### What is Python? State some programming language features of Python.

Python is a modern powerful interpreted language with objects, modules, threads, exceptions, and automatic memory managements.

Python was introduced to the world in the year 1991 by Guido van Rossum

Salient features of Python are

- Simple & Easy: Python is simple language & easy to learn.

- Free/open source: it means everybody can use python without purchasing license.

- High level language: when coding in Python one need not worry about low-level details.

- Portable: Python codes are Machine & platform independent.

- Extensible: Python program supports usage of C/ C++ codes.

- Embeddable Language: Python code can be embedded within C/C++ codes & can be used a scripting language.

- Standard Library: Python standard library contains prewritten tools for programming.

- Build-in Data Structure: contains lots of data structure like lists, numbers & dictionaries.

### Explain how python is interpreted.

Python program runs directly from the source code. Each type Python programs are executed code is required. Python converts source code written by the programmer into intermediate language which is again translated it into the native language / machine language that is executed. So Python is an Interpreted language.

### What are the rules for local and global variables in Python?

If a variable is defined outside function then it is implicitly global. If variable is assigned new value inside the function means it is local. If we want to make it global we need to explicitly define it as global. Variable referenced inside the function are implicit global. Following code snippet will explain further the difference

```

#!/usr/bin/python

# Filename: variable\_localglobal.py

def fun1(a):

print 'a:', a

a= 33;

print 'local a: ', a

a = 100

fun1(a)

print 'a outside fun1:', a

def fun2():

global b

print 'b: ', b

b = 33

print 'global b:', b

b =100

fun2()

print 'b outside fun2', b

```

Output

```

$ python variable\_localglobal.py

a: 100

local a: 33

a outside fun1: 100

b :100

global b: 33

b outside fun2: 33

```

### Explain the dictionary in Python.

Python's built-in data type is dictionary, which defines one-to-one relationships between keys and values.

- Dictionaries consist of pairs of keys and their corresponding values.

- Dictionaries are indexed by keys.

- Dictionary is similar to associative array or hash table of other languages.

As following example explains further- India, Angel & Cartoon are keys & their corresponding values are Bharat, Mother Teresa & Mickey respectively.

```

>>> dict = {'India': 'Bharat', 'Angel': ‘Mother Teresa’, 'Cartoon': 'Mickey'}

>>>print dict[India]

Bharat

>>>print dict[Angel]

Mother Teresa

```

### How do we share global variables across modules in Python?

We can create a config file & store the entire global variable to be shared across modules or script in it. By simply importing config, the entire global variable defined it will be available for use in other modules.

For example I want a, b & c to share between modules.

```

config.py :

a=0

b=0

c=0

module1.py:

import config

config.a = 1

config.b =2

config.c=3

print “ a, b & resp. are : “ , config.a, config.b, config.c

```

output of module1.py will be

1 2 3

### How can we pass optional or keyword parameters from one function to another in Python?

Gather the arguments using the `\*` and `\*\*` specifiers in the function's parameter list.

This gives us positional arguments as a `tuple` and the keyword arguments as a `dictionary`.

Then we can pass these arguments while calling another function by using \* and \*\*:

```

def fun1(a, \*tup, \*\*keywordArg):

...

keywordArg['width']='23.3c'

...

Fun2(a, \*tup, \*\*keywordArg)

```

### Explain indexing and slicing operation in sequences

Different types of sequences in python are strings, Unicode strings, lists, tuples, buffers, and xrange objects

Slicing & indexing operations are salient features of sequence.

indexing operation allows to access a particular item in the sequence directly ( similar to the array/list indexing) and the slicing operation allows to retrieve a part of the sequence.

The slicing operation is used by specifying the name of the sequence followed by an optional pair of numbers separated by a colon within square brackets say S[startno.:stopno].

The startno in the slicing operation indicates the position from where the slice starts and the stopno indicates where the slice will stop at.

If the startno is ommited, Python will start at the beginning of the sequence. If the stopno is ommited, Python will stop at the end of the sequence..

Following code will further explain indexing & slicing operation:

```

>>> cosmeticList =[‘lipsstick’,’facepowder’,eyeliner’,’blusher’,kajal’]

>>> print “Slicing operation :”,cosmeticList[2:]

Slicing operation :[‘eyeliner’,’blusher’,kajal’]

>>>print “Indexing operation :”,cosmeticList[0]

“Indexing operation :lipsstick

```

### What is a Lambda form? Explain about assert statement?

The lambda form:

Using lambda keyword tiny anonymous functions can be created.

