



FEU INSTITUTE OF TECHNOLOGY

COLLEGE OF COMPUTER STUDIES AND MULTIMEDIA ARTS

IT0011 (INTEGRATIVE PROGRAMMING TECHNOLOGIES)

EXERCISE

3

STRING AND FILE HANDLING

Student Name / Group Name:	CARL ALLEN TORNO	
Members (if Group):	Name	Role
Section:	TW21	
Professor:	DR. JAY-AR LALATA	

➤ **PROGRAM OUTCOME/S (PO) ADDRESSED BY THE LABORATORY EXERCISE**

- Analyze a problem and identify and define the computing requirements appropriate to its solution. [PO: B]

➤ **COURSE LEARNING OUTCOME/S (CLO) ADDRESSED BY THE LABORATORY EXERCISE**

- Able to manipulate string and numbers. Use file to persist some data, read and manipulate and read some formatted to process. [CLO: 1]

➤ **INTENDED LEARNING OUTCOME/S (ILO) OF THE LABORATORY EXERCISE**

At the end of this exercise, students must be able to:

- To manipulate string and numbers.
- Use file to persist some data.
- Read some formatted process.

➤ **BACKGROUND INFORMATION**

How To Use str.format()

`str.format()` is an improvement on %-formatting. It uses normal function call syntax and is [extensible through the `__format__\(\)` method](#) on the object being converted to a string.

With `str.format()`, the replacement fields are marked by curly braces:

```
Python >>>
>>> "Hello, {}. You are {}.".format(name, age)
'Hello, Eric. You are 74.'
```

You can reference variables in any order by referencing their index:

```
Python >>>
>>> "Hello, {1}. You are {0}.".format(age, name)
'Hello, Eric. You are 74.'
```

But if you insert the variable names, you get the added perk of being able to pass objects and then reference parameters and methods in between the braces:

```

Python >>>
>>> person = {'name': 'Eric', 'age': 74}
>>> "Hello, {name}. You are {age}.".format(name=person['name'], age=person['age'])
'Hello, Eric. You are 74.'

```

You can also use `**` to do this neat trick with dictionaries:

```

Python >>>
>>> person = {'name': 'Eric', 'age': 74}
>>> "Hello, {name}. You are {age}.".format(**person)
'Hello, Eric. You are 74.'

```

File Handling in Python

Python too supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files. The concept of file handling has stretched over various other languages, but the implementation is either complicated or lengthy, but unlike other concepts of Python, this concept here is also easy and short. Python treats file differently as text or binary and this is important. Each line of code includes a sequence of characters and they form text file. Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character. It ends the current line and tells the interpreter a new one has begun. Let's start with Reading and Writing files.

Working of `open()` function

We use **`open()`** function in Python to open a file in read or write mode. As explained above, `open()` will return a file object. To return a file object we use **`open()`** function along with two arguments, that accepts file name and the mode, whether to read or write. So, the syntax being: **`open(filename, mode)`**. There are three kinds of mode, that Python provides and how files can be opened:

- `"r"`, for reading.
- `"w"`, for writing.
- `"a"`, for appending.
- `"r+"`, for both reading and writing

One must keep in mind that the mode argument is not mandatory. If not passed, then Python will assume it to be `"r"` by default. Let's look at this program and try to analyze how the read mode works:

➤ LABORATORY ACTIVITY

INSTRUCTIONS:

Read and analyze the problems then create a Python program that satisfies the requirement of each problem.

ACTIVITY 3.1 Character Existence

Write a program that accepts a string message then counts the occurrences of all characters within the string including space. Store the counts on a file named "chars.txt"

Expected Output:

chars.txt ✕

```
1 i: 1 time(s)
2 | : 2 time(s)
3 l: 1 time(s)
4 o: 2 time(s)
5 v: 1 time(s)
6 e: 1 time(s)
7 p: 1 time(s)
8 y: 1 time(s)
9 t: 1 time(s)
10 h: 1 time(s)
11 n: 1 time(s)
12
```

Copy the source code directly from the IDE so that colors of the codes are preserved for readability. Snip the output as seen in the sample and paste it below.

Source Code: **ACTIVITY 3.1**

```
print("\n=====")
print(" ACTIVITY 3.1 Character Existence")

print("\n ENTER A STRING: ", end = "")
phrase = input()

# declaring a empty dictionary
chars = {}

for x in phrase:
    # checking index or key if it occurs again then add 1
    if x in chars:
        chars[x] += 1
    # if the key or index we are going to set it by 1 to be included in the dict
    else:
        chars[x] = 1

# Creating or opening chars.txt
file = open("chars.txt", 'w')

# accessing dictionary using for loop
# k represents the key or the character
# v is for the value or occurrences
for k,v in dict.items(chars):
    file.write(f"{k}: {v} times(s)\n")

file.close()
print(f" {phrase} has been added to chars.txt")
print("=====\n")
```

