**Level 1: Reading a Text File**

1. Open a new Python Repl and run the following program.

fileHandle = open("myfile.txt","r")

fileContents = fileHandle.read()

print(fileContents)

fileHandle.close()

1. Why does this program produce a run-time error?

It does not open because there is no file with that name. You must first create the file, than it is possible.

1. Add a text file to your project as follows:
   * Click on “Add File” icon in the files pane/window.
   * Type “myfile.txt” and return.
   * “myfile.txt” is now open in the editor pane/window.
   * Type some text into “myfile.txt”
   * Make sure to add several lines of text

1. Switch back to main.py pile and run the program.
   1. What gets printed out?

The several lines of text that had to be written in the file when creating it.

* 1. Explain the result.

The lines were printed out on to the output (black side).

1. Load and run the following program.

fileHandle = open("myfile.txt","r")

line = fileHandle.readline()

count = 1

while line :

print("Line ", count, " : ",line.strip())

line = fileHandle.readline()

count += 1

fileHandle.close()

1. Compare and contrast the output of the first and second program
   1. How is the read() function similar to the readline() function?

Both read and readline print out the text that was put int the folder

* 1. How is the read() function different from the readline() function?

The read function simple shows the text exactly how you wrote it but the readline function, shows the text line by line beginning with a “Line *n* :”

1. Research the Python open() function for file I/O (input / output).
   1. How do you specify which file to open?

In the brackets you specify which file you want to open by using the name of the file.

* 1. Modify the program to open a different file.

fileHandle = open("agentmyfile.txt")

fileContents = fileHandle.read()

print(fileContents)

fileHandle.close()

1. Research how to open a file in a sub-directory.
   1. Modify the second program to open a file in a sub-directory.
   2. Demo your program to Mr. Nestor
   3. List your program modifications below

fileHandle = open("Spies/agentmyfile.txt")

fileContents = fileHandle.read()

print(fileContents)

fileHandle.close()

**Level 2: Writing a Text File**

1. Research the Python open() function for file I/O (input / output).
   1. What does the file mode “r” mean?

Mode “r” is for reading.

* 1. What mode is used to open a file for writing?

Mode “w”

* 1. What other file modes can be used? List and explain their meanings.

Append means “a”

1. Load and run the following program.

print("Enter test to write to a file")

print("Type STOP to end the program")

print(" ")

lineNumber = 0

while True :

lineNumber += 1

userPrompt = "Enter Line " + str(lineNumber) + " : "

userText = input(userPrompt)

if userText == "STOP" :

break

print(userText)

1. Modify the program to open a text file for writing.
   1. Demo your program to Mr. Nestor
   2. List your program modifications below
2. fileHandle = open("Spies/agentmyfile.txt")
4. fileContents = fileHandle.read()
5. print(fileContents)
6. print("Enter test to write to a file")
7. print("Type STOP to end the program")
8. print(" ")
9. lineNumber = 0
10. while True :
11. lineNumber += 1
12. userPrompt = "Enter Line " + str(lineNumber) + " : "
13. userText = input(userPrompt)
14. if userText == "STOP" :
15. break
16. print(userText)
17. fileHandle.close()

1. Replace the line “print(userText)” with a command to write the value of “userText” to an open file.
   1. Verify that text was written to your file
   2. Demo your program to Mr. Nestor
   3. List your program modifications below
2. fileHandle = open("Spies/agentmyfile.txt")
4. fileContents = fileHandle.read()
5. print(fileContents)
6. print("Enter test to write to a file")
7. print("Type STOP to end the program")
8. print(" ")
9. lineNumber = 0
10. while True :
11. lineNumber += 1
12. userPrompt = "Enter Line " + str(lineNumber) + " : "
13. userText = input(userPrompt)
14. if userText == "STOP" :
15. break
16. fileHandle = open("userText","w")

**Level 3: Binary Files**

1. Add a folder to your repl workspace and call it “images”.
2. Locate and download a “BMP” format image file and add it to your images folder.
   1. The file must be a BMP file. JPG, GIF, PNG, etc. will not work
   2. Add the image by using “drag-and-drop” onto your images folder.
   3. You can use the “Penguin.bmp” file from the GitHub Topic B folder if you want
3. Load the following program
   1. Add it to your repl
   2. Modify the “open” command to read your image file
   3. Run the program and examine the data output.

