Linux & Bash Scripting Exercises

1. Create a file named file.txt, and create a user sample user. Change the ownership of the file to sampleuser.

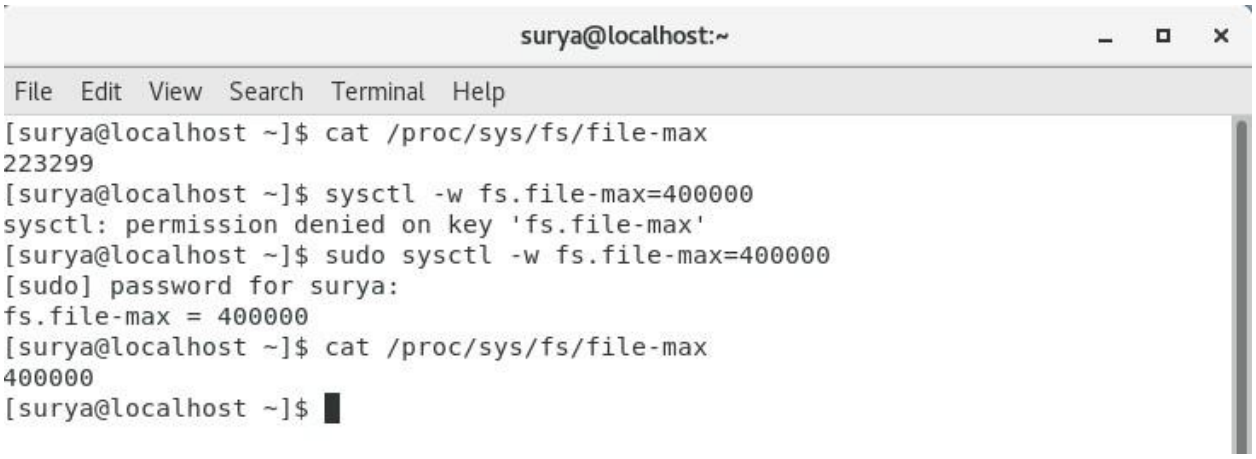
adduser suser
touch file.txt
chown suser file.txt
chgrp suser file.txt

