Question → 1.

Difference Between list and Tuple

List	Tuple.
(1) Lists are mutable.	1 Tuples are immutable
(2) Lists is a container to contain difference type of objective and is used to iterate objects.	2. Tuples is also similar to list but contains immutable objects
3. Syntax of list: List = ['a', 'b', 'c', 1, 2, 3]	3. Syntax of Tuple Tuples :- ['a', 'b', 'c', 1, 2]
4. List iteration is slower	4 Tuples processing is faster than List.
5. Lists consume more memory	5. Tuples consume less memory
6. Operation like insertion and deletion are better performed	6. Elements can be accessed better.

Question → 2. What is Decorator Explain in with an example.

Answer → A Decorator is just a function that takes another function as an argument, add some kind of functionality and then returns another function.

All of this without altering the source of the original function that you passed in.

```
def decorator_func(func):
    def wrapper_func
        print("wrapper_func Worked")
        return func()
    print("decorator_func worked")
    return wrapper_func
```

```
def show():
        print("show Worked")
decorator__show = decorator_func show
decorator__show()

Output:
decorator__func worked
wrapper__func Worked
Show Worked
# Alternative
@ decorator_func
def display():
        print("display worked")
display()
```

Output

decorator__func worked
wrapper__func Worked
display worked.

Question → 3. What is the difference between list Comprehension and Dict Comperhension.

Answer ⇒

List Comperhension	Dict Comperhension
Syntax: [expression for item in iterable. if conditional]	Syntax: { key: value for (key, value) in iterable. if conditional.
Example Common Way:	Ex. Common Way:
l = []	d = {}
for i in range (10):	for i in range (1, 10):
if i % 2:	sqr = i*i
l.append (i)	d(i) = i * i
print (l)	print (d)

Using list Comprehension:

```
ls = [i for i in range(10) if i%2]
```

Output:

[1, 3, 5, 7, 9]

Using Dict Comprehension:

```
d1 = {n: n*n for n in range(1, 10)}
print(d1)
```

Output:

{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}

Question → 4. How Memory Managed in python?

Answer →

Memory management in python involves a private heap containing all python objects and data structures Interpreter takes care of python heap and that the programmer has no access to it.

#

The allocation of heap space for python objects is done by Python memory manager. The core API of Python provides some tools for the programmer to code reliable and more robust program.

Python some tools for the programmer Python also has a build-in garbage collector which recycles all the unused memory. When an object in on longer referrenced by the program, the heap space it occupies can be freed. The garbage collector determiners object which are no longer referrenced by the program frees the occupied memory and meke it available to the heap space.

#

The gc module defines functions to enable / disable garbage collector:

- gc.enable() - Enables automatic garbage collection
- gc.disable() - Disable automatic garbage collection

Question → 5. What is the difference between Generator & Iterator?

Answer :-

Generator	Iterator
# Generator are iterators which can execute only once.	An iterator is an object which contains a countable number of values and it is used to iterator over iterable objects like list, tuples, sets
# Generator uses "yield" keyword	# Iterator are used mostly to iterate or conver other object to an iterator using iter() function.
# Generator are mostly used in loops to generator an iterator by returning all the value in the loop without affecting the iteration of the loop.	# Iterator uses iter() and next() functions.
# Every Generator is an iterator.	# Every iterator is not a generator.

Example :
def sqr(n):
for i in range(1, n+1):
yield i*i
a = sqr(3)
print(next(a))
print(next(a))
print(next(a))
Output
1
4
9
	Example :-
	iter_list = iter(['A', 'B', 'C']
	print(next(iter_list))
	print(next iter_list)
	print(next iter_list)
	
	Output
	A
	B
	C.

Question→6 What is 'init' Keyword in python?

Answer → __init__ .py file

The __init.py file lets the python interpreter know that a directory contains code for a python module. It can be blank. Without one, you cannot import modules from another folder into your project.

The role of the __init__ .py file is similar to the __init function in a python class. The file essentially the constructor of your package or directory without it being called such. It sets up how packages or functions will be imported into your other files.

__init__() function

The __init__ method is similar to constructors in C++ and Java. Constructors are used to initialize the object's state.

```
# A Sample class with init method
class Person:
    # init method or constructor
    def __init__(self, name):
        self.name = name
    # Sample Method
    def say_hi(self):
        print('Hello, my name is', self.name)
p = Person('Nitin')
p.say_hi()
```

Question→7. Difference Between Modules and Packages in Python.

Modules →

The modules is a simple Python file that contains collections of function and global variable and with having a .py extension file. It is an executable file and to organise all the modules we have the concept called Package in python.

A module is a single file (or files) that are imported under one import and used.

E.g. import <My_Module>
import numpy

#

Packages ⇒

The Package is a simple directory having collections of modules. This directory contains Python modules and also having __init.py file by which the interpreter interprets it as a Package. The package is simply a namespace. The package also contains — package inside it.
A package is a collection of modules in directories that give a package hierarchy
E.g. from my — package . abc import a.

Question ⇒ 8. What is the difference between Range() & X range() ?

Parameters	Range()	X range()
Return type	it returns a list of integers.	it returns a generator object
Memory Consumption	Since range() returns a list of elements, it takes more memory.	In Comparison to range(), it takes less memory.
Speed	Its execution speed is slower.	its execution speed is faster.
Python Version	Python 2, Python 3.	xrange no longer exists.
Operations	Since it returns a list, all kinds of arithmatic operation can be performed.	Such operations cannot be performed on x range().

Question → 9. What are Generators? Explane with an example.

Answer →

Generators are iterators which can execute only once.

Every generator is an iterator.

Generator uses "yield" keyword.

Generator are mostly used in loops to generator an iterator by returning all the values in the loop without affecting the iteration of the loop.

Example :-

```
def sqr(n):
for i in range(1, n+1);
    yield i*i
a = sqr(3)
print("The squre are:")
print(next(a))
print(next(a))
print(next(a))
```

Output

The squre are:
1
4
9.

Question → 10 What are inbuilt Data Types in pytho

Or

Explain Mutable & Immutable Data Types in Python.

Answer → A first fundamental distinction that python makes on data is about whether or not the value of an object changes. If the value can change, the object is called Mutable. While if the value cannot change, the object is called Immutable.

A first fundamantal distinction the python makes on data is about whether or not the value of an object changes.

Data Type	Mutable Or Immutable ?
Boolean (bool)	Immutable
Integer (int)	" " "
Float	" " "
String (str)	" " "
tuple	" " "
Frozenset	" " "
list	Mutable "
set	Mutable
dict	" "

Question 11 Explain Ternary Operator in Python?

Answer ⇒

[if _ true] if [expression] else [if _ false]

Ternary Operator Example:

```
age: 25
discount = 5 if age < 65 else 10
print (discount)
```

Question-12.

What is Inheritance in Python?

Answer ⇒ In Inheritance, the child class acquire the properties and can access all the data member and functions defined in the parent class. A child class can also provides its specific implementation to the function of the parent class.

In python, a derived class can inherit base class by just mentioning the base in the bracket after the derived class name.

Class A(B):

Example :-

```
Class A:
    def display (self):
        print ("A Display")
Class B (A):
    def show (self):
        print ("B show")
d = B()
d. show()
d. display()
```

Output:

B show

A Display.

Question-13 What is the difference between local & Global Variable?

Local Variable	Global Variable
# It is declared inside a function.	It is declared aside the function.
# If it not initialized, a garbage value is stored.	if it is not initialized zero is stored as defaul
# It is created when the fuaction starts execution and lost when the fun. terminat	It is created before the program's global execution starts and some global variable.
# Data sharing is not possible as data of the local variable can be accessed by only one function.	Data sharing is possible as multiple function can access the same global Variable.
# Parameters passing is requied for variable to access the value inother function.	Parameters passing is not necessary for a global variable as it is visible throughout the program.
# It is storage on the stack unless specified.	It is stored on a fixed location decided by the compiler.

Question → 14. What is Break, Continue & Pass statment ?

Answer ⇒ A break statment, when used inside the loop, will terminate the loop and exit. If used inside nested loops, it will break out from the current loop.

A continue statment will stop the current execution when used inside a loop, and the control will go back to the start of the loop.

A pass statment is a null statment When the python interpreter comes across the pass statment, it does nothing and is ignored.

