Assignment 1

Nicholas Alexeev Edward Hazelton

July 2020

1 Linux ssh login

Nothing to do here.

2 Getting to know the bash shell

2.1 Using ssh, log into the remote system titanic at quirin.is-a-geek.org

I logged into the remote system.

```
Linux titanic 4.19.118-v7l+ #1311 SMP Mon Apr 27 14:26:42 BST 2020
    armv7l

The programs included with the Debian GNU/Linux system are free
    software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jun 29 15:57:20 2020 from 206.174.94.250
naalexeev@titanic:~ $
```

2.2 On that system find out

2.2.1 how many users are logged in concurrently with you

I ran users.

```
naalexeev@titanic:~ $ users
amwilliams24 enfasoformoso enfasoformoso jfrippe mddimatulac
    naalexeev
```

2.2.2 how long the system is already running since last rebooted

I used uptime.

```
naalexeev@titanic:~ $ uptime
 11:18:10 up 5 days, 22:46,  6 users,  load average: 0.08, 0.06,
    0.01
```

2.2.3 what processors it has

I read the file /proc/cpuinfo. The server has a ARMv7 processor

```
naalexeev@titanic:~ $ cat /proc/cpuinfo
processor : 0
model name  : ARMv7 Processor rev 3 (v7l)
BogoMIPS  : 108.00
Features  : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva
     idivt vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part  : 0xd08
CPU revision  : 3

processor : 1
model name  : ARMv7 Processor rev 3 (v7l)
BogoMIPS  : 108.00
Features  : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva
     idivt vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part  : 0xd08
CPU revision  : 3

processor : 2
model name  : ARMv7 Processor rev 3 (v7l)
BogoMIPS  : 108.00
Features  : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva
     idivt vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part  : 0xd08
CPU revision  : 3

processor : 3
model name  : ARMv7 Processor rev 3 (v7l)
BogoMIPS  : 108.00
Features  : half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva
     idivt vfpd32 lpae evtstrm crc32
CPU implementer : 0x41
CPU architecture: 7
CPU variant : 0x0
CPU part  : 0xd08
CPU revision  : 3

Hardware  : BCM2835
Revision  : c03111
Serial    : 10000000900d629c
Model   : Raspberry Pi 4 Model B Rev 1.1
```

2.2.4 how much RAM is available

I ran the program free. The program has 2.5 Gibibytes free.

```
naalexeev@titanic:~ $ free -h
               total        used        free      shared  buff/cache
       available
Mem:           3.8Gi       299Mi       2.5Gi       128Mi       1.0Gi
           3.3Gi
Swap:           99Mi          0B        99Mi
```

2.3 Copy the file nanpa, found in the home directory of the clauteruser, to your account or use scp to copythe file back home onto your private computer.

I used cp to copy the file.

```
naalexeev@titanic:~ $ cp /home/clauter/nanpa ~/napa
```

2.4 On titanic or a private Linux installation, open the file nanpa using less. The file contains quite a comprehensive list of North American phone number prefixes (first 6 digits, excluding+1), followed by the location this phone number prefix is attached to. For example, for 907519, the location Anchorage AK is listed. Still inside less, find the entries for 907519,503526 and a couple of other phone numbers you know in the country, as such your home phone, your parents' phone, the phone of a loved one etc.

I used / to forward search for 907519, then ? to backward search for 503526, then searched for a couple other phone number prefixes using / and ? as needed.

```
907519Anchorage AK
503526Beaverton OR
907891Anchorage AK
508829Holden MA
```

2.5 Find out how many lines connecting prefixes to locations are contained in the filenanpa. Which Linux command line tool do you use to count lines?

I used wc -l nanpa to get the number of lines in nanpa

```
eshazelton@titanic:~ $ wc -l nanpa
166482 nanpa
```

2.6 List the first 17 lines of the nanpa file on command line. Also list the last 42 lines of the file. You can use the Linux tools head and tail for this task.

I used head nanpa -n 17 to get the first 17 lines in the file.

```
eshazelton@titanic:~ $ head nanpa -n 17
201200Jersey City NJ
201202Hackensack NJ
201203Hackensack NJ
201204Jersey City NJ
201205Jersey City NJ
201206Hackensack NJ
201207Newark NJ
201208Jersey City NJ
201209Jersey City NJ
201210Union City NJ
201212Hackensack NJ
201213Morristown NJ
201214Hackensack NJ
201215Bayonne NJ
201216Jersey City NJ
201217Jersey City NJ
201218Hackensack NJ
```

I used ‘tail nanpa -n 42‘ to get the last 42 lines in the file.

```
eshazelton@titanic:~ $ tail nanpa -n 42
989921Saginaw MI
989922Bay City MI
989923Midland MI
989924Alma MI
989925Manistee Ri MI
989926Midland MI
989928Saginaw MI
989929Bay City MI
989932Durand MI
989934Rose City MI
989935Clare MI
989936Owosso MI
989938Grace Harbo MI
989939Chester MI
989941Midland MI
989942West Branch MI
989943Middleton MI
989944Mount Pleas MI
989945Ovid MI
989946Standish MI
989948Midland MI
989949McBrides MI
989953Mount Pleas MI
989954Mount Pleas MI
989956Mount Pleas MI
989962Minden City MI
989963Elkton MI
989964Saginaw MI
989965Gladwin MI
989966Vanderbilt MI
```

```
989967Remus MI
989968Alma MI
989971Saginaw MI
989975Bad Axe MI
989977Sebewaing MI
989979St Johns MI
989980Saginaw MI
989981Hubbardston MI
989983Vanderbilt MI
989984East Tawas MI
989992Saginaw MI
989996Saginaw MI
```

2.7 Write a short bash script findlocation that takes a 6-digit prefix in argument and displays the corresponding location. If the script receives no argument or the argument is not a6-digit prefix made only out of the digits 0 thru 9, the script must return an exit condition code signaling failure (e.g. by executing exit 1). If the script receives a correctly formatted argument but the prefix is not found in the nanpa file, the script must return an exit condition code signaling failure. Otherwise, the script must display the appropriate location (on stdout).The location must not be prefixed by the prefix nor followed by superfluous spaces. This means you have to format the line found the nanpa file before displaying it. You may use grep and sed for this script.

See findlocation.sh

3 Rewrite head and tail

I rewrote head and tail using only direct syscalls. I first wrote a simple standard library and then used the standard library as a foundation for head.c and tail.c. A major pain point was manual memory management. The memory management was error prone and it required significant debugging. See head.c, tail.c and my_stdlib.h for the source code.

4 File Searching

I rewrote findlocation in c. The main difficulty was string manipulation. While using another programming language is not in the scope of the course if I were to write the program for a real world deployment I would use rust to write the

program because it would allow me to mix high level abstractions and low level concepts such as pointers. An improvement that could be incorporated easily is an argument parser that provides helpful error messages.