Shell Scripting 2020: Week 5

Stefan Ciprian Voinea
Student number: 015383372

December 5, 2020

37. **Counting in the shell**

Contents of the `task37_bc_average.sh` file:

```
#!/bin/bash

sum=0
for i in $@
do
    sum=$(($i + $sum))
done

result=$(echo "scale=2; 1.0 * $sum / $#" | bc -l)

echo "The mean of these " $# " values is " $result
```

Output of the execution:

```
 cip  ~/Desktop/UNI/ShellScripting2020/Week5  master ±  ./task37_bc_average.sh 1 2 3 4 5
The mean of these  5  values is  3.00
 cip  ~/Desktop/UNI/ShellScripting2020/Week5  master ±  ./task37_bc_average.sh 123 3534
The mean of these  2  values is  1828.50
 cip  ~/Desktop/UNI/ShellScripting2020/Week5  master ±  ./task37_bc_average.sh 123 353 4564 123123 789789 21321 123
The mean of these  7  values is  134199.42
 cip  ~/Desktop/UNI/ShellScripting2020/Week5  master ±  |
```

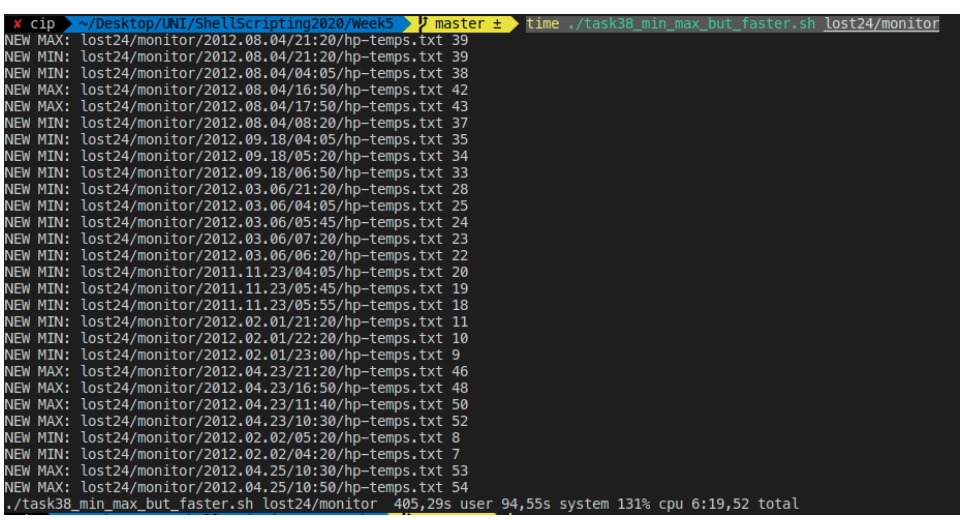
38. **Gone in 10 seconds**

Contents of the `task38_min_max_but_faster.sh` file:

```
#!/bin/bash

dir=$1

max_file=""
max=0

min_file=""
min=99

while read file
do
    temp=`grep "PROCESSOR_ZONE *[0-9]\+C" $file | cut -b 32-33`
    temp=`echo $temp | sed -e "s/C//g"`

    if [ $temp -gt $max ]
    then
        max=$temp
        max_file=$file
        echo "NEW MAX: $max_file $max"
    fi
    if [ $temp -lt $min ]
    then
        min=$temp
        min_file=$file
        echo "NEW MIN: $min_file $min"
    fi

done < <( for file in `find $dir -type f -name "*hp-temps.txt"` ; do echo $file ; done )
```

Unfortunately I was not able to solve this exercise in order to make it run under 10 seconds. I measured the run time by executing the command as:

