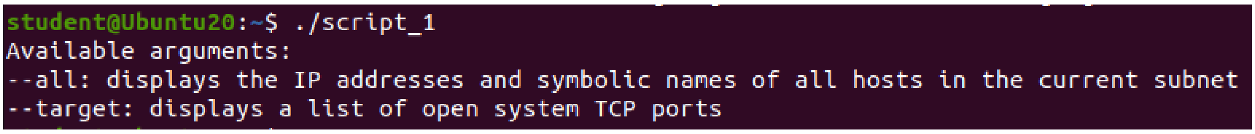
Linux administration with bash. Home task

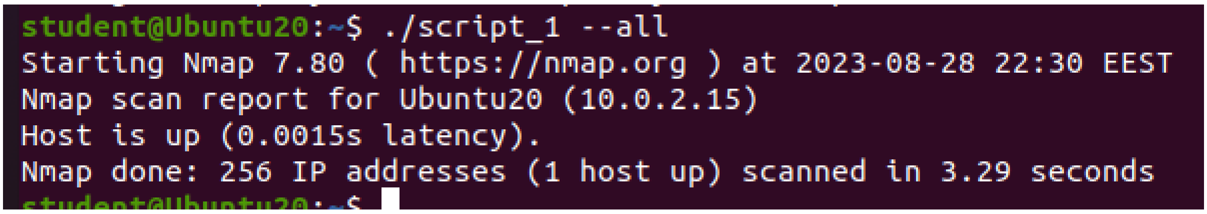
A. Create a script that uses the following keys:

1. When starting without parameters, it will display a list of possible keys and their description.



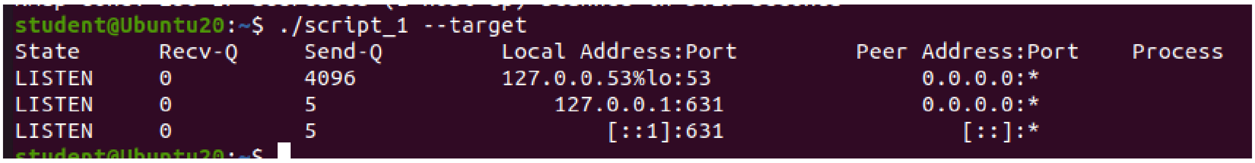
```
student@Ubuntu20:~$ ./script_1
Available arguments:
--all: displays the IP addresses and symbolic names of all hosts in the current subnet
--target: displays a list of open system TCP ports
```

