# 

# Computer Architecture and Technology Convergence Assignment

Table of Contents

[Computer Architecture and Technology Convergence Assignment 1](#_Toc7614000)

[Q1: Binary Arithmetic: 2](#_Toc7614001)

[Q1.1. 2](#_Toc7614002)

[Q1.2. 2](#_Toc7614003)

[Q1.3. 4](#_Toc7614004)

[Q1.4. 5](#_Toc7614005)

[Q1.5. 6](#_Toc7614006)

[Q2: Linux Assignment 7](#_Toc7614008)

[Q2.1: 7](#_Toc7614009)

[Q2.2: 10](#_Toc7614010)

[Q2.3: 22](#_Toc7614103)

[Q2.3.1: 22](#_Toc7614104)

Q2.3.2: 23

[Q2.4: 24](#_Toc7614105)

[References 26](#_Toc7614106)

## Q1: Binary Arithmetic:

Feel free to use any resources you need for the tasks below, but make sure to show workings.

Q1.1. Add 11011 to 1011. Show your work (in particular, show where you get carries, and where you don't). You can check your work by translating the numbers into decimal, but I want to see the addition algorithm in base 2 instead of base ten. Hint: You can use MS Word tables to show calculations. Go to Insert->Table to insert a grid of desired size. References: (McGinley, 2019)(McGinley, 2019).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Carry |  |  | 1 | 1 | - | 1 | 1 | - |
|  |  |  |  | 1 | 1 | 0 | 1 | 1 |
| Sign: + |  |  |  |  | 1 | 0 | 1 | 1 |
| Total: |  |  | 1 | 0 | 0 | 1 | 1 | 0 |

Q1.2. Rewrite the following base-10 numbers as 8-bit two's complement integers: -31, & -59.

Using 2s complements means that all numbers are stored as positive binary numbers. Half of the numbers are encoded as positive and the other half negative. How they are interpreted is through use of denoting the left-most bit as the determinant of sign whereby:

* Positive Numbers left-most-bit ==0
* Negative Numbers left-most-bit == 1

To calculate what the value of a binary number if it is interpreted as an 8-bit two’s complement integer?

1. Firstly, we find the equivalent binary number to the positive value of the decimal. (keep dividing by 2 to get binary(what’s left over)\_ then reverse this number).
2. If the decimal was positive no change is needed. But in this case both do.
3. If the decimal was negative – invert all the bits and add 1 ( with carries as needed).

References: (McGinley, 2019)

|  |
| --- |
| -31 |
| Step 1 | Step 2 | Step 2 |
| Convert positive to binary i.e. 31. | No change if positive. | a. Invert all bits  b. add 1 |
| 31/2  15 \_\_\_\_\_\_\_\_1  15/2  7 \_\_\_\_\_\_\_\_\_1  7/2  3 \_\_\_\_\_\_\_\_\_1  3/2  1 \_\_\_\_\_\_\_\_\_1  ½  0 \_\_\_\_\_\_\_\_\_1 | Step 2 is Skipped as the numbers sign is negative. | 00011111  a. 11100000  b. 11100001 |
| 8-Bit #  Ans: 00011111 |  | 8-bit 2s Compliment  Ans: 11100001 |
|  |  |  |
| -59 |
| Step 1 | Step 2 | Step 2 |
| Convert positive to binary i.e. 59. | No change if positive. | a. Invert all bits  b. add 1 |
| 59/2  29 \_\_\_\_\_\_\_\_1  29/2  14 \_\_\_\_\_\_\_\_1  14/2  7 \_\_\_\_\_\_\_\_\_0  6/2  3 \_\_\_\_\_\_\_\_\_1  3/2  1 \_\_\_\_\_\_\_\_\_1  ½  0 \_\_\_\_\_\_\_\_\_1 | Step 2 is Skipped as the numbers sign is negative. | 00111011  a. 11000100  b. 11000101 |
| 8-Bit #  Ans: 00111011 |  | 8-bit 2s Compliment  Ans: 11000101 |

Q1.3. What does the bit pattern 11101001 represent if you interpret it as an 8-bit two's complement integer?

Now we need to do what we did in the previous question however carry it out in reverse to obtain the binary number for both positive and negative values of the 2s compliment.

(Steps 1&2)

Following this we must convert the 2 binary numbers to decimal (Base 10). (Step 3)

References: (McGinley, 2019) (McGinley, 2019).

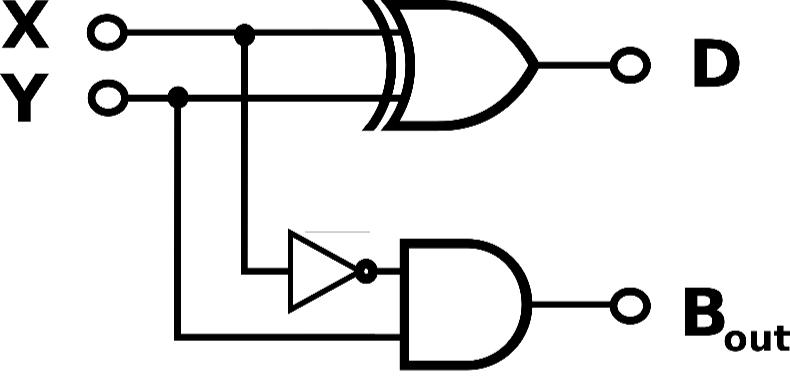
|  |  |
| --- | --- |
| Step 1 | Step 2 |
| If the number was originally positive in sign. | a. Invert all bits  b. remove 1 |
| Value of the 2s compliment does not change. | 11101001  a. 00010110  b. 00010111 |
| 8-Bit Positive binary #  Ans 1: 11101001 | Ans 2: For a negative Binary #: 00010111 |

|  |
| --- |
| Step 3: Conversion of binary values to decimal. |
| Ans 1: For the 8-Bit Positive binary value of the 2s Compliment : 11101001 |
| 11101001 = (1 × 2⁷) + (1 × 2⁶) + (1 × 2⁵) + (0 × 2⁴) + (1 × 2³) + (0 × 2²) + (0 × 2¹) + (1 × 2⁰) = 233 |
| Ans 1: For the 8-Bit Negative binary value of the 2s Compliment : 00010111 |
| 00010111 = (0 × 2⁷) + (0 × 2⁶) + (0 × 2⁵) + (1 × 2⁴) + (0 × 2³) + (1 × 2²) + (1 × 2¹) + (1 × 2⁰) = 23 = -23  #Note: Remember the sign on this value will be negative. |

**Answers:**

233, -23

Q1.4. Draw up the truth table for the circuit below (inputs are X and Y and outputs are B and D). From observing the result, what function do you think this circuit performs?



Firstly, I wanted to figure out the Boolean Logic Equations so as I could understand what was going on in this circuit diagram. From this I drew out the diagrams in logic.ly to easily see what would happen in the truth table. References: (McGinley, 2019), (Academo.org, 2019) (Logic.ly, 2019) (Das, 2019)

|  |  |  |
| --- | --- | --- |
| D = (X' Y + XY') = X ⊕ Y | XOR GATE | Accepts 2 inputs. If both are the same output = 0. If inputs differ output = 1. |
| B = X' Y | AND GATE | 2 Inputs: Input X through a NOT GATE (X'). Input Y straight into AND GATE. |

**Truth Table**

|  |  |  |  |
| --- | --- | --- | --- |
| Inputs | | Outputs | |
| X | Y | D | Bout |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

The circuit seems to perform a subtraction between 2 bits. In the diagram of the provided circuit above it shows 2 inputs and 2 outputs. It can be seen two produces the differences between the 2 binary input bits at the output.

