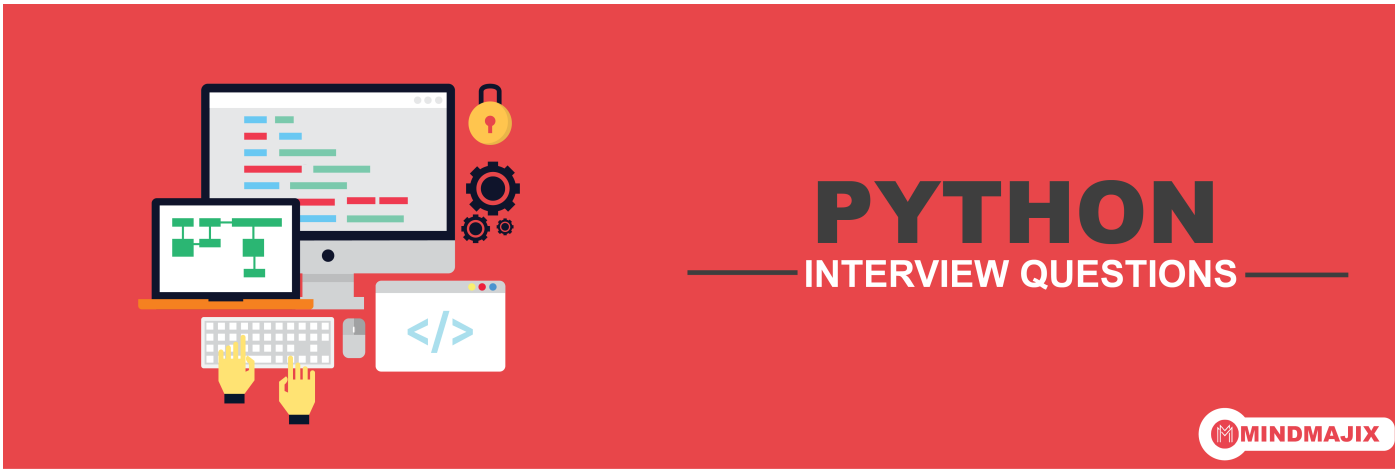
Blog

Python Interview Questions

★★★★☆ (4.0) | 59012 Ratings



If you're looking for Python Interview Questions and Answers for Experienced & Freshers, you are at right place. There are lot of opportunities from many reputed companies in the world. According to research Python has a market share of about 4.0%. So, You still have opportunities to move ahead in your career in Python. Mindmajix offers advanced Python Interview Questions 2019 that helps you in cracking your interview & acquire your dream career as Python Developer.

Q1. What are the differences between Python and Java?

Python Vs Java		
Comparison	**Python**	**Java**
Performance Speed	Fast	Not as much as Python
Indentation	Must be followed	Using proper flower braces is enough
Typing	Dynamically typed	Static typed
Accessability	Simple and compact	Not as much as Python
Platforms	Not compatible to many	Platform independent
Database Access	Weak compared to JAVA	Strong (JDBC)

Q2. How is Python executed?

Python files are compiled to bytecode. which is then executed by the host.
Alternate Answer:
Type python .pv at the command line.

Q3. What is the difference between .py and .pyc files?

.py files are Python source files. .pyc files are the compiled bvtecode files that is generated by the Python compiler

Q4. How do you invoke the Python interpreter for interactive use?

python or pythonx.y where x.y are the version of the Python interpreter desired.

Q5. How are Phon blocks defined?

By indents or tabs. This is different from most other languages which use symbols to define blocks. Indents in Python are significant.

Q6. What is the Pthon interpreter prompt?

Three greater-than signs: >>> Also, when the int
erpreter is waiting for more input the prompt changes to three periods

Q7. How do you execute a Python Script?

From the command line, type python .py or pythonx.y
.py where the x.y is the version of the Python interpreter desired.

Q8. Explain the use of try: except raise, and finally:

Try, except and finally blocks are used in Python error handling. Code is executed in the try block until an error occurs. One can use a generic except block, which will receive control after all errors, or one can use specific exception handling blocks for various error types. Control is transferred to the appropriate except block. In all cases, the finally block is executed. Raise may be used to raise your own exceptions.