2. The --all key displays the IP addresses and symbolic names of all hosts in the current subnet



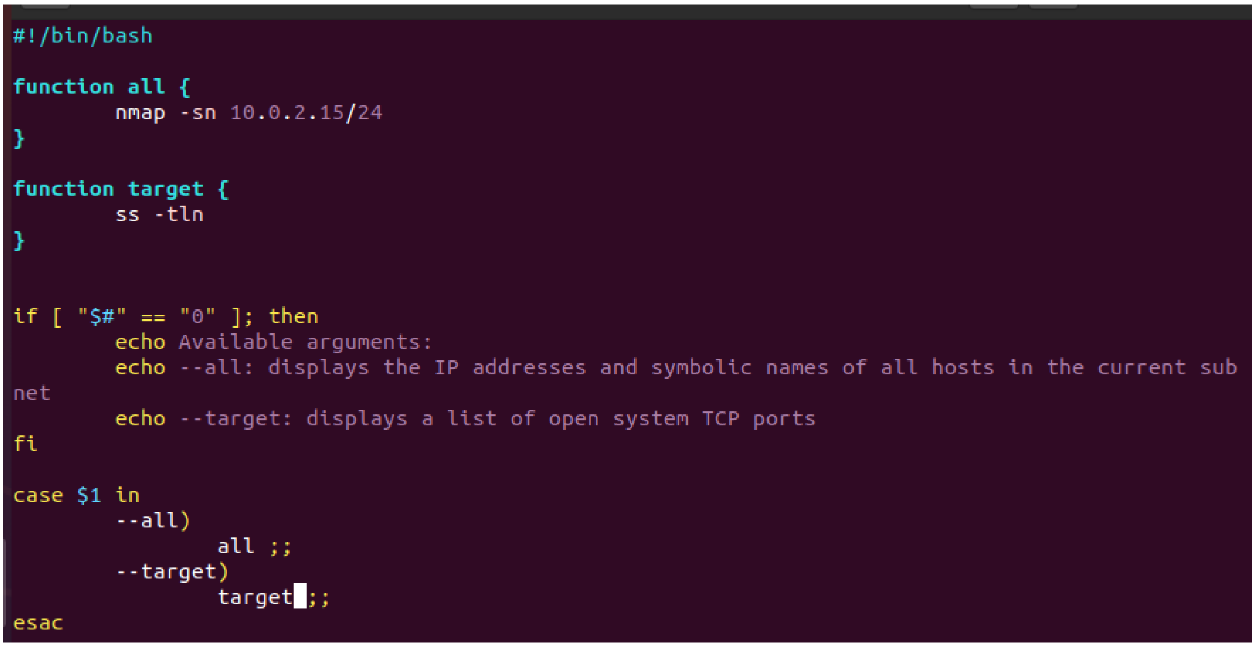
```
student@Ubuntu20:~$ ./script_1 --all
Starting Nmap 7.80 ( https://nmap.org ) at 2023-08-28 22:30 EEST
Nmap scan report for Ubuntu20 (10.0.2.15)
Host is up (0.0015s latency).
Nmap done: 256 IP addresses (1 host up) scanned in 3.29 seconds
student@Ubuntu20:~$
```

3. The --target key displays a list of open system TCP ports.



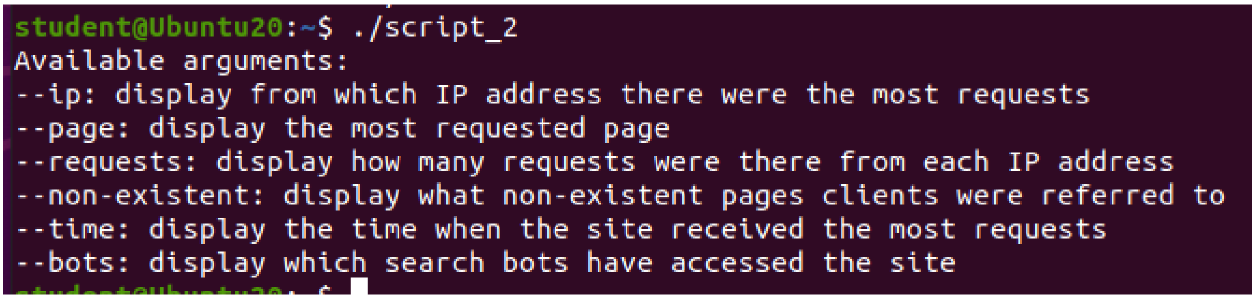
```
student@Ubuntu20:~$ ./script_1 --target
State      Recv-Q     Send-Q         Local Address:Port          Peer Address:Port    Process
LISTEN     0          4096           127.0.0.53%lo:53                 0.0.0.0:*
LISTEN     0          5                  127.0.0.1:631                0.0.0.0:*
LISTEN     0          5                      [::1]:631                   [::]:*
student@Ubuntu20:~$
```

The code that performs the functionality of each of the subtasks must be placed in a separate function

