**Lab Exercise 8**

Q1. Create an abstract class Appliance with two abstract methods:

turn\_on() and turn\_off()

Then, create two subclasses: Fan and Light

Each subclass must implement the two abstract methods and demonstrate their use.

Q2: Create an abstract class Shape with an abstract method area() and a regular method display\_shape(). Implement with Circle and Rectangle class.

Q3. Create a decorator called greet\_decorator that prints "Hello!" before calling the actual function.

Q4: Create a decorator log\_arguments that prints the function arguments before calling it.

Q5. Create a log\_method\_call decorator that logs method name when called from a class.

**[Hint:** func.\_\_name\_\_ may be used to return calling method name**]**

**[Exception Handling]**

Q1: Write a function to compute 5/0 and use try/except to catch the exceptions.

Q2: Write a Python program to access the array element whose index is out of bound and handle the corresponding exception

Q3: Write a program that asks the user for a file name and tries to open the file. If the file doesn't exist, handle the FileNotFoundError. Also, handle the case where the user enters non-numeric data for the file content using a ValueError.

Q4: Write a custom exception InvalidAgeError that will be raised if a user inputs an age less than 0 or greater than 150. Implement exception handling to handle this custom exception.

Q5: Write a Python program that asks the user for two numbers, performs division, and uses a try block to catch ZeroDivisionError. If no exception occurs, the else block should print the result. The finally block should always print "Operation Complete."

Q6: Write a program that attempts to convert a string into an integer using int(). If the input is not a valid integer, handle the ValueError. If the input is an empty string, handle the TypeError.

Q7: Write a function that calls another function which raises an exception. In the outer function, catch that exception and raise a new exception with additional information, preserving the original exception.

Q8: Write a program that handles both IndexError and KeyError using a single except block. The program should simulate accessing an element in a list and a key in a dictionary and handle both exceptions gracefully.

Q9: Write a Python program that reads a file and processes each line as an integer. If the conversion fails due to non-numeric data, use a nested try-except to handle the exception for that line without terminating the program.

Q10: Try to open and write to a file that is not writable:

Q11: Write a python program to raise an error and stop the program if x is lower than 0.

Q12: Write a python program to raise a TypeError if x is not an integer: