1 File System Operations

Manipulating files and directories is surprising useful when undertaking complex projects. The most important file system commands are located in the modules os and shutil. This workshop assumes that

```
import os
import shutil
```

have been included.

1.1 Changing the Working Directory

The working directory is where files can be created and accessed without any path information. `os.getcwd()` can be used to determine the current working directory, and `os.chdir(path)` can be used to change the working directory, where path is a directory, such as `/temp` or `c:`
`temp`.Alternatively, path can can be .. to more up the directory tree.

```
pwd = os.getcwd()
os.chdir('c:\\temp')
os.chdir(r'c:\temp') # Raw string, no need to escape \
os.chdir('c:/temp') # Identical
os.chdir('..') # Walk up the directory tree
os.getcwd() # Now in 'c:\\'
```

1.2 Creating and Deleting Directories

Directories can be created using `os.mkdir(dirname)`, although it must be the case that the higher level directories exist (e.g. to create /home/username/Python/temp, it /home/username/Python already exists). `os.makedirs(dirname)` works similar to `os.mkdir(dirname)`, except that is will create any higher level directories needed to create the target directory. Empty directories can be deleted using `os.rmdir(dirname)` – if the directory is not empty, an error occurs. `shutil.rmtree(dirname)` works similarly to `os.rmdir(dirname)`, except that it will delete the directory, and any files or other directories contained in the directory.

```
os.mkdir('c:\\temp\\test')
os.makedirs('c:/temp/test/level2/level3') # mkdir will fail
os.rmdir('c:\\temp\\test\\level2\\level3')
shutil.rmtree('c:\\temp\\test') # rmdir fails, since not empty
```

1.3 Listing the Contents of a Directory

The contents of a directory can be retrieved in a list using os.listdir(dirname), or simply `os.listdir('.')` to list the current working directory. The list returned contains all files and directories. `os.path.isdir( name )` can be used to determine whether a value in the list is a directory, and `os.path.isfile(name)` can be used to determine if it is a file. `os.path` contains other useful functions for working with directory listings and file attributes.

```
os.chdir('c:\\temp')
files = os.listdir('.')
for f in files:
if os.path.isdir(f):
print(f, ' is a directory.')
elif os.path.isfile(f):
print(f, ' is a file.')
else:
print(f, ' is a something else.')
```

A more sophisticated listing which accepts wildcards and is similar to `dir` (Windows) and `ls` (Linux) can be constructed using the *glob* module.

```
import glob
files = glob.glob('c:\\temp\\*.txt')
for file in files:
print(file)
```

1.4 Copying, Moving and Deleting Files

File contents can be copied using `shutil.copy( src , dest )`, `shutil.copy2( src , dest )` or `shutil.copyfile( src , dest )`. These functions are all similar, and the

differences are:

- `shutil.copy` will accept either a filename or a directory as dest. If a directory is given, the a file is created in the directory with the same name as the original file
- `shutil.copyfile` requires a filename for dest.
- `shutil.copy2` is identical to `shutil.copy` except that metadata, such as last access times, is also copied.

Finally, `shutil.copytree( src , dest )` will copy an entire directory tree, starting from the directory src to the directory dest, which must not exist. shutil.move(src,dest) is similar to `shutil.copytree`, except that it moves a file or directory tree to a new location. If preserving file metadata (such as permissions or file streams) is important, it is better use system commands (copy or move on Windows, cp or mv on Linux) as an external program.

```
os.chdir('c:\\temp\\python')
# Make an empty file
f = file('file.ext','w')
f.close()
# Copies file.ext to 'c:\temp\'
shutil.copy('file.ext','c:\\temp\\')
# Copies file.ext to 'c:\temp\\python\file2.ext'
shutil.copy('file.ext','file2.ext')
# Copies file.ext to 'c:\\temp\\file3.ext', plus metadata
shutil.copy2('file.ext','file3.ext')
shutil.copytree('c:\\temp\\python\\','c:\\temp\\newdir\\')
shutil.move('c:\\temp\\newdir\\','c:\\temp\\newdir2\\')
```