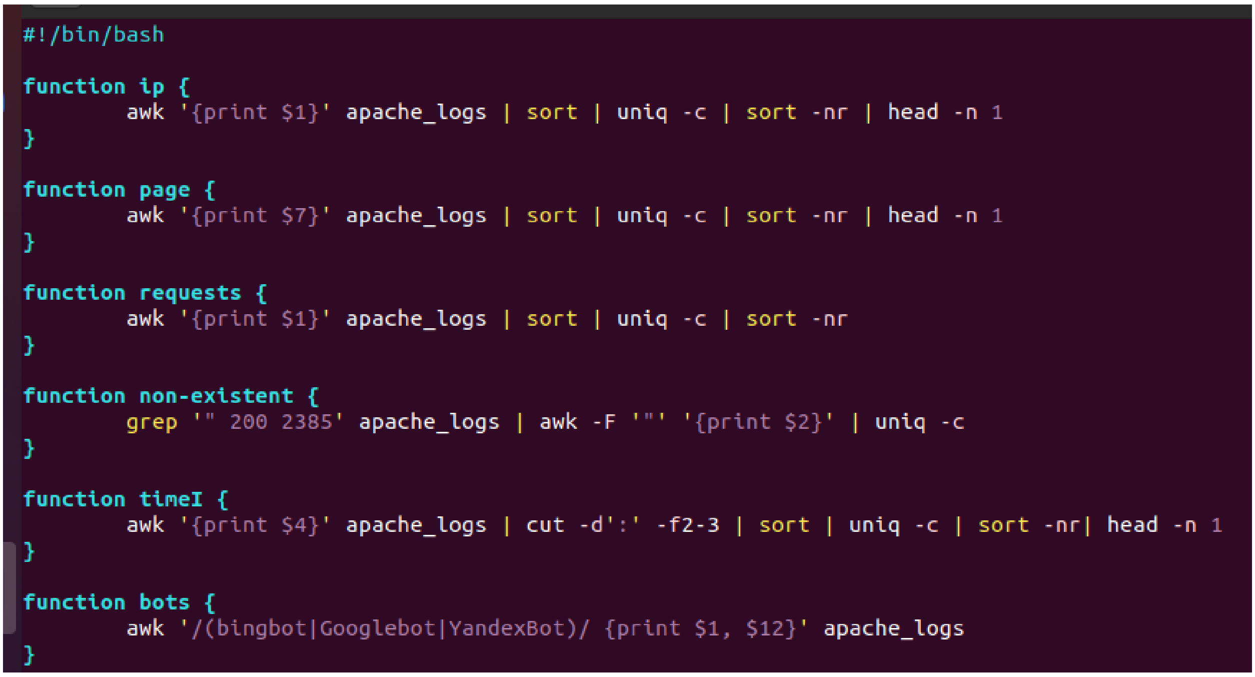


```
#!/bin/bash

function all {
        nmap -sn 10.0.2.15/24
}

function target {
        ss -tln
}


if [ "$#" == "0" ]; then
        echo Available arguments:
        echo --all: displays the IP addresses and symbolic names of all hosts in the current sub
net
        echo --target: displays a list of open system TCP ports
fi

case $1 in
        --all)
                all ;;
        --target)
                target;;
esac
```

