Lab #6: Script Language

Objectives: the following subjects will be covered:

- Bash scripting language
- GPIO

Part1: *Advanced script*

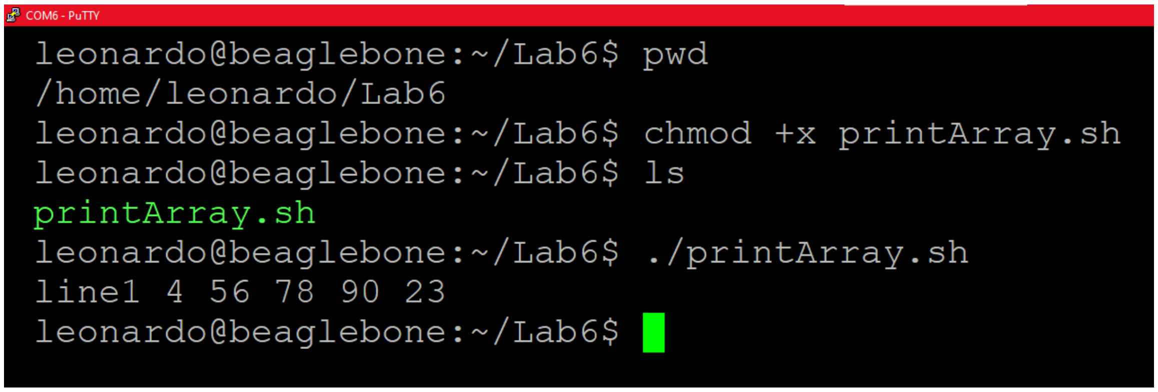
Array and for-loop:

1. In a script file, create an array of values separated by a space. See example below:

```
line1 4 56 78 90 23
```

2. Write a script to print all values as follows:

```
root@beaglebone:/home# ./printArray.sh
line1 4 56 78 90 23
```

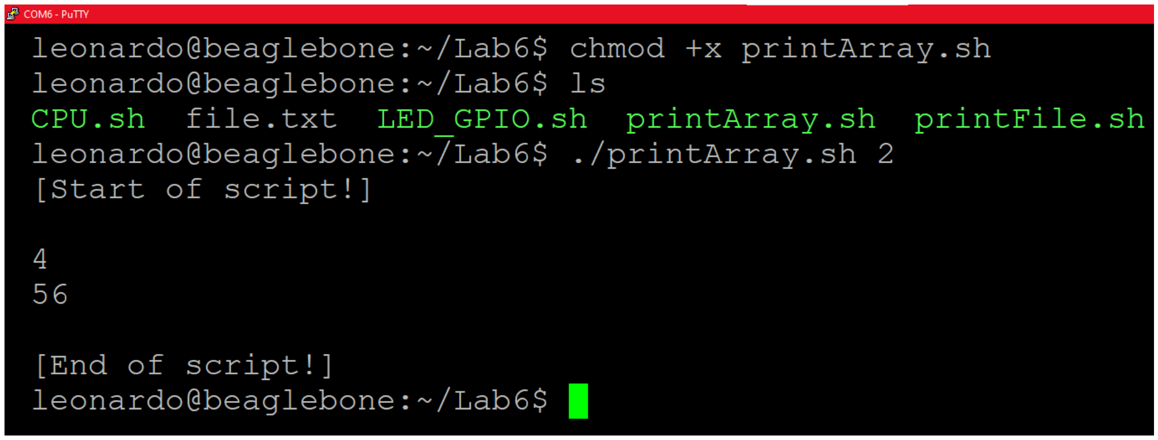


```
leonardo@beaglebone:~/Lab6$ pwd
/home/leonardo/Lab6
leonardo@beaglebone:~/Lab6$ chmod +x printArray.sh
leonardo@beaglebone:~/Lab6$ ls
printArray.sh
leonardo@beaglebone:~/Lab6$ ./printArray.sh
line1 4 56 78 90 23
leonardo@beaglebone:~/Lab6$ 
```