Break statment Ex.

```
for in range (10):
    if i == 7:
        break
    print (i, end = " ")
```

Output
0, 1, 2, 3, 4, 5, 6

Continue statment Ex.

```
for i in range (10):
    if i == 7:
        continue
    print (i, end = " ")
```

Output
0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Pass statment Ex.

```
def my_func():
    print (pass inside function)
    pass
my_func()
```

Output
pass inside function

Question - 15 What is self self Keyword in python ?

Answer = The 'self' parameter is a reference to the current instance of the class, and is used to access variable that belongs to the class.

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def info(self)
        print(f"My name {}. I am {self.age}
            years old")

c = Person("Nitin"), 23
c-info()
```

Output:

My name is Nitin. I am 23 year old.

Question 16 Difference Between Pickling and Unpickling?

Answer

Pickling ⇒ In python, the Pickling module accepts my Python object, transforms it into a string representation, and dumps is into a file by using the dump function, This process is known as pickling. The function used for this process is pickle.dump()

Unpickling ⇒

The process of retrieving the original python object from the string representation is called unpickling.

The function used for this process is pickling.load()

- They are inverse of each other.
- Pickling, also called serialization, involves converting a Python object into a series of bytes which can be written out to a file.

- Unpickling, or de-serialization, does the opposite it converts a series of bytes into the python object it represents.

Question → 17 Explain Function of list, Set, Tuple And Dictionary?

Function of List

Sort(): Sorts the list in ascending order.
#
append(): Adds a single element of a list
#
extend(): Adds multiple elements of a list.
#
max(list):- It return as item from the list with max value.
(min) list: It return an item from the list with min value
len(): it gives the total length of the list.
list(seq): It gives the total Convert a tuples into a list.
type: list: it return the class type of an object.

Function of Tuple

Camp(tuple)1, tuple2 - Compares elements of the both tuples.
#
len(): total length of the tuple.
#
Max(): Returns item from the tuple with max value.
#
tuple(seq): Convert a list into tuple.
#
sum(): return are arithmatic sum of all the items in the tuple.
any(): if even one item in the tuple has a Boolean value of true. Otherwise it returns false.
sorted(): a sorted version
of the tuple.
count(): it takes one argument and returns the number of times and items appears in the tuple.

function of Dictionary

Clear(): Return Removes all the elements from the dictionary.

Copy(): Return a copy of the dictionary.

fromkeys(): Return a dictionary with the specified keys and value.

get(): Return the value of the specified key.

items(): Return a list containaning a tople for each key value pair.

Keys(): Return a list containing a tople for each key value pair.

pop(): Removes the element with the specified key.

popitem(): Removes the last inserted key-value pair

exist: inset the key, with the specified value.

cmp(): Compare two dictionaryies.

function of set

Add: Adds an elements to the set.

clear(): Removes all the a copy of the set.

difference() Return a set containing the diff b/w two or more setp.

difference() Removes the specified item items in the set that are also included in another

discard(): Remov the specified item.

intersection(): Returns a set that is the intersections of two or more sets.

isdisjoint(): Return whether two sets have a intersection or not

pop(): Returns whether this set contains mother set or not.

Removes(): Removes the specified element.

update(): update the set with another set, or any other iterable.

Question-18 What are python Iterators?

Answer ⇒ An iterator is an object which contains a countable number and it is used to iterate over iterable objects lik list, tuple, sets etc

Iterators are used mostly to iterate or convert other objects to an iterator using iter() function.

Iterator uses iter() and next() function.

Every iterator is not a generator.

Example →

```
iter_list = iter(['A', 'B', 'C'])
print(next(iter_list))
print(next(iter_list))
print(next(iter_list))
```

Output

A
B
C

Example Question →19 Explain Type Conversion in python [int(), float(), ord(), oct(), str() etc.)]

Answer ⇒ float() – Return a floating point number from a number or string

Example

```
a = '100'
d = float(a)
print(d)
print(type(d))
```

Output 100.0 <class 'float'>

oct() – Returns its octal representation in a string format.

Example :

```
a = 100
d = oct(a)
print (d)
print (type (d))
```

Output

```
0144
<class 'str'>
```

ord() – Returns the integer of the Unicode point of the character in the Unicode case or the byte value in the case of an 8-bit argument

Example →

