COMP2041 Week 3 Tutorial

Question 1

- run shell: `sh myScript`
- problem: you might not have the current directory in your PATH
- solution: add . to the end of your PATH (via PATH=$PATH:.) or type the command name as ./myScript
- problem: the myScript file might not be executable
- solution: make the file executable (via chmod +x myScript)
  or execute it via the command sh myScript (also fixes the first problem)
- problem: you might have gotten the #!/bin/sh line wrong
- solution: check the line to make sure there are no spurious spaces or spelling mistakes and then check that the shell is actually called /bin/sh on your system
- problem: the myScript file has been transferred from a Windows-based computer in binary mode, and there's a ^M ('\r' in the C context) after /bin/sh
- solution: run the standard command dos2unix MyScript which will remove the pesky ^Ms.

Shell Variables

- $0 the name of the command
- $1 the first command-line argument
- $2 the second command-line argument
- $3 the third command-line argument
- $# count of command-line arguments
- $* all of the command-line arguments (together)
- $@ all of the command-line arguments (separately)
- $? exit status of the most recent command
- $$ process ID of this shell
- single-quote (’) grouping, turns off all transformations
- double-quote (") grouping, no transformations except $ and ‘
- backquote (‘) no grouping, capture command results

`quotes.sh`

Before Anything Else

- `shebang` or `hashbang`
- wildcard in `*, ?, []`
- talk about test, `[]` , `[[]]` `link.txt`

- talk about `expr.sh`
- talk about `arg1.sh arg2.sh`

Question 2

```
start=1
step=1

if [ $# -eq 1 ]
then
    stop=$1
elif [ $# -eq 2 ]
then
    start=$1
    stop=$2
elif [ $# -eq 3 ]
    start=$1
    step=`expr $2 - $1`
    stop=$3
fi

while [ $start -le $stop ]
do
    echo "$start"
    start=`expr $start + $step`
done
```

Question 3

```
#!/bin/sh

for file in *.html
do
    # note use of -i to ignore case and -w to ignore white space
    # however tags containing newlines won't be detected
    # why dev/null?
    # use -q instead
    # LAB HINT for checking a thing is integer
    if egrep -i '</?blink>' $file >/dev/null
    then
        echo  "Removing $file because it uses the <blink> tag"
        rm "$file"
    fi
done
```

Question 4

```
for file in *.c
do
    echo "$file includes:"
    egrep '^#include' "$file"|  # find '#include lines
    sed 's/[">][^">]*$//'|      # remove the last '"' or '>' and anything after it
    sed 's/^.*["<]/    /'       # remove the first '"' or '>' and anything before i
done
```

Question 5

No, we need to sort it based on family name field, but every person has different number of initials. One possibility is to make family name as first field.

Question 6

a. `head -3 /etc/passwd`
b. `egrep '^(cs|se|bi|en)[0-9]' /etc/passwd`
c. `grep '/bin/bash' passwd | cut -d':' -f1`
d. `cut -d':' -f1,2 passwd | tr ':' '\t' > passwords.txt`

Question 7

- the script doesn't concatenate files named on the command line, just standard input
- it doesn't implement all of the cat options (e.g. no -m)
- the appearance of lines may be altered (space at start of line is removed, and runs of multiple spaces will be compressed to a single space)

```
#!/bin/sh -x    for debugging
test -r "$f" or [ -r "$f" ]
man test
```

```
#!/bin/sh -x
for f in "$@"
do
    if [ ! -r "$f" ]
    then
        echo "No such file: $f"
    else
        while read line
        do
            echo "$line"
        done <$f
    fi
done
```

Question 8

```
#!/bin/sh

for f in "$@"
do
    echo "==== `echo $f | sed 's/\.txt//'` ===="
    cat "$f"
done
```

Question 9

```
#!/bin/sh

while read zid name init
do
    mark=`grep $zid Marks | cut -d' ' -f2`
    echo "$mark $name $init"
done
```

```
sort -n students | join marks - | cut -d' ' -f2,4,5 | sort -k2
```

Question 10

```
#!/bin/sh
while read stid mark extras
do
    case "$mark" in
    [0-9]*) ;;
    *)      echo "$stid ?? ($mark)"
            continue
            ;;
    esac

    if test $mark -lt 50
    then
        echo $stid FL
    elif test $mark -le 100
    then
        echo $stid HD
    else
        echo "$stid ?? ($mark)"
    fi
done
```

Question 11

```
#!/bin/sh

current_month=`date | cut -c8-10`

while test "$current_month = `date | cut -c8-10`"
do
    date
    sleep 5
done
```

Question 12

```
wc -l *.tex             // different lines
echo `wc -l *.tex`      // make it in one line, no difference in content
```

Question 13

```
#!/bin/sh

LIMIT=50

for f in *
do
    if test -d "$f"
    then
        continue
    fi

    count=`wc -c < "$f"`
    if [ $count -gt $LIMIT ]
    then
        echo "$f"
    fi
done
```

Question 14

```
-d deletes space
-c complement
-s squeeze multiple occurrences to one
```

a. 2
b. 11
c. `4 BIG 1 BUT 2 IS 1 NOT 1 SO 2 THIS`

replaces not alphanumeric to new line, and squeeze all consecutive new lines with just one

Question 15

a. a b c
b. a b c
c. Y Y
d. x2
e. `can use ${x}x`
f. Y Y
g. $y `single quotes prevent expansion`
h. Y: command not found
i. a b c
j. a b c