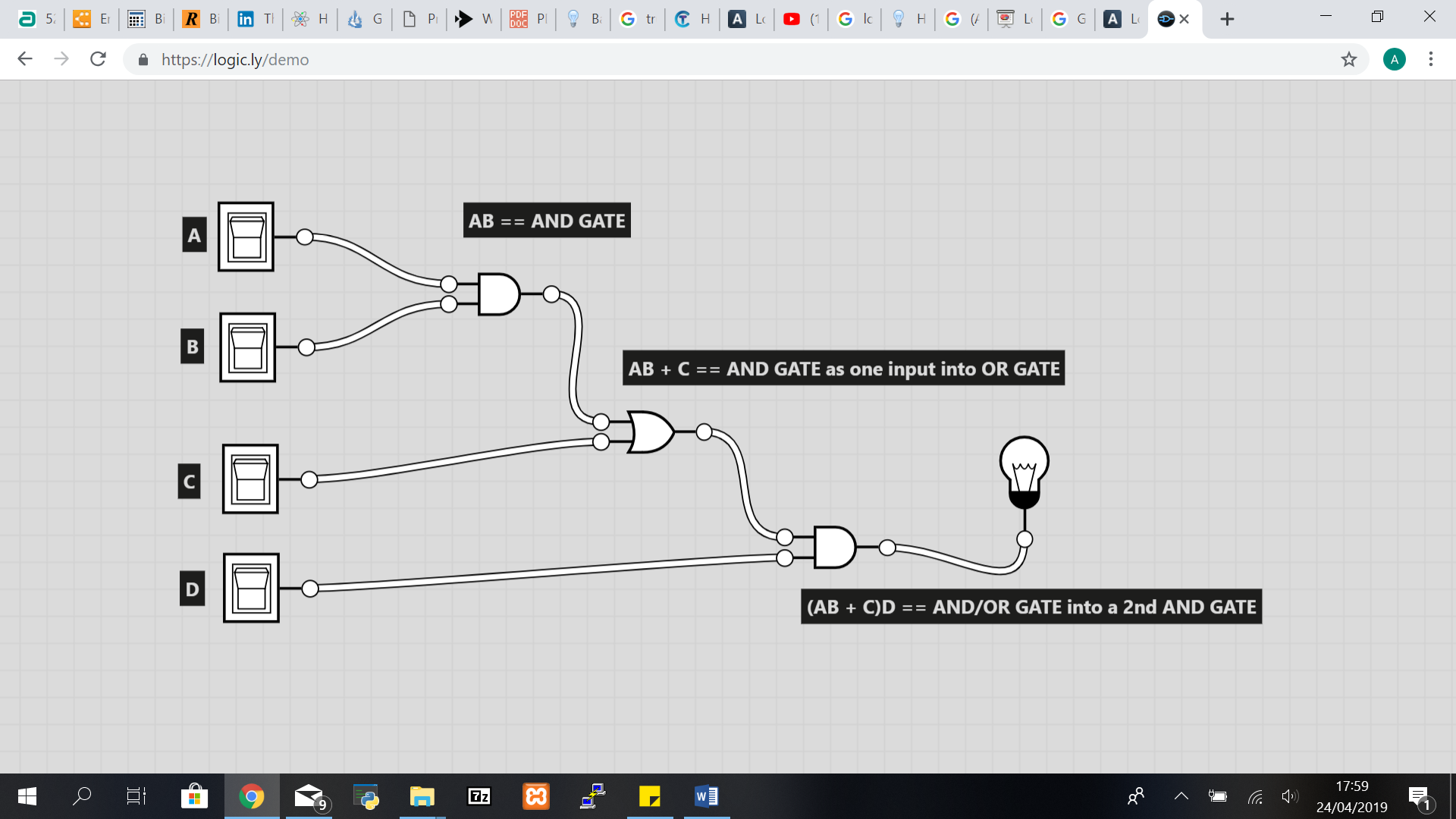
Q1.5. Draw the circuit diagram for the Boolean logic equation: (AB + C)D (Hint: you can use an online logic gate simulator such as: <https://academo.org/demos/logic-gate-simulator/> and screen-capture your drawing)

I used the logic.ly/demo circuit simulator to draw out the Boolean expression first as I found it was easier with the logic simulator as it allowed the labelling of each of the inputs, outputs and gates in-between. For the purpose of curiosity, I then carried this out using the recommended academo to see the differences between the two simulators.

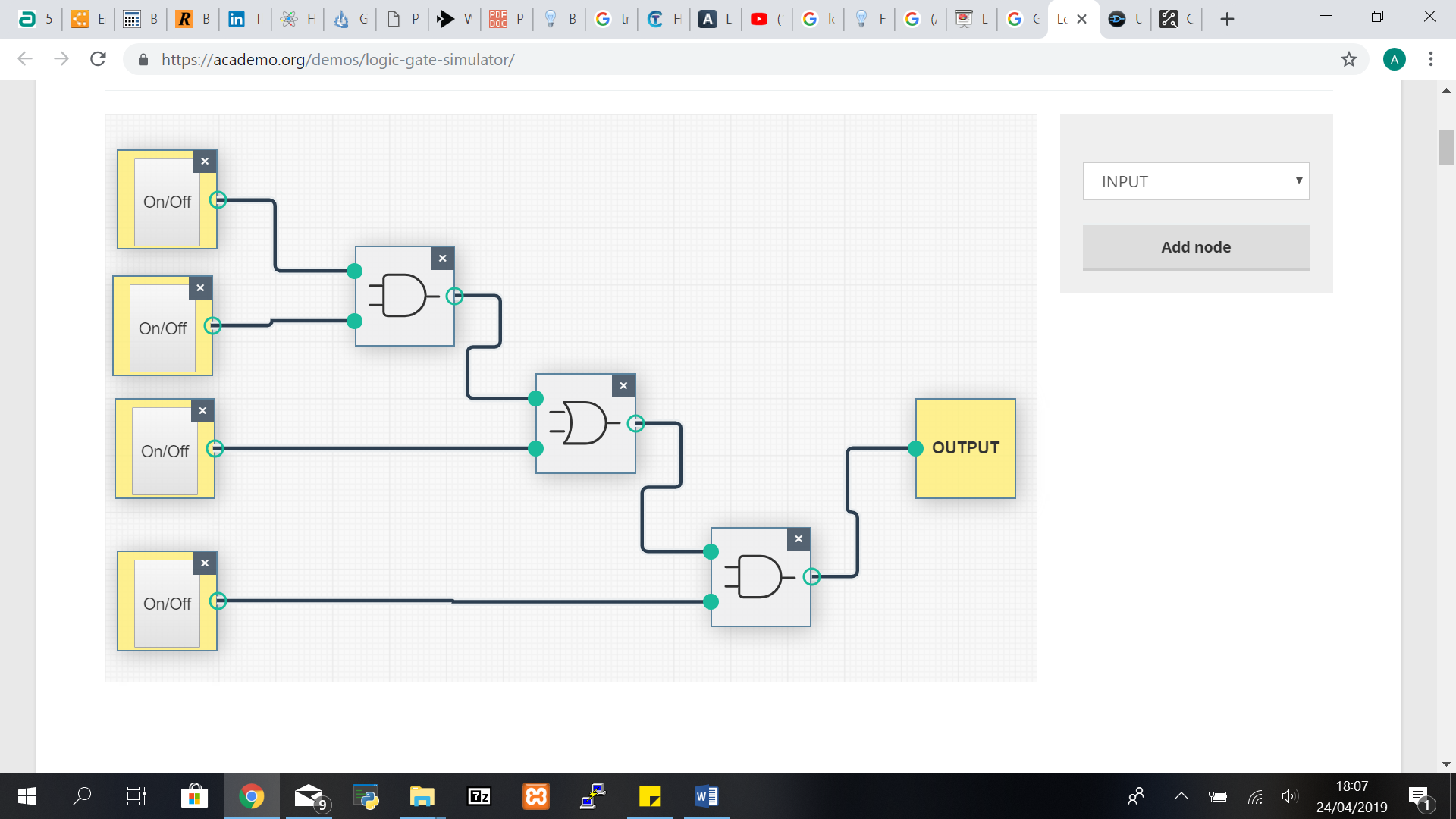
References: (McGinley, 2019). (Academo.org, 2019) (Logic.ly, 2019)