```
a = 'A'
d = ord (a)
print = (d)
print = (type (d))
```

Output

```
65
<class 'int'>
```

eval → Parses the expression argument and evaluates it as python expression.

Example →

```
a = '100 + 2 + 3'
d = eval (a)
print (d)
print (type (d))
```

Output

```
105
<class 'int'>
```

str() - Convert a value (integer or float) into a string.

Example →

```
a = 100
d = str(a)
print(d)
print(type(d))
```

Output

```
100
<class 'str'>
```

Question → 20 What does * args and * kwargs mean? Explain.

Answer ⇒ When you are not clear how many argument you need to pass to a particular function, then we use * args and ** kwarge.

The * args keyword represents a varied number of argument. It is used to add together the value of multiples arguments.

The * kwarge keyword represents an arbitrary number of argument that are passed to a function ** kwarage keywords are stored in a dictionary. You can access each item by referring to the keyword you associated with an argument when you passed the argument.

* args Python Ex.

```
def sum(*args):
    total = 0
    for a in args:
        total = total + a
    print(total)
```

Output 15

** kwarge Python Ex.

```
def show(**kwarge)
    print(kwarge)
show(A=1, B=2, C=3)
```

Output ⇒ {'A': 1, 'B': 2, 'C': 3}

Question ⇒ 21 What is Open and With Statment?
Answer ⇒ - Both statment are used in case of file handling.
- With the 'With' statment, You get better syntax and exceptions handling

```
f = open ("nitin.txt")
content = f.read()
print (content)
f.close()
with open ("nitin.txt") as if:
    content = f.read()
    print (content)
```

Question ⇒ 22 What are the different ways to read / write a file in python?
Answer ⇒ Read only ('r') ⇒ Open text file for reading. The handle is positioned at the beginning of the file. If the file does not exist, raise I/O error. This is also the default mode in which file is opened.
Read and Write ('r+') ⇒ Open the file for reading and writing. The Handle is positioned at the beginning of the file. Raise I/O error if the file does not existes.
Write Only ('w') ⇒ Open the file for writing For existing file, the date is truncated and over-written. The Handle is positioned at the beginning of the file. Creates the file does not exists.

Write and Read ('w+') ⇒ Open the file reading and writing. For existing file, data is truncated and over written. The handle is positional at the beginning of the file.

Append Only ('a') ⇒ Open the file for writing. The file is created if it does not exist. The Handle is positioned at the end of the file. The data being written will be Inserted at the end, after the existing data.

Appened and Read ('a+') ⇒ Open the file for reading and writing. The file is created if it is created if it does not exist. The handle is positioned at the end of the file. The data being written will be inserted at the end, after existing data.

Text Modle ⇒ meaning /n characters will be translated to the host OS line ending when writing to a file, and back again when reading.

#

Exclusive creation ('x') ⇒ file is created and opened for writing - but only if it doesn't already exist otherwise you get a file Exists Error.

Binary mode ⇒ appended to the mode opens the file in binary mode, so there are also modes like 'rb', 'wb' and 'r+b'

Question-23 What is PythonPath ?

Answer ⇒ PYTHONPATH is an environment variable which you can set to add additional directories where python will look for modules and package.

The 'PYTHONPATH' variable holds a string with the name of various directories the need to be added to the sys. path directory list by Python.

The primary use of this variable is to allow users to import modules that are not made installable yet.

Question ⇒ 24.

How Exception is Handled in Python?

Answer ⇒ Try: This block will test the exceptional error to occur.

Except :- Here you can handle the errors.

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

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

```
try
  # some Code ..... !
except
      # Optional Block
      # Handling of exception (if required)
else:
    # some code....
    # execute if no exception
finally
    # Some code..... (always executed)
```

Question → 25 What is the difference b/w Python 2.0 & Python 3.0?

Python3	Pyton2.
# In this version, Rules of ordering comparisons have been simplified.	Rules of ordering comparision are very complex.
# The new Range() function introduced to perform iterations.	# In python2, the xrange() is used for iterating.
# It should be enclosed in parenthesis.	The value It should be enclosed in notations.
# The value of variable never changes.	The value of the global Variable will changes while using it inside for-loop.
# Not difficult to port python 2 to python 3 but it is never reliable.	# Python version 3 is not backwardly compatible with python 2.
# Many recent developed are creating libraries which you can only use with python.	Many older-libraries created for python 2 is not forword-compatible.

26. Question What is 'Pip' in python?
Answer → The basic syntax of pip commands in command prompt is:
pip 'arguments'

Question → 27 Where Python is used?

Web Applications
Desktop Ap.
Database Ap.
Networking Ap.
Machine Learning
Artificial intelligence
Data Analysis.
IOT Application.
Games and many more.....!

Question → 28 How to use F string and Format or Replacement Operator?

Answer → How to using f - string.