Output: **ACTIVITY 3.1**

TERMINAL OUTPUT

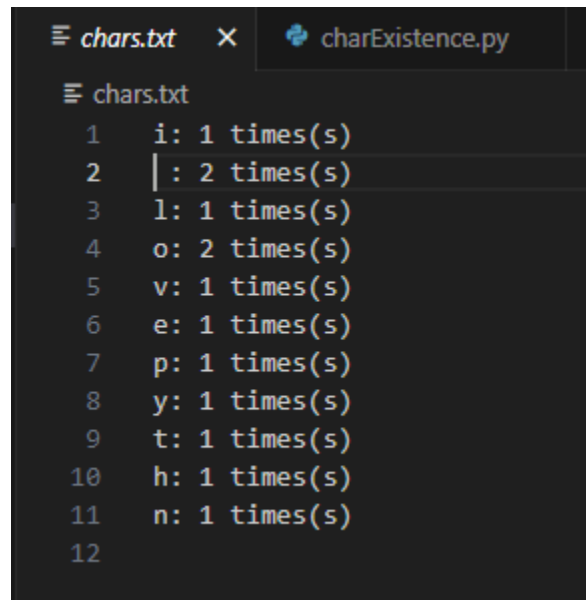
```
PS C:\Users\202210168\Documents\Torno> & "C:/Program Files/Python311/python.exe" c:/Users/202210168/Documents/Torno/charExistence.py

=====
ACTIVITY 3.1 Character Existence

ENTER A STRING: i love python
i love python has been added to chars.txt
=====

PS C:\Users\202210168\Documents\Torno> |
```

GENERATED TEXT FILE



The screenshot shows a text editor with two tabs: `chars.txt` and `charExistence.py`. The `chars.txt` tab is active and displays the following content:

```
1 i: 1 times(s)
2 | : 2 times(s)
3 l: 1 times(s)
4 o: 2 times(s)
5 v: 1 times(s)
6 e: 1 times(s)
7 p: 1 times(s)
8 y: 1 times(s)
9 t: 1 times(s)
10 h: 1 times(s)
11 n: 1 times(s)
12
```

ACTIVITY 3.2 Not my T

Create a text file that contains the text below, then save it as "bgyo.txt." Write a Python program that counts the number of lines from the text file that does not start with the letter "T."

Content of bgyo.txt

```
BGYO is a five-member boy group.  
The group is a two-year old PPop group.  
Akira is the lead vocalist  
JL is the main vocalist.  
ToreMori is the ship name for JL and Aki.  
Nate is the main dancer.  
Mikki is the rapper.  
Gelo is the leader and lead dancer. |
```

Expected Output:

Number of lines not starting with 'T': 6

Copy the source code directly from the IDE so that colors of the codes are preserved for readability. Snip the output as seen in the sample and paste it below.

Source Code: **ACTIVITY 3.2**

```
file = open("bgyo.txt", 'r')

# setting a counter to indicate the lines that doesnt start with T
isnotT = 0

print("\n=====")
print(" ACTIVITY 3.2 NOT MY T")

# accessing the file bgyo.txt
for line in file:
    # checking if the line not starts with T
    if line and not line.startswith("T"):
        isnotT += 1 # increament the counter to one if the line doesnt start with
T

file.close()

# displaying how many line doesnt starts with T
print(f" Number of lines not starting with 'T': {isnotT}")
print("=====\\n")
```

Output:

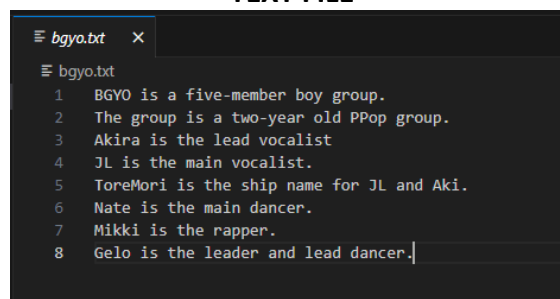
TERMINAL OUTPUT

```
PS C:\Users\202210168\Documents\Torno> & "C:/Program Files/Python311/python.exe" c:/Users/202210168/Documents/Torno/notmyT.py

=====
ACTIVITY 3.2 NOT MY T
Number of lines not starting with 'T': 6
=====

PS C:\Users\202210168\Documents\Torno> █
```

TEXT FILE



```
bgyo.txt
1 BGYO is a five-member boy group.
2 The group is a two-year old PPop group.
3 Akira is the lead vocalist
4 JL is the main vocalist.
5 ToreMori is the ship name for JL and Aki.
6 Nate is the main dancer.
7 Mikki is the rapper.
8 Gelo is the leader and lead dancer.]
```


➤ QUESTION AND ANSWER

Briefly answer the questions below. Avoid erasures. For group activity, specify the name of GROUP MEMBER/s who answered the question. Do not forget to include the source for all NON-ORIGINAL IDEAS.

- | |
|---|
| <ul style="list-style-type: none">• What are the different types of Python strings?<ul style="list-style-type: none">➤ According to copahost there are 2 types of string which are bytes and str where Bytes are sequences of raw binary data and STR are sequences of Unicode and enclosed either on double or single quotation. |
| <ul style="list-style-type: none">• What is the importance of escape character?<ul style="list-style-type: none">➤ Escape character is a special character that can either modify a string add that has a special function which starts a backslash like add a newline using \n. |
| <ul style="list-style-type: none">• What is file handling in Python?<ul style="list-style-type: none">➤ It is a process of creating, removing, modifying, appending a file which are built in functions that can interact with files using python. |

➤ REFERENCES

- Muller, A. C. & Guido, S. (2017). Introduction to Machine Learning with Python. O'Reilly Media, Inc
- Eric Matthes (2016). Python Crash Course. No Starch Press, Inc.
- <http://onefeu.instructure.com>