**#(AB + C)D Circuit Diagram using logic.ly/demo**

## 



**(AB + C)D Circuit Diagram using academo.org/demos**



## Q2: Linux Assignment

### Q2.1:

Enter the commands below at the Linux terminal on the AWS VM (on which you completed your Linux Homework) and try to interpret the output.

For the submission:

In your own words, write a brief description of what each command does. Make sure to use Google as a resource and don't be afraid to experiment (as a normal user you cannot do much harm). Also include a screenshot of the output of the history command. Copy the screenshot into your “.docx” document for submission.

Commands:

1. echo hello world



o passwd



o date\*



o hostname\* o arch\*



o uname -a\*



1. dmesg | more(you may need to press q to quit) o uptime\*



o whoami\* o who\*



o last



o finger\*



1. w\*



1. top\* (you may need to press q to quit) o echo $SHELL



o echo {con,pre}{sent,fer}{s,ed}



1. man ls (you may need to press q to quit)



1. man who (you may need to press q to quit) o clear



o cal 2000



1. cal 9 1752(do you notice anything unusual. Why is this the case?)



1. yes please(you may need to press Ctrl-c to quit) o time sleep 5



o history\*



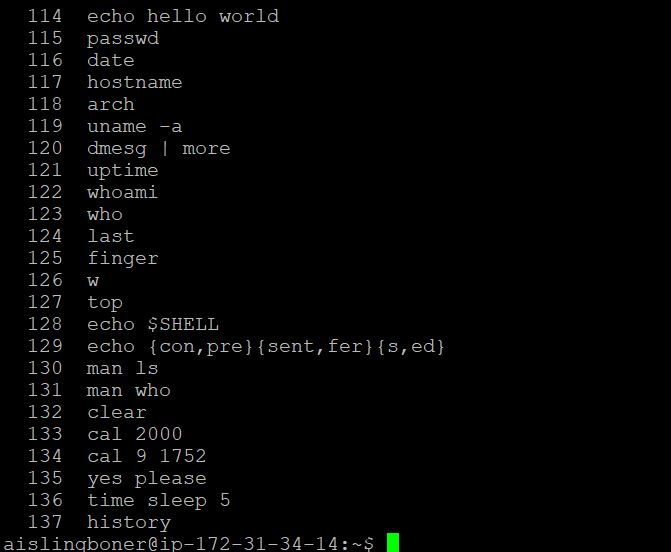
## 

**Tested Commands for Q2.1:**

References: (Maker Pro, 2019), (Files.fosswire.com, 2019), (GeeksforGeeks, 2019)

|  |  |  |
| --- | --- | --- |
| # | Commands: | Output  Interpretation: |
| 1 | echo hello world | Prints hello world. |
| 2 | passwd | Asks you to Change your password. |
| 3 | date\* | Displays todays date. |
| 4 | hostname\* | Displays ip address of host. |
| 5 | arch\* | Prints the Architecture of the system. |
| 6 | uname -a\* | Displays all information about the system usernames beginning with a. |
| 7 | dmesg | more  (you may need to press q to quit) | Writes kernel message buffer in Linux to output. |
| 8 | uptime\* | Used to show how long the system was active (running). |
| 9 | whoami\* | Displays the current user’s username. |
| 10 | who\* | Shows which users are currently logged on to the system. |
| 11 | last | Shows the last logged in users. |
| 12 | finger\* | Used to find User Details. |
| 13 | w\* | Provides a quick summary of every user logged into a computer, what there doing, and the load all this activity is having on the computer. (similar to who) |
| 14 | top\* (you may need to press q to quit) | Provides a real time view of the system’s top processes that are running. |
| 15 | echo $SHELL | Shows your default Shell (The hidden folder for the shell). |
| 16 | echo {con,pre}{sent,fer}  {s,ed} | Combines the syllables in brackets to create words. |
| 17 | man ls (you may need to press q to quit) | Displays the manual info about the “ls” command. |
| 18 | man who (you may need to press q to quit) | Displays the manual info about the “who” command. |
| 19 | clear | Clears the previous commands and presents a black screen. |
| 20 | cal 2000 | Displays the calendar for the year 2000. |
| 21 | cal 9 1752  (do you notice anything unusual. Why is this the case?) | Displays the calendar for September 1752.  Its missing 2 weeks of its dates for that month. |
| 22 | yes please  (you may need to press Ctrl-c to quit) | Continuously Repeats “please”. |
| 23 | time sleep 5 | Shows the system idle time. The real, user and system. |
| 24 | history\* | Shows the history of commands entered. |

Screenshot of the output of the history command:



### Q2.2:

This is a research project. Use Google to help you identify a solution.

For each of the commands marked with an \*, group them into a shell script so that you can automate execution of the commands. Write the shell script using the Vim text editor.

Once you have verified that the script works, add output redirection to append the output of each command to a file named as follows: **firstnameSurname.txt** (replacing firstname and surname with your own details). When writing to this text file, make liberal use of the echo command within the shell script to format the output nicely – i.e. insert blank lines or other demarcations and headings to make your file easily readable.