```
name = 'Nitin'
role = 'Python Developer'
print (f"Hello, My name is {name} and
    I'm {role}")
```

Output

Hello, My name is Nitin and I'm python Developer.

```
# How, To Use format Operator
name = 'Nitin'
role = 'Python Developer'
print = ('Hello My name is {} and I'm {}
    formate.(name, role)
```

Output

Hello, My name is Nitin and I'm python Developer.

Question 29 How to Get list of all keys in a Dictionary ?

Case -1, 2 : Using list

```
dct = {'A': 1, 'B': 2, 'C': 3}
all keys = list(dct.keys())
print(all - keys)
```

shortcut for above Code:

```
dct = {'A': 1, 'B': 2, 'C': 3}
all_key = list(dct)
print(all - keys)
```

Output

['A', 'B', 'C']

Case -3, 4 ; Using Iterable Unpackling Operator

```
d = {'A': 1, 'B': 2, 'C': 3}
X = (* d.keys())
print(X)
```

shortcut for Above Code:

```
d = {'A': 4, 'B': 2, 'C': 3}
print(X)
```

Output

['A', 'B', 'C']

Case : 5 Using keys() function.

```
d = {'A': 1, 'B': 2, 'C': 3}
x = d.keys()
print = (k for k in x)
```

Output

('A', 'B', 'C')

Case: 6, 7 Using Iterable Unpacking Operator →

```
d = {'A': 1, 'B': 2, 'C': 3}
*x = d.keys()
print(x)
```

Shortcut for Above Code:

```
d = {'A':1, 'B':2, 'C':3}
*x = d
print(x)
```

Output

('A', 'B', 'C')

Question ⇒ 30 Difference blw Abstraction and Encapsulation

Abstraction	Encapsulation
# Abstraction is implemented to hide unnecessary data and withdrawing relevant data.	Encapsulation is the mechanism of hiding the code and the data together from the outside world or misuse.
# Abstraction works on the design level.	Encapsulation works on the application level.
# It highlights what of an object instead of how the object works is.	It focuses on the inner details of how the object works. Modifications can be done later to the setting.
# Abstraction is supported in Java with the interface and the abstract class.	Encapsulation is supported using e.g. public, private and secure accesses modification systems.
# In a nutshell, abstraction is hiding implementation with the help of an interface and an abstract class.	In a nutshell, encapsulation is hiding the data with the help of getters and setters.

Question-31. Does python support Multiple Inheritance ? (Diamond Problem)

Answer ⇒ Yes, Python supports Multiple Inheritance.

What is Diamond Problem? →

Java does not allow is multiple inheritance where one class can inherit properties from more than one class. It is known as the diamond problem.

```
class A
{
public void display()
{
   System.out.println("class A");
}
}
```

```
class B extend A
{
  @ Override
   public void display()
    System.out.println("class B");
}
}
```

```
Class C extends A
{
  @ Override
   public void display()
   {
     System out println("class C");
   }
}
```

// not supported in Java

```
public class D extends B,C.
{
    Public static void main (String args[])
    {
    D d = new ();
    // created ambiguity which display
    () method to call d. Display ();
    }
```

In the above figure, we find that class D is trying to inherit from class B and C that

Multiple Inheritance in python :-

```
class A:
    def abc (self):
        print ("a")
Class B (A)
    def abc (self):
        print (b)
class C(A)
    def abc (self):
        print (c)
class D(B,C)
    def abc (self)
        print pass
d=D
d. abc ()
```

Output

b

Question: 32. How to initialise Empty List, Tuple, Dict & set.

Answer ⇒ Empty List:

```
a = []
```

Empty Tuple

```
a = ()
```

Empty Dict:

```
a = { }
```

Empty Set:

```
a = set()
```

Question=33 What is the difference between .py & .pyc file?

Answer ⇒ .py files contain the source code of a program. pyc file contains the bytecode of your program.

python complain the .py files and saves it as .pyc files. so it can reference them is subsequent invocations.

The pyc contain the complained bytecode of python source files. This code is then executed by python's virtual machine.

Question=34. How Slicing Works in String Manipulation Explain?

Syntax: Str — Object [Start — Position: End Position: Step]

```
s = 'HelloWorld'
```

Indexing

H	E	L	L	O	W	O	R	L	D
0	1	2	3	4	5	6	7	8	9
-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(s[:])
```

Output
Hellowworld

Indexing

H	E	L	L	O	W	O	R	L	D
0	1	2	3	4	5	6	7	8	9
-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

print (s[2:5])

Output
llo

H	E	L	L	O	W	O	R	L	D
0	1	2	3	4	5	6	7	8	9
-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

print (s[2:8:2])

output
lwo

print (s[8:1:-1])

Output
lroWoll

print (s[-4:2])

Output
or

print (s[::-1])

Output
dlroWolleH

[::-1] → Used for Palindrome

Question 35 Can you Concatenat two Tuples. if Yes, How is it Possible? Since it is Immutable? How to Concatenat two Tuple:

t1 =(1,2,3)
t2 → (7,9,10)
t1 =(t1 + t2
print (" After concatenation is :(", t1)

Output

After concatenation is: (1,2,3,7,9,10)

Why Tuple is Immutable and list is Mutable?

tuple_1 = (1,2,3)
print (id (tuple_1) # 140180965800128
tuple_2 = (7,9,10)
print (id (tuple_2) # 140180965665600
tuple_1 =tuple_1 + tuple_2
tuple_3 =tuple_1
list_1 = [1,2,3]
print (id (list_1) # 140180965602048
list_2 =[7,9,10]
print (id (list_2) # 140180965601408
list_1 extend (list_2)
list_3 = list_1
print (The list after concatenation is:" list_1)
The list after concatenation is: (1,2,3,7,9,10)
print (id list_3) # 140180965602048

tuple_3 → # 140180966177280

Question 36. Difference Blw. python Arrays and Lists.

LIST	ARRAY
The list can store the value of different Types.	It can only consist of value of same type.
The list cannot handle the direct arithmatic operations.	It can directly handle arithmatic operations.
The list are less compatible than the array to store the data.	An array are much compatible than the list.
The consumes a large memory.	It is a more compact in memory size comparatively list.
It is suitable for storing the longer sequence of the data item.	It is suitable for storing shorter sequence of data items.
We can print the entire list using explict looping	We can print the entire list without using explicit looping.

Question 37 What is a, __a, __a in Python?

Answer — a

Leading double underscore tell python interpreter to rewrite name in order to avoid conflict in subclass.

Interpreter changes variable name with class extension and that feature known at the Mangling.

In Mangling python interpreter modify variable name with ——

PAGE:
DATE: / /

- So Multiple time it use as the private member b/c another class can not access that variable directly.
- Main purpose for __ is to variable in class only if you want to use it outside of the class you can make public api.

__ a __

- Name with start with __ and ends with same considers special methods in python.
- Python provides this methods to use it use overloading depending on the user.
- Python provides this convention to differentiate blw the user defined with the module's function.

Question=38 How to Read Multiple Values from Single Input ?

Answer ⇒ By using split()

```
x = List (map (int, Input ("Enter a multiplevalue
    split ()))
print ("List of Values: ", x)
x = [int (x) for x in input ["Enter multiple
Value :") split ()"]
    print (Number of list is :" x )
x = [int(x) for x in input [" Enter multiple
value :"). split (",")]
    print (" Number of list is :", x]
```

Question 39 How to Copy and Delet a Dictionary

Answer → Delete By Using clear():

```
d1 = {'A':1, 'B':2, 'C':3}
di.clear()
print(d1) # {}
```

Delete By using pop():

```
d1 = {'A':1, 'B':2, 'C':3}
print(d1) # {'A':1, 'B':2, 'C':3}
di.pop('A')
print(d1) # {'13':2, 'C':3}
```