It is a very powerful feature of Python which declares a one-line unknown small function on the fly. The lambda is used to create new function objects and then return them at runtime.

The general format for lambda form is:

lambda parameter(s): expression using the parameter(s)

For instance k is lambda function-

```

>>> k= lambda y: y + y

>>> k(30)

60

>>> k(40)

80

```

The assert statement:

The build-in assert statement of python introduced in version 1.5 is used to assert that something is true.

Programmers often place assertions at the beginning of a function to check for valid input, and after function call to check for valid output. Assert statement can be removed after testing of program is over.

If assert evaluates to be false, an AssertionError exception is raised. AssertionError exceptions can be handled with the try-except statement.

The general syntax for assert statement is:

assert Expression[, Arguments]

### Explain the role of repr function.

Python can convert any value to a string by making use of two functions repr() or str().

The str() function returns representations of values which are human-readable, while repr() generates representations which can be read by the interpreter.

repr() returns a machine-readable representation of values, suitable for an exec command.

Following code sniipets shows working of repr() & str() :

```

def fun():

y=2333.3

x=str(y)

z=repr(y)

print " y :",y

print "str(y) :",x

print "repr(y):",z

fun()

-------------

output

y : 2333.3

str(y) : 2333.3

repr(y) : 2333.3000000000002

```

### Explain pickling and unpickling.

pickle is a standard module which serializes & de-serializes a python object structure.

pickle module accepts any python object converts it into a string representation & dumps it into a file(by using dump() function) which can be used later, process is called pickling. Whereas unpickling is process of retrieving original python object from the stored string representation for use.

### What is LIST comprehensions features of Python used for?

LIST comprehensions features were introduced in Python version 2.0, it creates a new list based on existing list.

It maps a list into another list by applying a function to each of the elements of the existing list.

List comprehensions creates lists without using map() , filter() or lambda form.

### How is memory managed in python?

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 also has a build-in garbage collector which recycles all the unused memory. When an object is no longer referenced by the program, the heap space it occupies can be freed. The garbage collector determines objects which are no longer referenced by the sprogram frees the occupied memory and make it available to the heap space.

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

gc.enable() -Enables automatic garbage collection.

gc.disable() - Disables automatic garbage collection.

### How do you make a higher order function in Python?A higher-order function accepts one or more functions as input and returns a new function. Sometimes it is required to use function as data

To make high order function , we need to import functools module

The functools.partial() function is used often for high order function.

### Explain how to copy an object in Python.

There are two ways in which objects can be copied in python.

Shallow copy & Deep copy.

Shallow copies duplicate as minute as possible whereas Deep copies duplicate everything.

If a is object to be copied then

- copy.copy(a) returns a shallow copy of a.

- copy.deepcopy(a) returns a deep copy of a.

### How can I find the methods or attributes of an object in python?

Built-in dir() function of Python ,on an instance shows the instance variables as well as the methods and class attributes defined by the instance's class and all its base classes alphabetically. So by any object as argument to dir() we can find all the methods & attributes of the object’s class.

Following code snippet shows dir() at work :

```

class Employee:

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

self.name=name

self.empCode=empCode

self.pay=pay

print("dir() listing all the Methods & attributes of class Employee")

print dir(e)

-----------------------------------------------------

Output

dir() listing all the Methods & attributes of class Employee

[ '\_\_init\_\_', 'empCode', 'name', 'pay']

```

### How do I convert a string to a number?

Python contains several built-in functions to convert values from one data type to another data type.

The int function takes string and coverts it to an integer.

```

>>>s = "1234" # s is string

>>>i = int(s) # string converted to int

>>>print i+2

-------------------------

1236

The float function converts strings into float number.

>>>s = "1234.22" # s is string

>>>i = float(s) # string converted to float

>>>print i

-------------------------

1234.22

```

### What is a negative index in python?

Python arrays & list items can be accessed with positive or negative numbers (also known as index).

For instance our array/list is of size n, then for positive index 0 is the first index, 1 second, last index will be n-1. For negative index, -n is the first index, -(n-1) second, last negative index will be – 1.

A negative index accesses elements from the end of the list counting backwards.

An example to show negative index in python

```

>>> import array

>>> a= [1, 2, 3]

>>> print a[-3]

1

>>> print a[-2]

2

>>> print a[-1]

3

```

### How an array in Python?

The array module contains methods for creating arrays of fixed types with homogeneous data types. Arrays are slower then list.

