Lab *05 – Scripting*

Name: Christian Fernandez
Course/Section: IS-1003-ON3
Date: 10/28/2025

INTRODUCTION

In this lab we will create a script that automates a set of commands and a main driver so that we are able to execute the commands again in the future without having to type everything out again.

BREAKPOINT 1

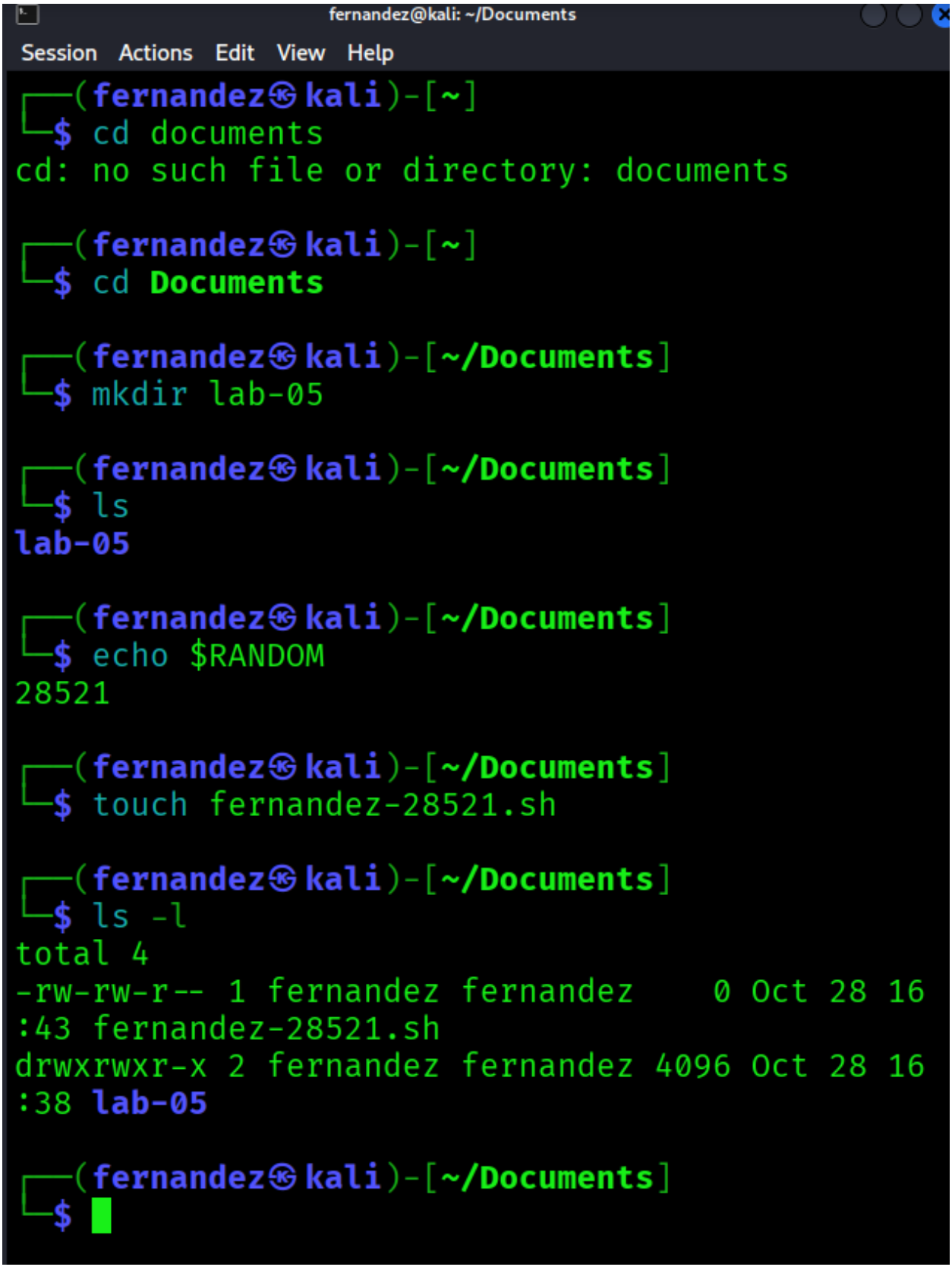

Figure 1: Dir and random number file created 16:50 10/28/2025

For this part of the Lab I created a directory call “lab-05” and a file using the command *echo $RANDOM* to create a random number and then used the number to create a file inside the new directory.