Delete By Using del():

```
del d1['B']
print(d1) # {'C':3}
```

Copy A Dictionary Using copy():

```
d2 = {'A':1, 'B':2, 'C':3}
print(d2) # {'A':1, 'B':2, 'C':3}
```

Copy A Dictionary Using '=':

```
d2 = {'A':1, 'B':2, 'C':3}
print(d2) # {'A':1, 'B':2, 'C':3}
d3 = d2
print(d3) # {'A':1, 'B':2, 'C':3}
```

Benefit of Using Copy():

```
d2 = {'A':1, 'B':2, 'C':3}
print(d2) # {'A':1, 'B':2, 'C':3}
d3 = d2.copy()
print(d2) # {'A':1 'C':3}
print(d3) # {'A':1, 'B':2, 'C':3}
```

Question ⇒ to difference Blw Anonymous and Lambda Function

Ans ⇒ Lambda function :- It can have any number of arguments but only one experession.

The expression is evaluated and returned.

Lambda functions can be used wherever function objects are required.

Anonymous function :-

In python Anonymous functions is a functions that is defined without a name.

While normal functionl are defined using the def keyword Anonymous fun. are defined using the lambda keyword.

Hence, anonymous functions are also called lambda functions.

Question ⇒ 47 How to achieve Multiprocessing and Multithreading in python?

Multithreading →

It is a technique where multipli threads are spawned by process todo different tasks, at about the same time, just one after the other.

The gives you the illusion that the threads are running is pardlel, but they are actually run in a concurrent manner.

In python, the Global Interpreter Lock (Gil) privents the threads from runing simultaneously.

Multiprocessing :- →

* It is a technique where parallelism in its from is achieved.

* Each process can have many threads running in its own memory space.

* In python, each process has its own instance of python interpreter doing the job of executing the instructions

Question 42. What is GIL Explain?

Answer → The Global Interpreter Lock (GIL) of python only one thread to be executed at a time. It is often a hurble, as it does not allow multi-threading In python to save time.

* The python Global Interpreter Lock or GIL in simple words, is a mutex the allows only one thread to hold the control of the python interpreter.

* Basically, GIL in python doesn't allow multi threading which some thing be considered as a disadvantage.

* Since the GIL allows one thread to execute at a time even in a multi ac architecture with more than one CPU more the GIL has gained a reputation as 'infamous' feature of python.

Question #3 How Class & Objects are created in python?

Answer

* Python is an object oriented programming language.

* Almost everything in Python is an object, with its properties and methods.

* A Class is like an object constructor, or a "blueprint" for creating objects.

Create a Class:

To create a class use the keyword 'class'

```
Class Myclass
    x=5
```

Create Object

Now we can use the class named My class to create objects

```
obj : My class()
    print (obj.x)
```

Question → 44. What is Namespace and What is it's types?

Answer → In python we deal with variables, function, libraries and modules etc.

* In such scenario, we need to learn about how all these names managed by a python program.

This is the concept of namespace.

Categories of Namespace : Following are the three categories of Namespace.

Local Namespace →

All the name of the following functions and Variables be declared by a program are held in this namespace This namespace exists as long as the program runs.

Global namespace →

This namespace hold all the name of functions and other variables that are included in the modules being used in the python program. It includes all are part of the local namespace.

Built-in Nameespace →

This is the highest level of namespace which is available with default name available as part of the python interpreter that is loaded as the programing environment. It include Global Namespace which in turn included the local namespace. We can access all the names defined in the built-in namespace. ces follows

```
builtin — name - dir (-builtins-)
    for name in builtin — names:
        print (name)
```

Question → 45 → explain Recursion by Reversing a list.

Answer → def reversed list (lst):
if not lst:
return []
return [lst[-1]] + reversed list [lst[:-1]]
print (reversed list ([1,2,3,4,5])

Output:
[5, 4, 3, 2, 1]

Question → 46. What are Unittests in Python?

Answer → Unit Testing → is the first level of software testing where the smallest testable parts of a shoftware are tested. This is used to validate that each unit of the software perfos ms as designed. The unittests test framework is python's xunit style framework. This is how you can import it →

Imported unittest →

* Unit testing is a shoftware testing method by which individual units of source code are put under various tests to determine whether they are fit for use. It determines and ascrtains the quality of your face.

