Exception

When an error occurs within a method, the method creates an object and hands it off to the runtime system. The object, called an *exception object*, contains information about the error, including its type and the state of the program when the error occurred. Creating an exception object and handing it to the runtime system is called *throwing an exception,which avoids your program to crash.*

Use of Exception

Exceptions are convenient in many ways for handling errors and special conditions in a program. When you think that you have a code which can produce an error then you can use exception handling.

Raising an Exception

You can raise an exception in your own program by using the raise exception statement.

Raising an exception breaks current code execution and returns the exception back until it is handled.

Set up exception handling blocks

To use exception handling in Python, you first need to have a catch-all except clause. The words "try" and "except" are Python keywords and are used to catch exceptions.

Sytax

```
try:

   some statements here

except:

   exception handling
```

- ➢ The code within the try clause will be executed statement by statement.

- ➢ If an exception occurs, the rest of the try block will be skipped and the except clause will be executed.

Example 1

```
try:

   print (1/0)

except ZeroDivisionError:

   print ("You can't divide by zero")
```

Try ... except ... else clause

The else clause in a try , except statement must follow all except clauses, and is useful for code that must be executed if the try clause does not raise an exception.

Syntax

```
try:

    data = something_that_can_go_wrong

except IOError:
    handle_the_exception_error

else:
    doing_different_exception_handling
```

Example

```
try:

            result = 10 / 5

except ZeroDivisionError:

            print("division by zero!")

else:

            print("result is", result)
```

Try ... finally clause

If an error is encountered, a try block code execution is stopped and transferred down to the except block.In addition to using an except block after the try block, you can also use the finally block.The code in the finally block will be executed regardless of whether an exception occurs.The finally clause is optional only.

Example 1

```
def divide(x, y):

            try:

                     result = x / y

            except ZeroDivisionError:

             print("division by zero!")

            else:

             print("result is", result)

            finally:

             print("executing finally clause")
```

```
divide(4,0)
```

Common exceptions errors in Python

IOError

If the file cannot be opened.

Example

```
try:
   fh = open("non_existing_file")
except IOError:
   print ("The file does not exist")
```

ImportError
If python cannot find the module

Example

```
try:
         import randm
except ImportError:
   print ("module not exist")
```

ZeroDivisionError

Raised when the second argument of a division or modulo operation is zero.

Example

```
try:

         result = 10 / 0

except ZeroDivisionError:

         print("division by zero!")
```

Ouput

```
division by zero!
```

TypeError

Raised when an operation or function is applied to an object of inappropriate type.

Example

```
try:

         print ('a'+1)
```

```
except TypeError:
    print ("cannot concatenate 'str' and 'int' objects")
```

Output:

```
cannot concatenate 'str' and 'int' objects
```

KeyError

Raised when a mapping (dictionary) key is not found in the set of existing keys.

Example

```
try:
    mydict={1:'a',2:'b',4:'d'}
    print (mydict[11])
except KeyError:
    print ("Key not found")
```

Ouput

```
Key not found
```

IndexError

Raised when a sequence subscript is out of range.

Example

```
try:
    mylist=[23,44,2,3,1]
    for i in range(0,10):
        print (mylist[i])
except IndexError:
    print (" Index out of range")
```

Ouput

```
23
44
```

```
2

3

1

 Index out of range
```

Exercise

1. Division of two numbers using exception handling?

2. Write a program to print factorial values of each element in the list?Use Exceptions if the list contains other than integer values?