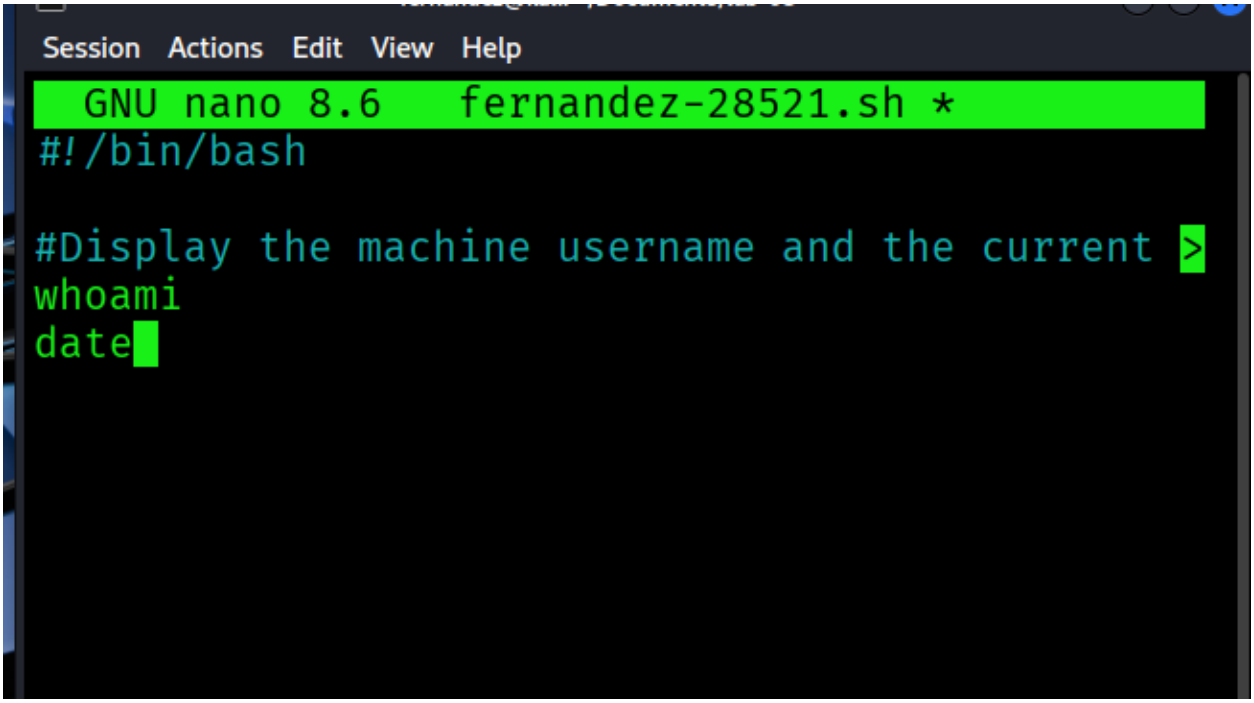

Figure 2: nano file creation 17:11 10/28/2025

In the above screen shot I opened my file I created with the nano command and added the text above.

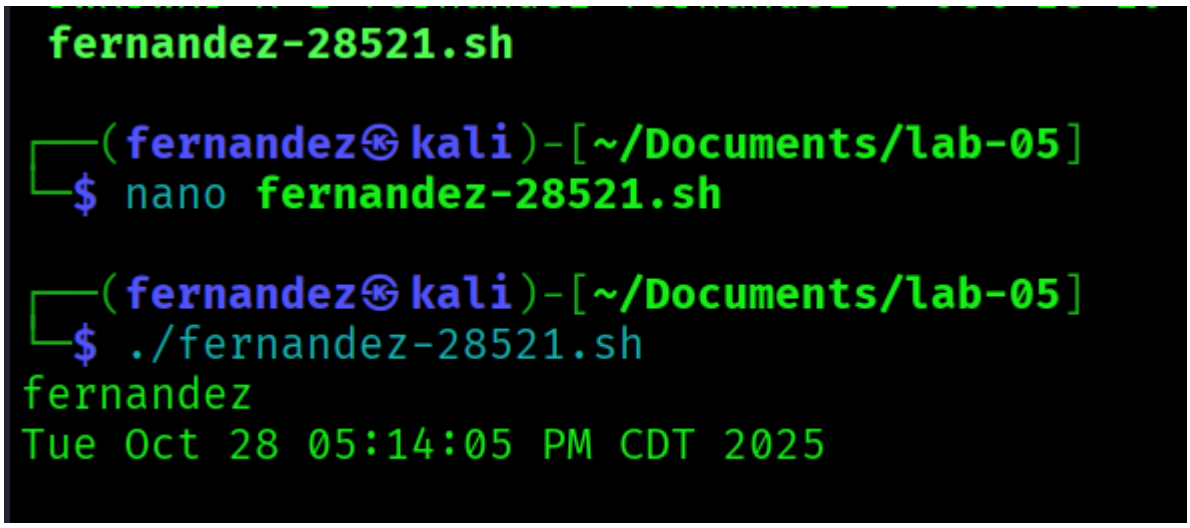

Figure 3:output 17:19 10/28/2025

From the screenshot above we can see that the program we wrote with nano provides the correct output so no need for debugging.