* Generally, when the development process is complete, the developer codes criteria, or the result that are known to be potentially practical and useful, into the test script to verify a particulare unit, s

correctness. During test case execution various frameworks log tests that fail any criteria and report them in a summary.

* The developers are expected to write automated test scripts, which ensures that each and every section or a unit meets its design and behaves as expected.

* Though writing manual tests for your code is definitely a tedious and time consuming task, python's built-in unit testing framework has made life a lot easier.

* The unit test framework in python is called, which comes packaged with python.

Question 47. How to use Map, Filter and and Reduce Function in python?

Answer → Filter() function →

The filter() function takes a function object and an iterable and creates a new list. As the name suggest, filter() function a new list that contains only elements that satisfy a certain condition, i.e. function we passed returns True.

The syntax is:

filter() function, iterables(s))

```
fruit = ["Apple", "Banana", "Pear"]

filter___object = filter(lambda x: x[0]="A
    print [list(filter___object])
```

Output,

['Apple', 'Apricot']

* Reduce() function →

The reduce() function works differently than map() and filter(). It does not return a new list based on the function and iterable we've passed, it return a single value, Also, in python 3 reduce() isn't a built-in function anyone. and it can be found in the functools module.

The syntax is:

```
from functools import reduce
    list: [2,4,7,3]
    print (reduce) (lambda x, y: x+y. list)
    print ("with an inital value" + str
reduce (lambda x,y: x+y, list, 10)
```

Output

16

With an initial value: 26

Question → 48 What is the difference blw Shallow Copy & Deep Copy.?

Answer → Shallow Copy →

Shallow Copy ies duplicate as little as possible, A shallow copy of a collection is a copy of the collection structure, not the elements with a shallow copy, two collections now share the individual elements.

Shallow copying is creating a new object and then copying the non static fields of the current to the new object if the field is a value type, a bit by bit copy of the field is performed. If the freild as a reference type, the reference is copied but the referred object is not, therefore the original object and its clone refere to the seme object.

Deep Copy →

Deep Copies duplicate everything. A deep copy of a collection is two collections with all of the elements in the original collection duplicated.

Deep copy is creating a new object and then copying the non-static fields of the current objects to the new object. if a field is a value type, a bit by bit copy of the field is performed if a field is a reference type, a new copy of the referred object is

performed. A deep copy of an object is a new object with entirely new instance variables. it does not share objects with the old. While performing Deep Copy the classes to be cloned must be flagged as [Serializable].

Question ⇒ 50 What does term MONKEY PATCHING refer to in python?

Answer

In python, the term monkey patch refers to dyamic (or run-time) modifications of a class or module. In python, we can actully change the behaviour of code run-time.

```
# monkey.py
Class A:
    def func(self)
    print ("func() is called")
```

module

Whe We have above (monkey) is below code and change behaviour of func() at run-time by assignment different value.

```
Import monkey
def monkey_func (self):
    print ("monkey_func() is called)
# replacing address of "func" with "monkey"
obj = monkey.A()
```

called function "func" whose address got replaced
with function "monkey_func()".

obj.func()

Example.

Output: monkey_func() is called.

Question → 51 What is Operator Overloading & Dunder Method?

* Dunder method in python are special methods.

* In python, we sometimes see method name with a dobule underscore (__) such as the __init__ method that every class has. These methods are called "dundre" methods.

* In python, Dunder methods are used for operator overloading and customizing some other function's behaviour.

Some Example →

+	—	add — (self, other)
-	—	sub — (" , ")
*	—	mul — (" , ")
/	—	truediv — (" , ")
//	—	floordiv — (" , ")
%	—	mod — (" , ")
**	—	pow — (" , ")
>>	—	rshift — (" , ")
<<	—	lshift — (" , ")

& — cmd — (self, other)
| — or — (self, other)
^ — xor — (self, other)

Question → 52 Draw pattern →

Answer # This is the example of print simple pyramid pattern

n - int (input) ("Enter the number of rows")

other loop to handle number of rows

for in range (0, n):

inner loop to handle number of columns

values is changing according to outer loop for j in rang (0, i+1):

printing stars

print ("* ", end = "")

ending linear after each row

print ()

Output

```
*
* *
* * *
* * * *
* * * * *
```