11.EXCEPTION.

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

Register No.: 231901020 **Name: KAVIYA.V**

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.

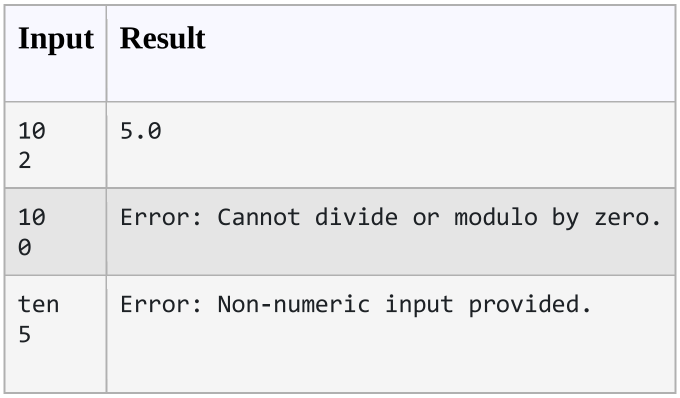
Input Format: Two lines of input, each containing a number.

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

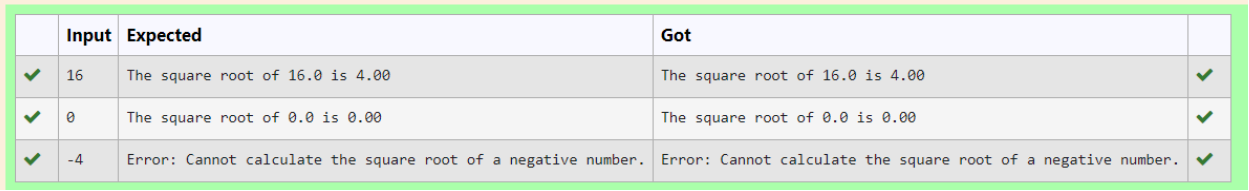
CODING:

```
def safe_division():
    try:
        num1 = float(input())
        num2 = float(input())

        result = num1 / num2

        if num2 == 0:
            raise ZeroDivisionError

        print(result)

    except ValueError:
        print("Error: Non-numeric input provided.")
    except ZeroDivisionError:
        print("Error: Cannot divide or modulo by zero.")

if __name__ == "__main__":
    safe_division()
```

For example:



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



	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. : 11.2 **Date: 02.06.24**

Register No.: 231901020 **Name: KAVIYA.V**

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:

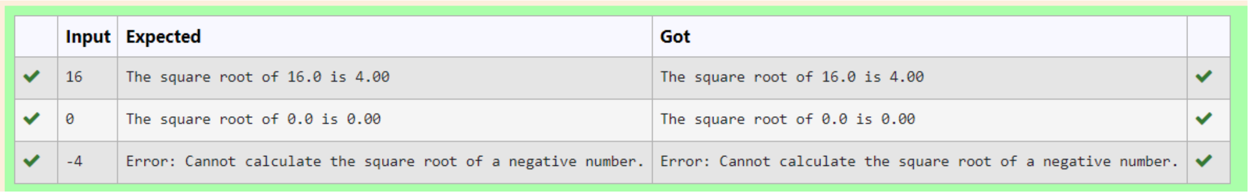
Print the square root of the number or an error message if an exception occurs.

CODING:

```
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()
```

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



	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. :	**11.3**	**Date:**	**02.06.24**
Register No.:	**231901020**	**Name:**	**KAVIYA.V**

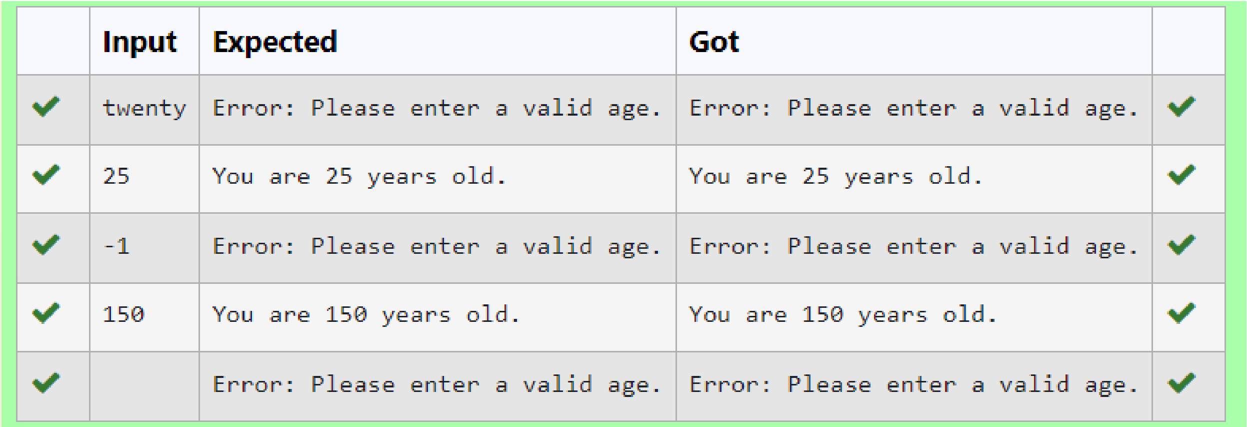
Write a Python program that asks the user for their age and prints a message based on the age. Ensure that the program handles cases where the input is not a valid integer.

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.

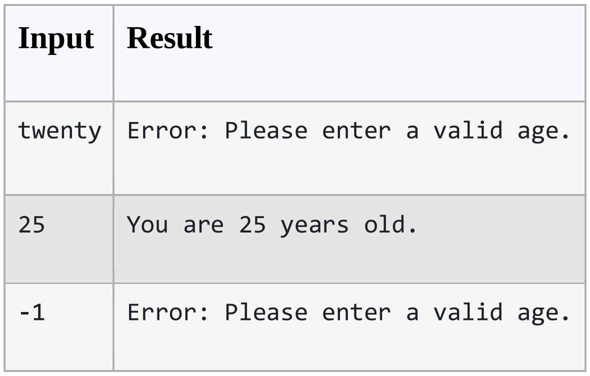
CODING:

```
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()
```



	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.	✓

For example:



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

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

Register No.: 231901020 **Name: KAVIYA.V**

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.

CODING:

```
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()

For example:

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

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

Register No.: 231901020 **Name: KAVIYA.V**

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.

CODING:

```
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()
```

For example:

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