Array of characters, integers, floating point numbers can be created using array module.

array(typecode[, intializer])

Returns a new array whose items are constrained by typecode, and initialized from the optional initialized value. Where the typecode can be for instance ‘c’ for character value, ‘d’ for double, ‘f’ for float.

### Explain how to create a multidimensional list.

There are two ways in which Multidimensional list can be created:

By direct initializing the list as shown below to create multidimlist below

```

>>>multidimlist = [ [227, 122, 223],[222, 321, 192],[21, 122, 444]]

>>>print multidimlist[0]

>>>print multidimlist[1][2]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Output

[227, 122, 223]

192

```

The second approach is to create a list of the desired length first and then fill in each element with a newly created lists demonstrated below :

```

>>>list=[0]\*3

>>>for i in range(3):

>>> list[i]=[0]\*2

>>>for i in range (3):

>>> for j in range(2):

>>> list[i][j] = i+j

>>>print list

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Output

[[0, 1], [1, 2], [2, 3]]

```

### Explain how to overload constructors (or methods) in Python.

`\_init\_\_ ()` is a first method defined in a class. when an instance of a class is created, python calls \_\_init\_\_() to initialize the attribute of the object.

Following example demonstrate further:

```

class Employee:

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

self.name=name

self.empCode=empCode

self.pay=pay

e1 = Employee("Sarah",99,30000.00)

e2 = Employee("Asrar",100,60000.00)

print("Employee Details:")

print(" Name:",e1.name,"Code:", e1.empCode,"Pay:", e1.pay)

print(" Name:",e2.name,"Code:", e2.empCode,"Pay:", e2.pay)

---------------------------------------------------------------

Output

Employee Details:

(' Name:', 'Sarah', 'Code:', 99, 'Pay:', 30000.0)

(' Name:', 'Asrar', 'Code:', 100, 'Pay:', 60000.0)

```

### Desc to send mail from a Python script.

The smtplib module defines an SMTP client session object that can be used to send mail to any Internet machine.

A sample email is demonstrated below.

```

import smtplib

SERVER = smtplib.SMTP(‘smtp.server.domain’)

FROM = sender@mail.com

TO = ["user@mail.com"] # must be a list

SUBJECT = "Hello!"

TEXT = "This message was sent with Python's smtplib."

# Main message

message = """

From: Sarah Naaz < sender@mail.com >

To: CarreerRide user@mail.com

Subject: SMTP email msg

This is a test email. Acknowledge the email by responding.

""" % (FROM, ", ".join(TO), SUBJECT, TEXT)

server = smtplib.SMTP(SERVER)

server.sendmail(FROM, TO, message)

server.quit()

```

### Desc how to generate random numbers in Python.

The standard module random implements a random number generator.

There are also many other in this module, such as:

```

uniform(a, b) returns a floating point number in the range [a, b].

randint(a, b)returns a random integer number in the range [a, b].

random()returns a floating point number in the range [0, 1].

```

Following code snippet show usage of all the three functions of module random:

Note: output of this code will be different evertime it is executed.

```

import random

i = random.randint(1,99)# i randomly initialized by integer between range 1 & 99

j= random.uniform(1,999)# j randomly initialized by float between range 1 & 999

k= random.random()# k randomly initialized by float between range 0 & 1

print("i :" ,i)

print("j :" ,j)

print("k :" ,k)

\_\_\_\_\_\_\_\_\_\_

Output -

('i :', 64)

('j :', 701.85008797642115)

('k :', 0.18173593240301023)

Output-

('i :', 83)

('j :', 56.817584548210945)

('k :', 0.9946957743038618)

```

### How do we make python scripts executable?

Python scripts can be executed in two ways:

Suppose I want to execute script1.py

We can open the script1.py in IDE editor & run the script in the frontmost window of the python IDE by hitting the run all button.

Second way is using command prompt by making sure PATH is set appropriately directly type script name else type

```

>>>python script1.py

```

```

```

```

### Desc how exceptions are handled in python.

Errors detected during execution of program are called exceptions. Exceptions can be handled using the try..except statement.

We basically put our usual statements within the try-block and put all our error handlers in the except-block.

