**11 -Exceptions**

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

**Register No.: Name:**

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() |

#Exception

try:

n=int(input())

if n>=1 and n<=100:

print("Valid input.")

else:

raise Exception

except ValueError:

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

except:

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

**Ex. No. : 11.2 Date:**

**Register No.: Name:**

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 |

#Square root exceptions

try:

n=input()

if '.' in n:

n=float(n)

else:

n=int(n)

if n>=0 and '.' not in str(n):

print("The square root of %.1f"%n,"is %.2f"%(n\*\*0.5))

#print("The square root of",n,"is",round((n\*\*0.5),2))

elif '.' in str(n):

print("The square root of",n,"is",round((n\*\*0.5),2))

elif n<0:

raise Exception

except ValueError:

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

except:

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

**Ex. No. : 11.3 Date:**

**Register No.: Name:**

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. |

#Age exception

try:

n=int(input())

if n>=0:

print("You are %d years old."%n)

else:

raise Exception

except:

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

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

**Register No.: Name:**

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. |

#Age exception

try:

n=int(input())

if n>=0:

print("You are %d years old."%n)

else:

raise Exception

except:

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

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

**Register No.: Name:**

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. |

#Division Exception

try:

a=float(input())

b=float(input())

c=a/b

except ValueError:

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

except ZeroDivisionError:

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

else:

print(a/b)