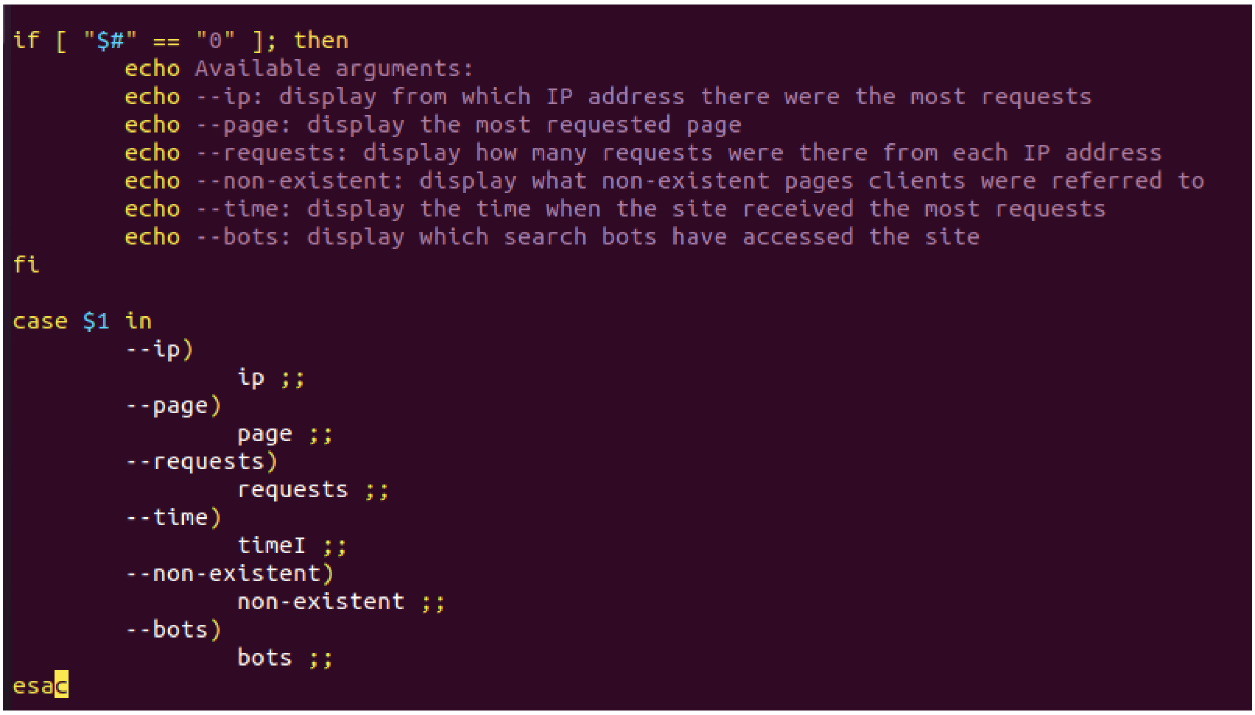
B. Using Apache log example create a script to answer the following questions:



```
student@Ubuntu20:~$ ./script_2
Available arguments:
--ip: display from which IP address there were the most requests
--page: display the most requested page
--requests: display how many requests were there from each IP address
--non-existent: display what non-existent pages clients were referred to
--time: display the time when the site received the most requests
--bots: display which search bots have accessed the site
student@Ubuntu20:~$
```

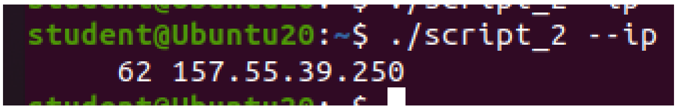


```
#!/bin/bash

function ip {
        awk '{print $1}' apache_logs | sort | uniq -c | sort -nr | head -n 1
}

function page {
        awk '{print $7}' apache_logs | sort | uniq -c | sort -nr | head -n 1
}

function requests {
        awk '{print $1}' apache_logs | sort | uniq -c | sort -nr
}

function non-existent {
        grep '" 200 2385' apache_logs | awk -F '"' '{print $2}' | uniq -c
}

function timeI {
        awk '{print $4}' apache_logs | cut -d':' -f2-3 | sort | uniq -c | sort -nr| head -n 1
}

function bots {
        awk '/(bingbot|Googlebot|YandexBot)/ {print $1, $12}' apache_logs
}
```



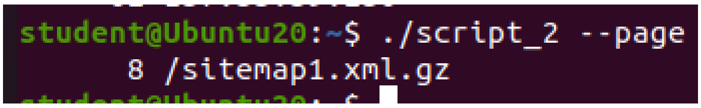
```
if [ "$#" == "0" ]; then
        echo Available arguments:
        echo --ip: display from which IP address there were the most requests
        echo --page: display the most requested page
        echo --requests: display how many requests were there from each IP address
        echo --non-existent: display what non-existent pages clients were referred to
        echo --time: display the time when the site received the most requests
        echo --bots: display which search bots have accessed the site
fi

case $1 in
        --ip)
                ip ;;
        --page)
                page ;;
        --requests)
                requests ;;
        --time)
                timeI ;;
        --non-existent)
                non-existent ;;
        --bots)
                bots ;;
esac
```

