**1)What is an Exception in python? Write the difference between Exceptions and Syntax errors.**

Ans- An exception is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions. In general, when a Python script encounters a situation that it cannot cope with, it raises an exception. An exception is a Python object that represents an error.

An error is an issue in a program that prevents the program from completing its task. In comparison, an exception is a condition that interrupts the normal flow of the program. Both errors and exceptions are a type of runtime error, which means they occur during the execution of a program.

**2) What happens when an exception is not handled? Explain with an example.**

Ans- If an exception is not caught (with a catch block), the runtime system will abort the program (i.e. crash) and an exception message will print to the console.

try:

fh = open("testfile", "r")

fh.write("This is my test file for exception handling!!")

except IOError:

print "Error: can\'t find file or read data"

else:

print "Written content in the file successfully"

**3) Which Python statements are used to catch and handle exceptions? Explain with an example.**

Ans- The try and except Block: Handling Exceptions. The try and except block in Python is used to catch and handle exceptions. Python executes code following the try statement as a “normal” part of the program. The code that follows the except statement is the program's response to any exceptions in the preceding try clause

def fun(a):

    if a < 4:

        # throws ZeroDivisionError for a = 3

        b = a/(a-3)

    # throws NameError if a >= 4

    print("Value of b = ", b)

try:

    fun(3)

    fun(5)

# note that braces () are necessary here for

# multiple exceptions

except ZeroDivisionError:

    print("ZeroDivisionError Occurred and Handled")

except NameError:

    print("NameError Occurred and Handled")

**4)Explain with an example:**

**a. try and else**

**b. finally**

**c. raise**

Ans- In the try clause, all statements are executed until an exception is encountered. except is used to catch and handle the exception(s) that are encountered in the try clause. else lets you code sections that should run only when no exceptions are encountered in the try clause.

It defines a block of code to run when the try... except...else block is final. The finally block will be executed no matter if the try block raises an error or not. This can be useful to close objects and clean up resources.

The raise keyword is used to raise an exception. You can define what kind of error to raise, and the text to print to the user.

**5) What are Custom Exceptions in python? Why do we need Custom Exceptions? Explain with an example.**

Ans-

Custom exceptions provide you the flexibility to add attributes and methods that are not part of a standard Python exception. These can store additional information, like an application-specific error code, or provide utility methods that can be used to handle or present the exception to a user.

# define Python user-defined exceptions

class InvalidAgeException(Exception):

"Raised when the input value is less than 18"

pass

# you need to guess this number

number = 18

try:

input\_num = int(input("Enter a number: "))

if input\_num < number:

raise InvalidAgeException

else:

print("Eligible to Vote")

except InvalidAgeException:

print("Exception occurred: Invalid Age")

**6) Create a custom exception class. Use this class to handle an exception.**

Ans-

class FahrenheitError(Exception):

min\_f = 32

max\_f = 212

def \_\_init\_\_(self, f, \*args):

super().\_\_init\_\_(args)

self.f = f

def \_\_str\_\_(self):

return f'The {self.f} is not in a valid range {self.min\_f, self.max\_f}'

def fahrenheit\_to\_celsius(f: float) -> float:

if f < FahrenheitError.min\_f or f > FahrenheitError.max\_f:

raise FahrenheitError(f)

return (f - 32) \* 5 / 9

if \_\_name\_\_ == '\_\_main\_\_':

f = input('Enter a temperature in Fahrenheit:')

try:

f = float(f)

except ValueError as ex:

print(ex)

else:

try:

c = fahrenheit\_to\_celsius(float(f))

except FahrenheitError as ex:

print(ex)

else:

print(f'{f} Fahrenheit = {c:.4f} Celsius')