2. Switch to sample user in terminal then change the permission of the file.txt to the user and group as rwx

sudo su suser
chmod +rwx file.txt

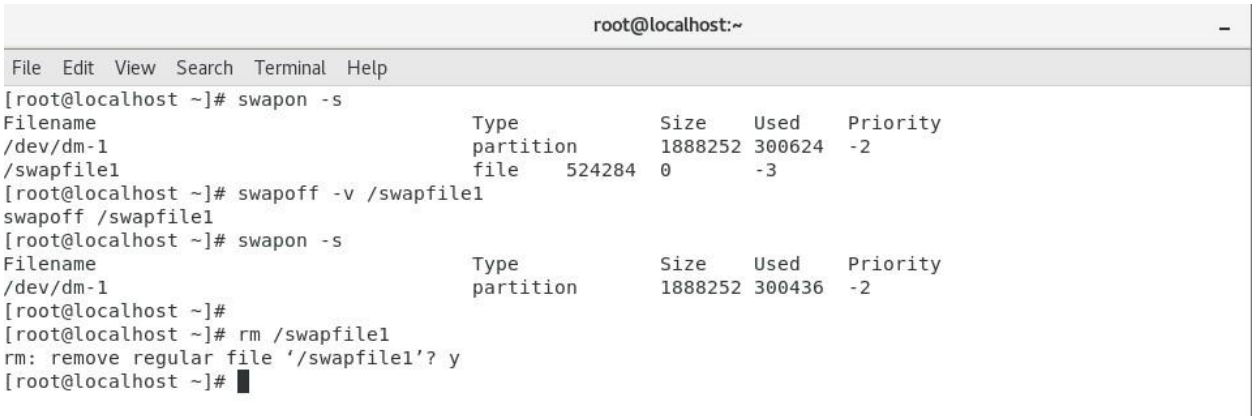
3. Increase the open file limit

cat /proc/sys/fs/file-max
sysctl -w fs.file-max=400000



4. Increase the swapsize by adding new swap storage

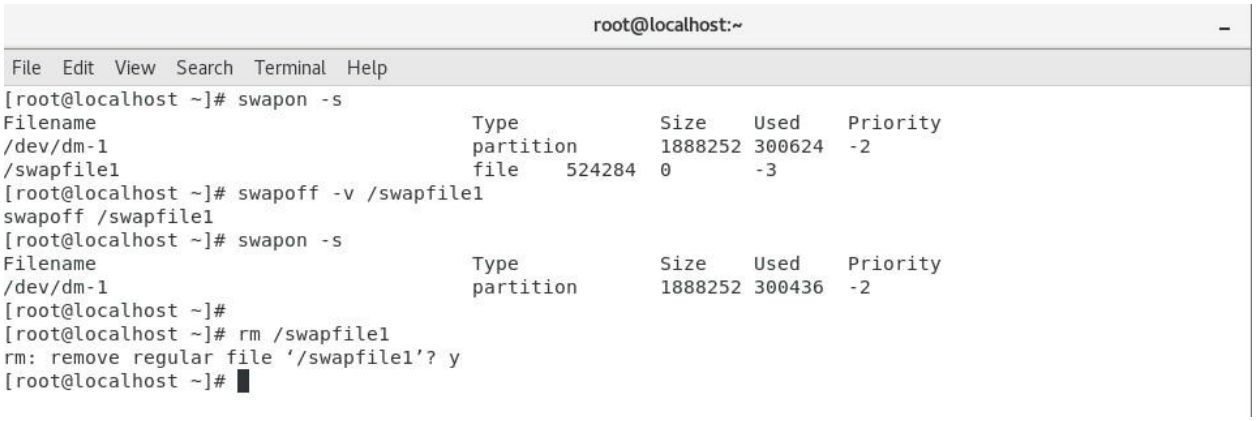
dd if=/dev/zero of=/swapfile1 bs=1024 count=524288
chmod 600 /swapfile1
mkswap /swapfile1
Swapon /swapfile1



5.Remove the swap storage

swapoff -v /swapfile1
rm /swapfile1

To check use => swapon -s



6. Write an script that will read the file content.txt that contains the names,age,job title and print the value in this order “name(age) – jobtitle”
Contents.txt file contents
Raj 30 Developer
Ram 25 Ops
Gokul 21 Intern
Sundar 45 CEO
Anil 38 SRE
Output example
Raj(30) – Developer

touch content.txt
awk '{print $1,"("$2")","- " $3}' content.txt

7. Write an script that will get the company name as arguments(command line arguments) and print their products by parsing the following json file

```
{
“Ibm”: [“RedHat”,”Mainframe”,”IBM cloud”,”RHEL”],
“google”: [“k8s”,”kaggle”,”firebase”],
“microsoft”: [“windows”,”azure”,”office365”],
“amazon”: [“AWS”,”AWS Gov”,”Amplify”],
“softwareag”: [“webmethods”,”AIRS”,”ADABAS”,”CumIOT”]
}
```

Step 1: touch pro.txt
Step 2: In product-script.sh

```
echo `jq ‘.’$1’’ pro.json`
```

Step 3: Execute the script and give an input as an argument
sh product-script.sh amazon

8. Create an script that will calculate the sum of prime numbers between 0 to an given number(to be given as an argument) name it child.sh, create another script named parent.sh which check if the child script exists in current dir and if exists change permission to execute for current user and call the child script from the parent script passing the number as argument

Child.sh

```
#!/bin/bash
#Author : Surya
#Description : Sum of n Prime numbers
#Creation date : 17-06-2022

for((i=2; i<=$1; i++))
do
 for  ((j=2; j<=$(($i/2)); j++))
 do
  if [ $(($i % $j)) -eq 0 ]
   then
    j=$i
   break
  fi
 done
 if [ $j -ne  $i ]
  then
  sum=$(($sum + $i))
 Fi
done
echo "Sum is $(($sum+2))"
```

Parent.sh

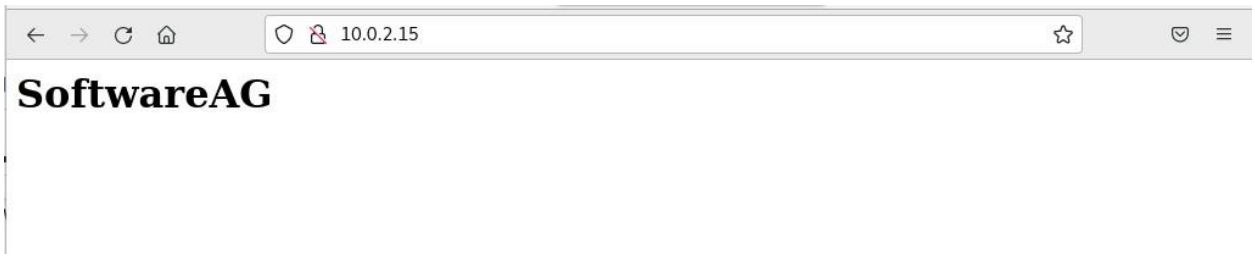
```
#!/bin/bash
#Author : Surya
#Description : Script to check if file exists or not.
#Creation date : 17-06-2022

if [ -e 'child.sh' ]
then
  chmod a+x child.sh
  sh child.sh $1
else
 echo File not exit
fi
```