Q9. Illustrate the proper use of Python error handling.

Code Example:

```
try:
….#This can be any code
except:
…# error handling code goes here
finally.
…# code that will be executed regardless of exception handling goes here.
```

Q10. What happens if an error occurs that is not handled in the except block?

The program tenuinates. and an execution trace is sent to sys.stderr.

Q11. How are modules used in a Python program?

Modules are brought in via the import statement.

Q12. How do you create a Python function?

Functions are defined using the def statement. An example might be def foo(bar):

Q13. How is a Python class created?

Classes are created using the class statement. An example might be class aa rdva rk(fooba r):

Q14. How is a Python class instantiated?

The class is instantiated by calling it directly. An example might be
myclass =aardvark(5)

Q15. In a class definition, what does the __ init_O function do?

It overrides the any initialization from an inherited class, and is called when the class is instantiated.

Q16. How does a function return values?

Functions return values using the return statement.

Q17. What happens when a function doesn't have a return statement? Is this valid?

Yes, this is valid. The function will then return a None object. The end of a function is defined by the block of code being executed (i.e., the indenting) not by any explicit keyword.

Q18. What is the lambda operator?

The lambda operator is used to create anonymous functions. It is mostly used in cases where one wishes to pass functions as parameters. or assign them to variable names.

Q19. Explain the difference between local and global namespaces.

Local namespaces are created within a function. when that function is called. Global name spaces are created when the program starts.

Q20. Name the four main types of namespaces in Python?

- Global,
- Local,
- Module and
- Class namespaces.

Q21. When would you use triple quotes as a delimiter?

Triple quotes '''" or '" are string delimiters that can span multiple lines in Python. Triple quotes are usually used when spanning multiple lines, or enclosing a string that has a mix of single and double quotes contained therein.

Q22. What are the two major loop statements?

for and while

Q23. Under what circumstances would von use a while statement rather than for?

The while statement is used for simple repetitive looping and the for statement is used when one wishes to iterate through a list of items, such as database records, characters in a string, etc.

Q24. What happens if.ou put an else statement after after block?

The code in the else block is executed after the for loop completes, unless a break is encountered in the for loop execution. in which case the else block is not executed.

Q25. Explain the use of break and continue in Python looping.

The break statement stops the execution of the current loop. and transfers control to the next block. The continue statement ends the current block's execution and jumps to the next iteration of the loop.

Q26. When would you use a continue statement in a for loop?

When processing a particular item was complete; to move on to the next, without executing further processing in the block. The continue statement says, "I'm done processing this item, move on to the next item."

Q27. When would you use a break statement in a for loop?

When the loop has served its purpose. As an example. after finding the item in a list searched for, there is no need to keep looping. The break statement says, I'm done in this loop; move on to the next block of code."

Q28. What is the structure of afor loop?

for in : ... The ellipsis represents a code block to be executed, once for each item in the sequence. Within the block, the item is available as the current item from the entire list.

Q29. What is the structure of a while loop?

while : … The ellipsis represents a code block to be executed. until the condition becomes false. The condition is an expression that is considered true unless it evaluates to o, null or false.

Q30. Use a for loop and illustrate how you would define and print the characters in a string out, one per line.

```
myString = “I Love Python”
for myChar hi myString:
print myChar
```

Q31. Given the string “I LoveQPython” use afor loop and illustrate printing each character tip to, but not including the Q.

```
inyString = “I Love Pijtlzon”
for myCizar in myString:
fmyC’har ==
break
print myChar
```

Q32. Given the string “I Love Python” print out each character except for the spaces, using a for loop.

```
inyString = I Love Python”
for myCizar in myString:
fmyChar == ‘’ ‘’:
continue
print myChar
```

Q33. Illustrate how to execute a ioop ten times.

i=1

while i < 10:

Q34. How to use GUI that comes with Python to test your code?

That is just an editor and a graphical version of the interactive shell. You write or load code and run it, or type it into the shell.
There is no automated testing.

Q35. What is Python good for?

Python is a high-level general-purpose programming language that can be applied to many different classes of problems.

