Modual-3

Q1. What is File function in python? What is keywords to create and write file.

1. **open()**: This function is used to open a file.
    - **Syntax**: file = open('filename', mode)
        - filename: The name of the file.
        - mode: The mode in which to open the file (e.g., read, write, append).

File Modes:

- 'r': Open a file for reading (default).
- 'w': Open a file for writing (creates a new file if it doesn't exist or truncates the file if it exists).
- 'a': Open a file for appending (adds content to the end of the file without truncating it).
- 'x': Create a new file and open it for writing (fails if the file exists)

Q.13What is Exception Handling in Python?

Ans-> Exception handling in Python is a process of resolving errors that occur in a program.This involves catching exceptions,

understanding what caused them, and then responding accordingly. Exceptions are errors that occur at runtime when the

program is being executed.

Q.14 How many except statements can a try-except block have? Name Some built-in exception classes:

Ans->

You can have multiple try blocks in a program but only one except

statements with each try block.

Ans-> Exception

base exception

indexerror

keyerror

Q.15 When will the else part of try-except-else be executed?

Ans-> if the code inside the try block doesn't raise an exception the else block executes.

Q.16 Can one block of except statements handle multiple exception?

Ans-> Yes, you can handle multiple execution in a single except block in python.

```
try:
    result=10/0
except(zerodivisionError, ValueError) as e:
    print("An error occured:",e)
```

Q.17 When is the finally block executed?

Ans->

A finally block always executes, regardless of wheather an exception is thrown.

ex.

```
try:
# you code that might eaise exceptions
except Some exception:
finally
code to execute regardless of exceptions
```

Q.18 What happens when „1“== 1 is executed?

Ans->

When the expression "1" == 1 is executed in Python, it checks whether the string "1" is equal to the integer

1. The result of this comparison is False, because the types are different: one is a string, and the other is an integer.

In Python, equality (==) compares both the value and the type.

Q21. What are oops concepts? Is multiple inheritance supported in java

Ans->

class

object

inheritance

- single
- multiple
- multilevel

encapsulation

polymorphism

  mrthod overloading

  method overriding

access specifire

  public

  private

Q.22How to Define a Class in Python? What Is Self? Give An Example Of

A Python Class

Ans->

To define a class, use the class keyword followed by the class name and a colon.

self allows you to access and modify attributes and call methods within that instance.

EX.

```
class test:
    a=40
    b=50

    def data(self):
        print(f'Sum of A + B: {self.a + self.b}')

result=test()

result.data()

#Output: Sum of A + B: 90
```

Q.25 Explain Inheritance in Python with an example? What is init? Or What

Is A Constructor In Python?

Ans->

There are three typea of inheritance:

1. Single inheritance

2. Multiple Inheritance

3. Multilevel Inheritance

1. Single Inheritance

```
class test:
  sum=0
  a=10
  b=10

  def getdata(self):
    self.sum=self.a+self.b

class result(test):

  def printdata(self):
    print(f'Sum of A+B: {self.sum}')

rst=result()
rst.getdata()
rst.printdata()
```

output= Sum of A+B: 20

2. Multiple Inheritance

```
class sub_1:
  a=10
  b=20
  def sumdata(self):
    print(f'Sum Of {self.a} + {self.b}: {self.a+self.b}')

class sub_2:
  c=50
  d=10
  def minusdata(self):
    print(f'Substraction Of {self.c} - {self.d}: {self.c-self.d}')

class result(sub_1,sub_2):

  def printdata(self):
    self.sumdata()
    self.minusdata()

rst=result()
rst.printdata()
```

#OUTPUT:

 # Sum Of 10 + 20: 30

 # Sum Of 50 - 10: 40

3. Multilevel Inheritnce

```
class sub_1:
  a=10
  b=20
  def sumdata(self):
    print(f'Sum Of {self.a} + {self.b}: {self.a+self.b}')

class sub_2(sub_1):
  def multidata(self):
    print(f'Multiplication Of {self.a} * {self.b}: {self.a*self.b}')

class sub_3(sub_2):

  def divisiondata(self):
    print(f'Division Of {self.a} / {self.b}: {self.a/self.b}')

s3=sub_3()
s3.sumdata()
```

s3.multidata()

s3.divisiondata()

#OUTPUT:

 # Sum Of 10 + 20: 30

 # Multiplication Of 10 * 20: 200

 # Division Of 10 / 20: 0.5

Init is a constructor method in python. when an object of class is created the init method is execute itself. EX.

```
class test:
    def __init__(self,name) -> None:
        print(f'My Name is {name}')

ts=test('keval')
```

#output: My Name is keval

Q .26 What is Instantiation in terms of OOP terminology?

Ans->

In object-oriented programming (OOP), instantiation refers to the process of creating an instance of a class. When you instantiate a class, you create an actual object in memory based on the class blueprint. This object (or instance) can then be used to access the class's attributes (variables) and methods (functions).

Q27. What is used to check whether an object o is an instance of class A?

ANSWER:

python has isinstance() method to check whether an object is instance or not.

```
class A:
    pass

obj = A()

*Check if 'obj' is an instance of class 'A'
if isinstance(obj, A):
    print("obj is an instance of class A")
else:
    print("obj is not an instance of class A")
```

Q28. What relationship is appropriate for Course and Faculty?

ANSWER:

Relationship between Course and Faculty is hierarchical Inheritance.

```
class Faculty:
    name='keval'
    sub_1='Python dev'
    sub_2='Java dev'
    sub_3='Php dev'
    def teacher(self):
        print('My Name Is: ',self.name)

class Subject_1(Faculty):
    def data_1(self):
        print(f'I am a {self.sub_1} ')

class Subject_2(Faculty):
    def data_2(self):
```

```
        print(f'I am a {self.sub_2}')

class Subject_3(Faculty):
    def data_3(self):
        print(f'I am a {self.sub_3} ')

ft=Faculty()
ft.teacher()
s1=Subject_1()
s1.data_1()
s2=Subject_2()
s2.data_2()
s3=Subject_3()
s3.data_3()
```

OUTPUT:

```
My Name Is:  deval
I am a Python dev
I am a Java dev
#I am a Php dev
```

Q29 What relationship is appropriate for Student and Person?

ANSWER:

Relationship between Student and Person is Multiple Inheritance.

```
class Student:
    def data_1(self):
        print('A am Student and i am learning Python.')

class Teacher:
    def data_2(self):
        print('A am Teacher and i teach IOS at Tops.')

class Person(Student,Teacher):
    def data_3(self):
        print('i am a keval')

ps=Person()
ps.data_1()
```

ps.data_2()

ps.data_3()

 # OUTPUT:

 # A am Student and i am learning Python.

 # A am Teacher and i teach IOS at Tops.

 # i am a keval