1 Writing to and reading from files

For data analytics we will usually be using two libraries called NumPy and Pandas which have their own simple methods for importing data, but here is the standard Python method. This standard method may sometimes have an advantage that data may be written and read one line at a time without having all data held in memory. This may be useful for very large data files.

1.1 File access modes

Files may be opened with different access control models

Text file access modes:

"r" Read (if file does not exist there will be an error)

"w" Write If the file exists contents are overwritten. If the file does not exist, it is created.

"a" Append - will add to file, or will create a file if no file exists

"r+" Read from and write to a file

"w+" Write to and read from a text file. If the file exists contents are overwritten. If the file does not exist, it is created.

"a+" Append and read from a text file. If the file exists, new data is appended to it. If the files doesn't exist, it's created.

1.2 Creating a file with write

This method where we open a file to save to with the 'w' argument will create a new file or overwrite an old one.

In this standard way of using a file we begin by opening a file and end by closing the file:

```
text_file = open('write_it.txt', 'w')
text_file.write('Line 1\n')
text_file.write('This is line 2\n')
text_file.write('That makes this line 3\n')
text_file.close()
\begin{verbatim}

An alternative to opne and closing the file is using the with statement:

\begin{verbatim}
with open('write_it.txt', 'w') as text_file:
    text_file.write('Line 1\n')
    text_file.write('This is line 2\n')
    text_file.write('That makes this line 3\n')
```

1.3 Adding to a file

Using the 'a' mode we will add to an existing file. We will use the with statement below, but the open/close method will work as well.

```
with open('write_it.txt', 'a') as text_file:
    text_file.write('This is a fourth line\n')
```

1.4 Writing multiple lines

Multiple lines may be written using a loop with the *write* method, or may we can use *writelines*:

```
'write_it.txt'
text_file = open('write_it.txt', 'w')
lines = ['Line 1\n',
         'This is line 2\n',
         'That makes this line 3\n']
text_file.writelines(lines)
text_file.close()
```

1.5 Reading a file

We can read a file in one line at a time (useful for processing very large files which cannot be held in memory), or we can read in all lines.

Here we read one line at a time:

```
text_file = open('write_it.txt', 'r')
text_file = open('write_it.txt', 'r')
print(text_file.readline())
print(text_file.readline())
print(text_file.readline())
text_file.close()

OUT:

Line 1

This is line 2

That makes this line 3
```

Or we can read the entire file in as a list:

```
text_file = open('write_it.txt', 'r')
lines = text_file.readlines()
text_file.close()
print (lines)

OUT:

['Line 1\n', 'This is line 2\n', 'That makes this line 3\n']
\end{verabtim}

A text file may be stepped through as part of a loop:

\begin{verbatim}
text_file = open('write_it.txt', 'r')
for line in text_file:
    print(line)
text_file.close()
OUT:

Line 1

This is line 2

That makes this line 3
```