Systems Programming & Scripting

Lecture 13: Shell Scripting Basics

Shell Scripting

- Shells allow the user to interact with the system kernel.
- They are programs that handle command lines and run other programs.
- A shell script (program) contains a list of command lines.
- Any program that can be typed in a shell terminal can be included in a shell script.

Different Shell Scripting Syntax sets

- There are two different shell syntax sets:
  - The Bourne shell: more flexible.
  - The C shell: similar to C syntax.
- We will be introduced to the Bourne shell.
- Within the Bourne shell syntax, there are different dialects.
- Bourne-compatible shells include: sh, bash, zsh and ksh.

A Basic Shell Script

```
#! /bin/sh

echo “Hello World”
```

- *#!* is the start of the script (interpreter line).
- *echo* prints its string argument to the standard output.
- Save the script in file *hello.sh*
- The script can be run from a terminal using the following command:

  ./hello.sh

Variables in a Shell Script

- A variable in a shell script is used to refer to a script or a character value.
  - The same variable can be used e.g. to hold a character value and then a numerical value.
- No need to declare variables before using them.
- Variables in scripting languages are usually *untyped*.

Quotation Marks in a Shell Script

- *Single quotation* marks: what is inside the quotation marks will be treated literally including special characters.
- *Double quotation* marks: used for strings that contain special characters that the shell should interpret.
- *Backslash* is used to escape a single character that otherwise will be treated as a special character.

Examples

v = ‘Hello $USER’

echo $v

Hello $USER

v = “Hello $USER”

echo $v

Hello John

v = “The price is \$10”

echo $v

The price is $10

Variables Syntax

- sh-style languages distinguish between a read use of a variable and a write use.
- In a write use, i.e. on the lhs of an assignment, the variable is used without change e.g.

```
i=1
```

- In a read use, i.e. when dereferencing a variable, the variable name should be preceded by a dollar sign.
  - The shell inserts the variable content at that point in the script.

```
echo “the value of i is $i”
```

Script Arguments

- When starting a script from the command line, values can be included in the command after the name of the script.
- Each value passed will be assigned to the special variables $1, $2, $3, …
- The name of the current running script is stored in $0.

Other Special Variables

- $# the number of arguments.
- $* the entire argument string.
- $? the return code of the last command issued.

Example

```
echo “My first name is $1”
echo “My surname is $2”
echo “Total number of arguments is $#”
```

- Assuming that the script is stored in name.sh
- Running the script using the command

  ./name.sh John Smith

  will display

  My first name is John

  My surname is Smith

  Total number of arguments 2

Doing More Than Displaying

```
cd $HOME
echo “removing temp files”
ls –l
rm tmp*
```

- This script automates the following operations:
  - Will switch to the home directory of the user.
  - Lists all files in the directory.
  - Deletes the files that start with the word *tmp.*

Pipes & Filters

- In Linux, pipes connect the standard output of one command to the standard input of another command.
- The vertical bar (|) is used to pipe the commands.
- Example:
  - grep peter students.txt | lpr
  - Will print every line in students.txt that contains the *peter*