1. From which ip were the most requests?



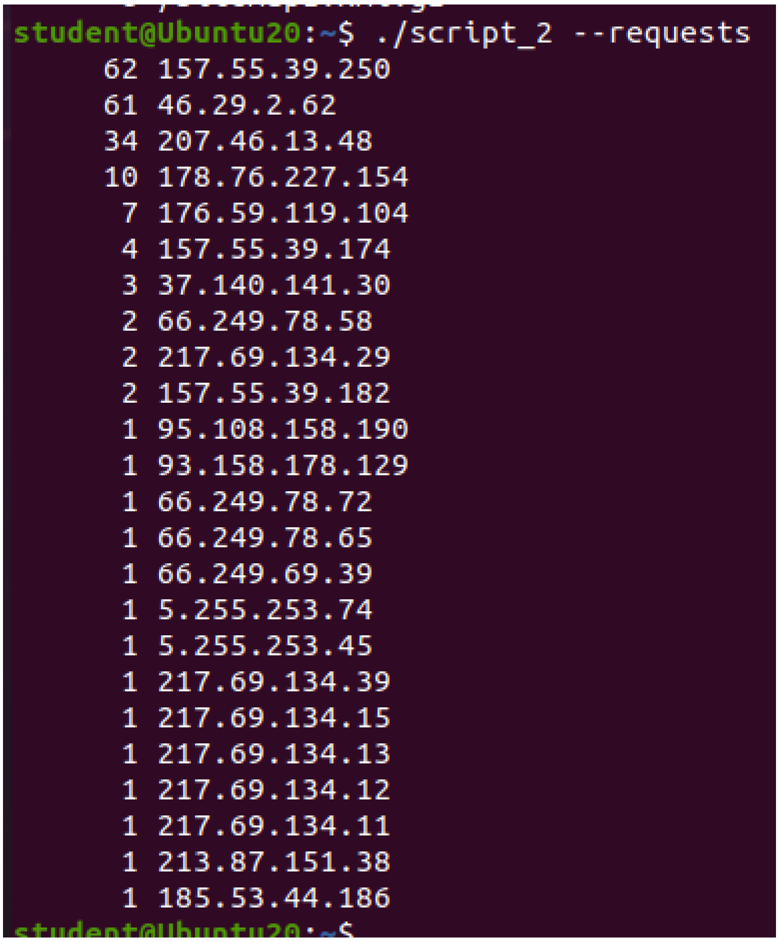
```
student@Ubuntu20:~$ ./script_2 --ip
     62 157.55.39.250
```

2. What is the most requested page?



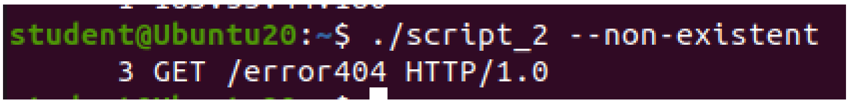
```
student@Ubuntu20:~$ ./script_2 --page
      8 /sitemap1.xml.gz
```