```

try…except demo code:

>>> while True:

try:

x = int(raw\_input("Enter no. of your choice: "))

break

except ValueError:

print "Oops! Not a valid number. Attempt again"

Enter no. of your choice: 12ww

Oops! Not a valid number. Attempt again

Enter no. of your choice: hi there

Oops! Not a valid number. Attempt again

Enter no. of your choice: 22

>>>

```

### What is used to create Unicode string in Python?

Unicode is a system to represent characters from all the world's different languages.

Two methods can be used to create unicode string:

unicode() method is unicode(string[, encoding, errors]), its arguments should be 8-bit strings. The first argument is converted to Unicode using the specified encoding, if encoding argument left, the ASCII encoding is used for the conversion.

encode() method is encode([encoding], [errors='strict']), which returns an 8-bit string version of the Unicode string.

### When to use list vs. tuple vs. dictionary vs. set?

List is like array, it can be used to store homogeneous as well as heterogeneous data type (It can store same data type as well as different data type). List are faster compared to array. Individual element of List data can be accessed using indexing & can be manipulated.

```

List Code Snippet:

list = ["Sarah",29,30000.00]

for i in range (3):

print list[i]

------------------

Output

Sarah

29

30000.0

```

Tuples are similar to lists, but there data can not be changed once created throught the execution of program. Individual element of Tuples can be accessed using indexing.

Tuples Code Snippet: The Days

```

days = ("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday")

print days

------------------------------

('Sunday', 'Mondays', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday')

```

Sets stores unordered values & have no index. And unlike Tuples and Lists, Sets can have no duplicate data, It is similar to Mathematical sets.

add() function can be used to add element to a set.

update() function can be used to add a group of elements to a set.

Copy() function can be used to create clone of set.

Set Code Snippet:

```

disneyLand = set (['Minnie Mouse', 'Donald Duck', 'Daisy Duck', 'Goofy'])

disneyLand.add('Pluto')

print disneyLand

-----------------------------------------------

Output

set(['Goofy', 'Daisy Duck', 'Donald Duck', 'Minnie Mouse', ’Pluto’])

Dictionary are similar to what their name is. In a dictionary, In python, the word is called a 'key', and the definition a 'value'. Dictionaries consist of pairs of keys and their corresponding values.

Dictionary Code Snippet:

>>> dict = {'India': 'Bharat', 'Angel': ‘Mother Teresa’, 'Cartoon': 'Mickey'}

>>>print dict[India]

Bharat

>>>print dict[Angel]

Mother Teresa

```

### Explain the disadvantages of python

Disadvantages of Python are:

Python isn't the best for memory intensive tasks.

Python is interpreted language & is slow compared to C/C++ or java.

Python not a great choice for a high-graphic 3d game that takes up a lot of CPU.

Python is evolving continuously, with constant evolution there is little substantial documentation available for the language.

### What are the ways to write a function using call by reference?

Arguments in python are passed as an assignment. This assignment creates an object that has no relationship between an argument name in source and target. The procedure to write the function using call by reference includes:

The tuple result can be returned to the object which called it. The example below shows it:

```

def function(a, b):

a = 'value'

b = b + 1

# a and b are local variables that are used to assign the new objects

return a, b

# This is the function that is used to return the value stored in b

```

• The use of global variables allows the function to be called as reference but this is not the safe method to call any function.

• The use of mutable (they are the classes that consists of changeable objects) objects are used to pass the function by reference.

```

def function(a):

a[0] = 'string'

a[1] = a[1] + 1

# The ‘a’ array give reference to the mutable list and it changes the changes that are shared

args = ['string', 10]

func1(args)

print args[0], args[1]

#This prints the value stored in the array of ‘a’

```

### What are the commands that are used to copy an object in Python?

The command that is used to copy an object in python includes:

• copy.copy() function: This makes a copy of the file from source to destination. It returns a shallow copy of the parameter that is passed.

• copy.deepcopy(): This also creates a copy of the object from source to destination. It returns a deep copy of the parameter that is passed to the function.

The dictionary consists of all the objects and the copy() method which is used as:

newdict = olddict.copy()

The assignment statement doesn’t copy any object but it creates a binding between the target and the object that is used for the mutable items. Copy is required to keep a copy of it using the modules that is provided to give generic and shallow operations.

### What is the difference between deep and shallow copy?

• Shallow copy is used when a new instance type gets created and it keeps the values that are copied in the new instance. Whereas, deep copy is used to store the values that are already copied.

• Shallow copy is used to copy the reference pointers just like it copies the values. These references point to the original objects and the changes made in any member of the class will also affect the original copy of it. Whereas, deep copy doesn’t copy the reference pointers to the objects. Deep copy makes the reference to an object and the new object that is pointed by some other object gets stored. The changes made in the original copy won’t affect any other copy that uses the object.

• Shallow copy allows faster execution of the program and it depends on the size of the data that is used. Whereas, deep copy makes it slower due to making certain copies for each object that is been called.

### Write a program to find out the name of an object in python.

The object doesn’t have any name and there is no way the can be found out for objects. The assignment is used to bind a name to the value that includes the name of the object that has to be bound by a value. If the value is callable then the statements are made true and then the program followed can be used to find the reference name of an object.

