Practical No 14

Aim

Write a Python program that takes two numbers as input and performs division. Implement exception handling to manage division by zero and invalid input errors gracefully.

Apparatus / Software Required

- Python Interpreter (Python 3.14.2)

Theory

In this practical, we develop a Python program that performs division of two numbers entered by the user. While performing arithmetic operations, certain errors may occur during runtime, such as:

- Division by zero
- Entering non-numeric values

These errors can cause the program to crash. To avoid this, Python provides **exception handling**, which allows us to handle such errors gracefully without stopping the program execution.

Exception handling is implemented using:

- `try` block → contains code that may generate an error
- `except` block → handles specific errors
- `finally` block → executes regardless of error occurrence (optional)

By using exception handling, the program becomes more robust, user-friendly, and reliable.

Concepts Required

Before performing this practical, the following concepts should be known:

1. Input and Output Functions

- `input()` for taking user input
- `print()` for displaying output

2. Type Conversion

- Converting input to numeric type using `int()` or `float()`

3. Arithmetic Operators

- Division operator `/`

4. Exception Handling

- `try`, `except`, `finally`

5. Common Exceptions

- `ZeroDivisionError` → when dividing by zero
- `ValueError` → when invalid input is given

Algorithm

1. Start
2. Take first number as input
3. Take second number as input
4. Use `try` block:
    - Convert inputs into numbers
    - Perform division
5. Use `except` blocks:
    - If `ZeroDivisionError` occurs → display "Cannot divide by zero"
    - If `ValueError` occurs → display "Invalid input"
6. Use `finally` block (optional):
    - Display completion message
7. If no error, display the result
8. Stop

Program Code

Output

Conclusion