How it works...

The `mktemp` command is very straightforward. It generates a random file and returns its filename (or directory names, in case of directories).

When providing custom templates, `X` will be replaced by a random alphanumeric character. Also note that there must be at least three `X` characters in the template for `mktemp` to work.

Splitting files and data

Splitting of files into many smaller pieces becomes essential in certain situations. Earlier, when memory was limited with devices such as floppy disks, it was crucial to split files into smaller file sizes to split files across many disks. However, nowadays we split files for other purposes, such as readability, for generating logs, sending files over e-mail, and so on. In this recipe we will see various ways of splitting files in different chunks.

How to do it...

Let's say we have a test file called `data.file`, which has a size of 100 KB. You can split this file into smaller files of 10k each by specifying the split size as follows:

```
$ split -b 10k data.file
$ ls
data.file  xaa  xab  xac  xad  xae  xaf  xag  xah  xai  xaj
```

It will split `data.file` into many files, each of a 10k chunk. The chunks will be named the manner `xab`, `xac`, `xad`, and so on. This means it will have alphabetic suffixes. To use the numeric suffixes, use an additional `-d` argument. It is also possible to specify a suffix length using `-a length`:

```
$ split -b 10k data.file -d -a 4
```

Instead of the `k` (kilobyte) suffix we can use `M` for MB, `G` for GB, `c` for byte, `w` for word, and so on.

```
$ ls
data.file x0009  x0019  x0029  x0039  x0049  x0059  x0069  x0079
```

There's more...

The `split` command has more options. Let's go through them.

Specifying a filename prefix for the split files

The previous split files have a filename prefix `x`. We can also use our own filename prefix by providing a prefix filename. The last command argument for the `split` command is `PREFIX`. It is in the format:

```
$ split [COMMAND_ARGS] PREFIX
```

Let's run the previous command with the prefix filename for split files:

```
$ split -b 10k data.file -d -a 4 split_file
$ ls
data.file       split_file0002  split_file0005  split_file0008  strtok.c
split_file0000  split_file0003  split_file0006  split_file0009
split_file0001  split_file0004  split_file0007
```

To split files based on the number of lines in each split rather than chunk size, use `-l no_of_lines` as follows:

```
$ split -l 10 data.file
# Splits into files of 10 lines each.
```

There is another interesting utility called `csplit`. It can be used to split logfile-based specified conditions and string match options. Let's see how to work with it.

`csplit` is a variant of the `split` utility. The `split` utility can only split files based on chunk size or based on the number of lines. `csplit` makes the split based on context based split. It can be used to split files based on the existence of a certain word or text content.

Look at the following example log:

```
$ cat server.log
SERVER-1
[connection] 192.168.0.1 success
[connection] 192.168.0.2 failed
[disconnect] 192.168.0.3 pending
[connection] 192.168.0.4 success
SERVER-2
[connection] 192.168.0.1 failed
[connection] 192.168.0.2 failed
[disconnect] 192.168.0.3 success
[connection] 192.168.0.4 failed
SERVER-3
[connection] 192.168.0.1 pending
```