```

class try:

pass

B = A

a = B()

b = a

print b

<\_\_main\_\_.try instance at 0x16D07CC>

print b

```

The class consists of name and the names are invoked by using the the variable B that creates an instance for the class try. The method is to find out from all the namespaces that the object exists and then print the name of the object.

### How can the ternary operators be used in python?

The ternary operator is the operator that is used to show the conditional statements. This consists of the true or false values with a statement that has to be evaluated for it. The operator will be given as:

```

[on\_true] if [expression] else [on\_false]

x, y = 25, 50

big = x if x < y else y

```

This is the lowest priority operator that is used in making a decision that is based on the values of true or false. The expression gets evaluated like if x<y else y, in this case if x<y is true then the value is returned as big=x and if it is incorrect then big=y will be sent as a result.

### How the string does get converted to a number?

• To convert the string into a number the built-in functions are used like int() constructor. It is a data type that is used like int (‘1’) ==1.

• float() is also used to show the number in the format as float(‘1’)=1.

• The number by default are interpreted as decimal and if it is represented by int(‘0x1’) then it gives an error as ValueError. In this the int(string,base) function takes the parameter to convert string to number in this the process will be like int(‘0x1’,16)==16. If the base parameter is defined as 0 then it is indicated by an octal and 0x indicates it as hexadecimal number.

• There is function eval() that can be used to convert string into number but it is a bit slower and present many security risks like \_\_import\_\_('os').system("rm -rf$HOME") - use of this will delete the home directory of the system.

### What is the function of negative index?

The sequences in python are indexed and it consists of the positive as well as negative numbers. The numbers that are positive uses ‘0’ that is uses as first index and ‘1’ as the second index and the process goes on like that. The index for the negative number starts from ‘-1’ that represents the last index in the sequence and ‘-2’ as the penultimate index and the sequence carries forward like the positive number. The negative index is used to remove any new-line spaces from the string and allow the string to except the last character that is given as S[:-1]. The negative index is also used to show the index to represent the string in correct order.

### Write a program to check whether the object is of a class or its subclass.

There is a method which is built-in to show the instances of an object that consists of many classes by providing a tuple in a table instead of individual classes. The method is given as isinstance(obj,cls) and in more details given as:

isinstance(obj, (class1, class2, ...)) that is used to check about the object’s presence in one of the classes. The built in types can also have many formats of the same function like isinstance(obj, str) or isinstance(obj, (int, long, float, complex)).

It is not preferred to use the class instead user-defined classes are made that allow easy object-oriented style to define the behavior of the object’s class. These perform different thing that is based on the class. The function differs from one class to another class. To find out the object of the particular class the following program is used:

```

def search(obj):

if isinstance(obj, box):

# This is the code that is given for the box and write the program in the object

elif isinstance(obj, Document):

# This is the code that searches the document and writes the values in it

elif

obj.search()

#This is the function used to search the object’s class.

```

### Why does delegation performed in Python?

Delegation is a technique that is used in object oriented programming. This is used to show the object and the behavior of the methods that are used. The class can be created to provide an implementation of the method that allows the method to be referenced. The delegate is having the parameter and the return value in an object. It also allows the methods to be passed as parameters and allow the defining of the callback methods that can be grouped together in multiple methods. These methods can be called from a single event. The example shows a class that captures the behavior of the file and converts data from lower to uppercase.

