Python Exception Handling

CSC 2310

In this lab you will perform python exception handling. You will use the code in the repository to perform these operations.

Pre-work

- Download the git repository using the following command:

```
% git clone https://gitlab.csc.tntech.edu/csc2310-fa22-
students/<%userid%>/<%userid%>-lab-07-exceptions.git
```

replacing `%userid%` with your own TNTech issued userid.

Problem Description

Concept

In this lab, we will look in to Python's Exception Handling and also look at custom exception handling.

Clone the project

First step is to clone (copy) a project hosted remotely. For this, you can run the command:

```
git clone https://gitlab.csc.tntech.edu/csc2310-fa22-
students/%userid%/%userid%-lab-05-git.git
```

Section 1 -

Once you have checked out the project, you need to fix the code to handle appropriate exceptions for the given input. DO NOT change the code in "__ *main* __".

- Exception handling for calculate_sqrt: Catch TypeErrors inside the calulate_sqrt when the input the not an integer. This should allow the program to continue to the next statement.
- Exception handling in divide_numbers(): Catch the ZeroDivisionError in divide_numbers. Write your own try/catch block to catch division by zero errors. It should print "ZeroDivisionError: cannot divide a number by 0". It will thencontinue to the next statement.
- Exception handling in custom multiplication: In our custom multiplication, we are not allowed to use negative numbers.
  - If the first parameter is negative, print "LogicalError: first parameter cannot be a negative number".

  - If the second parameter is negative, raise a ValueError and return this message: "LogicalError: second parameter cannot be a negative number"
  - If you raise the exception successfully, the code in "__ *main* __" will print "We received an expected logical Error".

- Exception handling in return element: Catch any IndexErrors. If you are trying to access an illegal position, return the last element of the array. If you handle the error successfully, both print statements should print 4. The second statemenet should also print "IndexError: index is out of range, returning the last element 4"
- Exception handling in a dictionary: If a key is not in a dictionary, it should print a KeyNotFoundError
- Custom Exception: For this exercise, modify the custom exception class to raise an error if number1 is less than number2. If you raise it successfully, the code is main should print "Successfully raised a custom exception"

Turn-in

By the end of the exercise, your code should print the following output. If should also exit with an exit code 0.

```

===Exception 1===
16
TypeError: pow() can not be applied to a string
===Exception 2===
2.0
ZeroDivisionError: cannot divide a number by 0
===Exception 3===
10
LogicalError: first parameter cannot be a negative number
-2
We received an expected logical Error
===Exception 4===
4
IndexError: index is out of range, returning the last element 4
4
===Exception 5===
two
KeyNotFoundError
===Exception 6===
Subtracting two values. Result = 2
Successfully raised a custom exception

Process finished with exit code 0
```

Push your code to gitlab. DO NOT submit it to iLearn. The lab is due one week after the beginning of the assignment.

This laboratory is worth 20 points.

Rubric

1. Completeness (3 pts):

- all exceptions are updated (3pts)

2. Correctness (14 pts):

- All exceptions are properly handled - (12pts)
- The code exits with an exit code 0 - (2pts)

3. Submission (3 pts):

- Code submitted using the specific standard (filename correct). Git submission is expected.