12th Feb Assignment

February 21, 2023

1 Assignment 11

Q1. What is an Exception in python? Write the difference etween Exceptions and syntax error?

Ans.The exception is an error that occur during the program execution

Difference between exception and syntax error

- Syntax error caused by using invalid syntax of program while exception is error during program execution
- Syntax error are not handled while the exception are handled using exception handling

Q2. What happened when an exception is not handled? Explain with an example.

Ans.If the exception is not handled by an except clause, the exception is re-raised after the finally clause has been executed.

[2]:
```
#example
try:
    raise KeyboardInterrupt
finally:
    print('Hello,Bye')
```

```
Hello,Bye
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[2], line 3
      1 #example
      2 try:
----> 3     raise KeyboardInterrupt
      4 finally:
      5     print('Hello,Bye')

KeyboardInterrupt:
```

Q3. Which Python statements are used to catch and handle exceptions? Explain with an example.

Ans. The try block in python is used to catch the exceptions in python while the except block is used to handle the exception in python.

[3]:
```
##example
try:
    10/0
except ZeroDivisionError as e:
    print('The error is:',e)
```

```
The error is: division by zero
```

Q4. Explain with an example:

- try and else
- finally
- raise

Ans.

- Try and else: Try block is used to handle the exceptions and else block execute when try executes without exception
- Finally: This block always executed
- raise: This is used to raise an exception

[6]:
```
#example of try and else
def add(x,y):
    try:
        result=x+y
    except Exceptions as e:
        print(e)
    else:
        print('The result is',result)
```

[7]:
```
add(2,3)
```

```
The result is 5
```

[8]:
```
#example of finally
def add(x,y):
    try:
        result=x+y
    except Exceptions as e:
        print(e)
    else:
        print('The result is',result)
    finally:
```

```
        print('The function is completed here')
```

[9]:
```
add(3,4)
```

```
The result is 7
The function is completed here
```

[11]:
```
# example of raise
x=int(input('Enter x'))

if x<0:
    raise Exception('The value should not negative')
```

```
Enter x -1
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
Cell In[11], line 5
      2 x=int(input('Enter x'))
      4 if x<0:
----> 5     raise Exception('The value should not negative')

Exception: The value should not negative
```

Q5. What are Custom Exceptions in python? Why do we need Custom Exceptions? Explain with an example.

The exceptions which are exceptions for user not for system is custom exception.

[12]:
```
#example
class ValidateAge(Exception):
    def _init_(self,message):
        self.message=message

def validate_age(age):
    if age<0:
        print('Age should not negative')
    elif age==0:
        print('Age should not be zero')
    else:
        print('Age is valid')
```

[18]:
```
#custom exception handling
try:
    age=int(input('Enter an age'))
    validate_age(age)
except ValidateAge as v:
```

```
    print(v)
```

```
Enter an age -3

Age should not negative
```

Q6.Create custom exception class. Use this class to handle the exception.

[20]:
```
class SalaryNotInRangeError(Exception):
    def _init_(self,message):
        self.message=message

def validate_salary(salary):
    if salary<5000:
        print('salary not in range')
    else:
        print('Salary in range')
```

[21]:
```
try:
    salary=int(input('Enter the salary'))
    validate_salary(salary)
except SalaryNotInRangeError as s:
    print(s)
```

```
Enter the salary 3000

salary not in range
```