Ex. No. : 11.1 **Date:17/05/2024**

Register No.: 231901004 **Name: VAKASH DURAI**

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.

Code:

```
def print_age_message():
    try:
        age =(input())
        if int(age) < 0 and age!='' :
            print("Error: Please enter a valid age.")
        else:
            print(f"You are {age} years old.")
    except ValueError:
        print("Error: Please enter a valid age.")
    except EOFError:
        print("Error: Please enter a valid age.")
print_age_message()
```

Ex. No. : 11.2 **Date:17/05/2024**

Register No.: 231901004 **Name: VAKASH DURAI**

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.

Code:

```
try:
    num1 = input()
    num2 = input()
    num1 = float(num1)
    num2 = float(num2)
    division_result = num1 / num2
    modulo_result = int(num1 % num2)
    print(f"Division result: {division_result}")
    print(f"Modulo result: {modulo_result}")
except ZeroDivisionError:
    print("Error: Cannot divide or modulo by zero.")
except ValueError:
    print("Error: Non-numeric input provided.")
```

Ex. No. : 11.3 Date:17/05/2024

Register No.: 231901004 Name: VAKASH DURAI

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.

Code:

```
'''n=int(input())
try:
    if n>=1 and n<=100:
        print("Valid input.")
    else:
        print("Error: Number out of allowed range")
except TypeError:
    print("Error: invalid literal for int()")'''
def get_number_from_user():
    try:
        user_input = input()
        number = int(user_input)
        if 1 <= number <= 100:
            print("Valid input.")
        else:
            print("Error: Number out of allowed range")
    except ValueError:
        print("Error: invalid literal for int()")
get_number_from_user()
```

Ex. No. : 11.4 **Date:17/05/2024**

Register No.: 231901004 **Name: VAKASH DURAI**

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.

Code:

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

Ex. No. : 11.5 **Date:17/05/2024**

Register No.: 231901004 **Name: VAKASH DURAI**

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.

Code:

```
"""try:
    n=int(input())
    if n<0:
        print("Error: Cannot calculate the square root of a negative number.")
    else:
        print("The square root of",float(n),"is",str(n**.5)+"0")
except ValueError:
    print("Error: could not convert string to float")"""

import math

def calculate_square_root():
    try:
        user_input = input()
        number = float(user_input)
        if number < 0:
            print("Error: Cannot calculate the square root of a negative number.")
        else:
            sqrt_result = math.sqrt(number)
```

```
            print(f"The square root of {number} is {sqrt_result:.2f}")
        except ValueError:
            print("Error: could not convert string to float")
    calculate_square_root()
```