For the submission:

Copy and paste the contents of this auto-created “.txt” file into your “.docx” document for submission. You are also required to upload the shell script which you wrote and the text file which it generated.

date\*



hostname\*



arch\*



uname -a\*



uptime\*



whoami\*



who\*



finger\*



w\*



top\* (you may need to press q to quit)



history\*



Shell Script used for Q2.2 (aislingBoner.sh) **– Uploaded for Submission.**

References: (YouTube, 2019), (Vim.org, 2019), (UNIX tutorials, VC++ tutorials, VM ware tutorials, 2019), (How-To Geek, 2019), (Radford.edu, 2019), (Vim.fandom.com, 2019), (Medium, 2019).

|  |  |
| --- | --- |
| #!/bin/bash################################################################Script Name : Project/projectQ2.2#Description : A Script which runs several commands.#Author : Aisling Boner#Student Number : G00376422#Email : aislingboner@gmail.com##############################################################echoecho "#########################################"echo "Research Project for Q2.2 "echo "#########################################"echo echo echoecho "-----------------------------------------"echo "Current Date & Time: "echo "-----------------------------------------"echodateechoechoecho "-----------------------------------------"echo "Host IP address: "echo "-----------------------------------------"echohostnameechoecho "-----------------------------------------"echo "Computer Architecture Specs: "echo "-----------------------------------------"echoarchechoechoecho "-----------------------------------------"echo "Computer Architecture Specs: "echo "-----------------------------------------"echouname -aechoechoecho "-----------------------------------------"echo "Current Activity Length Period: "echo "-----------------------------------------"echouptime | echoechoecho "-----------------------------------------"echo "Current User's Username: "echo "-----------------------------------------"echowhoamiechoechoecho "-----------------------------------------"echo "All Users Who are Logged in: "echo "-----------------------------------------"echowhoechoechoecho "-----------------------------------------"echo "User Details: "echo "-----------------------------------------"echofingerechoechoecho "-----------------------------------------"echo "Login,Activily & Computer Usage Summary: "echo "-----------------------------------------"echowechoechoecho "-----------------------------------------"echo "Top Proccesses Currently Running: "echo "-----------------------------------------"echotop#(you may need to press q to quit)echoechoecho "-----------------------------------------"echo "History of Commands Inputted by User: "echo "-----------------------------------------"echohistoryecho "#########################################"echo "#########################################" |

Shell Script Output to (aislingBoner.txt) – Uploaded in Submission.

##########################################

Research Project for Q2.2

##########################################

------------------------------------------

Current Date:

------------------------------------------

Tue Apr 30 18:10:06 UTC 2019

------------------------------------------

Host IP Address:

------------------------------------------

ip-172-31-34-14

------------------------------------------

Computer Architecture Specs:

------------------------------------------

x86\_64

------------------------------------------

System User's Information:

------------------------------------------

Linux ip-172-31-34-14 4.4.0-1075-aws #85-Ubuntu SMP Thu Jan 17 17:15:12 UTC 2019 x86\_64 x86\_64 GNU/Linux

------------------------------------------

Current Activity Length Period:

------------------------------------------

18:10:06 up 43 days, 7:21, 10 users, load average: 0.00, 0.00, 0.00

------------------------------------------

Current User's Username:

------------------------------------------

aislingboner

------------------------------------------

All Users that are Logged in:

------------------------------------------

aislingboner pts/0 2019-04-30 16:43 (37.228.235.97)

janhazincak pts/1 2019-04-30 18:00 (37.228.227.94)

alessandraguerrarios pts/2 2019-04-30 14:16 (87.198.171.156)

vanessaromano pts/3 2019-04-30 17:42 (146.247.40.253)

margaretlynch pts/5 2019-04-30 09:13 (89.127.21.72)

edwardryan pts/7 2019-04-30 16:59 (109.76.124.238)

lisaroessel pts/8 2019-04-30 16:18 (87.198.171.156)

petermcguinn pts/9 2019-04-30 17:49 (94.230.99.36)

marysteed pts/10 2019-04-30 12:41 (51.171.48.98)

petermcguinn pts/11 2019-04-30 17:12 (94.230.99.36)

------------------------------------------

User Details:

------------------------------------------

Login Name Tty Idle Login Time Office Office Phone

aislingboner pts/0 Apr 30 16:43 (37.228.235.97)

alessandraguerrarios pts/2 1:31 Apr 30 14:16 (87.198.171.156)

edwardryan pts/7 44 Apr 30 16:59 (109.76.124.238)

janhazincak pts/1 Apr 30 18:00 (37.228.227.94)

lisaroessel pts/8 33 Apr 30 16:18 (87.198.171.156)

margaretlynch pts/5 8:42 Apr 30 09:13 (89.127.21.72)

marysteed pts/10 1:23 Apr 30 12:41 (51.171.48.98)

petermcguinn pts/9 Apr 30 17:49 (94.230.99.36)

petermcguinn pts/11 21 Apr 30 17:12 (94.230.99.36)

vanessaromano pts/3 Apr 30 17:42 (146.247.40.253)

------------------------------------------

Login, Activity & Computer Usage Summary:

------------------------------------------

18:10:06 up 43 days, 7:21, 10 users, load average: 0.00, 0.00, 0.00

USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT

aislingb pts/0 37.228.235.97 16:43 4.00s 0.14s 0.00s w

janhazin pts/1 37.228.227.94 18:00 35.00s 0.06s 0.02s vim Irishman.sh

alessand pts/2 87.198.171.156 14:16 1:31m 0.09s 0.00s less alessandra

vanessar pts/3 146.247.40.253 17:42 6.00s 0.08s 0.08s -bash

margaret pts/5 89.127.21.72 09:13 8:42m 0.03s 0.03s -bash

edwardry pts/7 109.76.124.238 16:59 44:14 0.11s 0.11s -bash

lisaroes pts/8 87.198.171.156 16:18 33:06 0.07s 0.07s -bash

petermcg pts/9 94.230.99.36 17:49 30.00s 0.08s 0.08s -bash

marystee pts/10 51.171.48.98 12:41 1:23m 0.24s 0.24s -bash

petermcg pts/11 94.230.99.36 17:12 21:50 0.28s 0.03s vim

------------------------------------------

Top Processes Currently Active:

------------------------------------------

top - 18:10:06 up 43 days, 7:21, 10 users, load average: 0.00, 0.00, 0.00

Tasks: 163 total, 1 running, 161 sleeping, 1 stopped, 0 zombie

%Cpu(s): 0.1 us, 0.2 sy, 0.0 ni, 99.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st

KiB Mem : 1014436 total, 125148 free, 137064 used, 752224 buff/cache

KiB Swap: 0 total, 0 free, 0 used. 640236 avail Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND

1 root 20 0 185436 6048 4032 S 0.0 0.6 1:20.64 systemd

2 root 20 0 0 0 0 S 0.0 0.0 0:00.03 kthreadd

3 root 20 0 0 0 0 S 0.0 0.0 0:40.23 ksoftirqd/0

5 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kworker/0:0H

7 root 20 0 0 0 0 S 0.0 0.0 1:03.80 rcu\_sched

8 root 20 0 0 0 0 S 0.0 0.0 0:00.00 rcu\_bh

9 root rt 0 0 0 0 S 0.0 0.0 0:00.00 migration/0

10 root rt 0 0 0 0 S 0.0 0.0 0:18.43 watchdog/0

11 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kdevtmpfs

12 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 netns

13 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 perf

14 root 20 0 0 0 0 S 0.0 0.0 0:00.00 xenwatch

15 root 20 0 0 0 0 S 0.0 0.0 0:00.00 xenbus

17 root 20 0 0 0 0 S 0.0 0.0 0:01.03 khungtaskd

18 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 writeback

19 root 25 5 0 0 0 S 0.0 0.0 0:00.00 ksmd

20 root 39 19 0 0 0 S 0.0 0.0 0:08.66 khugepaged

21 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 crypto

22 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kintegrityd

23 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 bioset

24 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kblockd

25 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 ata\_sff

26 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 md

27 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 devfreq\_wq

30 root 20 0 0 0 0 S 0.0 0.0 0:01.06 kswapd0

31 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 vmstat

32 root 20 0 0 0 0 S 0.0 0.0 0:00.00 fsnotify\_ma+

33 root 20 0 0 0 0 S 0.0 0.0 0:00.00 ecryptfs-kt+

49 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kthrotld

50 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 bioset

51 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 bioset

52 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 bioset

53 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 bioset

------------------------------------------

History of Commands Inputted by the User:

