TASK5 and TASK6

**1.** Write a program in Python to allow the error of syntax to be handled using exception handling. **HINT:** Use SyntaxError

try:

# Prompt the user to enter a Python expression

expression = input("Enter a Python expression: ")

# Evaluate the expression

result = eval(expression)

print("Result:", result)

except SyntaxError:

print("Invalid syntax. Please check your expression.")

**2.** Write a program in Python to allow the user to open a file by using the argv module. If the entered name is incorrect throw an exception and ask them to enter the name again. Make sure to use read only mode.

import sys

while True:

try:

# Prompt the user to enter the file name as a command line argument

filename = sys.argv[1]

# Open the file in read-only mode

with open(filename, 'r') as file:

# Read and print the contents of the file

contents = file.read()

print("File Contents:")

print(contents)

# If no exception occurred, break out of the loop

break

except IndexError:

print("Please provide a file name as a command line argument.")

except FileNotFoundError:

print("File not found. Please enter a valid file name.")

continue

**3.** Write a program to handle an error if the user entered a number more than four digits it should return “The length is too short/long !!! Please provide only four digits”

number = input("Enter a number with four digits: ")

try:

# Check if the length of the input number is exactly four digits

if len(number) != 4:

raise ValueError("The length is too short/long !!! Please provide only four digits")

# Convert the number from string to integer

number = int(number)

print("Number:", number)

except ValueError as e:

print(e)

**4.** Create a login page backend to ask users to enter the username and password. Make sure to ask for a Re-Type Password and if the password is incorrect give chance to enter it again but it should not be more than 3 times.

import sys

while True:

try:

# Get the file name from command line arguments

file\_name = sys.argv[1]

# Open the file in read-only mode

file = open(file\_name, 'r')

# Read and print the contents of the file

contents = file.read()

print(contents)

# Close the file

file.close()

# Exit the loop if the file was successfully opened

break

except IndexError:

print("Please provide a file name as a command line argument.")

except FileNotFoundError:

print("File not found. Please enter a valid file name.")

break

**5.** Go through the link provided below to understand finally and raise concept: https://www.programiz.com/python-programming/exception-handling

**6.** Read doc.txt file using Python File handling concept and return only the even length string from the file. Consider the content of doc.txt as given below:  
Hello I am a file  
Where you need to return the data string

Which is of even length  
Make sure you return the content in The same link as it is present.

**TASK SIX GENERATORS, LIST COMPREHENSION AND DECORATORS**

**1.** Write a program in Python to find out the character in a string which is uppercase using list comprehension.

def find\_uppercase\_chars(string):

uppercase\_chars = [char for char in string if char.isupper()]

return uppercase\_chars

# Test the function

input\_string = "Hello World! OpenAI is Amazing"

uppercase\_chars = find\_uppercase\_chars(input\_string)

print(uppercase\_chars)

**2.** Write a program to construct a dictionary from the two lists containing the names of students and their corresponding subjects. The dictionary should map the students with their respective subjects. Let’s see how to do this using for loops and dictionary comprehension.  
**HINT -** Use Zip function also

**Sample input:** students = ['Smit', 'Jaya', 'Rayyan'] subjects = ['CSE', 'Networking', 'Operating System'] **Expected output:** {‘Smit’ : ’CSE’ , ’Jaya’ : ’Networking’ , ’Rayyan’ : ’Operating System’}

students = ['Smit', 'Jaya', 'Rayyan']

subjects = ['CSE', 'Networking', 'Operating System']

# Method 1: Using for loop

student\_subject\_dict = {}

for student, subject in zip(students, subjects):

student\_subject\_dict[student] = subject

print(student\_subject\_dict)

# Method 2: Using dictionary comprehension

student\_subject\_dict = {student: subject for student, subject in zip(students, subjects)}

print(student\_subject\_dict)

**3.** Learn More about Yield, next and Generators  
**4.** Write a program in Python using generators to reverse the string.

Input String = “Consultadd Training”

def reverse\_string(input\_string):

length = len(input\_string)

for i in range(length - 1, -1, -1):

yield input\_string[i]

input\_string = "Consultadd Training"

reversed\_string = ''.join(reverse\_string(input\_string))

print(reversed\_string)

Output: gniniarT ddatlusnoC

**5.** Write an example on decorators.

def uppercase\_decorator(function):

def wrapper():

result = function()

return result.upper()

return wrapper

def greeting():

return "hello, world!"

greet = uppercase\_decorator(greeting)

print(greet())

Output: HELLO, WORLD!