**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?

The program produces a run-time error because there is no text file created that has the name “myfile.txt”, so it doesn’t know how to open it.

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?
   2. Explain the result.

What I typed in “myfile.txt” gets printed out in the main.py pile. It prints out what I wrote in the text file because I asked it to open and print the file’s contents, which was *“Henlo, my name is Hasrat. I'm 14 years old and I like drawing and playing videogames. Art is my passion and I love drawing. When I'm older, I'd like to do something in the social sciences. also I like squids. okay bye”*

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?

“Read” is similar to the “readline” function because both of them open up the text file and read its contents.

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

“Read” is different from the “readline” function because “read” reads the contents of the file as it was written. “Readline” begins by stating the line number (“line 1”) and then reads the contents of the file, but it’s continuous, so there are no spaces when a new line begins, making it slightly difficult to read.

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

An example for how to open a specific file would be fileHandle = open("myfile.txt","r"). You use “open” and in brackets, with quotation marks, say the file name. Add “r” to specify that you want the file to be read.

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

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

To open another file) fileHandle = open (“myfilethesequel.txt”,”r”)

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

from pathlib import Path

fileHandle = Path("test/")

file\_to\_open = fileHandle / "myfile.txt"

f = open(file\_to\_open)

print(f.read())

**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?

“r” opens a file for reading only. It’s the default mode.

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

“w” opens a file for writing only. If the file does not exist, it creates a new file for writing.

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

“rb” - Opens a file for reading only in binary format.

“r+” - Opens a file for both reading and writing.

“rb+” - Opens a file for both reading and writing in binary format.

“wb” - Opens a file for writing only in binary format.

“w+” - Opens a file for both writing and reading.

”wb+” - Opens a file for both writing and reading in binary format.

“a” - Opens a file for appending.

“ab” - Opens a file for appending in binary format.

“a+” - Opens a file for both appending and reading.

“ab+” - Opens a file for both appending and reading in binary format.

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

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

print("Type STOP to end the program")

print(" ")

F = open("mychild.txt", "w")

lineNumber = 0

while True :

lineNumber += 1

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

userText = input(userPrompt)

if userText == "STOP" :

break

print(userText)

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

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

print("Type STOP to end the program")

print(" ")

F = open("mychild.txt", "w")

lineNumber = 0

while True :

lineNumber += 1

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

userText = input(userPrompt)

if userText == "STOP" :

break

F = 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("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

"""

On line 25, change it to:

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

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?

6 and ?

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?

Description = BM

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

The offset is similar to the ASCII Conversion Chart.

<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: (re-ordered) 000 000 001 031

Image Width: (raw) 031 001 000 000

* 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 Height: (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?

Image Width in Photoshop = 287

Image Height in Photoshop = 301

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

It compares to the output of the program because the width and height (in pixels) that was stated in the program is exactly the same as the width and height in Photoshop.

**Level 3: Binary Files**

t.b.d.

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