WEEK 11- EXCEPTION HANDLING

Problem Description:

Write a Python script that asks the user to enter a number within a specified range (e.g., 1 to 100). Handle 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())
   if n>=1 and n<=100:
      print("Valid input.")
   else:
      print("Error: Number out of allowed range")
except ValueError:
   print("Error: invalid literal for int()")
```

	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	

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:-

```
try:
    n=int(input())
    if n>=0:
        print("You are",n,"years old.")
    else:
        print("Error: Please enter a valid age.")
except ValueError:
    print("Error: Please enter a valid age.")
except Exception as e:
    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.	

Problem Description:

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:-

```
try:
  n=float(input())
  if n>=0:
    print(f"The square root of {n} is {n**.5:.2f}")
  else:
    print("Error: Cannot calculate the square root of a negative number.")
except ValueError:
  print("Error: could not convert string to float")
```

	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.	

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:-

```
#n=int(input())

#d=int(input())

try:

   n=int(input())

   d=int(input())

   div=n/d

   mod=n%d

   print("Division result:",div)

   print("Modulo result:",mod)

except ZeroDivisionError:

   print("Error: Cannot divide or modulo by zero.")

except ValueError:

   print("Error: Non-numeric input provided.")
```

	Input	Expected	Got	
	102	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.	🟩

Problem Description:

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
25	You are 25 years old.
rec	Error: Please enter a valid age.
-5	Error: Please enter a valid age.

Program:-

```
try:
    n=int(input())
    if n>=0:
        print("You are",n,"years old.")
    else:
        print("Error: Please enter a valid age.")
except ValueError:
    print("Error: Please enter a valid age.")
except Exception as e:
    print("Error: Please enter a valid age.")
```

	Input	Expected	Got	
	25	You are 25 years old.	You are 25 years old.	
	rec	Error: Please enter a valid age.	Error: Please enter a valid age.	
	!@#	Error: Please enter a valid age.	Error: Please enter a valid age.	