Lab Exercises 6 (Based on Modules & Packages)

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

```
with open("lab_ex6.txt", mode = 'r', encoding = 'utf-8') as f:
    print(f.read())
```

```
Given task to read the n lines of file.
As said we created a file name lab_ex6.
Which is a 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.

```python
def countLines(name):
    try:
        with open(name, 'r') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print(f"Error: The file '{name}' was not found.")
        return 0
def countChars(name):
    try:
        with open(name, 'r') as file:
            content = file.read()
            return len(content)
    except FileNotFoundError:
        print(f"Error: The file '{name}' was not found.")
        return 0
def test(name):
    line_count = countLines(name)
    char_count = countChars(name)

    print(f"Number of lines: {line_count}")
    print(f"Number of characters: {char_count}")
```

```python
import mymod
mymod.test('Lab_ex6.txt')
```

```
Number of lines: 3
Number of characters: 102
```

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.

```python
from mymod import countLines, countChars, test
print("Testing countLines and countChars by name import:")
print("Number of lines:", countLines('Lab_ex6.txt'))
print("Number of characters:", countChars('Lab_ex6.txt'))
test('Lab_ex6.txt')
```

```
Testing countLines and countChars by name import:
Number of lines: 3
Number of characters: 102
Number of lines: 3
Number of characters: 102
```

```
from mymod import *
print("Number of lines:", countLines('Lab_ex6.txt'))
print("Number of characters:", countChars('Lab_ex6.txt'))
test('Lab_ex6.txt')
```

```
Number of lines: 3
Number of characters: 102
Number of lines: 3
Number of characters: 102
```

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 "__main__", as shown in this unit. Try running your

module then, import the module and test its functions interactively.

```
import mymod
print(mymod.countLines('Lab_ex6.txt'))
print(mymod.countChars('Lab_ex6.txt'))
mymod.test('Lab_ex6.txt')
```

```
3
102
Number of lines: 3
Number of characters: 102
```

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

run myclient . 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 .

```
1 from mymod import countLines, countChars, test
2 print("Testing countLines and countChars:")
3 print("Number of lines in 'Lab_ex6.txt':", countLines('Lab_ex6.txt'))
4 print("Number of characters in 'example.txt':", countChars('Lab_ex6.txt'))
5 test('Lab_ex6.txt')
6
```

```
import myclient
mymod.countLines('Lab_ex6.txt')
```

```
3
```

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.

```
import mypkg.mymod
print("Testing countLines and countChars:")
print("Number of lines in 'example.txt':", mypkg.mymod.countLines('Lab_ex6.txt'))
print("Number of characters in 'Lab_ex6.txt':", mypkg.mymod.countChars('Lab_ex6.txt'))

mypkg.mymod.test('Lab_ex6.txt')
```

```
Testing countLines and countChars:
Number of lines in 'example.txt': 3
Number of characters in 'Lab_ex6.txt': 102
Number of lines: 3
Number of characters: 102
```

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.

```
import changer
print(changer.say())
```

```
Hello!
```

```
import changer
print(changer.say())
```

```
Hi!
```

```
import importlib
import changer

print(changer.say())
importlib.reload(changer)

print(changer.say())
```

```
Hello!
Hi!
```