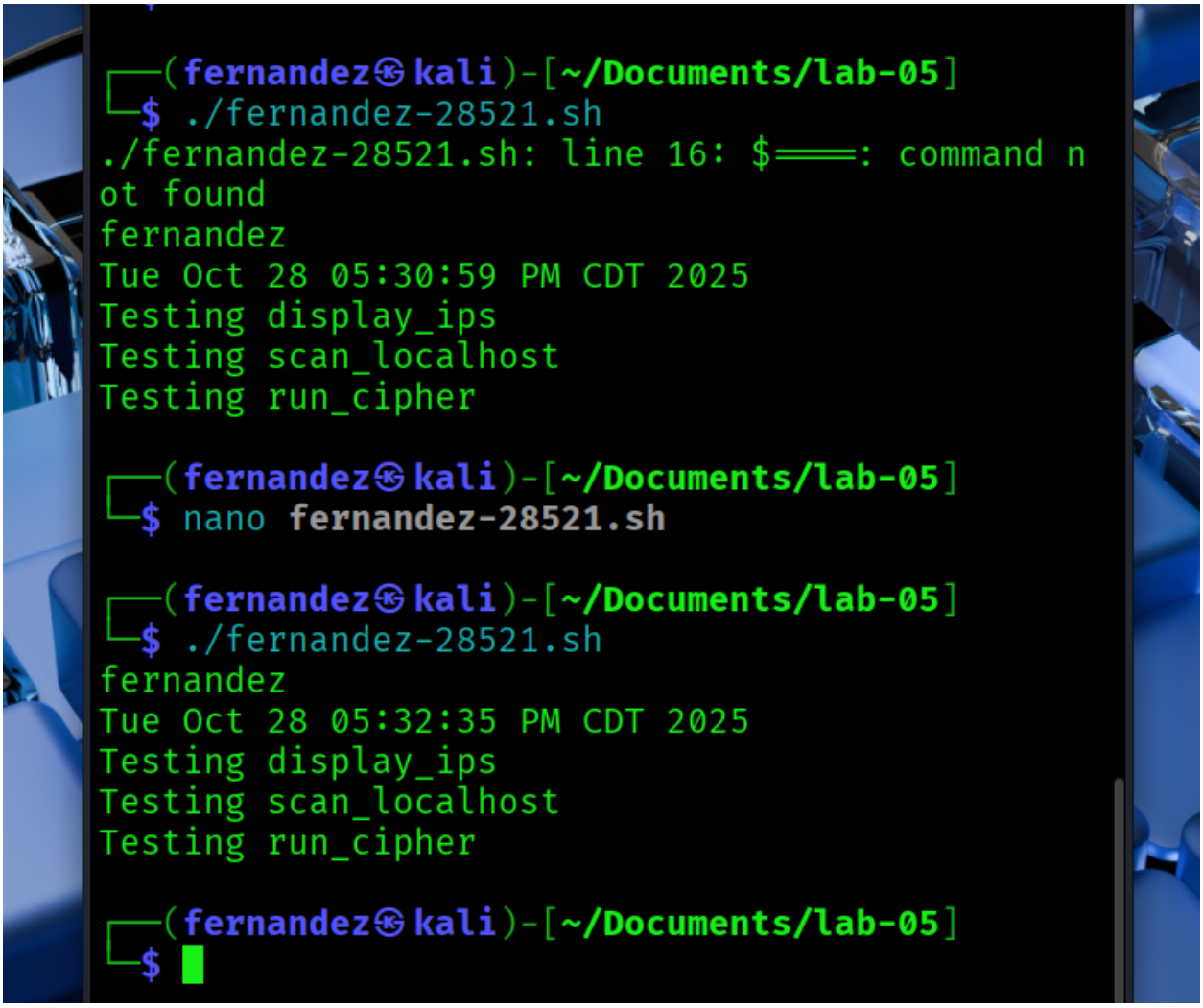

Figure 4program output 17:34 10/28/2025

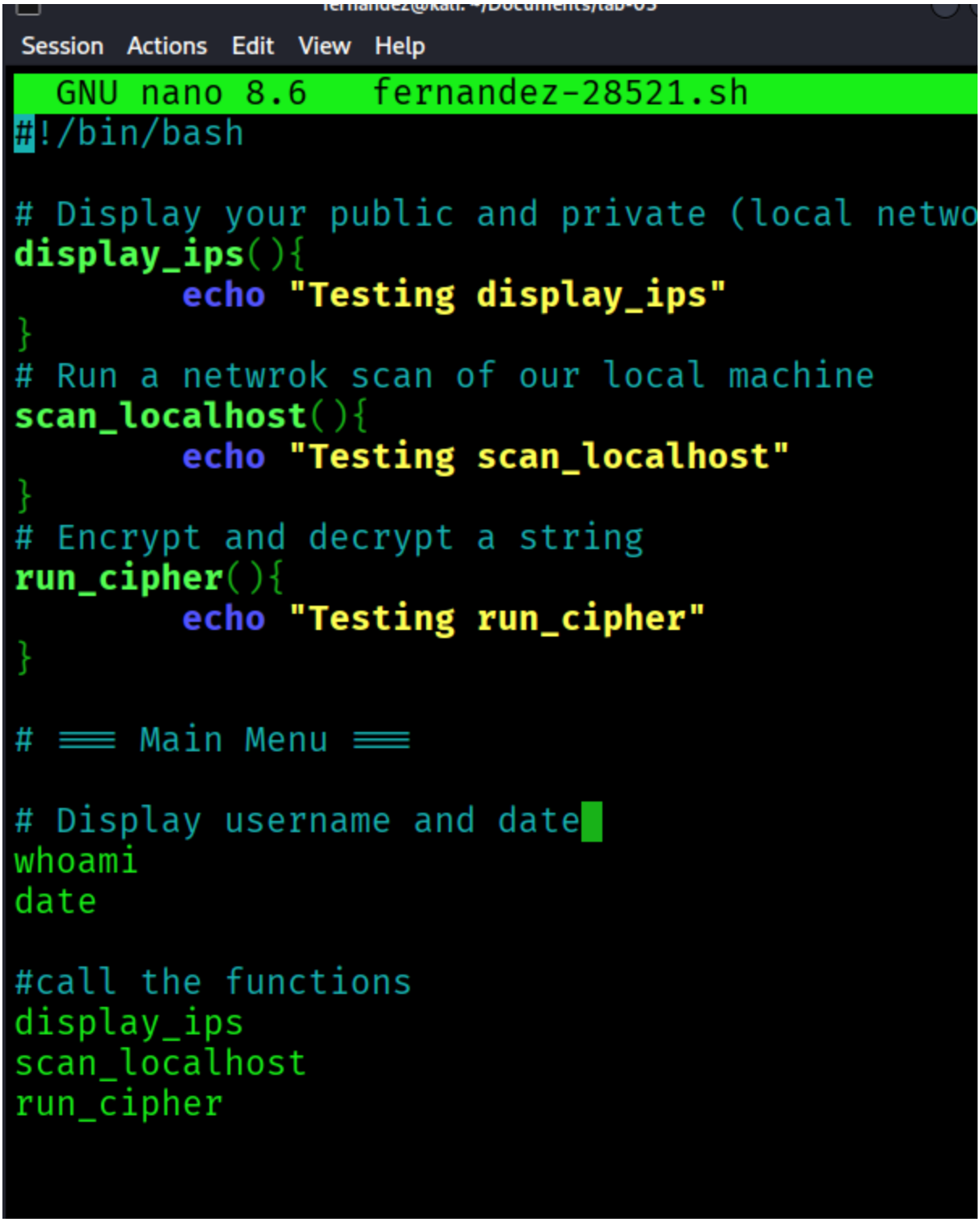

Figure 5:program input 17:35 10/28/2025

The above step shows the program and the output into the terminal. This part did require debugging because at line 16 I first put a $ instead of a #. The design of the program was key to keeping the script organized and the comments helped show what each function was used for.

