11- EXCEPTION HANDLING

Ex. No. : 11.1 **Date: 02.06.24**

Register No.: 230701385 **Name: S. Vishwak**

.

EXCEPTION HANDLING

To find whether a digit lies in the specified range(1-100). Handling exceptions for invalid inputs and out-of-range numbers .

Input Format:

User inputs a number.

Output Format:

Confirm the input or print an error message if it's invalid or out of range.

For example:

Input	Result
1	Valid input.
101	Error: Number out of allowed range
rec	Error: invalid literal for int()

Program:

```
try:
    n=int(input())
    assert n>0 and n<101
except ValueError:
    print("Error: invalid literal for int()")
except:
    print("Error: Number out of allowed range")
else:
    print("Valid input.")
```

	Input	Expected	Got	
✔	1	Valid input.	Valid input.	✔
✔	100	Valid input.	Valid input.	✔
✔	101	Error: Number out of allowed range	Error: Number out of allowed range	✔

Ex. No. : 11.2 **Date: 02.06.24**

Register No.: 230701385 **Name S. Vishwak**

.

EXCEPTION HANDLING

Write a Python program that performs division and modulo operations on two numbers provided by the user. Handle division by zero and non-numeric inputs.

Input Format:

Two lines of input, each containing a number.

Output Format:

Print the result of division and modulo operation, or an error message if an exception occurs.

For example:

Input	Result
10 2	Division result: 5.0 Modulo result: 0
7 3	Division result: 2.3333333333333335 Modulo result: 1
8 0	Error: Cannot divide or modulo by zero.

Program:

```
try:
    n=int(input())
    m=int(input())
    print("Division result:",n/m)
    print("Modulo result:",n%m)
except ZeroDivisionError:
    print("Error: Cannot divide or modulo by zero.")
except ValueError:
    print("Error: Non-numeric input provided.")
```

	Input	Expected	Got
✔	10 2	Division result: 5.0 Modulo result: 0	Division result: 5.0 Modulo result: 0
✔	7 3	Division result: 2.3333333333333335 Modulo result: 1	Division result: 2.3333333333333335 Modulo result: 1
✔	8 0	Error: Cannot divide or modulo by zero.	Error: Cannot divide or modulo by zero.
✔	abc 5	Error: Non-numeric input provided.	Error: Non-numeric input provided.

Ex. No. : 11.3 **Date: 02.06.24**

Register No.: 230701385 **Name: S. Vishwak**

.

EXCEPTION HANDLING

Write a Python program that asks the user for their age and prints a message based on the age. Ensure that the program handles cases where the input is not a valid integer.

Input Format: A single line input representing the user's age.

Output Format: Print a message based on the age or an error if the input is invalid.

For example:

Input	Result
twenty	Error: Please enter a valid age.
25	You are 25 years old.
-1	Error: Please enter a valid age.

Program:

```
def age(n):
    if(n>-1):
        print("You are",n,"years old.")
try:
    a=int(input())
    age(a)
except ValueError:
    print("Error: Please enter a valid age.")
except EOFError:
        print("Error: Please enter a valid age.")
else:
    if(a<0):
        print("Error: Please enter a valid age.")
```

	Input	Expected	Got	
✔	twenty	Error: Please enter a valid age.	Error: Please enter a valid age.	✔
✔	25	You are 25 years old.	You are 25 years old.	✔
✔	-1	Error: Please enter a valid age.	Error: Please enter a valid age.	✔
✔	150	You are 150 years old.	You are 150 years old.	✔
✔		Error: Please enter a valid age.	Error: Please enter a valid age.	✔

Ex. No. : 11.4 **Date: 02.06.24**

Register No.: 230701385 **Name: S. Vishwak**

.

EXCEPTION HANDLING

Develop a Python program that safely calculates the square root of a number provided by the user. Handle exceptions for negative inputs and non-numeric inputs.

Input Format:

User inputs a number.

Output Format:

Print the square root of the number or an error message if an exception occurs.

For example:

Input	Result
16	The square root of 16.0 is 4.00
-4	Error: Cannot calculate the square root of a negative number.
rec	Error: could not convert string to float

Program:

```
import math
def sqrt(a):
    if(a>-1):
        print("The square root of",a,'is',"%0.2f"%math.sqrt(a),sep=' ')
try:
    a=float(input())
    sqrt(a)
except ValueError:
    print("Error: could not convert string to float")
else:
    if(a<0):
        print("Error: Cannot calculate the square root of a negative number. ")
```

	Input	Expected	Got	
✓	16	The square root of 16.0 is 4.00	The square root of 16.0 is 4.00	✓
✓	0	The square root of 0.0 is 0.00	The square root of 0.0 is 0.00	✓
✓	-4	Error: Cannot calculate the square root of a negative number.	Error: Cannot calculate the square root of a negative number.	✓

Ex. No. : 11.5 **Date: 02.06.24**

Register No.: 230701385 **Name: S. Vishwak**

.

EXCEPTION HANDLING

Develop a Python program that safely performs division between two numbers provided by the user. Handle exceptions like division by zero and non-numeric inputs.

Input Format: Two lines of input, each containing a number.

Output Format: Print the result of the division or an error message if an exception occurs.

For example:

Input	Result
10 2	5.0
10 0	Error: Cannot divide or modulo by zero.
ten 5	Error: Non-numeric input provided.

Program:

```
try:
    n=float(input())
    d=int(input())
    print(n/d)
except ZeroDivisionError:
    print("Error: Cannot divide or modulo by zero.")
except ValueError:
    print("Error: Non-numeric input provided.")
```

	Input	Expected	Got	
✔	10 2	5.0	5.0	✔
✔	10 0	Error: Cannot divide or modulo by zero.	Error: Cannot divide or modulo by zero.	✔
✔	ten 5	Error: Non-numeric input provided.	Error: Non-numeric input provided.	✔