3. How many requests were there from each ip?



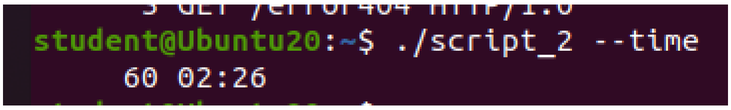
```
student@Ubuntu20:~$ ./script_2 --requests
     62 157.55.39.250
     61 46.29.2.62
     34 207.46.13.48
     10 178.76.227.154
      7 176.59.119.104
      4 157.55.39.174
      3 37.140.141.30
      2 66.249.78.58
      2 217.69.134.29
      2 157.55.39.182
      1 95.108.158.190
      1 93.158.178.129
      1 66.249.78.72
      1 66.249.78.65
      1 66.249.69.39
      1 5.255.253.74
      1 5.255.253.45
      1 217.69.134.39
      1 217.69.134.15
      1 217.69.134.13
      1 217.69.134.12
      1 217.69.134.11
      1 213.87.151.38
      1 185.53.44.186
student@Ubuntu20:~$
```

4. What non-existent pages were clients referred to?



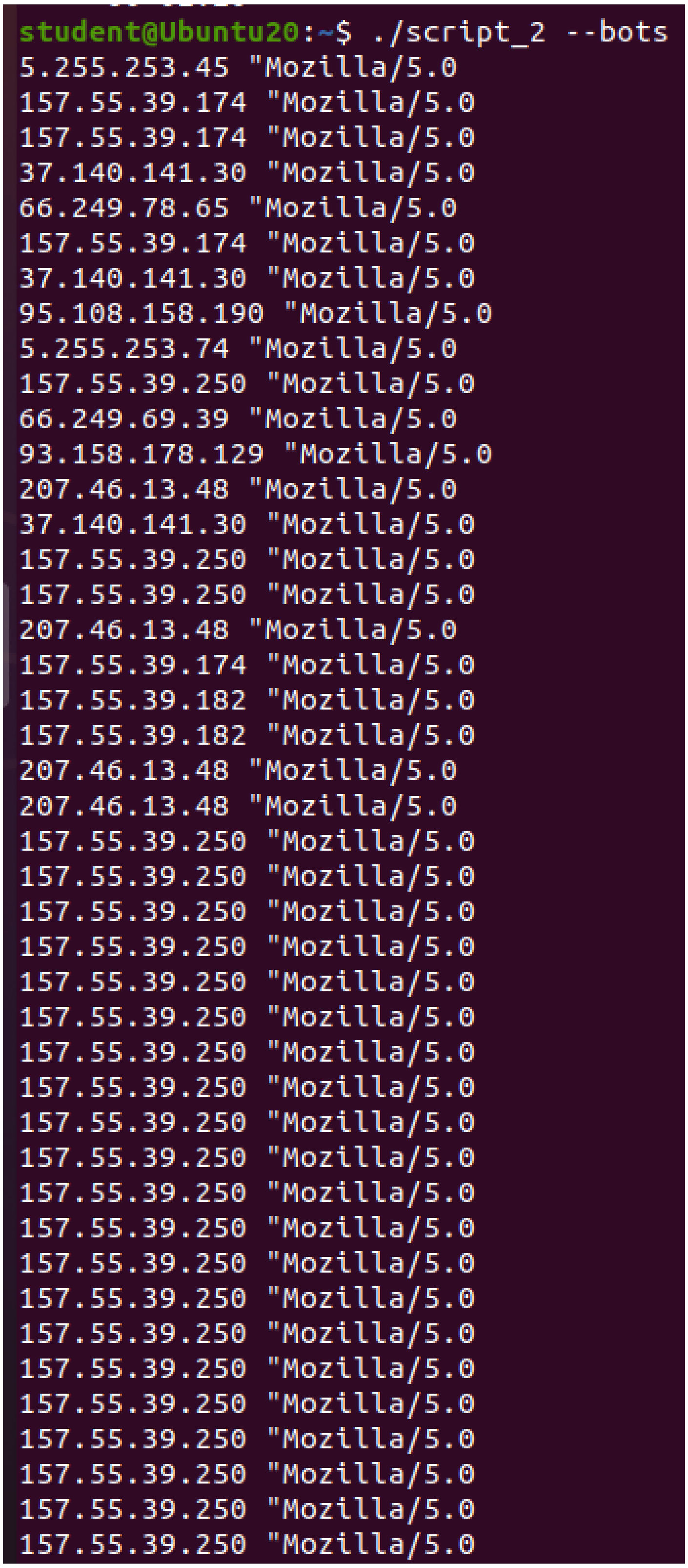
```
student@Ubuntu20:~$ ./script_2 --non-existent
      3 GET /error404 HTTP/1.0
```