"""

Function to convert 4 bytes (1 word) into a decimal integer

"""

def convertWordToInteger(dataWord) :

result = int(dataWord[3])

result += 256 \* int(dataWord[2])

result += 512 \* int(dataWord[1])

result += 1024 \*int(dataWord[0])

return result

"""

Function to display raw file data

Each data byte is displayed in row order

"""

def dumpRawData(rawData) :

idx = 0

for row in range(8) :

rowText = " ";

for col in range(8) :

rowText += str(rawData[idx]).zfill(3) + " "

idx += 1

print(rowText)

"""

Main program code begins here

- Start with opening and reading the data file

"""

handle = open("images/Penguin.bmp", "rb")

rawData = handle.read(64)

handle.close()

"""

Print out the RAW data contained at the start of the file

- This is the Header Information

- A BPM (Bitmap) Image has a well defined Header

- Each grouping of bytes has a specific meaning

"""

print(" ")

print("RAW Image Header Data (64 bytes)")

dumpRawData(rawData)

print(" ")

"""

According to the BMP specification the first two bytes

have the value "BM".

"""

print("First Two Bytes")

print(str(rawData[0]).zfill(3), str(rawData[1]).zfill(3))

print(" ")

"""

According to the BMP specification the image Width

is contained in the 4 bytes (1 word) biginning at

position 18

"""

print("Image Width Data")

dataText = str(rawData[18]).zfill(3) + " "

dataText += str(rawData[19]).zfill(3) + " "

dataText += str(rawData[20]).zfill(3) + " "

dataText += str(rawData[21]).zfill(3)

print("Image Width: (raw)", dataText)

dataText = str(rawData[21]).zfill(3) + " "

dataText += str(rawData[20]).zfill(3) + " "

dataText += str(rawData[19]).zfill(3) + " "

dataText += str(rawData[18]).zfill(3)

print("Image Width: (re-ordered)", dataText)

dataWord = [rawData[21],rawData[20],rawData[19],rawData[18]]

print("Image Width: (pixels)", convertWordToInteger(dataWord))

print(" ")

"""

According to the BMP specification the image Height

is contained in the 4 bytes (1 word) biginning at

position 22

"""

print("Image Height Data")

dataText = str(rawData[22]).zfill(3) + " "

dataText += str(rawData[23]).zfill(3) + " "

dataText += str(rawData[24]).zfill(3) + " "

dataText += str(rawData[25]).zfill(3)

print("Image Width: (raw)", dataText)

dataText = str(rawData[25]).zfill(3) + " "

dataText += str(rawData[24]).zfill(3) + " "

dataText += str(rawData[23]).zfill(3) + " "

dataText += str(rawData[22]).zfill(3)

print("Image Width: (re-ordered)", dataText)

dataWord = [rawData[25],rawData[24],rawData[23],rawData[22]]

print("Image Width: (pixels)", convertWordToInteger(dataWord))

print(" ")

"""

END OF PROGRAM

"""

1. Decode the meaning of the first two bytes of data in the header data of a BMP file.
   1. What are the values of the first two bytes?

066 077

* 1. Look up the values in an ASCII character table. Google “ASCII Character Table” or Download the ASCII Conversion Chart from the GitHub Topic B folder.
  2. What ASCII characters do these two bytes represent?

066 = 6

077 = ?

1. Open and examine the BMP file format specification for the “Signature” data field
   1. Open the URL listed below to access the document
   2. According to the document, the first two bytes of data are the “Signature”
   3. What is the description of the “Signature” in a BMP file?

‘BM’

* 1. How does this compare to your answer to question #4 above?

<http://www.ece.ualberta.ca/~elliott/ee552/studentAppNotes/2003_w/misc/bmp_file_format/bmp_file_format.htm>

1. Examine the BMP file format specification for the Width data field
   1. Locate the “Width” data field in the BMP specification document.
   2. What is the size, in bytes, of this field?

4 bytes

* 1. What is the value, in bytes, of this field for your image file? (Look at the program output)

Image Width: (raw) 031 001 000 000

Image Width: (re-ordered) 000 000 001 031

* 1. What is the value, in decimal, of this field for your image file? (Look at the program output)

Image Width: (pixels) 287

1. Examine the BMP file format specification for the Height data field
   1. Locate the “Height” data field in the BMP specification document.
   2. What is the size, in bytes, of this field?

4 bytes

* 1. What is the value, in bytes, of this field for your image file? (Look at the program output)

Image Width: (raw) 045 001 000 000

Image Width: (re-ordered) 000 000 001 045

* 1. What is the value, in decimal, of this field for your image file? (Look at the program output)

Image Width: (pixels) 301

1. Open your BMP image file in an application program like Paint or Photoshop.
   1. What is the size of your image file?

287x301

* 1. How does this compare to the output of the program?

It is the decimal value of the image