Part 1 Step 1, 2, and 3:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo ls /dev/nvme*
/dev/nvme0  /dev/nvme0n1  /dev/nvme0n1p1  /dev/nvme0n1p14  /dev/nvme0n1p15
```

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo fdisk -l
Disk /dev/nvme0n1: 24 GiB, 25769803776 bytes, 50331648 sectors
Disk model: Amazon Elastic Block Store
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: 71DFD449-F0E2-4249-B370-2373EC9CF964

Device           Start      End  Sectors  Size Type
/dev/nvme0n1p1  262144 50331614 50069471 23.9G Linux filesystem
/dev/nvme0n1p14   2048     8191     6144    3M BIOS boot
/dev/nvme0n1p15   8192   262143   253952  124M EFI System

Partition table entries are not in disk order.

┌──(root㉿kali.example.com)-[/home/student]
└─# sudo parted -l
Model: Amazon Elastic Block Store (nvme)
Disk /dev/nvme0n1: 25.8GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name  Flags
14      1049kB  4194kB  3146kB                     bios_grub
15      4194kB  134MB   130MB   fat16              boot, esp
 1      134MB   25.8GB  25.6GB  ext4
```

Part 2 Step 1:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo dd if=/dev/zero of=~/JSAMO006.vdi bs=1M count=200
200+0 records in
200+0 records out
209715200 bytes (210 MB, 200 MiB) copied, 0.129986 s, 1.6 GB/s
```

Part 2 Step 2:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo losetup -fP ~/JSAMO006.vdi
```

Part 2 Step 3:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo ls /dev/nvme*
/dev/nvme0  /dev/nvme0n1  /dev/nvme0n1p1  /dev/nvme0n1p14  /dev/nvme0n1p15

┌──(root㉿kali.example.com)-[/home/student]
└─# sudo fdisk -l
Disk /dev/nvme0n1: 24 GiB, 25769803776 bytes, 50331648 sectors
Disk model: Amazon Elastic Block Store
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes
Disklabel type: gpt
Disk identifier: 71DFD449-F0E2-4249-B370-2373EC9CF964

Device           Start      End  Sectors  Size Type
/dev/nvme0n1p1  262144 50331614 50069471 23.9G Linux filesystem
/dev/nvme0n1p14   2048     8191     6144    3M BIOS boot
/dev/nvme0n1p15   8192   262143   253952  124M EFI System

Partition table entries are not in disk order.

Disk /dev/loop0: 200 MiB, 209715200 bytes, 409600 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

┌──(root㉿kali.example.com)-[/home/student]
└─# sudo parted -l
Model: Amazon Elastic Block Store (nvme)
Disk /dev/nvme0n1: 25.8GB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name  Flags
14      1049kB  4194kB  3146kB                     bios_grub
15      4194kB  134MB   130MB   fat16              boot, esp
 1      134MB   25.8GB  25.6GB  ext4
```

Part 3 Step 1:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo mkfs.ext4 /dev/loop0
mke2fs 1.47.2 (1-Jan-2025)
Discarding device blocks: done
Creating filesystem with 204800 1k blocks and 51200 inodes
Filesystem UUID: 4ca78545-d98a-4072-9bac-67161575c832
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

Part 3 Step 2

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo mkdir /cyse

┌──(root㉿kali.example.com)-[/home/student]
└─# sudo mount /dev/loop0 /cyse
```

Part 3 Step 3:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.9G     0  1.9G   0% /dev
tmpfs           387M 1004K  386M   1% /run
/dev/nvme0n1p1   24G   14G  8.5G  63% /
tmpfs           1.9G  4.0K  1.9G   1% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           1.9G  8.0K  1.9G   1% /tmp
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-journald.service
/dev/nvme0n1p15 124M  286K  124M   1% /boot/efi
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-resolved.service
tmpfs           387M  108K  387M   1% /run/user/1000
tmpfs           1.0M     0  1.0M   0% /run/credentials/getty@tty1.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/serial-getty@ttyS0.service
tmpfs           387M  100K  387M   1% /run/user/0
/dev/loop0      182M   64K  168M   1% /cyse
```

Part 3 Step 4:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# echo Julian Samonte | sudo tee /cyse/JSAMO006.txt
Julian Samonte
```

Part 3 Step 5:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# sudo umount /cyse
```

Part 3 Step 6:

```
┌──(root㉿kali.example.com)-[/home/student]
└─# ls /cyse
```

Part 4:

1. Explain the purpose of using the `sudo` command with `ls /dev/sd*` and `ls /dev/nvme*`. Why is administrator privilege required in this context?

Using ‘sudo’ is important in making sure that the user can access all of the devices. Using sudo does not limit the user to certain devices within the computer. If sudo were not used it is possible that permission could be denied.

2. What is a loop device, and why do we use `losetup` to attach the virtual disk file as a loop device in this lab?

A loop device is a virtual device that lets a regular file act as if it were a hard drive. Losetup attaches a file to a loop device and it appears as a real disk. Losetup allows a file to act as a real disk in Linux so that we don’t have to do anything physically.

3. Why do we format the virtual disk using `mkfs.ext4`? Explain what this command does and why we chose the `ext4` filesystem specifically.

Mkfs is a front-end command for creating file systems and ext4 is the type of file system that is made with the command. An ext4 file system organizes data on a storage device and is a good file system for large files.

4. After mounting the virtual disk to `/cyse`, what changes should you observe in the output of `df-h`? Explain how `df` helps verify that the disk is mounted correctly.

The change in the output of ‘df-h’ is /dev/loop0 is added onto the list and in the mounted on category it lists “/cyse”.

5. Why is it important to unmount a directory (like `/cyse` in this lab) before detaching a virtual disk? What could happen if you detach a disk without unmounting it first?

When unmounting filesystems a file may be open from the filesystem or a process may be running from the filesystem. If we try to detach before unmounting it could cause data loss or corruption. Unmounting makes sure all of the files are closed and the data is secured.

6. After creating a file on the mounted virtual disk and then unmounting the disk, what do you expect to see when you check the contents of `/cyse`? Explain why this happens.

If you check the contents of ‘/cyse’ then it appears empty. This is because the files were created onto the virtual disk and not connected to ‘/cyse’ anymore.

7. How does using a virtual disk file differ from using a physical disk partition on your system? What are some advantages and disadvantages of using virtual disks in cybersecurity labs?

The virtual disk is just a file that requires a loop device and can be moved or deleted easily without having to deal with any of the real physical hardware. A physical disk partition is in the physical drive and requires the user to actually have to mess with the hardware. It is also harder to modify it. An advantage is that you can modify it, mount it, and even reset it without having to put the real system and its components at risk. It is perfect for labs and beginners that need to learn without the risk. Compared to a physical disk, a virtual disk is slower and has less space. It also requires the extra step of using ‘losetup’ in order to mount and access it.