Q1. What is Exception in python ? write difference between Exception and syntax error.

An exception in Python is an incident that happens while executing a program that causes the regular course of the program's commands to be disrupted. When a Python code comes across a condition it can't handle, it raises an exception. An object in Python that describes an error is called an exception.

An Error might indicate critical problems that a reasonable application should not try to catch, while an Exception might indicate conditions that an application should try to catch. Errors are a form of an unchecked exception and are irrecoverable like an OutOfMemoryError , which a programmer should not try to handle.

Q2. What happens when exception is not handled ? explain with example

When an exception occurred, if you don’t handle it, the program terminates abruptly and the code past the line that caused the exception will not get executed.

In [2]:

```
try:
    print('not handle')
    print(15/0)
except:
    j
```

```
not handle

---------------------------------------------------------------------------
-
ZeroDivisionError                         Traceback (most recent call las
t)
~\AppData\Local\Temp\ipykernel_11724\2479166901.py in <module>
      2     print('not handle')
----> 3     print(15/0)
      4 except:

ZeroDivisionError: division by zero

During handling of the above exception, another exception occurred:

NameError                                 Traceback (most recent call las
t)
~\AppData\Local\Temp\ipykernel_11724\2479166901.py in <module>
      3     print(15/0)
      4 except:
----> 5     j

NameError: name 'j' is not defined
```

Q3. Which python statement are used to catch and handle exceptions? write with an example.

try and except statement are used to handle exceptions.

In [5]:

```
try:
    print(2/0)
except Exception as e:
    print("error is ",e)
```

```
error is  division by zero
```

Q4. explain with example:

a. try and else

b. finally

c. raise

try :This block will test the excepted error to occur

else : If there is no exception then this block will be executed

finally :Finally block always gets executed either exception is generated or not

raise : 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.

In [10]:

```
try:
    print(9/3)
except Exception as e:
    print("code wrong so getting error",e)
else:
    print("condition is right")
finally:
    print("always execute")
```

```
3.0
condition is right
always execute
```

Q5. What are custom exception in python? why do we need custom exceptions ? explain with an example.

we can define custom exceptions by creating a new class that is derived from the built-in Exception class.Built-in exceptions offer information about Python-related problems, and custom exceptions will add information about project-related problems. That way, you can design your code (and traceback, if an exception in a way that combines Python code with the language of the project.

In [3]:

```
try:
    num = int(input('Enter a Number between 1 to 100 : '))
    if num > 100:
        raise Exception('Greater than 100')
    elif num < 1:
        raise Exception('More than 100')
except Exception as e:
    print(e)
```

```
Enter a Number between 1 to 100 : 189
Greater than 100
```

Q6.create a custom exception class .use this class to handle exception .

In [4]:

```
class age_validate(Exception):
    def __init__(self,msg):
        self.msg = msg

def validate_age(age):
    if age < 0:
        raise age_validate('Less than zero')
    elif age > 100:
        raise age_validate('More the 100')
try:
    age = int(input('Enter your Age::- '))
    validate_age(age)
except age_validate as e:
    print(e)
```

```
Enter your Age::- -52
Less than zero
```

In [ ]: