SI413: Programming Languages and Implementation

#!/bin/bash

Overview

- Written by Brian Fox for the GNU Project in 1989
- BASH stands for Bourne Again Shell
- BASH is a shell scripting language, perfect for writing command line programs
- Huge amount of online support
- Used to easily automate complex series of commands for easy reuse

Code Examples

```
#!/bin/bash
echo -n "Which fibonacci number do u want to see? "
read nSerial
a=0
b=1
count=2
fibonacci_number=$a
while [ $count -le $nSerial ]; do
        fibonacci_number=$[$a+$b]
        a=$b
        b=$fibonacci_number
        count=$[$count + 1]
done
echo "Fibonacci $nSerial = $fibonacci_number"
```

```
basil[115][~/413/proj2]$ ./fib.sh
Which fibonacci number do u want to see? 1
Fibonacci 1 = 0
basil[116][~/413/proj2]$ ./fib.sh 2
Which fibonacci number do u want to see? 2
Fibonacci 2 = 1
basil[117][~/413/proj2]$ ./fib.sh
Which fibonacci number do u want to see? 1
Fibonacci 1 = 0
basil[118][~/413/proj2]$ ./fib.sh
Which fibonacci number do u want to see? 2
Fibonacci 2 = 1
basil[118][~/413/proj2]$ ./fib.sh
Which fibonacci number do u want to see? 3
Fibonacci 3 = 2
basil[118][~/413/proj2]$ ./fib.sh
Which fibonacci number do u want to see? 4
Fibonacci 4 = 3
basil[118][~/413/proj2]$ ./fib.sh
Which fibonacci number do u want to see? 5
Fibonacci 5 = 5
basil[118][~/413/proj2]$ ./fib.sh
Which fibonacci number do u want to see? 6
Fibonacci 6 = 8
basil[118][~/413/proj2]$
```

```
#!/bin/bash
echo Hello World
```
→ Hello world script

```
diff <(find dir1) <(find dir2)
```
→ Find difference between the contents of 2 directories

```
if [ $file1 -nt  $file2 ]
```
→ Checks if file1 has been modified more recently than file2

Features

- No explicit types
- Supports arrays: no size declaration required
- Redirect stdin and stdout to files
- Flexible parameter passing with functions
- Extensive string manipulation
    - tr command
- Tight integration with operating system
    - Commands executed on the command line can be executed in the shell script

Cool Stuff

- Variables global unless declared otherwise
- Read and write to sockets
- Process substitution
- Multifunctional test command
- Debugging: #!/bin/bash –x
- Can execute most Bourne shell scripts without modification
- Doesn’t support floating point math
- Only supports 1-D arrays

Gotchas

- Use of whitespace in variable assignments
- Mixing up –eq and =
- Assuming uninitialized variables are zero