Starter - Files

These tasks are designed to refresh the reading and research you have undertaken at home prior to this lesson. If you have not completed the R&R assignment then please speak to your teacher before attempting these exercises.

Modes and encoding

You have been introduced the the various file opening modes that are used in Python. Refresh your knowledge of these concepts by attempting the below tasks.

Task 1

Mode	Explanation
`a`	Append: Appends input onto end of file - last character of last line
`w`	Write: will overwrite file with new input
`r`	Read: will read in each line to python

Task 2

You must set the encoding parameter when opening a file. Identify the encoding method you should use and in the space provided explain why:

Encoding	Explanation
UTF-8	This allows the file created to be accessed on any machine running any OS

Reading from a file

The screenshot below shows the result of reading in the names of students from a file and then printing them to the screen.

The code for printing the students to the screen is shown below:

```
for index, student in enumerate(student_file):
    print("{0:>2}. {1}".format(index+1, student))
```

Task 3

For the above code, explain what each of the following sections of code do:

Section	Explanation
`index`	Indexes each line in the file to allow the output to be enumerated
`{0:>2}`	Right aligns and adds a two character column width to the index value

Task 4

In the above screenshot there are gaps between each student in the list. It should look like the screenshot below.

Attempt to explain why there are gaps between each student and then suggest how the above code could be improved to remove them.

When printing lines in python, a new line is automatically added onto the end of each print statement. However, when writing files we must add that new line in and so when printing in python reads these out to us, it prints two new lines. To fix this you must remove python's new line function by adding "end =' ' "

Exception Handling

Exception Handling is used to deal with **known errors** in a more elegant manor than crashing the program. Take a look at the following code:

```
try:
    score = int(input("Please enter your score: ")
except:
    print("Please enter an integer value only")
print(Your score was {0}".format(score))
```

Task 5

Identify and **explain** (without running the code) all of the errors in this code

Error	Explanation
EOF(Missing bracket in second line)	Will crash the the program
Error type not specified	Not specifying type of error so it will accept all errors

Error	Explanation
final print	needs to be inside “try:” or it may print an empty variable and crash
EOF(Missing “)	Will crash the the program