The language comes with a large standard library that covers areas such as string processing like regular expressions, Unicode, calculating differences between files, Internet protocols like HTTP, FTP, SMTP, XML-RPC, POP, IMAP, CGI programming, software engineering like unit testing, logging, profiling, parsing Python code, and operating system interfaces like system calls, file systems, TCP/IP sockets.

Q36. How does the Python version numbering scheme work?

Python versions are numbered A.B.C or A.B.

- A is the major version number. It is only incremented for major changes in the language.
- B is the minor version number, incremented for less earth-shattering changes.
- C is the micro-level. It is incremented for each bug fix release.

Not all releases are bug fix releases. In the run-up to a new major release, ‘A’ series of development releases are made denoted as alpha, beta, or release candidate.

- Alphas are early releases in which interfaces aren’t finalized yet; it’s not unexpected to see an interface change between two alpha releases.
- Betas are more stable, preserving existing interfaces but possibly adding new modules, and release candidates are frozen, making no changes except as needed to fix critical bugs.

- Alpha, beta and release candidate versions have an additional suffix.
- The suffix for an alpha version is “aN” for some small number N,
- The suffix for a beta version is “bN” for some small number N,
- And the suffix for a release candidate version is “cN” for some small number N.

In other words, all versions labeled 2.0aN precede the versions labeled 2.0bN, which precede versions labeled 2.0cN, and those precede 2.0.

You may also find version numbers with a “+” suffix, e.g. “2.2+”. These are unreleased versions, built directly from the subversion trunk. In practice, after a final minor release is made, the subversion trunk is incremented to the next minor version, which becomes the “a0” version, e.g. “2.4a0”.

Q37. Where is math.py (socket.py, regex.py, etc.) source file?

If you can’t find a source file for a module, it may be a built-in or dynamically loaded module implemented in C, C++ or other compiled language. In this case you may not have the source file or it may be something like mathmodule.c, somewhere in a C source directory (not on the Python Path).

There are (at least) three kinds of modules in Python:

- Modules written in Python (.py);
- Modules written in C and dynamically loaded (.dll, .pyd, .so, .sl, etc);
- Modules written in C and linked with the interpreter; to get a list of these, type;
- Import sys print sys.builtin_module_names;

Q38. How do I make a Python script executable on UNIX?

You need to do two things:

- The script file’s mode must be executable and the first line must begin with “#!” followed by the path of the Python interpreter. The first is done by executing chmod +x scriptfile or perhaps chmod 755 ‘script’ file.
- The second can be done in a number of ways.

The most straightforward way is to write:

#!/usr/local/bin/python

As the very first line of your file, using the pathname for where the Python interpreter is installed on your platform. If you would like the script to be independent of where the Python interpreter lives, you can use the “env” program. Almost all UNIX variants support the following, assuming the python interpreter is in a directory on the users $PATH:

#! /usr/bin/env python

Don’t do this for CGI scripts. The $PATH variable for CGI scripts is often minimal, so you need to use the actual absolute pathname of the interpreter. Occasionally, a user’s environment is so full that the /usr/bin/env program fails; or there’s no env program at all. In that case, you can try the following hack (due to Alex Rezinsky):

```
#! /bin/sh
“””:”
exec python $0 ${1+”$@”}
“””
```

The minor disadvantage is that this defines the script’s __doc__ string. However, you can fix that by adding:

__doc__ = “””...Whatever...”””

Q39. Why don’t my signal handlers work?

The most common problem is that the signal handler is declared with the wrong argument list. It is called as:
handler (signum, frame)
So it should be declared with two arguments:
def handler(signum, frame):

Q40. How do I test a Python program or component?

Python comes with two testing frameworks:

The documentation test modulefinds examples in the documentation strings for a module and runs them, comparing the output with the expected output given in the documentation string.
The unit test moduleis a fancier testing framework modeled on Java and Smalltalk testing frameworks.

For testing, it helps to write the program so that it may be easily tested by using good modular design. Your program should have almost all functionality encapsulated in either functions or class methods. And this sometimes has the surprising and delightful effect of making the program run faster because local variable accesses are faster than global accesses.

Furthermore the program should avoid depending on mutating global variables, since this makes testing much more difficult to do.
The “global main logic” of your program may be as simple as:

if __name__==”__main__”:
main_logic()

at the bottom of the main module of your program.
Once your program is organized as a tractable collection of functions and class behaviors, you should write test functions that exercise the behaviors.

