More File Manipulation Utilities

Part 1: Compression and Uncompression

Compress and Uncompress

There is a compression utility in Unix. This reduces the size of a file by encoding it in a compact form. When compressed, the information in the file cannot be accessed without uncompressing it. Compression is useful as it can save on disk space and when fetching a file over a network reduces transmission times.

In order to compress a file using the compress utility only the filename is required. There are various options but these are not normally necessary. For example, in order to compress the file largefile the you would issue the following command.

```
compress largefile
```

The tile created by this command is called largefile.Z. Compress adds the characters .Z to the filename of the compressed version in order to indicate that it is compressed. The size of the compressed file depends on the file contents and the size of the original file.

The uncompress command can be used to uncompress this file.

```
uncompress largefile
```

Note that the .Z is not needed.

You can use the Is command to see how by how much the file shrinks and grows when it is compressed. The compressed files are binary files, so care should be taken when transferring these over networks. Appropriate options should be used with ftp and these should not be sent as email unless suitable ascii encoding is used.

gzip/gunzip

There are similar utilities in use which were not traditionally part of Unix but are universally available on current Unix and Linux systems. The most common is called gzip, which creates files with .gz added to the filename. The command name to uncompress these is called gunzip. These commands provide better compression ratios and a wide range of extra functions and can handle standard compressed files.

Part 2: tar, the tape archive command

Tar archives and extracts tiles from an archive file. Traditionally, this tile refers to a tape drive, but tar s often used to create and extract riles from archive tiles. The data or programs stored n such files must be extracted before they can be used. Many packages available on the Internet are stored as tar riles normally these are compressed). Archive tiles offer several advantages over separate riles. A single archive is much easier to move than ¡collection or tiles. Secondly, tar can archive entire directory subtrees with all their files making transfer even more convenient.

Tar has many options. However only a few are necessary in order to create archives and examine and extract files from archives. Unlike compress, tar will leave the original files and directories intact by default.

In order to create a tar file, the c (create) option and the f (file) option which specifies the pathname to be created are needed. In addition, the pathname or pathnames of the files and directories which make up the archive are needed. The y option is recommended, as this displays each pathname as it is added. For example,

```
tar cvf mfiles.tar m*
```

will create a tar file containing all tiles and directories in the current directory which start with “m”. The directories will include their contents including their subdirectories.

```
tar cvf project.tar project1
```

creates a tar file containing the subdirectory project and its contents. It is worth noting that tar will use absolute or relative pathnames. It is strongly recommended that relative pathnames are always used because systems vary in their filestore layout and you (or anyone else) may not have the access to create complete arbitrary subtrees on systems even if this was desirable.

The tar part of the filename is only a convention, but it is one that we would recommend that you use. If you wished to create a tape archive on a real tape, the pathname of the device would be used instead of the filename. The pathname of the device differs between Unix systems. Several may be available on the same systems which may refer to different drives or may refer to the same drive with different options, such as whether the tape is rewound after writing. Most of the devices such as tape drives are in a directory called /dev or one of its subdirectories depending on the version of Unix being used. An example of writing a tape would look something like

```
tar cvf /dev/rstO project1
```

This example will not work with the course usernames. as we do not have the facilities necessary for entire classes to write to tapes.

In order to look at the contents of a tar file or tape archive, the **t** (table) option is used. Use of the **v** (verbose) option is recommended as this displays all file details rather than just the filenames For example

```
tar tvf mfiles.tar
```

Extraction of tiles is achieved with the x option. For example

```
tar xvf mfiles.tar
```

By default, all files in the archive are extracted. However a filenames are given, only the specified tiles are extracted. For example.

```
tar xvf mfiles.tar m1 m2 m3 m4
```

Extraction does not destroy the copies of the files in the archive. There are other options available performing a variety of functions such as adding or replacing selected tiles.

Part 3 The make command

The make command is extremely useful for setting up and installing software, or for creation of files from various sources. Instructions are stored in a file which is normally called Makefile. This course will not cover the make command in detail, but it is covered briefly because you may need it in order to get software to work. The use of make on an existing Makefile is much simpler than setting up a new makefile.

The make command is either given on its own

```
make
```

or can take options defined within the Makefile. These options refer to “targets” which make will attempt to make according to the rules in the Makefile. These rules contain filenames and commands. For example.

```
make install
```

This option is commonly used to install software. When installing software, it may be necessary to have “root” access if software is to be installed in system directories.

A very simple example of use of make is included in the demonstration filestore.

makedemo.tar.Z

This file is a small package containing a simple program which has been put into a tar file and compressed. This is typical of the way many public domain software packages are distributed for Unix systems and the purpose of this exercise is to demonstrate briefly the steps involved in building and installing such a package.

First you must uncompress the file,

```
uncompress makedemo.tar
```

At this point it is usually worth checking the contents of the tar file before extracting any files as some tar files will extract files to the current directory, which may be inconvenient. If this is the case, it may be wise to create a new subdirectory and make it your working directory before extracting the directory and files. In this example, all files will be extracted to a subdirectory which will be created for you.

```
tar xvf makedemo.tar
```

Finally, make the extracted directory your working directory and issue the make command. This will create a program called file called hello. The install option will install this in your SHOME/bin directory. which is on your PATH.

```
make
```

```
make install
```

You can then move to your home directory and issue the command

```
hello
```

The program will run. In real situations, there will be normally be a README tile which will give instructions. Some packages will require you to edit various configuration tiles before making and installing.