`time ./task38_min_max_but_faster.sh lost24/monitor`



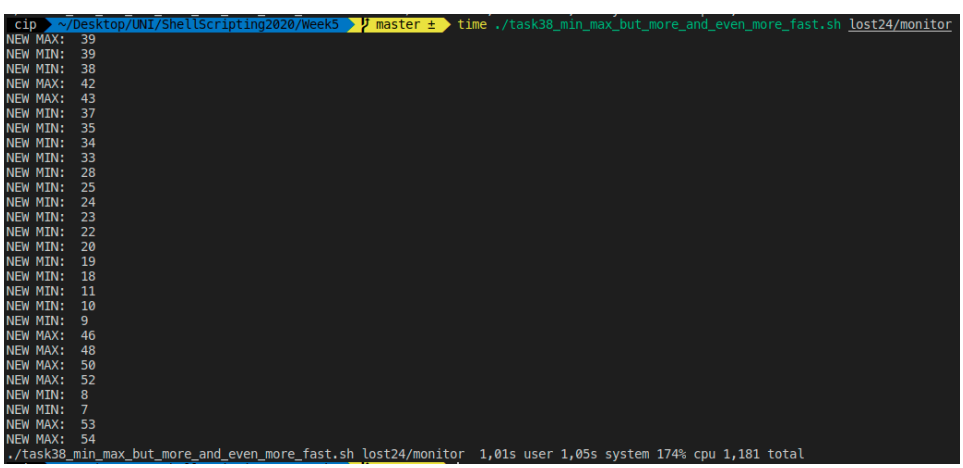
Then something happend and I got a better idea on how to execute the script, here it is ...

Contents of the `task38_min_max_but_more_and_even_more_fast.sh` file:

```
#!/bin/bash

input_dir=$1

max_file=""
max=0

min_file=""
min=99

while read temp
do
    if [ $temp -gt $max ]
    then
        max=$temp
        max_file=$file
        echo "NEW MAX: $max_file $max"
    fi
    if [ $temp -lt $min ]
    then
        min=$temp
        min_file=$file
        echo "NEW MIN: $min_file $min"
    fi

done < <( find $input_dir -name '*temps.txt' -exec grep 'PROCESSOR' {} \+ | cut -b 75-79 |
↪  sed -e "s/C//g" | sed -e "s/\///g" )
```

I measured the run time by executing the command as:

```
time ./task38_min_max_but_more_and_even_more_fast.sh lost24/monitor
```



39. **Hipstafy-dropbox**

Contents of the `task39_dropbox.sh` file:

```
#!/bin/bash

while read line
do
    ./task39_dropbox_hipstafy.sh "$line"
done < <( inotifywait -qm --format "%f" --event create dropbox/. )
```

Contents of the `task39_dropbox_hipstafy.sh` file:

```
#!/bin/bash

input_dir="dropbox"
input_file="$1"

output_dir="$input_dir/hipstafied"
mkdir -p $output_dir

image_name=`basename $input_dir/$input_file`
image_prefix=${image_name%.jpg}
output_file=$output_dir/$image_prefix-hipstah.jpg
echo "Converting ... $output_file"
convert -sepia-tone 60% +polaroid $input_dir/$input_file $output_file
```

Example input file:

Example output file:

40. **Summoning deamons**

Contents of the `task40_summon_task39.sh` file:

```
#!/bin/bash

pid_file="task40_summon_task39_pid.txt"

while true
do

    echo "1) Start"
    echo "2) Stop"
    echo "3) Status"
    echo "4) Restart"

    printf "What do you want to do? "
    read -r choice

    case $choice in

        1) echo "Starting daemon"
            deamon=nohup $(./task39_dropbox.sh) & echo $! > $pid_file
            echo ""
            ;;
        2) echo "Stopping daemon"
            kill `cat $pid_file`
            break
            echo ""
            ;;
        3) echo "Getting daemon status"
            if pgrep -F $pid_file
            then
                echo "Daemon running"
            else
                echo "Daemon not running"
            fi
            ;;
        4) echo "Restarting daemon"
            kill `cat $pid_file`
            deamon=nohup $(./task39_dropbox.sh) & echo $! > $pid_file
            ;;
        *) echo "Invalid choice. Please try again"
            ;;
    esac

    echo ""

done
```