VIDEO 15 : File I/O & Tuples

This time we'll cover how to read and write files and we'll investigate what a tuple is.

Writing Text to a File

I'll jump directly into the code needed to write text to a file.

CODE

```
# The os module provides methods for file processing
import os

# You can create or use an already created file with open

# If you use w (write) for mode then the file is
# overwritten.
# If you use a (append) you add to the end of the file

# Text is stored using unicode where numbers represent
# all possible characters

# We start the code with with which guarantees the file
# will be closed if the program crashes
with open("mydata.txt", mode="w", encoding="utf-8") as myFile:

    # You can write to the file with write
    # It doesn't add a newline
    myFile.write("Some random text\nMore random text\nAnd some more")
```

Reading Text from a File

Now we'll read text from a file and I'll show you how to perform some common directory procedures.

CODE

```
import os

# Open the file for reading
# You don't have to provide a mode because it is
# read by default
with open("mydata.txt", encoding="utf-8") as my_file:

    # We can read data in a few ways
    # 1. read() reads everything into 1 string
    # 2. readline() reads everything including the first newline
    # 3. readlines() returns a list of every line which includes
    # each newline

    # Use read() to get everything at once
    print(my_file.read())
```

```
# Find out if the file is closed
print(my_file.closed)

# Get the file name
print(my_file.name)

# Get the access mode of the file
print(my_file.mode)

# Rename our file
os.rename("mydata.txt", "mydata2.txt")

# Delete a file
# os.remove("mydata.dat")

# Create a directory
# os.mkdir("mydir")

# Change directories
# os.chdir("mydir")

# Display current directory
print("Current Directory :", os.getcwd())

# Remove a directory, but 1st move back 1 directory
# os.chdir("..")
# os.rmdir("mydir")
```

Read One Line at a Time

You can read one line at a time with readline().

CODE

```
import os

# Open the file
with open("mydata2.txt", encoding="utf-8") as my_file:

    lineNum = 1

    # We'll use a while loop that loops until the data
    # read is empty
    while True:
        line = my_file.readline()

        # line is empty so exit
        if not line:
            break

        print("Line", lineNum, " :", line, end="")

        lineNum += 1
```

Python Problem for you to Solve

For this problem I want you to cycle through each line of text and output the number of words and the average word length. Here is sample output.

Line 1
Number of Words : 3
Avg Word Length : 4.7
Line 2
Number of Words : 3
Avg Word Length : 4.7

We'll use the file we previously worked with.

Solution

```
import os

with open("mydata2.txt", encoding="utf-8") as my_file:

    line_num = 1

    while True:
        line = my_file.readline()

        # line is empty so exit
        if not line:
            break

        print("Line", line_num)

        # Put the words in a list using the space as
        # the boundary between words
        word_list = line.split()

        # Get the number of words with len()
        print("Number of Words :", len(word_list))

        # Incremented for each character
        char_count = 0

        for word in word_list:
            for char in word:
                char_count += 1

        # Divide to find the answer
        avg_num_chars = char_count/len(word_list)

        # Use format to limit to 2 decimals
        print("Avg Word Length : {:.2}".format(avg_num_chars))

        lineNum += 1
```

Tuples

Now as a bonus I'll cover tuples. A Tuple is like a list, but their values can't be changed. Tuples are surrounded with parentheses instead of square brackets. Here is some sample code.

CODE

```
my_tuple = (1, 2, 3, 5, 8)

# Get a value with an index
print("1st Value :", my_tuple[0])

# Get a slice from the 1st index up to but not including
# the 3rd
print(my_tuple[0:3])

# Get the number of items in a Tuple
print("Tuple Length :", len(my_tuple))

# Join or concatenate tuples
more_fibs = my_tuple + (13, 21, 34)

# Check if a value is in a Tuple
print("34 in Tuple :", 34 in more_fibs)

# Iterate through a tuple
for i in more_fibs:
    print(i)

# Convert a List into a Tuple
a_list = [55, 89, 144]
a_tuple = tuple(a_list)

# Convert a Tuple into a List
a_list = list(a_tuple)

# Get max and minimum value
print("Min :", min(a_tuple))
print("Max :", max(a_tuple))
```

I hope you have enjoyed this tutorial. In the next part I'll cover Classes, Objects, Self, __init__, Getters, Setters, Properties, and then create 2 warriors that fight to the death!!!