EMAT10007 – Introduction to Computer Programming

Exercises – Week 7. Reading and Writing Files

7.1 Reading and Writing Files

Essential Questions

Exercise 1 - Writing files

1. Write the following table to a new file called samples.txt:

Sample	Mass(kg)
A	0.600
B	0.455
C	0.550
D	0.505
E	0.550

   Hint: Remember to close the file!
2. Add the new entries: (Sample F: Mass = 0.505 kg, Sample G: Mass = 0.535 kg) to the table in samples.txt.

Exercise 2 - Reading files

1. Write a Python program to read the contents of the table you have just created in samples.txt and print its contents to the console in Spyder.
2. We can convert the iterable file object returned by `open` as a list of lists by casting using `list(<object_name>)`.
   Use this method and then print the third row of the table (excluding the headings).
3. Print the numerical data in the column Mass as a list of `float` values, shown in grams (g)
   Hint: Exclude the column heading from the list. Convert string data to numerical data.

Exercise 3 - Reading and writing files

1. Edit the table so that the column headings are in upper case letters.
   Hint: You can do this manually or use Python's built-in `upper` method (example below) which can be used to convert string data to uppercase letters (there is an equivalent `lower` method for converting to lower case).

```
txt = "Hello"
x = txt.upper()
print(x)
```

   >> HELLO

2. Write a program that:

- opens the file sample_data/price_per_item.csv and prints the contents to display the data
- adds another line (you can make up a new entry to the list of foods and prices).
- prints the new contents to confirm the new line has been added

Advanced Questions

(A) Write a program that edits the Mass of sample B to be 0.485 kg

(B) Write a program that edits the table saved in samples.txt so that all Mass data is rounded to 2 decimal places (the nearest 10g)
Hint: Use the built-in `round` function.

7.2 Imported Modules for Reading and Writing Files:

Essential Questions

Exercise 4 - Writing csv files

1. Use the `csv` module to write the table in Exercise 1.1 to a csv file, samples.csv.
2. Add the new entries: (Sample F: Mass = 0.505 kg, Sample G: Mass = 0.535 kg) to the table in samples.csv.

Exercise 5 - Reading csv files

1. Read the data in sample_data/douglas_data.csv and print it to the Console in Spyder.
2. Print the maximum value (**Hint:** Python built in `max` function) of `density` in the data set.

3. The Python function `sorted()` takes an iterable (e.g. list, string) as an argument and returns it as a sorted list. The argument `reverse` (default value `False` determines whether the items are sorted in ascending or descending (reverse) order.

```
nums = [8, 1, 2]
print(sorted(nums, reverse=True))
```

>> `[8, 2, 1]`
Read the data in sample.csv (created in Exercise 4). Print the values in the Mass column of the imported data in ascending order.

Advanced Questions: Reading and writing a csv file

Advanced Questions

(A) `sorted()` can be also used to sort one list using the values in another list by using `zip` to group the two lists element wise.

```
nums = [8, 1, 2]
txt = [‘a’, ‘b’, ‘c’]
lists = zip(nums, text)
s_lists = sorted(lists) # [(1, ‘b’), (2, ‘c’), (8, ‘a’)]
s_nums = [i[0] for i in s_lists] # [1, 2, 3]
s_txt = [i[1] for i in s_lists] # [‘b’, ‘c’, ‘a’]
```

Using this example, write a program that:

- asks the user's for a player's name and score.
- add them to the high score table in sample_data/scores.txt so that the scores remain in descending order.
- prints a message to the Console to tell the user if the new score is the highest in the table e.g. `‘[Player name] got a new high score of [score]!’`

Hint Remember to open the file using a mode specifier that lets you read *and* write e.g. ‘’r+‘.

Hint Remember, numerical data is exported fro ma csv file as string data.

Hint Remember that after reading, the position is at the end of the file. To return to the beginning of the file to overwrite its contents use ‘seek(0)‘ or ‘truncate()‘

(B) Imagine you have not seen the contents of the file samples.csv (created in Exercise 4). The volume of each sample in cm^3 is: A = 336, B = 231, C = 350, D = 272, E = 300, F = 312, G = 255. Write a program that:

- reads the contents of the samples.csv file .
- finds the density of each sample in kg/m^3.
- writes two new columns to the table: volume(cm3) and density(kg/m3)

Hint Remember to open the file using a mode specifier that lets you read *and* write e.g. ‘’r+‘.

Hint Remember, numerical data is exported fro ma csv file as string data.

Hint Remember that after reading, the position is at the end of the file. To return to the beginning of the file to overwrite its contents use ‘seek(0)‘ or ‘truncate()‘