

- Assignment using equals sign without spaces
  - `i=42`
  - `q=”What is the answer”`
- Preface a variable by a dollar sign ($) to reference its value
  - `echo $q $i`
  - `a=”The answer is $i”`
- Optionally, enclose in braces
  - `a2=”The answers are ${i}s”`

Arithmetics

- All values held in variables are **strings**
  - But shell will treat them as numbers when appropriate (using 0 if necessary)
- Three ways of performing **integer** arithmetics
  - Method 1: `` i=`expr $i + 1` ``
  - Method 2: `((i=i+1))` or `i=$((i+1))`
  - Method 3: `let “i = i + 1”`
    - Quotes permit the use of spaces
    - No $ signs needed with `let` or inside `((...))`
- Example: arithmetics.sh

For Loop

```
for variable in list
do
...
done
```

- List can be created from
  - Content of an array
  - File pattern
  - Result of a command
- Example: loops.sh

Other Constructs

- `case` statement
- `while` loop
- `until` loop
- `break` and `continue`
- Linux Pocket Guide p 171-175
- Also possible to define functions but we will not discuss them in this class

Arrays

- One dimensional arrays only
- Arrays do not have “fixed sizes” and can be sparse
- Make an array: `foo=(x y z)`
- Set element: `foo[2]=hi`
- Get element: `${foo[2]}`
- Get number of elements: `${#foo[*]}`
- All elements separated by spaces `${foo[*]}`
- Example: arrays.sh

Readings

- Linux Pocket Guide
  - Section on Shell Variables (p. 23-24)
  - Programming with shell scripts (p. 166-179)
    - Especially sections that show for-loops and other programming language constructs
- Online Bash Reference Manual
  - The pointer to the manual is on the class website
  - Section 6.5 Arithmetics
  - Section 6.7 Arrays