

Week 11

Exceptions

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.

```
def safe_divide(a, b):
```

```
    try:
```

```
        result = a / b
```

```

except ZeroDivisionError:
    result = "Error: Cannot divide or modulo by zero."
except TypeError:
    result = "Error: Non-numeric input provided."
return result
try:
    num1 = float(input())
    num2 = float(input())
    print(safe_divide(num1, num2))
except ValueError:
    print("Error: Non-numeric input provided.")

```

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.

Answer:

```

try:
    n = input()
    l = int(n)
    if l < 0:
        print("Error: Please enter a valid age.")

```

```
else:
    print("You are", l, "years old.")
except (ValueError, EOFError):
    print("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

Answer:

```
import math
def calculate_square_root():
    user_input = input()
```

```

try:
    num = float(user_input)
    if num < 0:
        print("Error: Cannot calculate the square root of a negative number.")
    else:
        sqrt = math.sqrt(num)
        print(f"The square root of {num} is {sqrt:.2f}")
except ValueError:
    print("Error: could not convert string to float")

calculate_square_root()

```

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.

Answer:

```
def age_message():
```

```

try:
    age = int(input())
    if age < 0:
        raise ValueError("Invalid age")
    print("You are {} years old.".format(age))
except ValueError:
    print("Error: Please enter a valid age.")
except EOFError:
    print("Error: Please enter a valid age.")

age_message()

```

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.

Answer:

```

try:
    a = int(input())

```

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