Exercise 11.1 The tmpfs Special Filesystem

tmpfs is one of many special filesystems used under **Linux**. Some of these are not really used as filesystems, but just take advantage of the filesystem abstraction. However, **tmpfs** is a real filesystem that applications can do I/O on.

Essentially, **tmpfs** functions as a **ramdisk**; it resides purely in memory. But it has some nice properties that old-fashioned conventional ramdisk implementations did not have:

1. The filesystem adjusts its size (and thus the memory that is used) dynamically; it starts at zero and expands as necessary up to the maximum size it was mounted with.
2. If your RAM gets exhausted, **tmpfs** can utilize swap space. (You still can't try to put more in the filesystem than its maximum capacity allows, however.)
3. **tmpfs** does not require having a normal filesystem placed in it, such as **ext3** or **vfat**; it has its own methods for dealing with files and I/O that are aware that it is really just space in memory (it is not actually a block device), and as such are optimized for speed. Thus there is no need to pre-format the filesystem with a `mkfs` command; you merely just have to mount it and use it.

Mount a new instance of **tmpfs** anywhere on your directory structure with a command like:

```
$ sudo mkdir /mnt/tmpfs
$ sudo mount -t tmpfs none /mnt/tmpfs
```

See how much space the filesystem has been given and how much it is using:

```
$ df -h /mnt/tmpfs
```

You should see it has been allotted a default value of half of your RAM; however, the usage is zero, and will only start to grow as you place files on `/mnt/tmpfs`.

You could change the allotted size as a mount option as in:

```
$ sudo mount -t tmpfs -o size=1G none /mnt/tmpfs
```

You might try filling it up until you reach full capacity and see what happens. Do not forget to unmount when you are done with:

```
$ sudo umount /mnt/tmpfs
```

Virtually all modern **Linux** distributions mount an instance of **tmpfs** at `/dev/shm`:

```
$ df -h /dev/shm
Filesystem     Type   Size  Used Avail Use% Mounted on
tmpfs          tmpfs  3.9G   24M  3.9G   1% /dev/shm
```

Many applications use this such as when they are using **POSIX** shared memory as an inter-process communication mechanism. Any user can create, read and write files in `/dev/shm`, so it is a good place to create temporary files in memory.

Create some files in `/dev/shm` and note how the filesystem is filling up with **df**.

In addition, many distributions mount multiple instances of **tmpfs**; for example, on a **RHEL 7** system:

```
$ df -h | grep tmpfs
devtmpfs                                        devtmpfs  3.9G     0  3.9G   0% /dev
tmpfs                                           tmpfs     3.9G   24M  3.9G   1% /dev/shm
tmpfs                                           tmpfs     3.9G  9.2M  3.9G   1% /run
tmpfs                                           tmpfs     3.9G     0  3.9G   0% /sys/fs/cgroup
/tmp/vmware-coop/564d9ea7-8e8e-29c0-2682-e5d3de3a51d8 tmpfs     3.3G     0  3.3G   0% /tmp/vmware-coop/
                                                                                  564d9ea7-8e8e-29c0-2682-e5d3de3a51d8
/tmp/vmware-coop/564d7668-ec55-ee45-f33e-c8e97e956190 tmpfs     2.3G  2.0G  256M  89% /tmp/vmware-coop/
                                                                                  564d7668-ec55-ee45-f33e-c8e97e956190
none                                            tmpfs     1.0G  1.0G     0 100% /tmp/ohno
```

Notice this was run on a system with 8 GB of ram, so clearly you can't have all these **tmpfs** filesystems actually using the 4 GB they have each been allotted!

Some distributions (such as **Fedora**) may (by default) mount `/tmp` as a **tmpfs** system; in such cases one has to avoid putting large files in `/tmp` to avoid running out of memory. Or one can disable this behavior as we discussed earlier when describing `/tmp`.