BREAKPOINT 2

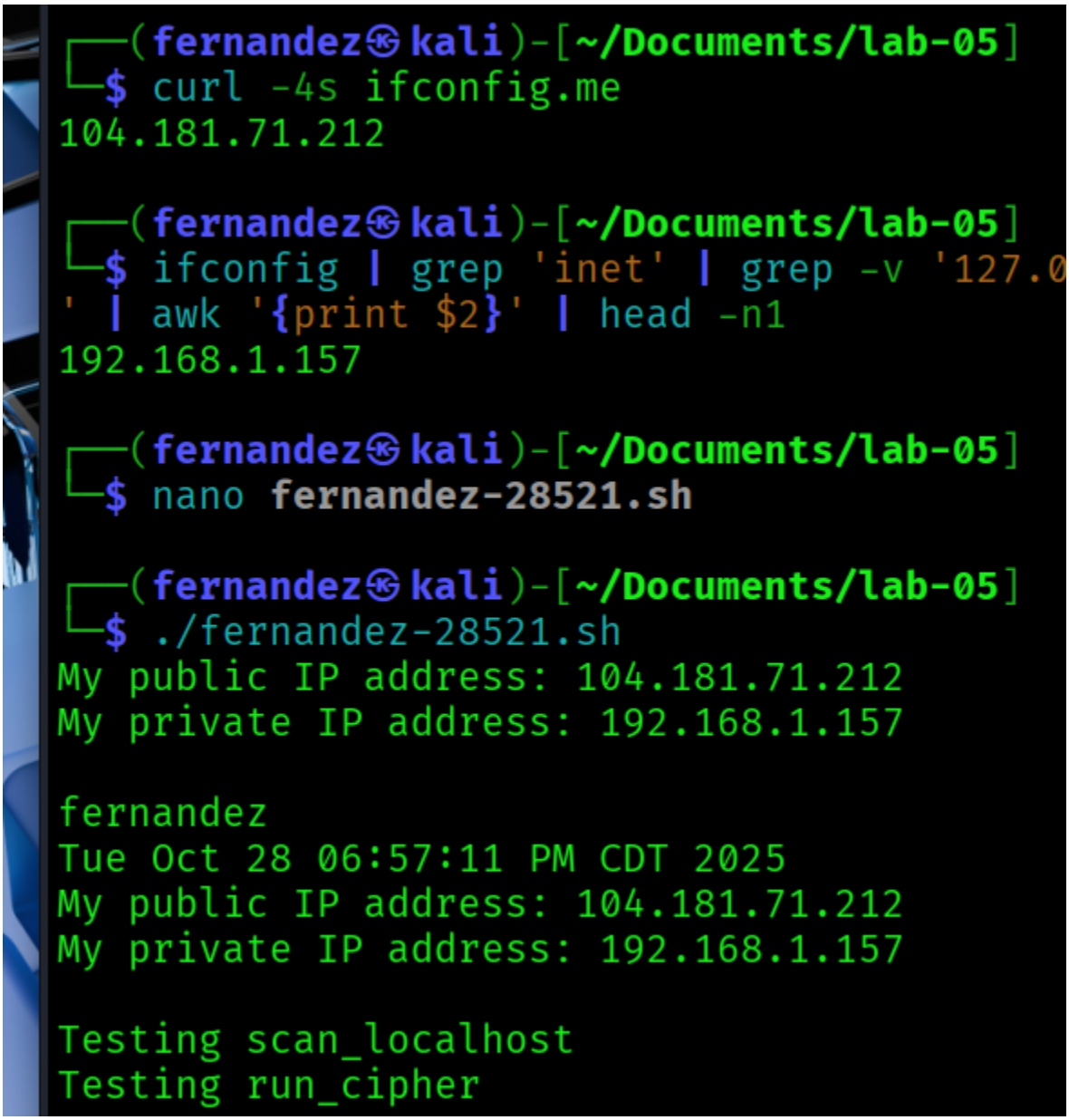

Figure 6:display_ips 18:57 10/28/2025

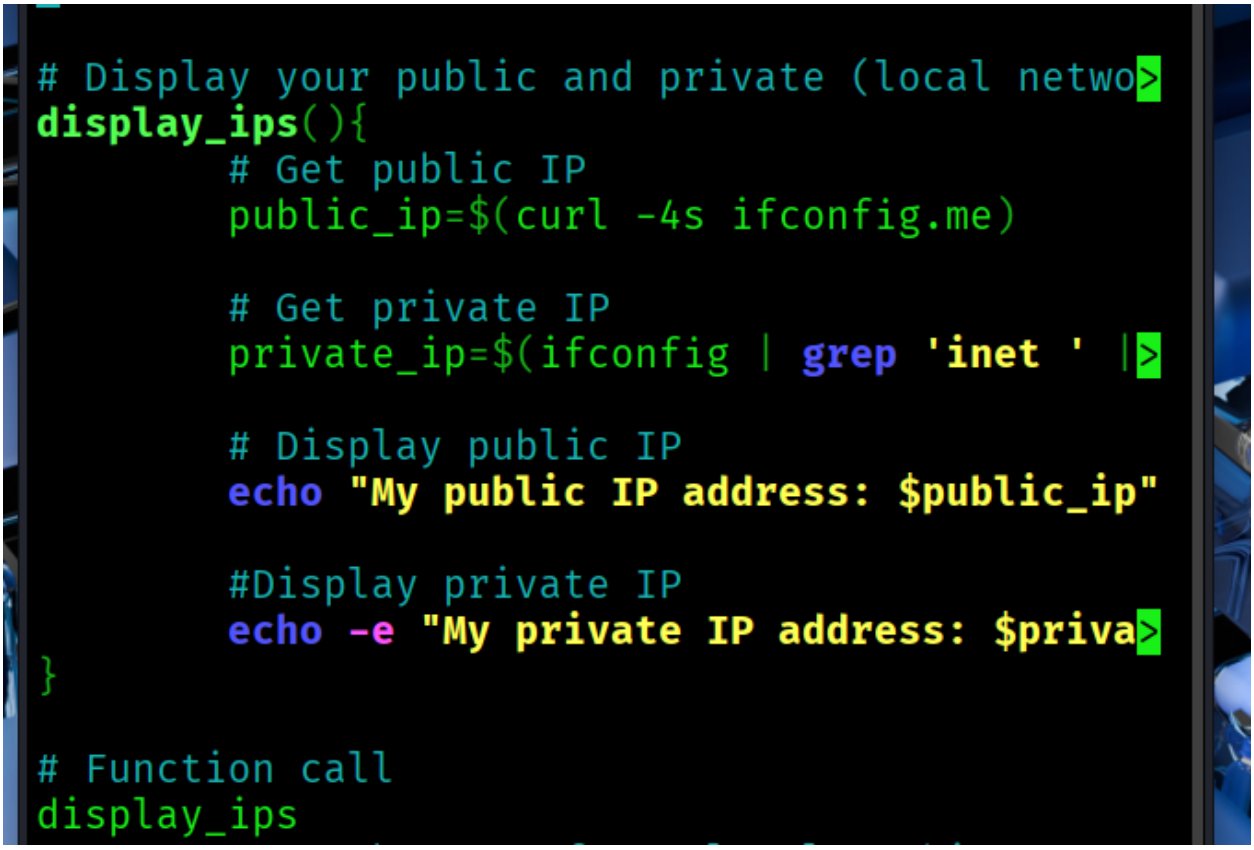

Figure 7:Display_ips pt 2 19:00 10/28/2025

In the above section I finished the display_ips function. The output is also shown above and did not require any debugging. In the example you used a file header to show what the content of the script are going to be. Function comments are also used to keep the script organized and helps the reader understand what each part of the script does.

BREAKPOINT 3

```
scan_localhost(){
        target= "127.0.0.1" # Scan localhost o>

        # present safe scanning options
        echo "=== Nmap scan options for localh>
        echo "[1] Basic Ping scan (Host Discov>
        echo "[2] TCP Connect Scan  (Top 10 Po>
        echo "[3] Service Version Detection (R>
        echo "[4] OS detection [requires sudo]>
        echo "[5] Full port scan (all ports)"
        echo "[6] return to main menu"
        echo -n "please choose an option:  "
        read choice

        #perform the type of scan based on opt>
        case $choice in
        1) nmap -sn "$target"
        2) nmap --top-ports 10 -sT "$target" ;;
        3) nmap -sV "$target" ;;
        4) sudo nmap -O "$target" ;;
        5) nmap -p- "$target" ;;
        6) echo -e "Returning to main menu ... >
        esac
}
#function call
scan_localhost
```

Figure 8: nmap script 19:32 10/28/2025

The above script did require some debugging as I forgot to add the " ;;" in line 37 which was stopping the script from running correctly. The program allows the user to automate nmap commands by making a selection from the list and running the nmap script. The curl command is used to transfer data and the nmap command is used when using the nmap took which is a pentesting tool that can be used for recon such as port scans.