Due to size of script, script is attached separately to this submission

3. Modify the script to print only some values:

```
root@beaglebone:/home# ./printArray.sh 3
4 78 23
```



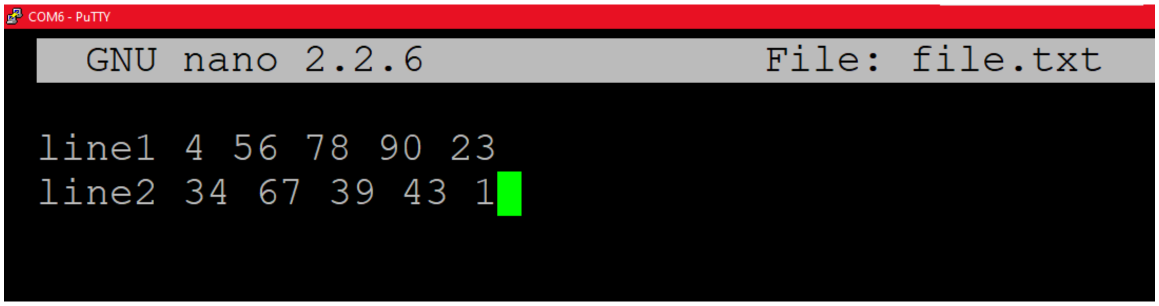
```
leonardo@beaglebone:~/Lab6$ chmod +x printArray.sh
leonardo@beaglebone:~/Lab6$ ls
CPU.sh  file.txt  LED_GPIO.sh  printArray.sh  printFile.sh
leonardo@beaglebone:~/Lab6$ ./printArray.sh 2
[Start of script!]

4
56

[End of script!]
leonardo@beaglebone:~/Lab6$ 
```

Due to size of script, script is attached separately to this submission

CAT command:

4. Create a file “file.txt” that contains the following values:

```
line1 4 56 78 90 23
line2 34 67 39 43 1
```



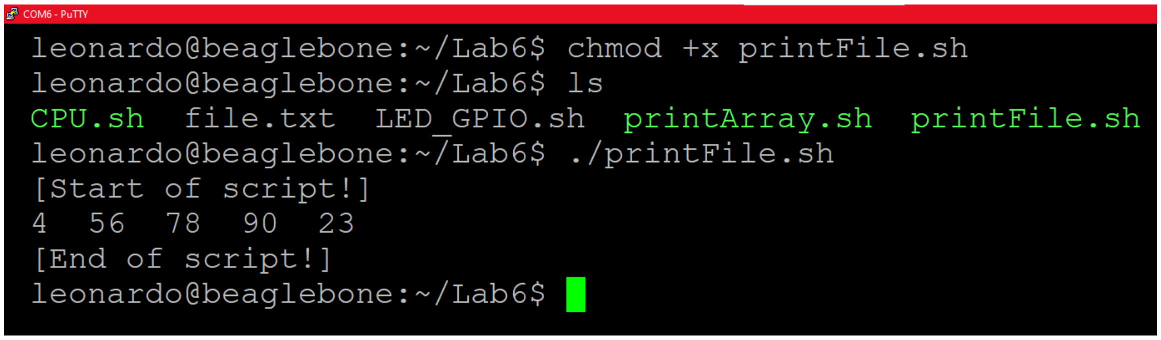
COM6 - PuTTY

```
  GNU nano 2.2.6                File: file.txt

line1 4 56 78 90 23
line2 34 67 39 43 1
```

File “file.txt” that contains two lines of text

5. Modify the previous script file using CAT to copy the file content to an array.
   Print only numerical values of the first line to the screen. Words like `line1` and `line2` must be suppressed.

```
root@beaglebone:/home# ./printFile.sh
4 56 78 90 23
```



