11-Exception Handling

1. 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.

PROGRAM:

import math try:

a=float(input()) if a>=0:

b=a\*\*0.5 c="%.2f"%b

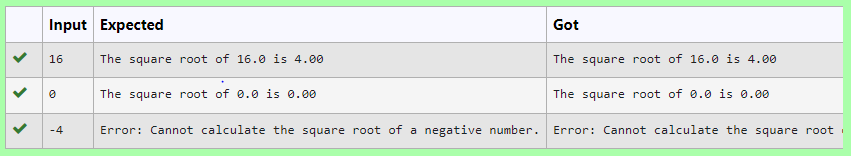
print("The square root of",float(a),"is",c) else:

print("Error: Cannot calculate the square root of a negative number.") except EOFError:

print("Error: could not convert string to float") except ValueError:

print("Error: could not convert string to float")

OUTPUT:



1. 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.

PROGRAM:

try:

a=input() b=input() c=int(a)/int(b) d=int(a)%int(b)

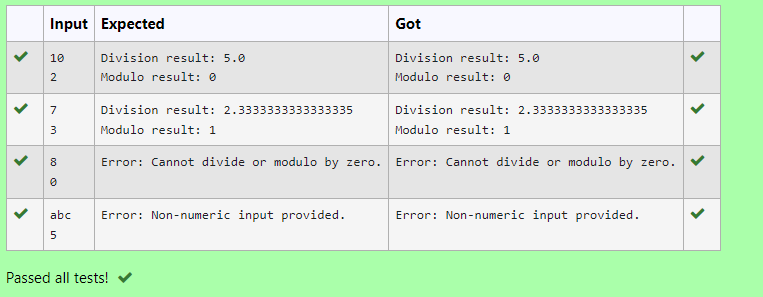
except ZeroDivisionError:

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

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

print("Division result:",c) print("Modulo result:",d)

OUTPUT:



1. 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.

try:

a=input() b=input() c=float(a)/float(b)

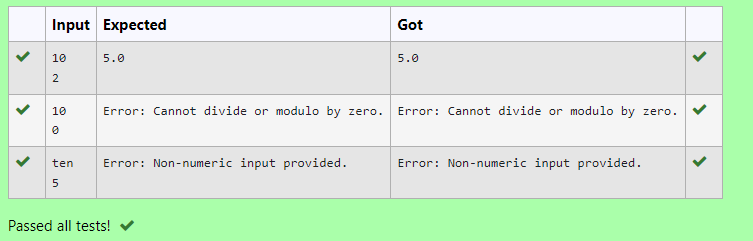
except ZeroDivisionError:

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

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

print(c)

OUTPUT:



1. 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.

try:

n=input()

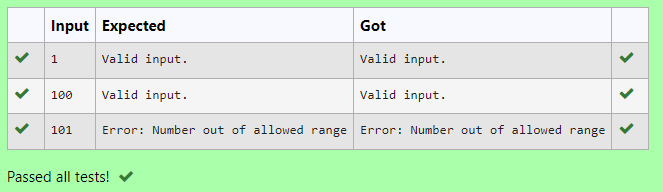
if(int(n)>0 and int(n)<101): print("Valid input.")

else:

print("Error: Number out of allowed range") except:

print("Error: invalid literal for int()")

OUTPUT:



1. 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.

PROGRAM:

try:

a=input()

if int(a)>=0:

print("You are",a,"years old.") else:

print("Error: Please enter a valid age.") except:

print("Error: Please enter a valid age.")

OUTPUT:

