**Lab Exercises 6 (Based on Modules, Packages &amp; File handling)**

1. Write a Python program to read an entire text file.

2.   Write a program that counts lines and characters in a file. With your favorite text editor, code

a Python module called mymod.py, which exports three top-level names:

a) A countLines(name) function that reads an input file and counts the number of lines

in it

b) A countChars(name) function that reads an input file and counts the number of

characters in it

c) A test(name) function that calls both counting functions with a given input filename.

All three mymod functions should expect a filename string to be passed in.

Now, test your module interactively, using import and name qualification to fetch your

exports.

3. Test your mymod module from Exercise 2 interactively, by using from to load the exports

directly, first by name, then using the from\* variant to fetch everything.

4.    Now, add a line in your mymod module that calls the test function automatically only when

the module is run as a script, not when it is imported The line you add will probably test the

value of \_\_name\_\_ for the string &quot;\_\_main\_\_&quot;, as shown in this unit. Try running your

module from the system command line; then, import the module and test its functions

interactively.

5.    Write a second module, myclient.py, which imports mymod and tests its functions;

run myclient from the system command line. If myclient uses from to fetch from mymod,

will mymod’s functions be accessible from the top level of myclient? What if it imports

with import instead? Try coding both variations in myclient and test interactively, by

importing myclient .

6.      Package imports. Finally, import your file from a package. Create a subdirectory

called mypkg nested in a directory on your module import search path, move

the mymod.py module file you created in exercises 2 or 4 into the new directory, and try to

import it with a package import of the form: import mypkg.mymod.

7.   Experiment with module reloads: perform the tests in the changer.py example, changing the

called function’s message and/or behavior repeatedly, without stopping the Python

interpreter. Depending on your system, you might be able to edit changer in another window.

**ANSWERS**

#Write a Python program to read an entire text file.  
file\_name = input("Enter the name of the text file: ")  
  
try:  
 with open(file\_name, 'r') as file:  
 file\_contents = file.read()  
 if file\_contents:  
 print("File Contents:")  
 print(file\_contents)  
 else:  
 print("The file is empty.")  
  
except FileNotFoundError:  
 print(f"The file '{file\_name}' was not found.")  
except PermissionError:  
 print(f"You don't have permission to read '{file\_name}'.")  
except Exception as e:  
 print(f"An error occurred: {e}")

OUTPUT :

C:\Users\DELL\PycharmProjects\pythonProject5\venv\Scripts\python.exe C:\Users\DELL\PycharmProjects\pythonProject5\mypackage\read.py

Enter the name of the text file: C:/Users/DELL/Desktop/SCHOLARSHIP.txt

File Contents:

MY APPLICATION ID NUMBER 2122SJS1002130043. MY AADHAR SEEDING STATUS IS LINKED. MY APPLICATION FORM IS CLEAR BUT SCHOLARSHIP AMOUNT NOT RECIVED.21-22

MY APPLICATION ID NUMBER 2223SJS1002809423. MY AADHAR SEEDING STATUS IS LINKED. MY APPLICATION FORM IS CLEAR BUT SCHOLARSHIP AMOUNT NOT RECIVED.

Process finished with exit code 0

-------------------------------------------------------------------------------------------------------------------------------------

def countLines(name):  
 try:  
 with open(name, 'r') as file:  
 line\_count = sum(1 for line in file)  
 return line\_count  
 except FileNotFoundError:  
 return -1 # File not found  
  
def countChars(name):  
 try:  
 with open(name, 'r') as file:  
 char\_count = len(file.read())  
 return char\_count  
 except FileNotFoundError:  
 return -1 # File not found  
  
def test(name):  
 line\_count = countLines(name)  
 char\_count = countChars(name)  
  
 if line\_count == -1:  
 print(f"File '{name}' not found.")  
 else:  
 print(f"Number of lines in '{name}': {line\_count}")  
  
 if char\_count == -1:  
 print(f"File '{name}' not found.")  
 else:  
 print(f"Number of characters in '{name}': {char\_count}")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 test("C:/Users/DELL/Desktop/SCHOLARSHIP.txt")

OUTPUT:

C:\Users\DELL\PycharmProjects\pythonProject5\venv\Scripts\python.exe C:\Users\DELL\PycharmProjects\pythonProject5\mypackage\mypkg\mymod.py

Number of lines in 'C:/Users/DELL/Desktop/SCHOLARSHIP.txt': 4

Number of characters in 'C:/Users/DELL/Desktop/SCHOLARSHIP.txt': 297

Process finished with exit code 0

from mypkg import mymod  
  
# Testing mymod's functions  
line\_count = mymod.countLines("C:/Users/DELL/Desktop/SCHOLARSHIP.txt")  
char\_count = mymod.countChars("C:/Users/DELL/Desktop/SCHOLARSHIP.txt")  
  
mymod.test("C:/Users/DELL/Desktop/SCHOLARSHIP.txt")  
  
print(f"Number of lines: {line\_count}")  
print(f"Number of characters: {char\_count}")

OUTPUT:

C:\Users\DELL\PycharmProjects\pythonProject5\venv\Scripts\python.exe C:\Users\DELL\PycharmProjects\pythonProject5\mypackage\myclient.py

Number of lines in 'C:/Users/DELL/Desktop/SCHOLARSHIP.txt': 4

Number of characters in 'C:/Users/DELL/Desktop/SCHOLARSHIP.txt': 297

Number of lines: 4

Number of characters: 297

Process finished with exit code 0

# changer.py  
def greeting():  
 return "Hello, World!"  
# changer.py  
def greeting():  
 return "Hi there!"

OUTPUT:

C:\Users\DELL\PycharmProjects\pythonProject5\venv\Scripts\python.exe C:\Users\DELL\PycharmProjects\pythonProject5\changer.py

Process finished with exit code 0

# main.py  
import changer  
  
print(changer.greeting())

OUTPUT:

C:\Users\DELL\PycharmProjects\pythonProject5\venv\Scripts\python.exe C:\Users\DELL\PycharmProjects\pythonProject5\main.py

Hi there!

Process finished with exit code 0

--------------------------------------------------------------------------------------------------------------------------------------