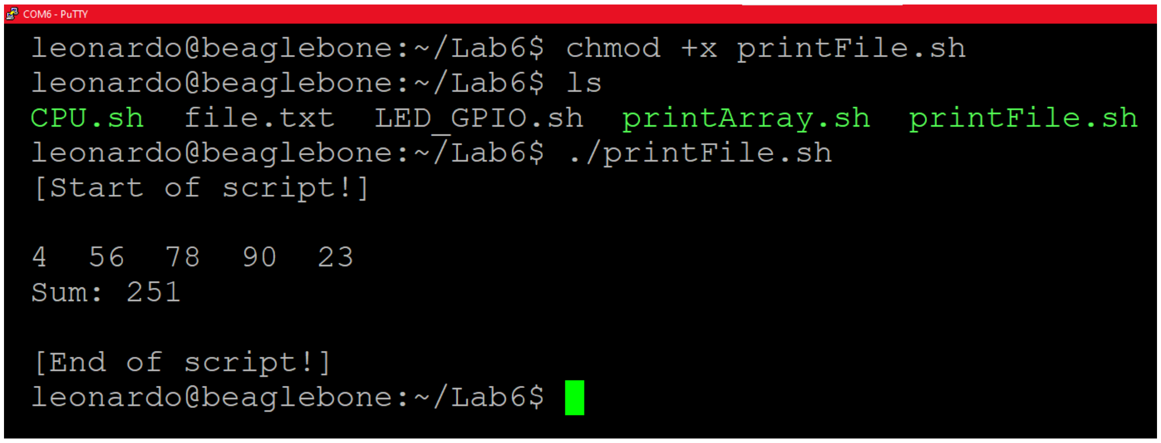
COM6 - PuTTY

```
leonardo@beaglebone:~/Lab6$ chmod +x printFile.sh
leonardo@beaglebone:~/Lab6$ ls
CPU.sh  file.txt  LED_GPIO.sh  printArray.sh  printFile.sh
leonardo@beaglebone:~/Lab6$ ./printFile.sh
[Start of script!]
4  56  78  90  23
[End of script!]
leonardo@beaglebone:~/Lab6$
```

Due to size of script, script is attached separately to this submission

6. Add some code to your script file to print the sum of all values for the first line only:

```
root@beaglebone:/home# ./printFile.sh
4 56 78 90 23
sum: 251
```



COM6 - PuTTY

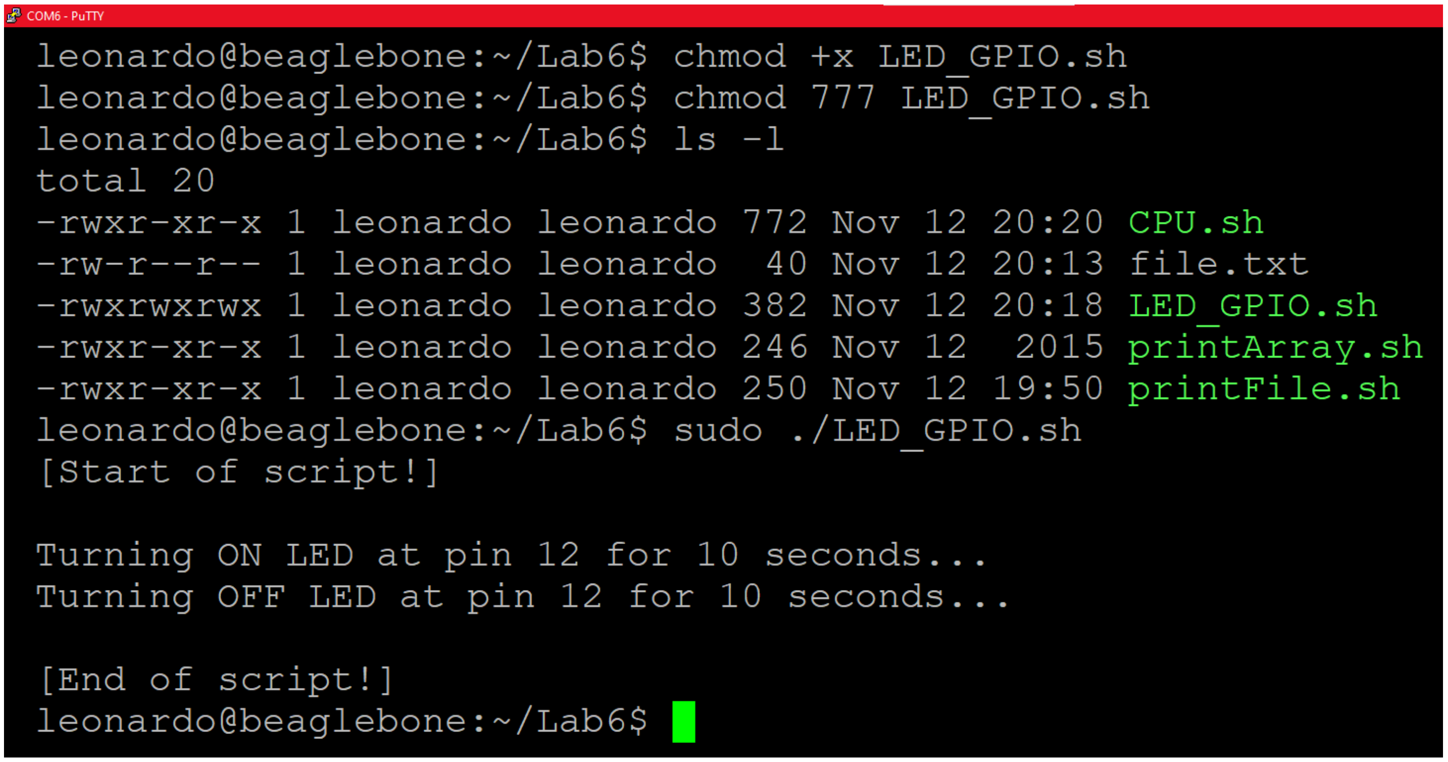
```
leonardo@beaglebone:~/Lab6$ chmod +x printFile.sh
leonardo@beaglebone:~/Lab6$ ls
CPU.sh  file.txt  LED_GPIO.sh  printArray.sh  printFile.sh
leonardo@beaglebone:~/Lab6$ ./printFile.sh
[Start of script!]

4  56  78  90  23
Sum: 251

[End of script!]
leonardo@beaglebone:~/Lab6$
```

Due to size of script, script is attached separately to this submission

Part2: GPIO

1. Write a script file to turn on an LED using a GPIO available.

   Remember to
   - Export
   - Set direction
   - Set the value
   - Wait a few seconds
   - Turn off LED and
   - unexport



COM6 - PuTTY

```
leonardo@beaglebone:~/Lab6$ chmod +x LED_GPIO.sh
leonardo@beaglebone:~/Lab6$ chmod 777 LED_GPIO.sh
leonardo@beaglebone:~/Lab6$ ls -l
total 20
-rwxr-xr-x 1 leonardo leonardo 772 Nov 12 20:20 CPU.sh
-rw-r--r-- 1 leonardo leonardo  40 Nov 12 20:13 file.txt
-rwxrwxrwx 1 leonardo leonardo 382 Nov 12 20:18 LED_GPIO.sh
-rwxr-xr-x 1 leonardo leonardo 246 Nov 12  2015 printArray.sh
-rwxr-xr-x 1 leonardo leonardo 250 Nov 12 19:50 printFile.sh
leonardo@beaglebone:~/Lab6$ sudo ./LED_GPIO.sh
[Start of script!]

Turning ON LED at pin 12 for 10 seconds...
Turning OFF LED at pin 12 for 10 seconds...

[End of script!]
leonardo@beaglebone:~/Lab6$ 
```

