**Module – 4 (Advance python programming)**

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

Ans: Python **file object provides methods and attributes to access and manipulate files**. Using file objects, we can read or write any files.

• Write a Python program to read an entire text file.

Ans:

import os

d=open(r'Z:\Python\Module-4\file operation.txt','r')

print(d.read())

d.close()

• Write a Python program to append text to a file and display the text.

Ans:

text="\nappended text"

d=open('file operation.txt','a')

d.write(text)

d.close()

• Write a Python program to read first n lines of a file.

Ans:

#readlines

with open(r'file operation.txt','r') as d:

#for reading N lines

for i in range(3):

print(d.readline())

d.close()

• Write a Python program to read last n lines of a file.

#readlines

with open(r'file operation.txt','r') as d:

#for reading N lines

for i in range(3):

print(d.readline())

d.close()  
• Write a Python program to read a file line by line and store it into a list.

Ans:

with open("file operation.txt") as f:

content\_list = f.readlines()

# print the list

print(content\_list)

# remove new line characters

content\_list = [x.strip() for x in content\_list]

print(content\_list)

• Write a Python program to read a file line by line store it into a variable.

Ans:

#As variable

def file\_read(fname):

with open (fname, "r") as myfile:

data=myfile.readlines()

print(data)

file\_read('file operation.txt')

• Write a python program to find the longest words.

Ans:

#find longest word

def longest\_word(filename):

with open(filename, 'r') as infile:

words = infile.read().split()

max\_len = len(max(words, key=len))

return [word for word in words if len(word) == max\_len]

print(longest\_word('file operation.txt'))  
• Write a Python program to count the number of lines in a text file.  
Ans:

#Count lines

with open(r"file operation.txt", 'r') as fp:

lines = len(fp.readlines())

print('Total Number of lines:', lines)

• Write a Python program to count the frequency of words in a file.

Ans:

from collections import Counter

def word\_count(fname):

with open(fname) as f:

return Counter(f.read().split())

print("Number of words in the file :",word\_count("test.txt"))

• Write a Python program to write a list to a file.

Ans:

#write a list to a file

items = ['Mango', 'Orange', 'Apple', 'Lemon']

file = open('file operation.txt','w')

for item in items:

file.write(item+"\n")

file.close()  
• Write a Python program to copy the contents of a file to another file.

Ans:

#Copy content from second file

with open('file operation.txt','r') as firstfile, open('copy.txt','a') as secondfile:

# read content from first file

for line in firstfile:

# append content to second file

secondfile.write(line)  
• Explain Exception handling? What is an Error in Python?

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.When a Python script raises an exception, it must either handle the exception immediately otherwise it terminates and quits.

If you have some *suspicious* code that may raise an exception, you can defend your program by placing the suspicious code in a try: block. After the try: block, include an except: statement, followed by a block of code which handles the problem as elegantly as possible.

Error is A syntax error occurs in Python when the interpreter is unable to parse the code due to the code violating Python language rules, such as inappropriate indentation, erroneous keyword usage, or incorrect operator use.

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

Ans:As many except statement as you want for one try block.

Built-in Exceptions

The table below shows built-in exceptions that are usually raised in Python:

|  |  |
| --- | --- |
| Exception | Description |
| ArithmeticError | Raised when an error occurs in numeric calculations |
| AssertionError | Raised when an assert statement fails |
| AttributeError | Raised when attribute reference or assignment fails |
| Exception | Base class for all exceptions |
| EOFError | Raised when the input() method hits an "end of file" condition (EOF) |
| FloatingPointError | Raised when a floating point calculation fails |
| GeneratorExit | Raised when a generator is closed (with the close() method) |
| ImportError | Raised when an imported module does not exist |
| IndentationError | Raised when indentation is not correct |
| IndexError | Raised when an index of a sequence does not exist |
| KeyError | Raised when a key does not exist in a dictionary |
| KeyboardInterrupt | Raised when the user presses Ctrl+c, Ctrl+z or Delete |
| LookupError | Raised when errors raised cant be found |
| MemoryError | Raised when a program runs out of memory |
| NameError | Raised when a variable does not exist |
| NotImplementedError | Raised when an abstract method requires an inherited class to override the method |
| OSError | Raised when a system related operation causes an error |
| OverflowError | Raised when the result of a numeric calculation is too large |
| ReferenceError | Raised when a weak reference object does not exist |
| RuntimeError | Raised when an error occurs that do not belong to any specific exceptions |
| StopIteration | Raised when the next() method of an iterator has no further values |
| SyntaxError | Raised when a syntax error occurs |
| TabError | Raised when indentation consists of tabs or spaces |
| SystemError | Raised when a system error occurs |
| SystemExit | Raised when the sys.exit() function is called |
| TypeError | Raised when two different types are combined |
| UnboundLocalError | Raised when a local variable is referenced before assignment |
| UnicodeError | Raised when a unicode problem occurs |
| UnicodeEncodeError | Raised when a unicode encoding problem occurs |
| UnicodeDecodeError | Raised when a unicode decoding problem occurs |
| UnicodeTranslateError | Raised when a unicode translation problem occurs |
| ValueError | Raised when there is a wrong value in a specified data type |
| ZeroDivisionError | Raised when the second operator in a division is zero |

• When will the else part of try-except-else be executed?

