Lab Report 5 Date:2081/03/14

Experiment 1: Error handling

Title: Programming to learn about Error Handling.

Objective:The objective of this lab-work is to ensure that program can gracefully handle unexpected situations, preventing crashes and providing a way to manage errors effectively.

Theory: In Python, error handling is managed using exceptions. When an error occurs during the execution of a program, Python raises an exception, which can be caught and handled to prevent the program from crashing. This is done using try,except, else and finally blocks

1. WAP to handle division by zero exception..

*n=int(input("enter a number"))*

*try:*

*x=n/0*

*except:*

*print("DIVISIONBYZERO!")*

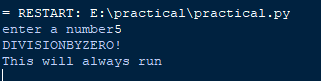
*else:*

*print("no error occured")*

*finally:*

*print("This will always run")*

Output:



1. Create a program to handle multiple exceptions like file not found, index out of range, etc.

*def handleexception():*

*try:*

*with open("Nofile.txt","r")as file:*

*content=file.read()*

*numbers=[1,2,3]*

*print(numbers[5])*

*user\_input="abc"*

*result=int(user\_input)*

*except FileNotFoundError:*

*print("NoFile.txt was not found")*

*except IndexError:*

*print("Error: Index out of range")*

*except ValueError:*

*print("Error: Invalid value entered. Could not convert to an int")*

*else:*

*print("No error occured")*

*finally:*

*print("This always Executes")*

*handleexception()*

Output:



1. Implement a custom exception class in python.

*class MyCustomError(Exception):*

*pass*

*def check\_value(value):*

*if value < 0:*

*raise MyCustomError("Value cannot be negative!")*

*return print(value," is not valid")*

*try:*

*print(check\_value(-5))*

*except MyCustomError as e:*

*print(f"Error: {e}")*

Output:



Conclusion: In the above page we have done the Error Handling programming with its output .