Due to size of script, script is attached separately to this submission

2. (BONUS) Improve the code to read from a sensor and light up a LED using GPIOs

Part3: Small script application

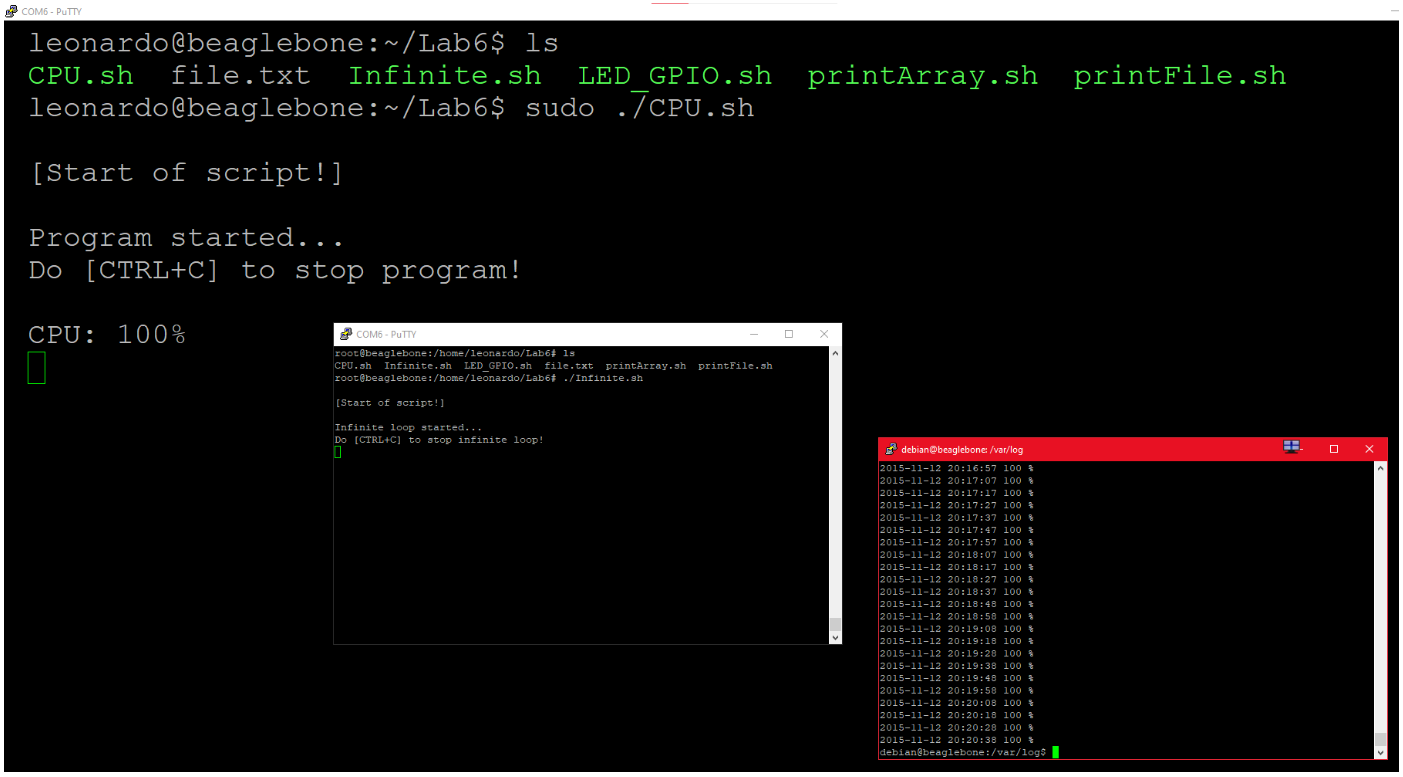
The following scrip reads the CPU loads from `/proc/stat` and displays it on the console:

```
#!/bin/bash
# by serge hould

PREV_TOTAL=0
PREV_IDLE=0
MAX=90

while true; do
        # Get the total CPU statistic.
        CPU=($(cat "/proc/stat"))
        IDLE=${CPU[4]} # Just the idle CPU time.

        # Calculate the total CPU time.
        TOTAL=0
        let "TOTAL=${CPU[1]}+${CPU[2]}+${CPU[3]}+${CPU[4]}"
         # Calculate the CPU usage since we last checked.
        let "DIFF_IDLE=$IDLE-$PREV_IDLE"
        let "DIFF_TOTAL=$TOTAL-$PREV_TOTAL"
        let "DIFF_USAGE=(1000*($DIFF_TOTAL-$DIFF_IDLE)/$DIFF_TOTAL+5)/10"
        echo -en "\b\b"
        echo -en "\rCPU: $DIFF_USAGE%"

        # Remember the total and idle CPU times for the next check.
        PREV_TOTAL="$TOTAL"
        PREV_IDLE="$IDLE"
        # Wait before checking again.
        sleep 1
done
```

3. Improve the script so that if the CPU is overloaded (>90%) the script turns on all user LEDs and logs a message to the `/var/log/syslog` every 10 seconds. The message logged must include a date and time and load in %.