BREAKPOINT 4

```
ease choose an option:  6
turning to main menu ...

rnandez
e Oct 28 07:42:42 PM CDT 2025
 public IP address: 104.181.71.212
 private IP address: 192.168.1.157

fernandez-28521.sh: line 22: 127.0.0.1: command not found
≡ Nmap scan options for localhost () ≡≡
] Basic Ping scan (Host Discovery)
] TCP Connect Scan  (Top 10 Ports)
] Service Version Detection (Running Software)
] OS detection [requires sudo] (operating system)
] Full port scan (all ports)
] return to main menu
ease choose an option:  6
turning to main menu ...

ts encrypt a string! Enter a string of your choice (or '0' to quit):
 Hello
crypted: Khoor
crypted: Hello
ts encrypt a string! Enter a string of your choice (or '0' to quit):
```

Figure 9: encryption script 19:43 10/28/2025

```
Encrypt and decrypt a string
un_cipher(){
This is an unformatted cipher function; please  try to pretty it up!
      run=true

      #This wile loop runs until the user types a 0 to quit
      while $run; do
              read -p "Lets encrypt a string! Enter a string of you>
              if [[ "$input" == "0" ]]; then
                      echo "Than you! Bye!!"
                      run=false
              else
                      encrypted=$(echo "$input" | tr 'a-zA-Z' 'd-za>
                      decrypted=$(echo "$encrypted" | tr 'a-zA-Z' '>
                      echo "encrypted: $encrypted"
                      echo "decrypted: $decrypted"
              fi
      done


=== Main Menu ===
```

Figure 10: encryption script 19:45 10/28/2025

The script above is used to automate the encryption process for a string. It uses the looped Caesar cipher demo which applies a ROT-3 shift to encrypt strings. And it will continue to do so until the use selects the “quit” option.

```
encrypted=$(echo "$input" | tr 'a-z' 'n-za-m')
decrypted=$(echo "$encrypted" | tr 'a-z' 'n-z>
echo "encrypted: $encrypted"
echo "decrypted: $decrypted"
```

Figure 11: ROT 13 19:51 10/28/2025

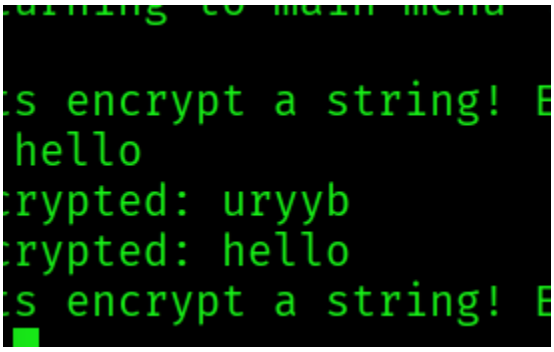

Figure 12:ROT 13 output 19:52 10/28/2025

Above, the change in script from a rot 3 to a rot 5 encryption is shown. For this cipher I changed the arguments for “tr” to ‘a-z’ ‘n-za-m’.