------------------------------------------

1 ls

2 /Music ls

3 ls Music/

4 cd Music

5 mkdir Guns N Roses/

6 ls

7 ../

8 rm Music/

9 rm /Music

10 ls

11 tree

12 rmdir Music/

13 cd Music

14 ls

15 rm

16 rm Guns

17 rmdir Guns

18 ls

19 rmdir John

20 lx

21 ls

22 rmdir N

23 Rmdir Roses

24 ls

25 rmdir Roses

26 rmdir Mayer

27 ls

28 cd ../

29 ls

30 $clear

31 ls

32 cd Music

33 ls

34 mkdir Queen

35 mkdir Metallica

36 mkdir Slipknotcd../

37 ls

38 rmdir Slipknotcd..

39 ls

40 mkdir AltJ

41 ls

42 /AltJ

43 cd AltJ

44 mkdir An Awesome Wave

45 ls

46 rmdir An

47 ls

48 rmdir Awesome

49 rmdir Wave

50 ls

51 mkdir 'An Awesome Wave'

52 ls

53 cd An Awesome wave

54 ls

55 tree

56 cd AltJ

57 ls

58 cd music

59 cd Music

60 cd AltJ

61 MKDIR Relaxer

62 mkdir Relaxer

63 ls

64 ../

65 cd ../

66 cd../

67 ls

68 tree

69 cd Metallica

70 mkdir Master Of Puppets

71 mkdir Ride the Lightning

72 cd ../

73 tree

74 cd Music

75 cd Queen

76 mkdir Jazz

77 mkdir Innuendo

78 cd ../

79 tree

80 cd Music

81 cd Metallica

82 deltree

83 rmdir

84 rmdir Metallica

85 cd ../

86 rmdir Metallica

87 cd Metallica

88 rmdir Lightning

89 rmdir the

90 rmdir Master

91 rmdir of

92 rmdir Of

93 rmdir Ride

94 cd ../

95 tree

96 cd Music/Metallica

97 rmdir Puppets

98 cd ../

99 rmdir Metallica

100 cd ../

101 tree

102 cd Music

103 mkdir 'John Mayer'

104 ls

105 cd John Mayer

106 cd 'John Mayer'

107 mkdir Continuum

108 mkdir 'Battle Studies'

