

Assignment 2: Shell Scripts

Complete assignment due: Thursday, Sept. 11th, 9 p.m.

Problem 1: Hidden Argument

Write a Korn shell script named **hideArg.ksh** that will print those command-line arguments that do not match the first command-line argument. After printing the non-hidden arguments, the script will print a count of the number of hidden arguments found among the 2nd and succeeding arguments.

Some examples:

```
$ ksh hideArg.ksh one two three one four one six
two
three
four
six
There were 2 hidden arguments
$ ksh hideArg.ksh one two "three one" "four one" six
two
three one
four one
six
There were no hidden arguments
$ ksh hideArg.ksh one two three one "four one" six
two
three
four one
six
There was 1 hidden argument
$ ksh hideArg.ksh
There were no hidden arguments
$
```

Turnin: Use the **turnin** program to turn in your completed **hideArg.ksh** script. The command is:

```
turnin 352assign2 hideArg.ksh
```

See the man page for the **turnin** program for details on what turnin can do and how you can confirm that your file was turned in.

Problem 2: One Pipe.

Write a Korn shell script named **onePipe.ksh**. The script will take each pair of command-line arguments and execute them using a pipe to connect the output of the first of the pair to the input of the second. For each pair of commands:

- Print the pair as a pipe.

- Show the result of executing the pair of commands
- State whether the first command produced output on standard error. If there is standard error output from the first command, print it.
- Print the exit status of the second command.
- After each pair of commands, print: a blank line, a row of 20 dashes, and a blank line.

If the number of command-line arguments is not divisible by 2, print an error message and a Usage statement and exit (see examples below). A script can send output to stderr by using `>&2`. For example:

```
echo "Text sent to stderr" >&2
```

will send the string to stderr.

If there are no command-line arguments, the script will simply exit.

Some examples:

```
$ ksh onePipe.ksh "cal 2008" wc
```

```
Executing: cal 2008 | wc
```

```
35      463      2019
```

```
cal 2008 produced nothing on standard error
```

```
Exit status of wc: 0
```

```
-----
```

```
$ ksh onePipe.ksh "cal -z 2008" wc
```

```
Executing: cal -z 2008 | wc
```

```
0        0        0
```

```
cal -z 2008 produced the following error output:
```

```
cal: invalid option -- 'z'
```

```
usage: cal [-lsmjyV] [[month] year]
```

```
Exit status of wc: 0
```

```
-----
```

```
$ ksh onePipe.ksh "cal -3 2008" "wc -z"
```

```
Executing: cal -3 2008 | wc -z
```

```
wc: invalid option -- 'z'
```

```
Try `wc --help' for more information.
```

```
cal -3 2008 produced nothing on standard error
```

```
Exit status of wc -z: 1
```

```
-----
```

```
$ ksh onePipe.ksh "cat /home/cs352/fall08/books/gettysburg.txt" "wc" "ls -l /home/cs352/fall08/books/gettysburg.txt" "wc" "ls -l ~cs352/fall08/books" "fgrep 352"
```

```
Executing: cat /home/cs352/fall08/books/gettysburg.txt | wc
```

```
23      280      1495
```

```
cat /home/cs352/fall08/books/gettysburg.txt produced nothing on standard
error
Exit status of wc: 0
```

```
-----
Executing: ls -l /home/cs352/fall08/books/gettysburg.txt | wc
          1          8          92
```

```
ls -l /home/cs352/fall08/books/gettysburg.txt produced nothing on standard
error
Exit status of wc: 0
```

```
-----
Executing: ls -l ~cs352/fall08/books | fgrep 352
```

```
ls -l ~cs352/fall08/books produced the following error output:
ls: cannot access ~cs352/fall08/books: No such file or directory
Exit status of fgrep 352: 1
```

```
-----
$ ksh onePipe.ksh "cal -3 10 2008"
Arguments must be paired
Usage: onePipe.ksh cmd1 cmd2 [cmd1 cmd2] ...
$ ksh onePipe.ksh
$ cp onePipe.ksh sample.ksh
$ sample.ksh "ls -z"
Arguments must be paired
Usage: sample.ksh cmd1 cmd2 [cmd1 cmd2] ...
$
```

Turnin: Use the **turnin** program to turn in your completed **onePipe.ksh** script. The command is:

```
turnin 352assign2 onePipe.ksh
```

See the man page for the **turnin** program for details on what turnin can do and how you can confirm that your file was turned in.

Final Summary:

There are two files that you will need to turnin for full credit on this assignment:

```
hideArg.ksh
```

```
onePipe.ksh
```