```

class upcase:

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

self.\_out = out

def write(self, s):

self.\_outfile.write(s.upper())

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

return getattr(self.\_out, name)

```

The write() method that is used in the upcase class converts the string to the uppercase before calling another method. The delegation is being given using the self.\_\_outfile object.

### What is the function of “self”?

“Self” is a variable that represent the instance of the object to itself. In most of the object oriented programming language, this is passed as to the methods as a hidden parameters that is defined by an object. But, in python it is declare it and pass it explicitly. It is the first argument that gets created in the instance of the class A and the parameters to the methods are passed automatically. It refers to separate instance of the variable for individual objects. This is the first argument that is used in the class instance and the “self” method is defined explicitly to all the methods that are used and present. The variables are referred as “self.xxx”.

### How is “self” explicitly defined in a method?

“Self” is a reference variable and an instance attribute that is used instead of the local variable inside the class. The function or the variable of the self like self.x or self.meth() can be used in case the class is not known. There are no variables declared as local. It doesn’t have any syntax and it allow the reference to be passed explicity or call the method for the class that is in use. The use of writebaseclass.methodname(self, <argument list>) shows that the method of \_init\_() can be extended to the base class methods. This also solves the problem that is syntactic by using the assignment and the local variables. This tells a way to the interpreter the values that are to be used for the instance variables and local variables. The use of explicit self.var solves the problem mentioned above.

### What is the use of join() for a string rather than list or tuple method?

The functions and the methods that are used for the functionality uses the string module. This string module is represented as by using the join function in it:

> ", ".join(['1', '2', '4', '8', '16']) that results in "1, 2, 4, 8, 16"

The string variable that is used provide a fixed string literal to allow the names that are used to be bounded to the strings. join() is a string method that is used to provide a separator string to use the function over the sequence of the string and insert the function to an adjacent elements. The method uses any number of arguments that follow some rules that has to be put up for the sequence objects that the class defines for itself. The join is used for the string module that is used to join the string characters together as it is given in the program. The example is given as:

```

string.join(['1', '2', '4', '8', '16'], ", ")

```

### What is the process of compilation and linking in python?

The compiling and linking allows the new extensions to be compiled properly without any error and the linking can be done only when it passes the compiled procedure. If the dynamic loading is used then it depends on the style that is being provided with the system. The python interpreter can be used to provide the dynamic loading of the configuration setup files and will rebuild the interpreter. The steps that is required in this as:

• Create a file with any name and in any lanugage that is supported by the compiler of your system. For example comp.c

• Place this file in the Modules/ directory of the distribution which is getting used.

• Add a line in the file Setup.local that is present in the Modules/ directory.

• Run the file using spam comp.o

• After successful run of this rebuild the interpreter by using the make command on the top-level directory.

• If the file is changed then run rebuildMakefile by using the command as ‘make Makefile’.

```

### How does global value mutation used for thread-safety?

The global interpreter lock is used to allow the running of the thread one at a time. This is internal to the program only and used to distribute the functionality along all the virtual machines that are used. Python allows the switching between the threads to be performed by using the byte code instructions that are used to provide platform-independence. The sys.setcheckinterval() method is used that allow the switching to occur during the implementation of the program and the instruction. This provides the understanding in the field of accounting to use the byte code implementation that makes it portable to use. The atomicity can be provided such that the shared variables can be given as built-in data types.

### Write a program to read and write the binary data using python?

The module that is used to write and read the binary data is known as struct. This module allows the functionality and with it many functionalities to be used that consists of the string class. This class contains the binary data that is in the form of numbers that gets converted in python objects for use and vice versa. The program can read or write the binary data is:

```

import struct

f = open(file-name, "rb")

# This Open() method allows the file to get opened in binary mode to make it portable for # use.

s = f.read(8)

x, y, z = struct.unpack(">hhl", s)

```

The ‘>” is used to show the format string that allows the string to be converted in big-endian data form. For homogenous list of data the array module can be used that will allow the data to be kept more organized fashion.

### What are the different ways to generate random numbers?

Random module is the standard module that is used to generate the random number.

The method is defined as:

```

import random

random.random()

```

The statement random.random() method return the floating point number that is in the range of [0, 1). The function generates the random float numbers. The methods that are used with the random class are the bound methods of the hidden instances. The instances of the Random can be done to show the multi-threading programs that creates different instance of individual threads. The other random generators that are used in this are:

- randrange(a, b): it chooses an integer and define the range in-between [a, b). It returns the elements by selecting it randomly from the range that is specified. It doesn’t build a range object.

- uniform(a, b): it chooses a floating point number that is defined in the range of [a,b).Iyt returns the floating point number

- normalvariate(mean, sdev): it is used for the normal distribution where the mu is a mean and the sdev is a sigma that is used for standard deviation.

- The Random class that is used and instantiated creates an independent multiple random number generators.