109 cd ../

110 tree

111 echo hello world

112 passwd

113 clear

114 echo hello world

115 passwd

116 date

117 hostname

118 arch

119 uname -a

120 dmesg | more

121 uptime

122 whoami

123 who

124 last

125 finger

126 w

127 top

128 echo $SHELL

129 echo {con,pre}{sent,fer}{s,ed}

130 man ls

131 man who

132 clear

133 cal 2000

134 cal 9 1752

135 yes please

136 time sleep 5

137 history

138 ./morning

139 #!/bin/bash

140 clear

141 chmod 775

142 echo "Good morning, worl

143 clear

144 q

145 cat / etc/shells

146 ./first.sh

147 bash First.sh

148 chmod +x project.sh

149 clear#

150 clear

151 cd

152 ls

153 ../

154 mkdir ProjectQ2.2

155 ls

156 cd ProjectQ2.2/

157 vim project.sh

158 echo SHELL

159 echo $SHELL

160 ./

161 man ls

162 info bash

163 cat>first

164 ./first

165 chmod +x first

166 ./first

167 first

168 cat >first

169 clear

170 cat > first

171 ./first

172 clear

173 cat >ProjectQ2.2

174 rmdir ProjectQ2.2

175 ls

176 cd ProjectQ2.2

177 ls

178 rmdir

179 rmdir --help

180 cd ./

181 cd ../

182 rmdir -p

183 ls

184 ProjectQ2.2 remove script

185 clear

186 rm ProjectQ2.2/

187 ../

188 ls

189 cd ProjectQ2.2/

190 rm

191 ../

192 rm

193 cd ./

194 ../

195 history

196 ls

197 cd ../

198 ./

199 cd ../

200 clear

201 history

202 clear

203 ../

204 cd ../

205 ls

206 cd home

207 ls

208 cd aislingboner/

209 ls

210 cd first

211 rmdir first

212 rm first

213 ls

214 rmdir ProjectQ2.2/

215 cd ProjectQ2.2/

216 ls

217 rm

218 rm --help

219 rm -r

220 cd ./

221 cd ../

222 ls

223 rm -r ProjectQ2.2/

224 ls

225 cat > projectQ2.2

226 ./projectQ2.2

227 chmod +x projectQ2.2

228 ./projectQ2.2

229 ls

230 cd projectQ2.2

231 ./projectQ2.2

232 sudo apt-get install vim

233 vim

234 clear

235 vim test

236 : echo $MYVIMRC

237 VIM

238 VIM test

239 clear

240 vim

241 vim ProjectQ2.2

242 ls

243 ./ProjectQ2.2

244 ./projectQ2.2

245 cd ../

246 ls

247 cd aislingboner/

248 ls

249 rmdir ProjectQ2.2

250 rm ProjectQ2.2

251 ls

252 vim projectQ2.2

253 ./projectQ2.2

254 hostname

255 vim projectQ2.2

256 ./projectQ2.2

257 chmod 755 projectQ2.2

258 vim projectQ2.2

259 ./projectQ2.2

260 vim projectQ2.2

261 ./projectQ2.2

262 which prove

263 which projectQ2.2

264 touch~/bin/projectQ2.2

265 ls ~bin/

266 clear

267 touch ~/bin/datecp

268 ls ~/bin/

269 vim projectQ2.2

270 ./projectQ2.2

271 vim projectQ2.2

272 ./projectQ2.2

273 vim projectQ2.2

274 ./projectQ2.2

275 vim projectQ2.2

276 ./projectQ2.2

277 vim projectQ2.2

278 ./projectQ2.2

279 vim projectQ2.2

280 ./projectQ2.2

281 vim projectQ2.2

282 ./projectQ2.2

283 vim projectQ.sh

284 ls

285 cd ../

286 ls

287 cd aislingboner/

288 ls

289 projectQ2.2 > C:\Users\aisli\Desktop\Project\aislingBoner.txt

290 projectQ2.2 >aislingBoner.txt

291 ./projectQ2.2

292 ls

293 cd aislingBoner.txt

294 cd bash

295 cd ../

296 ls

297 cd aislingboner/

298 ls

299 rm aislingBoner.txt

300 ls

301 rm C:UseraisliDesktopProjectaislingBoner.txt

302 ls

303 cd C:UseraisliDesktopProjectaislingBoner.txt

304 ./projectQ2.2

305 ls

306 rm C:UseraisliDesktopProjectaislingBoner.txt

307 ls

308 cd C:UseraisliDesktopProjectaislingBoner.txt

309 ls -l

310 cdmod 755 C:UseraisliDesktopProjectaislingBoner.txt

311 chmod 755 C:UseraisliDesktopProjectaislingBoner.txt

312 chmod 755 C:User\aisli\Desktop\Project\aislingBoner.txt

313 projectQ2.2>aislingBoner.txt

314 projectQ2.2

315 /bin/bash/projectQ2.2

316 ./projectQ2.2

317 cd bash

318 locate .bash

319 ls .bash

320 cd ../

321 cd aislingboner

322 ls

323 rm aislingBoner.txt

324 ls

325 clear

326 vim projectQ2\_2.sh

327 ls

328 mkdir Project

329 ls

330 mv projectQ2.2 Project

331 ls

332 cd Project/

333 ls

334 ./

335 ./projectQ2.2

336 cd ./

337 ls

338 cd ./

339 cd ../

340 clear

341 ./projectQ2.2

342 ls

343 cd Project

344 ./projectQ2.2

345 ../

346 cd ../

347 vim projectQ2.2

348 vim Project/projectQ2.2

349 clear

350 cd

351 ls

352 rm projectQ2\_2.sh

353 ls

354 mkdir Remove

355 ls

356 mv .txt Remove

357 ls .txt

358 rm -rf ./\*.txt

359 ls

360 rmdir Remove

361 ls

362 Project>aislingBoner.txt

363 cd Project

364 projectQ2.2>aislingBoner.txt

365 ls

366 ./projectQ2.2

##########################################

##########################################

## Q2.3:

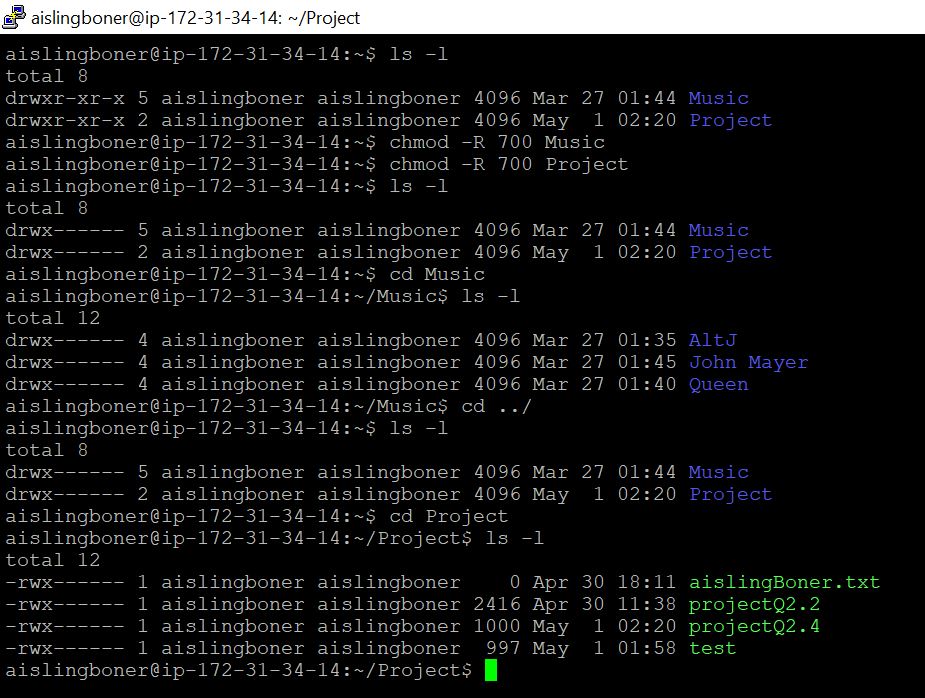
### Q2.3.1:

When a user account is created on Linux, it is public to all users of the machine by default. That is, anyone can view your personal files. Change the access permissions using the ‘chmod’ command so that your personal folder is fully accessible to you (read, write and execute privileges) and totally inaccessible to the group and all other users.

Look at the lecture notes to work out how to do this.

Run the “ls –l” command to verify that the permissions have been set correctly (**Screenshot the** **result from this for submission**).

**Screenshot Image for changing access permissions in my personal folders to only be fully accessible to me (read, write, execute):**



References: (Pluralsight.com, 2019), (Maketecheasier.com, 2019), (Access.redhat.com, 2019).

Q2.3.2:

Because there is no GUI installed on the VM, all programs have to run instead in text mode.

Use the ‘lynx’ text-based browser program on the VM. To run it, type:

lynx [www.google.com](http://www.google.com/)

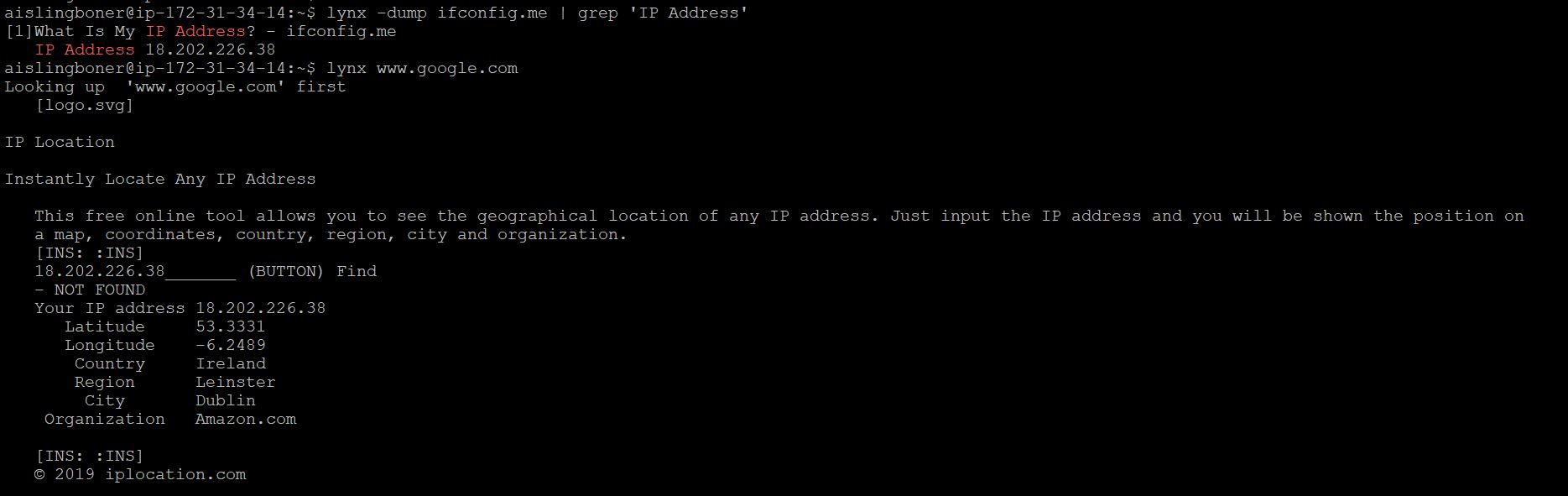
In the lynx browser, search for an online IP location service to determine the city and country where the VM is located. Type the IP address of the VM into the IP locator website and it will tell you the VM’s location (**Screenshot the result from this for submission**).

