Practical No 7

Aim: To demonstrate exception handling using try, except, finally blocks and handle multiple exceptions.

Theory:

An **exception** is an error that occurs during program execution, which interrupts the normal flow of instructions.

Example:

- Dividing by zero → ZeroDivisionError
- Converting a string to an integer when it's not numeric → ValueError

Keywords in Exception Handling

- **try**
  Block where we write code that may raise an error.
- **except**
  Block that handles the error if it occurs.
  We can have **multiple except blocks** for different exceptions.
- **finally** (optional)
  Always executes, whether there is an exception or not.
  Useful for **cleanup code** like closing files or database connections.

Basic try–except

If something goes wrong in the try block, Python jumps to the except block.

```
try:
   num = int(input("Enter a number: "))
   print("You entered:", num)
except ValueError:
   print("That was not a valid number!")
```

Handling Multiple Exceptions

Add more than one except block for different error types.

```
try:
   a = int(input("Enter first number: "))
   b = int(input("Enter second number: "))
   result = a / b
   print("Result =", result)

except ValueError:
   print("Invalid input! Please enter numbers only.")
```

```
except ZeroDivisionError:
    print("Division by zero is not allowed.")
```

Using finally

The finally block always runs, whether an exception occurs or not. It is useful for cleanup (like closing files, releasing resources, etc.).

```
try:
    num = int(input("Enter a number: "))
    print("Square =", num ** 2)
except ValueError:
    print("Please enter a valid number.")
finally:
    print("Execution completed (this runs no matter what).")
```

Handling Multiple Exceptions in a Single Block

```
try:
    a = int(input("Enter numerator: "))
    b = int(input("Enter denominator: "))
    print("Result =", a / b)
except (ValueError, ZeroDivisionError) as e:
    print("Error occurred:", e)
finally:
    print("Program finished.")
```

Program:

1. Write a program that asks the user for two integers and performs division. Use:

   except blocks to handle ValueError and ZeroDivisionError.

   else block to display the result only when no error occurs.

```
Program:

try:
    a = int(input("Enter number 1: "))
    b = int(input("Enter number 2: "))
    c = a / b

except(ValueError , ZeroDivisionError) as e:
    print("Error Occured: " , e)
else:
    print("Result : " , c)
```

2. Write a program that accepts user input and tries to convert it into an integer. Handle the exception when the input is non-numeric.

Program

```
try:
    number = int(input("Enter a number: "))
    print("You entered the number:", number)
except ValueError:
    print("Invalid input! Please enter a numeric value.")
```

Output:

1

```
PS C:\Users\STUDENT\Pictures> py main.py
Enter number 1: 12
Enter number 2: 6
Result :  2.0
PS C:\Users\STUDENT\Pictures> py main.py
Enter number 1: 12
Enter number 2: 0
Error Occured:  division by zero
PS C:\Users\STUDENT\Pictures> py main.py
Enter number 1: 12
Enter number 2: s
Error Occured:  invalid literal for int() with base 10: 's'
PS C:\Users\STUDENT\Pictures>
```

2

```
PS C:\Users\STUDENT\Pictures> py main.py
Enter a number: 28
You entered the number: 28
PS C:\Users\STUDENT\Pictures> py main.py
Enter a number: Satyam King
Invalid input! Please enter a numeric value.
PS C:\Users\STUDENT\Pictures>
```

Conclusion:

The above code is executed Successfully.