Ans: The else part is executed when no exception occurs.

• Can one block of except statements handle multiple exception?

Ans:Yes

• When is the finally block executed?

Ans: Every time try block executed.

• What happens when ‘1’== 1 is executed?

Ans:it simply evaluates to false and does not raise any exception.

• How Do You Handle Exceptions With Try/Except/Finally In Python?  
Explain with coding snippets.

Ans:

try:

numerator = 10

denominator = 0

result = numerator/denominator

print(result)

except:

print("Error: Denominator cannot be 0.")

finally:

print("This is finally block.")enter only odd numbers, else will  
raise an exception.

• What are oops concepts? Is multiple inheritance supported in java

Ans1; n Python, object-oriented Programming (OOPs) is a programming paradigm that uses objects and classes in programming. It aims to implement real-world entities like inheritance, polymorphisms, encapsulation, etc. in the programming. The main concept of OOPs is to bind the data and the functions that work on that together as a single unit so that no other part of the code can access this data.

**Main Concepts of Object-Oriented Programming (OOPs)**

* Class
* Objects
* Polymorphism
* Encapsulation
* Inheritance
* Data Abstraction

NO MULTIPLE INHERITANCE NOT SUPPORTED IN JAVA.

• How to Define a Class in Python? What Is Self? Give An Example Of  
A Python Class

Ans: To create a class, use the keyword class.

**SELF represents the instance of class**. This handy keyword allows you to access variables, attributes, and methods of a defined class in Python. The self parameter doesn't have to be named “self,” as you can call it by any other name.

# define a class

class Bike:

name = ""

gear = 0

# create object of class

bike1 = Bike()

# access attributes and assign new values

bike1.gear = 11

bike1.name = "Mountain Bike"

print(f"Name: {bike1.name}, Gears: {bike1.gear} ")

• Write a Python class named Rectangle constructed by a length and  
width and a method which will compute the area of a rectangle.

Ans:

class Rectangle():

def \_\_init\_\_(self, l, w):

self.length = l

self.width = w

def rectangle\_area(self):

return self.length\*self.width

newRectangle = Rectangle(12, 10)

print(newRectangle.rectangle\_area())

• Write a Python class named Circle constructed by a radius and two  
methods which will compute the area and the perimeter of a circle.

Ans:

class Circle():

def \_\_init\_\_(self, r):

self.radius = r

def area(self):

return self.radius\*\*2\*3.14

def perimeter(self):

return 2\*self.radius\*3.14

NewCircle = Circle(8)

print(NewCircle.area())

print(NewCircle.perimeter())

• Explain Inheritance in Python with an example? What is init? Or What  
Is A Constructor In Python?

Ans:

# A Python program to demonstrate inheritance

# Base or Super class. Note object in bracket.

# (Generally, object is made ancestor of all classes)

# In Python 3.x "class Person" is

# equivalent to "class Person(object)"

class Person(object):

# Constructor

def \_\_init\_\_(self, name):

self.name = name

# To get name

def getName(self):

return self.name

# To check if this person is an employee

def isEmployee(self):

return False

# Inherited or Subclass (Note Person in bracket)

class Employee(Person):

# Here we return true

def isEmployee(self):

return True

# Driver code

emp = Person("Geek1") # An Object of Person

print(emp.getName(), emp.isEmployee())

emp = Employee("Geek2") # An Object of Employee

print(emp.getName(), emp.isEmployee())

The \_\_init\_\_ function is called every time an object is created from a class. The \_\_init\_\_ method lets the class initialize the object's attributes and serves no other purpose. It is only used within classes.

The purpose of a python constructor is to assign values to the data members within the class when an object is initialized. The name of the constructor method is always \_\_init\_\_. In this example, the \_\_init\_\_ method is called when the Person object is created, and it sets the name and age attributes of the object.

• What is Instantiation in terms of OOP terminology?

Ans: Instantiation − The creation of an instance of a class. Method − A special kind of function that is defined in a class definition. Object − A unique instance of a data structure that's defined by its class. An object comprises both data members (class variables and instance variables) and methods.

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

Ans: Python isinstance() Function

The isinstance() function returns True if the specified object is of the specified type, otherwise False . If the type parameter is a tuple, this function will return True if the object is one of the types in the tuple.

• What relationship is appropriate for Course and Faculty?

Ans: Association(associated with each other)

• What relationship is appropriate for Student and Person?

Ans: Association(associated with each other)