9. Install the apache httpd server(centos preferred), and the home page should say “SoftwareAG”

```
sudo yum update httpd
sudo systemctl start httpd
sudo systemctl status httpd
```

Next step is to edit the index.html file in the /var/www/html directory to display SoftwareAG



10. Make the Https server an Linux systemd service and write an script to stop, start and restart the server via systemctl

Create a service file in /etc/systemd/system/ directory

```
[Unit]
Description=Http Server

[Service]
User=surya
ExecStart=/home/surya/Desktop/linuxex/http-server.sh
Restart=always
```

Script to start stop and restart the server

```
#!/bin/bash

echo Enter your choice
echo "1)Start server"
echo "2)Stop server"
echo "3)Restart server"
echo "4)Server status"

read choice

case $choice in

 1) sudo systemctl start http-server.service;;
 2) sudo systemctl stop http-server.service;;
 3) sudo systemctl restart http-server.service;;
 4) sudo systemctl status http-server.service;;

esac
```

Docker Exercises

1. Create an docker volume named sampledata

```
docker volume create sampledata
```

2. Spin up an ubuntu docker container, mount the sampledata docker volume. Pass the env variable os=ubuntu to the docker container when bringing it up

```
docker run -it -v sampledata:/sdirectory --env os=ubuntu ubuntu
```

3. Create an dir names /sampledir mount it to an nginx:1.19.0 docker container

```
docker run -d -it --name nginx -v nginxdir:/sampledir nginx:1.19.0
```

4. Create an docker file to create an image with httpd2 server in it, on running the image as an container the server should start up.

Dockerfile

```
FROM httpd
COPY /webfiles /var/www/html
docker build -t apache-image /home/surya/Desktop/dockerex/
```

5. Create an container out of the image you created in earlier step expose the 80 port to the parent host

```
docker run -d --name apache-server -p 300:80 apache-image
```

6. Create an bridge network and spin up two containers in the network one with the image you created

```
docker network create my-net
docker run -d --name apache-server1 apache-image
docker run -d --name apache-server2 apache-image
docker network connect my-net apache-server1
docker network connect my-net apache-server2
```

7. Create an docker compose file with an nginx and an redis server both connected via an bridge network

```
version: "3"
services:
 nginx:
  image: nginx
  networks:
   - bridgenet
 redis:
  image: redis
  networks:
   - bridgenet
networks:
  bridgenet:
```

Run the file using this command
docker-compose up

Scenario Based exercise

Write a Docker file to create an image with the apache httpd2 server and make the start of the httpd2 server as the container entry point.
Create a git repository with some html files for our web server, clone the repo to a local linux machine.
Create a bash script which will pull the contents from the GitHub Repo every 5 hours (use Linux Cron) to the local cloned repo, The script should log all these activities to an log file with timestamp (file which keep track of all script runs)
Spin up the container out of the image you create from the docker file, expose the port 80, mount the dir where you have cloned the git repo to the dir inside the container from where httpd2 server reads the HTML files (/var/www/html)
Do all the above steps inside an AWS linux EC2 machine and share the public ip/DNS of the VM, by allowing the traffic to reach your webpage

First I have launched an EC2 instance with Ubuntu AMI.

Next step is to install the git and docker in the ec2 instance by using the commands.

```
sudo apt-get update
sudo apt-get install docker
sudo apt-get install git
```

Docker file to create an image with the apache httpd2 server and make the start of the httpd2 server as the container entry point

```
FROM ubuntu:latest
RUN apt-get -y update
RUN apt-get install -y apache2
RUN apt-get install -y apache2-utils
EXPOSE 80
COPY dockerex/landingpageSAG/index.html /var/www/html/index.html
ENTRYPOINT ["/usr/sbin/apache2ctl"]
CMD ["-D","FOREGROUND"]
```

Next step is to clone the repository into the ec2 directory

sudo clone https://github.com/suryaprasaath/landingpageSAG.git

Bash script which will pull the contents from the GitHub Repo every 5 hours (use Linux Cron) to the local cloned repo

```
update.sh
git pull
date>>log.txt
```

```
crontab -l
* */5 * * * update.sh
```

Build the image

```
sudo docker build t apache-server dockerex/Dockerfile
```

Next step is to spin up the container

```
sudo docker run -p 80:80 -v /home/ubuntu/dockerex/landingpageSAG/:/var/www/html apache-server
```

ocker