WEEK 11:

.



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.

## For example:



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

try:

a=input() if(len(a)==0):

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

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

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

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

OUTOUT:

.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **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. |  |
| Passed all tests! | | | | | |
| **Correct** | | | | | |

2.

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. |

try:

a=input() if(len(a)==0):

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

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

else:

.



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

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

OUTPUT:



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **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. |  |
| Passed all tests! | | | | | |
| **Correct** | | | | | |

3.

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() |

def main(): min\_range = 1

max\_range = 100

try:

.



num = int(input())

if num < min\_range or num > max\_range:

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

print("Valid input.") except ValueError:

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

if \_\_name == " main ":

# OUTPUT:



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **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 |  |
| Passed all tests! | | | | | |
| **Correct** | | | | | |

Marks for this submission: 1.00/1.00.

# 4.

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 main(): try:

num1 = float(input()) num2 = float(input())

division\_result = num1 / num2 modulo\_result = num1 % num2

print(division\_result)

except ValueError:

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

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

if \_\_name == " main ": main()

# OUTPUT:

.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Input** | **Expected** | **Got** |  |
|  | 10  2 | 5.0 | 5.0 |  |
|  | 10  0 | Error: Cannot divide or modulo by zero. | Error: Cannot divide or modulo by zero. |  |
|  | ten 5 | Error: Non-numeric input provided. | Error: Non-numeric input provided. |  |
| Passed all tests! | | | | |
| **Correct** | | | | |

**5.**

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 |

try:

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

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

print("The square root of",a,"is {:.2f}".format(a\*\*0.5))

except:

.



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

# OUTPUT:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **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. |  |
| Passed all tests! | | | | |
| **Correct** | | | | |