**INF 1511 – Revision notes – Chapter 6**

1. **File Handling**

A file is a container for a sequence of data objects, represented as sequences of bytes.

The following are the three types of files in Python;

**Text files** are encoded and stored in a format that is viewable by many programs as

well as people. Text files are difficult to update in place.

**Binary files** are formatted to optimize processing speed. A binary file will typically place

data at known offsets, making it possible to access any particular byte using the seek()

method.

**Pickled files** are formatted to store objects. The objects are stored in binary format to

optimize performance.

**Opening a file**

Open (file\_name, mode)

The mode specified determines the actions that will be taken when the file is opened e.g. reading, writing, appending etc. Note the different modes on page 174 of the textbook.

Once the file is opened, several methods (actions) can be invoked including write, read, close etc. check the full list of methods in the textbook on page 175 and 185.

1. **Exception Handling**

Exception handling is used in programing to catch errors that may occur whilst a program is executed so that an error that can be understood by a user is displayed.

The try/except statement is used to execute code in the try block, should an error (exception) be encountered, the code for handling the exception is executed (usually information displayed for the user), the else statement(s) are executed when the try statements do not raise an error.

try:

statement(s)

except SomeException:

code for handling exception

[else:

statement(s)]

The try/finally statement is used to handle an error by executing code in the finally statement. The code in the finally statement is executed regardless of whether the statements in the try section are successful or not. This statement is usually used to ensure that some action is taken whether the try statement(s) succeed or not, e.g. closing a file or collecting garbage

try:

statement(s)

finally:

statement(s)

1. **Raising an exception**

The exceptions raised by Python include execution (runtime) errors such as dividing by 0 or trying to open a file that does not exist etc. if for example you are writing code for a business rule that says a manager can only travel once a month on business trips, and you want to raise an exception when a second trip is captured, then the following code can be used to raise an exception;

try:

if condition:

raise customException, statement for customException

except customException, e:

statements for customException

The if condition would in this case check if the manager has travelled already in the current month.

1. **The Assert Statement**

The assert statement is used to check that the values in variables are correct, e.g. a numerical value is entered and not a string. The syntax of the statement is as follows;

Assert (variable comparison)