Lab 12: Exception Handling in Student Result Processing System

Task: Developing a Student Result Processing System with Error Handling

Objective

The objective of this task is to demonstrate the use of Python's exception handling mechanisms to build a reliable student result processing system. Students will learn to handle invalid input, division by zero, logical errors, and ensure program execution completes properly.

Task Description

The program takes student marks as input and calculates results. Since users may enter incorrect data, the program must handle errors properly using Python exception handling. Students will practice using try, except, else, raise, and finally to make the program robust.

Task 1: Handle Invalid User Input

Problem: Users may enter non-numeric values for total marks or obtained marks. The program must handle this gracefully.

In [4]:
```
try:
    total_marks = int(input("Enter total marks: "))
    obtained_marks = int(input("Enter obtained marks: "))
except ValueError:
    print("Error: Please enter numeric values only")
```

```
Error: Please enter numeric values only
```

Task 2: Handle Division by Zero

Problem: If total marks is zero, calculating percentage will cause a runtime error.

In [5]:
```
try:
    percentage = (obtained_marks / total_marks) * 100
    print("Percentage:", percentage)
except ZeroDivisionError:
    print("Error: Total marks cannot be zero")
```

```
Percentage: 0.0
```

Task 3: Validate Logical Conditions Using raise

Problem: Obtained marks must not be negative or greater than total marks.

In [6]:
```
if obtained_marks < 0 or obtained_marks > total_marks:
    raise Exception("Invalid marks: Obtained marks must be between 0 and total
```

ask 4: Execute Code Only When No Error Occurs

Problem: Display percentage and pass/fail status only if no exception occurs

In [9]:
```
try:
    total_marks = int(input("Enter total marks: "))
    obtained_marks = int(input("Enter obtained marks: "))
    if obtained_marks < 0 or obtained_marks > total_marks:
        raise Exception("Invalid marks: Obtained marks must be between 0 and t
except ValueError:
    print("Error: Please enter numeric values only")
except ZeroDivisionError:
    print("Error: Total marks cannot be zero")
except Exception as e:
    print(e)
else:
    percentage = (obtained_marks / total_marks) * 100
    if percentage >= 50:
        print("Result: Pass")
    else:
        print("Result: Fail")
```

```
Result: Pass
```

Task 5: Guarantee Program Completion Using finally

Problem: Display a completion message regardless of errors.

In [14]:
```
try:
    total_marks = int(input("Enter total marks: "))
    obtained_marks = int(input("Enter obtained marks: "))
    percentage = obtained_marks / total_marks * 100
    print("Percentage:", percentage)
except ValueError:
    print("Error: Please enter numeric values only")
except ZeroDivisionError:
    print("Error: Total marks cannot be zero")
finally:
    print("Result processing completed.")
```

```
Percentage: 78.0
Result processing completed.
```