BREAKPOINT 5

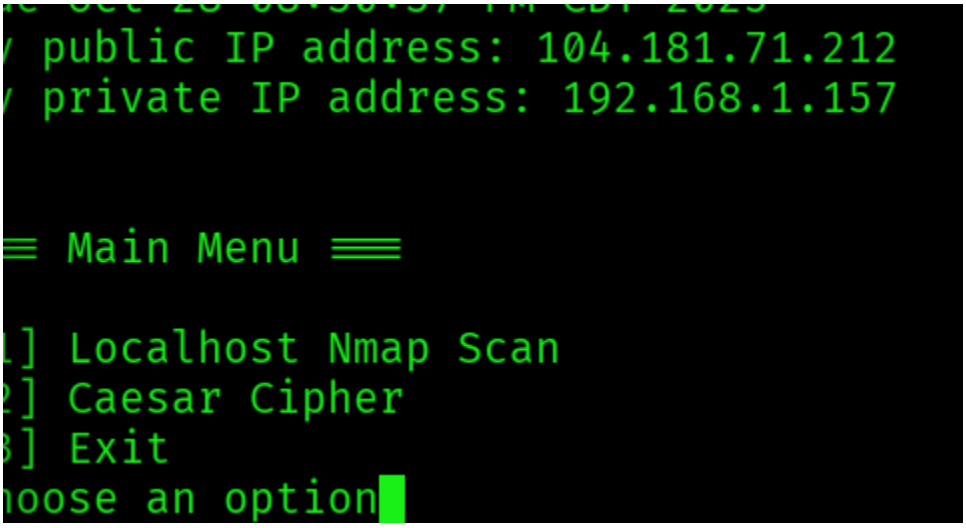

Figure 13: main menu 20:32 10/28/2025

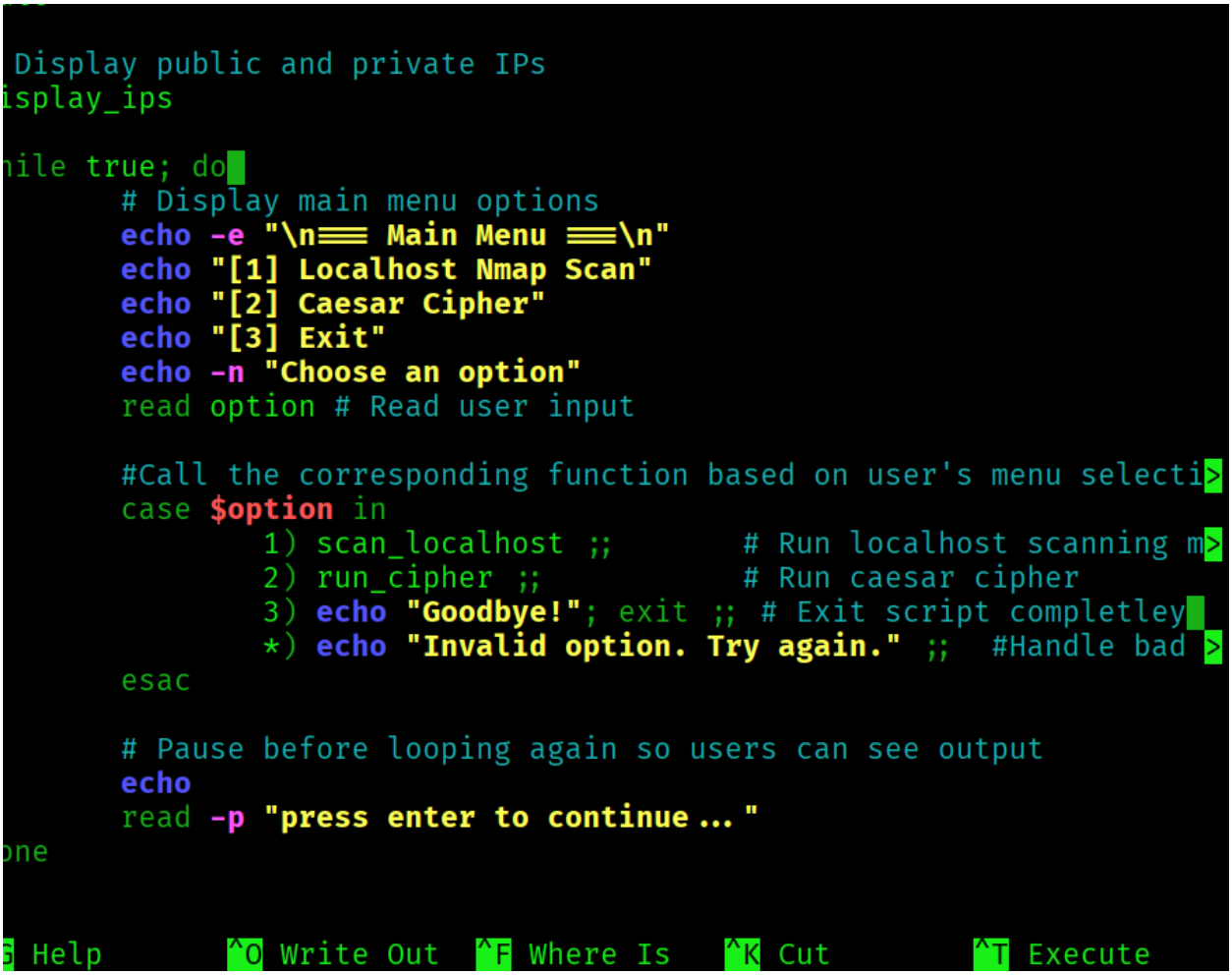

Figure 14: main menu script 20:34 10/28/2025

This part of the program is what directs the rest of the script. From here users can choose what tool they would like to use whether that is Nmap or the cipher. This just being the first version of my script I think for the update I would add some more cosmetics to it. In a way it reminds of the "Lazy script" that it available on github. With a few less tools but the same idea of automating tools.

Conclusion/Limitations

There were a few instances in this lab where debugging was necessary however it was never a big problem. For the most part it was missing characters.

References

R. Mitra, "*Lab-05 scripting*," The University of Texas at San Antonio (2025). Last accessed: *10/28/2025*.

Generative AI Searches

I did not use generative AI to complete this lab report.

Collaboration

I worked independently on this lab.