To overload the CPU, simply write a c program looping infinitely and call it from another terminal or write another script file to do the same.

```
void main(void){
    while(1);
}
```



COM6 - PuTTY

```
leonardo@beaglebone:~/Lab6$ ls
CPU.sh  file.txt  Infinite.sh  LED_GPIO.sh  printArray.sh  printFile.sh
leonardo@beaglebone:~/Lab6$ sudo ./CPU.sh

[Start of script!]

Program started...
Do [CTRL+C] to stop program!

CPU: 100%
```

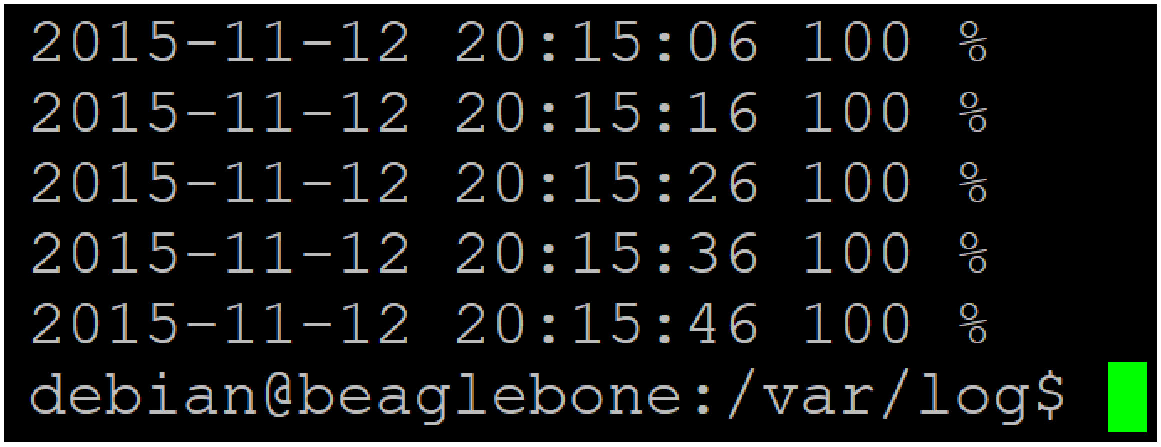
COM6 - PuTTY

```
root@beaglebone:/home/leonardo/Lab6# ls
CPU.sh  Infinite.sh  LED_GPIO.sh  file.txt  printArray.sh  printFile.sh
root@beaglebone:/home/leonardo/Lab6# ./Infinite.sh

[Start of script!]

Infinite loop started...
Do [CTRL+C] to stop infinite loop!
```

debian@beaglebone: /var/log

```
2015-11-12 20:16:57 100 %
2015-11-12 20:17:07 100 %
2015-11-12 20:17:17 100 %
2015-11-12 20:17:27 100 %
2015-11-12 20:17:37 100 %
2015-11-12 20:17:47 100 %
2015-11-12 20:17:57 100 %
2015-11-12 20:18:07 100 %
2015-11-12 20:18:17 100 %
2015-11-12 20:18:27 100 %
2015-11-12 20:18:37 100 %
2015-11-12 20:18:48 100 %
2015-11-12 20:18:58 100 %
2015-11-12 20:19:08 100 %
2015-11-12 20:19:18 100 %
2015-11-12 20:19:28 100 %
2015-11-12 20:19:38 100 %
2015-11-12 20:19:48 100 %
2015-11-12 20:19:58 100 %
2015-11-12 20:20:08 100 %
2015-11-12 20:20:18 100 %
2015-11-12 20:20:28 100 %
2015-11-12 20:20:38 100 %
debian@beaglebone:/var/log$
```

<u>First screen</u>: CPU script, <u>Second screen</u>: Infinite loop script, <u>Third screen</u>: `/var/log/syslog`
Due to size of scripts, scripts are attached separately to this submission



```
2015-11-12 20:15:06 100 %
2015-11-12 20:15:16 100 %
2015-11-12 20:15:26 100 %
2015-11-12 20:15:36 100 %
2015-11-12 20:15:46 100 %
debian@beaglebone:/var/log$
```

`/var/log/syslog` *showing CPU usage every 10 seconds*