For the submission:

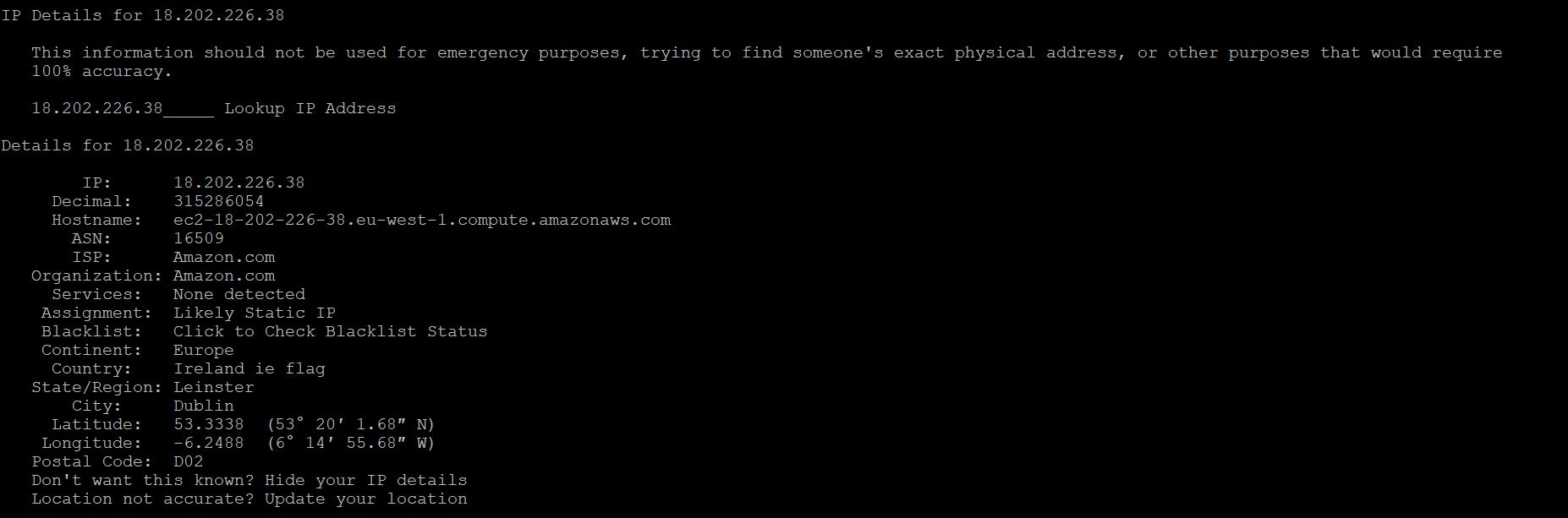
Copy the screenshots above into your “.docx” document for submission.

**Screenshot Image from my 1st Search for my IP Address Locator Website:**

References: (AddictiveTips, 2019), (YouTube, 2019), (Qiita, 2019), (Homes.chass.utoronto.ca, 2019).



**Screenshot Image from the 2nd Search for my IP Address using a different Locator Website:**



## Q2.4:

This is a research project. Use Google to help you identify a solution.

The objective of the task is to (using the Vim text editor) write a shell script program that behaves like an Irish person offering a cup of tea.

If the user types ‘y’ to the offer, the program displays “Great, I’ll make tea now” to the console.

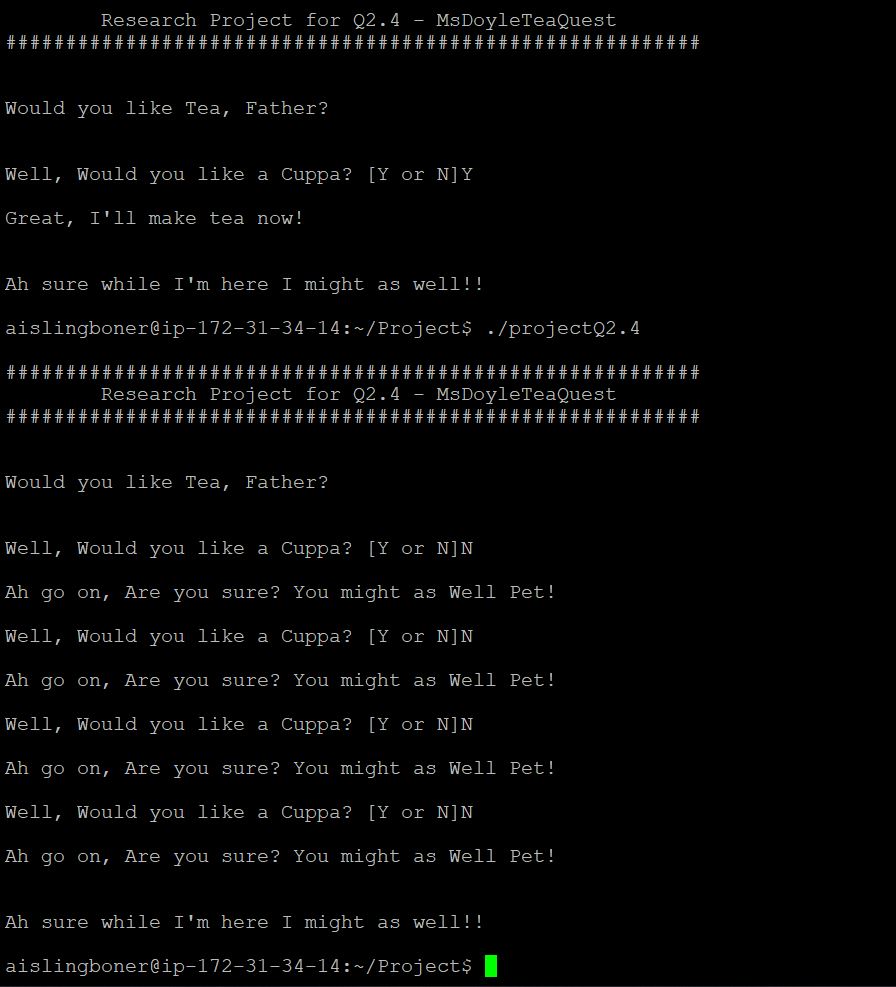
If the user types ‘n’ to the offer, the program asks the user “Are you sure” 4 more times before giving up. If at any point during the 4 follow up offers, the user changes their mind and presses ‘y’, the computer will print out “Great, I’ll make tea now” to the console.

In addition to shell scripting, this assignment examines your ability to use ‘while loops’ and’ if statements’ correctly. It also examines your ability to research and locate the information required online.

For the submission:

Capture a screenshot of the program in operation. Copy and paste the screenshot into your document for submission. Also, copy and paste the shell script code into your Word document for submission.