5. What time did site get the most requests?



```
student@Ubuntu20:~$ ./script_2 --time
     60 02:26
```

6. What search bots have accessed the site? (UA + IP)



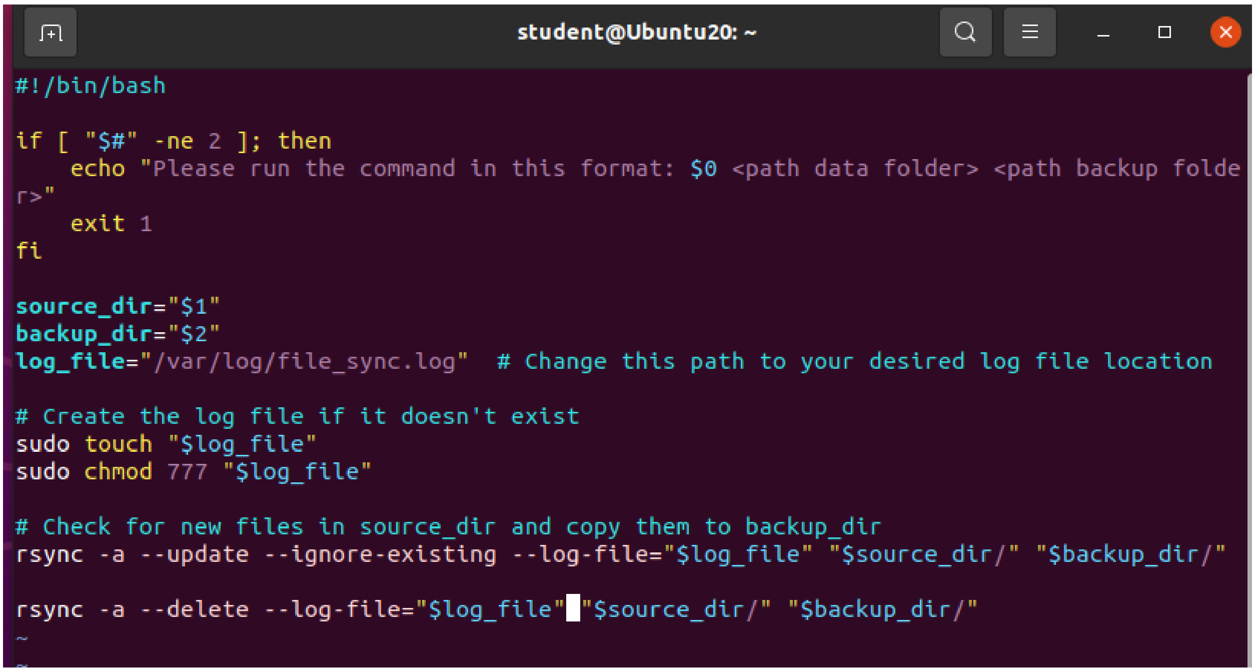
```
student@Ubuntu20:~$ ./script_2 --bots
5.255.253.45 "Mozilla/5.0
157.55.39.174 "Mozilla/5.0
157.55.39.174 "Mozilla/5.0
37.140.141.30 "Mozilla/5.0
66.249.78.65 "Mozilla/5.0
157.55.39.174 "Mozilla/5.0
37.140.141.30 "Mozilla/5.0
95.108.158.190 "Mozilla/5.0
5.255.253.74 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
66.249.69.39 "Mozilla/5.0
93.158.178.129 "Mozilla/5.0
207.46.13.48 "Mozilla/5.0
37.140.141.30 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
207.46.13.48 "Mozilla/5.0
157.55.39.174 "Mozilla/5.0
157.55.39.182 "Mozilla/5.0
157.55.39.182 "Mozilla/5.0
207.46.13.48 "Mozilla/5.0
207.46.13.48 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
157.55.39.250 "Mozilla/5.0
```

