Error handling in Python

In Python, there are (at least) two distinguishable kinds of errors: syntax errors and exceptions. Syntax errors, also known as parsing errors, are errors in the programming syntax. In the following example, the quotation mark is missing at the end of the string, hello world. This is a syntax error.

Example 1

```
>>> print("Hello World)

SyntaxError: EOL while scanning string literal
>>>
```

Example 2

Even if a statement or expression is syntactically correct, it may cause an error when an attempt is made to execute it. Errors detected during execution are called exceptions. In the following example, division by zero is an exception.

```
>>> 10 * (1/0)Traceback (most recent call last):
File "<pyshell#0>", line 1, in <module>
10 * (1/0)
ZeroDivisionError: division by zero
>>>
```

Error handling in Python is done through the use of exceptions that are caught in try blocks and handled in except blocks.

Let us look an example of how this is used. The following code not only accepts a user input and adds a new record but also displays a message if the operation was successful or not.

```
import sqlite3
MySchool=sqlite3.connect('schooltest.db')
```

```
mysid= int(input("Enter ID: "))
myname=input("Enter name: ")
myhouse=int(input("Enter house: "))
mymarks=float(input("Enter marks: "))
    #try block to catch exceptionstry:
    curschool=MySchool.cursor()
    curschool.execute("INSERT INTO student (StudentID, name, house, marks) VALUES (?,?,?,?)", (mysid, myname, myhouse, mymarks))
    MySchool.commit()
    print ("One record added successfully.")
#except block to handle exceptions    except:
    print ("Error in operation.")
    MySchool.rollback()

MySchool.close()
```

The connection class defines the commit() and rollback() methods. Changes in database are finalised only if the execute() method runs successfully by commit() method. Otherwise, any changes are undone by the rollback() method. You can try this yourself by saving this code as a .py file and executing it.

Try and Except

The try statement works as follows.

- First, the try clause (the statement(s) between the try and except keywords) is executed.
- If no exception occurs, the except clause is skipped and execution of the try statement is finished
- If an exception occurs during execution of the try clause, the rest of the clause is skipped. Then the except clause is executed, and then execution continues after the try statement.
- If an exception occurs which does not match the exception named in the except clause, it is passed on to outer try statements; if no handler is found, it is an unhandled exception and execution stops with a message.

Further Reading: https://docs.python.org/3/tutorial/errors.html