**Screenshot Image of Tea Programme Runnning:**



**Shell Script for Q2.4:**

References: (B., W. and G, 2019), (YouTube, 2019), (nixCraft, 2019), (H and W., 2019), (GeeksforGeeks, 2019), (Shellscripting Tutorial, YouTube, 2019),

#!/bin/bash

#

#################################################################

#Script Name : Project/MsDoyleTeaQuest

#Description : While Loop for Tea

#Author : Aisling Boner

#Student Number : G00376422

#Email : aislingboner@gmail.com

#################################################################

echo

echo "##########################################################"

echo " Research Project for Q2.4 - MsDoyleTeaQuest "

echo "##########################################################"

echo

echo

echo "Would you like Tea, Father?"

echo

echo

#While loop with Yes(Y) or No(N) response for User Executing.

count=0

while [ $count -le 3 ]

do

read -p "Well, Would you like a Cuppa? [Y or N]" RESP

if [ "$RESP" == "Y" ]; then

echo

echo "Great, I'll make tea now!"

echo

break

else

echo

echo "Ah go on, Are you sure? You might as Well Pet!"

echo

count=$((count+1))

fi

done

echo

echo "Ah sure while I'm here I might as well!!"

echo

## References

1. Academo.org. (2019). *Logic Gate Simulator | Academo.org - Free, interactive, education.*. [online] Available at: https://academo.org/demos/logic-gate-simulator/ [Accessed 1 May 2019].
2. Access.redhat.com. (2019). *Ownership and Permissions*. [online] Available at: https://access.redhat.com/documentation/en-US/Red\_Hat\_Enterprise\_Linux/4/html/Step\_by\_Step\_Guide/s1-navigating-ownership.html [Accessed 1 May 2019].
3. AddictiveTips. (2019). *How to use Lynx to browse the web from the Linux terminal*. [online] Available at: https://www.addictivetips.com/ubuntu-linux-tips/browse-the-web-from-the-linux-terminal/ [Accessed 1 May 2019].
4. B., K., W., D. and G, M. (2019). *BASH Syntax error near unexpected token 'done'*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/18367708/bash-syntax-error-near-unexpected-token-done [Accessed 1 May 2019].
5. Das, D. (2019). *Half Subtractor and Full Subtractor Theory with Diagram and Truth Table*. [online] CSETutor. Available at: https://www.csetutor.com/half-subtractor-and-full-subtractor-theory-diagram-truth-table/ [Accessed 1 May 2019].
6. ElProCus - Electronic Projects for Engineering Students. (2019). *Basic Logic Gates with Truth Tables - Digital Circuits*. [online] Available at: https://www.elprocus.com/basic-logic-gates-with-truth-tables/ [Accessed 1 May 2019].
7. Files.fosswire.com. (2019). [online] Available at: https://files.fosswire.com/2007/08/fwunixref.pdf [Accessed 1 May 2019].
8. GeeksforGeeks. (2019). *Conditional Statements | Shell Script - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/conditional-statements-shell-script/ [Accessed 1 May 2019].
9. GeeksforGeeks. (2019). *Linux Commands - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/linux-commands/ [Accessed 1 May 2019].
10. H, M. and W., D. (2019). *Looping a Bash Shell Script*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/5641309/looping-a-bash-shell-script [Accessed 1 May 2019].
11. Homes.chass.utoronto.ca. (2019). *Lynx Help for Beginners*. [online] Available at: http://homes.chass.utoronto.ca/~purslow/lhfb.html [Accessed 1 May 2019].
12. How-To Geek. (2019). *The Beginnerâ€™s Guide to Shell Scripting: The Basics*. [online] Available at: https://www.howtogeek.com/67469/the-beginners-guide-to-shell-scripting-the-basics/ [Accessed 1 May 2019].
13. Kau.edu.sa. (2019). [online] Available at: https://www.kau.edu.sa/files/830/files/60761\_linux.pdf [Accessed 1 May 2019].
14. Logic.ly. (2019). *Logicly Online Demo*. [online] Available at: https://logic.ly/demo [Accessed 1 May 2019].
15. Maker Pro. (2019). *Basic Linux Commands for Beginners | Linux*. [online] Available at: https://maker.pro/linux/tutorial/basic-linux-commands-for-beginners [Accessed 1 May 2019].
16. Maketecheasier.com. (2019). [online] Available at: https://www.maketecheasier.com/file-permissions-what-does-chmod-777-means/ [Accessed 1 May 2019].
17. McGinley, B. (2019). *Computer Architecture and Technology Convergence\_ Lecture 10*. Galway: GMIT, p.Video 1.
18. McGinley, B. (2019). *Computer Architecture and Technology Convergence \_ Lecture 4*. Galway: GMIT, p.Number Conversion, Number Systems & Binary numbers.
19. McGinley, B. (2019). *Computer Architecture and Technology Convergence\_ Lecture 5*. Galway: GMIT, p.Binary Arithmetic, Representing Real Numbers.
20. McGinley, B. (2019). *Computer Architecture and Technology Convergence*. Galway: GMIT, p.Logic Circuits, Logic Gates.
21. Medium. (2019). *Writing Shell Scripts â€” The Beginnerâ€™s Guide*. [online] Available at: https://medium.com/tech-tajawal/writing-shell-scripts-the-beginners-guide-4778e2c4f609 [Accessed 1 May 2019].
22. nixCraft. (2019). *Bash While Loop Examples - nixCraft*. [online] Available at: https://www.cyberciti.biz/faq/bash-while-loop/ [Accessed 1 May 2019].
23. Pluralsight.com. (2019). *Linux File Permissions*. [online] Available at: https://www.pluralsight.com/blog/it-ops/linux-file-permissions [Accessed 1 May 2019].
24. Qiita. (2019). *lynx cheatsheet - Qiita*. [online] Available at: https://qiita.com/legokichi/items/eabd0e5557d9f6729363 [Accessed 1 May 2019].
25. Radford.edu. (2019). *VIM Editor Commands*. [online] Available at: https://www.radford.edu/~mhtay/CPSC120/VIM\_Editor\_Commands.htm [Accessed 1 May 2019].
26. UNIX tutorials, VC++ tutorials, VM ware tutorials. (2019). *Bash Script Example*. [online] Available at: https://www.tutorialkart.com/bash-shell-scripting/bash-script-example/ [Accessed 1 May 2019].
27. Vim.fandom.com. (2019). *Vim Tips Wiki*. [online] Available at: https://vim.fandom.com/wiki/Vim\_Tips\_Wiki [Accessed 1 May 2019].
28. Vim.org. (2019). *download : vim online*. [online] Available at: https://www.vim.org/download.php [Accessed 1 May 2019].
29. YouTube. (2019). *BASH Scripting Lesson 5 using WHILE loops*. [online] Available at: https://www.youtube.com/watch?v=C2LPwIvIEjo [Accessed 1 May 2019].
30. YouTube. (2019). *Lynx - Text-based Web Browser - Surf the Web From a Terminal*. [online] Available at: https://www.youtube.com/watch?v=qtkqXCq4I9o [Accessed 1 May 2019].
31. YouTube. (2019). *Shell Scripting Tutorial for Beginners 15 - WHILE Loops*. [online] Available at: https://www.youtube.com/watch?v=iwEteYwKDfE [Accessed 1 May 2019].
32. YouTube. (2019). *Vim Basics in 8 Minutes*. [online] Available at: https://www.youtube.com/watch?v=ggSyF1SVFr4 [Accessed 1 May 2019].