C. Create a data backup script that takes the following data as parameters:

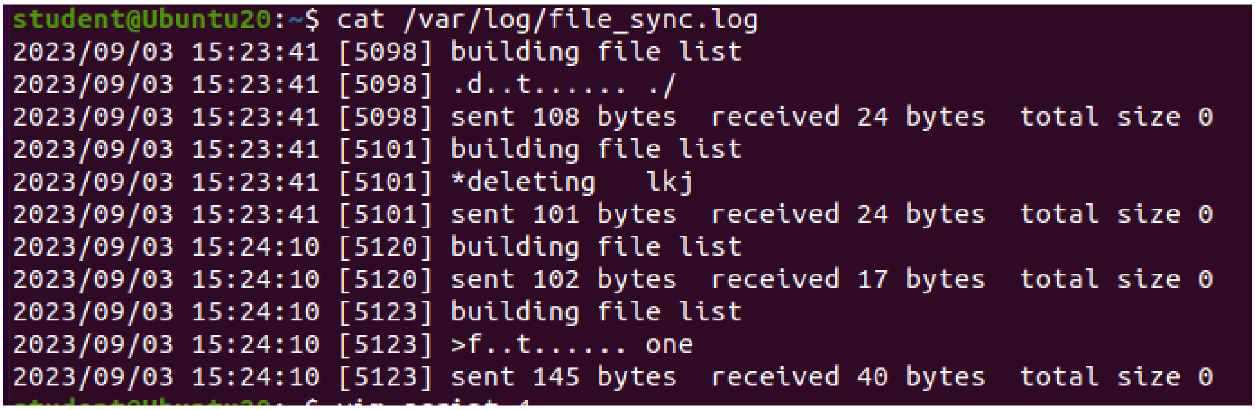
1. Path to the syncing directory.

2. The path to the directory where the copies of the files will be stored.

In case of adding new or deleting old files, the script must add a corresponding entry to the log file indicating the time, type of operation and file name. [The command to run the script must be added to crontab with a run frequency of one minute]



```
#!/bin/bash

if [ "$#" -ne 2 ]; then
    echo "Please run the command in this format: $0 <path data folder> <path backup folde
r>"
    exit 1
fi

source_dir="$1"
backup_dir="$2"
log_file="/var/log/file_sync.log"  # Change this path to your desired log file location

# Create the log file if it doesn't exist
sudo touch "$log_file"
sudo chmod 777 "$log_file"

# Check for new files in source_dir and copy them to backup_dir
rsync -a --update --ignore-existing --log-file="$log_file" "$source_dir/" "$backup_dir/"

rsync -a --delete --log-file="$log_file" "$source_dir/" "$backup_dir/"
~
~
```



```
student@Ubuntu20:~$ cat /var/log/file_sync.log
2023/09/03 15:23:41 [5098] building file list
2023/09/03 15:23:41 [5098] .d..t...... ./
2023/09/03 15:23:41 [5098] sent 108 bytes  received 24 bytes  total size 0
2023/09/03 15:23:41 [5101] building file list
2023/09/03 15:23:41 [5101] *deleting   lkj
2023/09/03 15:23:41 [5101] sent 101 bytes  received 24 bytes  total size 0
2023/09/03 15:24:10 [5120] building file list
2023/09/03 15:24:10 [5120] sent 102 bytes  received 17 bytes  total size 0
2023/09/03 15:24:10 [5123] building file list
2023/09/03 15:24:10 [5123] >f..t...... one
2023/09/03 15:24:10 [5123] sent 145 bytes  received 40 bytes  total size 0
```

::