**PYTHON ACTIVITY 18):**

1)Unix-based operating systems usually include a tool named head. It displays the

first 10 lines of a file whose name is provided as a command line parameter. Write

a Python program that provides the same behavior. Display an appropriate error

message if the file requested by the user does not exist or if the command line

parameter is omitted.

2)Unix-based operating systems typically include a tool named cat, which is short

for concatenate. Its purpose is to concatenate and display one or more files whose

names are provided as command line parameters. The files are displayed in the same

order that they appear on the command line.

Create a Python program that performs this task. It should generate an appropriate

error message for any file that cannot be displayed, and then proceed to the next file.

Display an appropriate error message if your program is started without any command

line parameters.

3)In this exercise you will create a Python program that identifies the longest word(s)

in a file. Your program should output an appropriate message that includes the length

of the longest word, along with all of the words of that length that occurred in the

file. Treat any group of non-white space characters as a word, even if it includes

numbers or punctuation marks.

4)Create a program that sums all of the numbers entered by the user while ignoring

any lines entered by the user that are not valid numbers. Your program should dis-

play the current sum after each number is entered. It should display an appropriate

error message after any invalid input, and then continue to sum any additional num-

bers entered by the user. Your program should exit when the user enters a blank

line. Ensure that your program works correctly for both integers and floating point

numbers.

5)While 80 characters is a common width for a terminal window, some terminals are

narrow or wider. This can present challenges when displaying documents containing

paragraphs of text. The lines might be too long and wrap, making them difficult to

read, or they might be too short and fail to make use of the available space.

Write a program that opens a file and displays it so that each line is filled as full as

possible. If you read a line that is too long then your program should break it up into

words and add words to the current line until it is full. Then your program should

start a new line and display the remaining words. Similarly, if you read a line that is

too short then you will need to use words from the next line of the file to finish filling

the current line of output.

**For example, consider a file containing the following lines**

**from “Alice’s Adventures in Wonderland”:**

Alice was

beginning to get very tired of sitting by her

sister

on the bank, and of having nothing to do: once

or twice she had peeped into the book her sister

was reading, but it had

no

pictures or conversations in it,"and what is

the use of a book," thought Alice, "without

pictures or conversations?"

**When formatted for a line length of 50 characters, it should be displayed as:**

Alice was beginning to get very tired of sitting

by her sister on the bank, and of having nothing

to do: once or twice she had peeped into the book

her sister was reading, but it had no pictures or

conversations in it, "and what is the use of a

book," thought Alice, "without pictures or

conversations?"

Ensure that your program works correctly for files containing multiple paragraphs

of text. You can detect the end of one paragraph and the beginning of the next by

looking for lines that are empty once the end of line marker has been removed. You

may perform error checking if you want to, but it is not required for this exercise.