A test suite can be associated with each module which automates a sequence of tests.
You can make coding much more pleasant by writing your test functions in parallel with the “production code”, since this makes it easy to find bugs and even design flaws earlier.
“Support modules” that are not intended to be the main module of a program may include a self-test of the module.

if __name__ == “__main__”:
self_test()

Even programs that interact with complex external interfaces may be tested when the external interfaces are unavailable by using “fake” interfaces implemented in Python.

Q41. How do I find undefined g++ symbols __builtin_new or __pure_virtual?

To dynamically load g++ extension modules, you must:
Recompile Python
Re-link it using g++ (change LINKCC in the python Modules Makefile)
Link your extension module using g++ (e.g., “g++ -shared -o mymodule.so mymodule.o”).

Q42. How do I send mail from a Python script?

Use the standard library module smtplib. Here’s a very simple interactive mail sender that uses it. This method will work on any host that supports an SMTP listener.

```
import sys, smtplib
fromaddr = raw_input("From: ")
toaddrs = raw_input("To: ").split(',')
print "Enter message, end with ^D:"
msg = ''
while 1:
line = sys.stdin.readline()
if not line:
break
msg = msg + line
# The actual mail send
server = smtplib.SMTP('localhost')
server.sendmail(fromaddr, toaddrs, msg)
server.quit()
```

A UNIX-only alternative uses send mail. The location of the send mail program varies between systems; sometimes it is /usr/lib/sendmail, sometime /usr/sbin/sendmail. The send mail manual page will help you out. Here's some sample code:

```
SENDMAIL = "/usr/sbin/sendmail" # sendmail location
import os
p = os.popen("%s -t -i" % SENDMAIL, "w")
p.write("To: receiver@example.comn")
p.write("Subject: testn")
p.write("n") # blank line separating headers from body
p.write("Some textn")
p.write("some more textn")
sts = p.close()
if sts != 0:
print "Sendmail exit status", sts
```

Q43. How can I mimic CGI form submission (METHOD=POST)? I would like to retrieve web pages that are the result of posting a form. Is there existing code that would let me do this easily?

Yes. Here's a simple example that uses httplib:

```
#!/usr/local/bin/python
import httplib, sys, time
### build the query string
qs = "First=Josephine&MI=Q&Last=Public"
### connect and send the server a path
httpobj = httplib.HTTP('www.some-server.out-there', 80)
httpobj.putrequest('POST', '/cgi-bin/some-cgi-script')
### now generate the rest of the HTTP headers…
httpobj.putheader('Accept', '*/*')
httpobj.putheader('Connection', 'Keep-Alive')
httpobj.putheader('Content-type', 'application/x-www-form-urlencoded')
httpobj.putheader('Content-length', '%d' % len(qs))
httpobj.endheaders()
httpobj.send(qs)
### find out what the server said in response…
reply, msg, hdrs = httpobj.getreply()
if reply != 200:
sys.stdout.write(httpobj.getfile().read())
Note that in general for URL-encoded POST operations, query strings must be quoted by using urllib.quote(). For example
>>> from urllib import quote
>>> x = quote("Guy Steele, Jr.")
>>> x
'Guy%20Steele,%20Jr.'
>>> query_string = "name="+x
>>> query_string
'name=Guy%20Steele,%20Jr.'
```

Q44. Why is that none of my threads are not running? How can I make it work?

As soon as the main thread exits, all threads are killed. Your main thread is running too quickly, giving the threads no time to do any work.

A simple fix is to add a sleep to the end of the program that's long enough for all the threads to finish:

```
import threading, time
def thread_task(name, n):
for i in range(n): print name, i
for i in range(10)
```

Q45. Installation of Python 3.6.1

Download the required 3.6.1 python, executable installer file from the www.python.org.com website.
Installation Process:

- Click on the downloaded executable installer
- Click On 'Run'
- Click on Customize Installation
- Click on 'Next'
- Select the installation location by clicking on browse button
- Click on 'Install'
- Click on 'Yes'
- Click on 'Close'

Path: Path is an environment variable of operating system by using e=which we can make it available the softwares which are installed in the directory to all other directions of the operating system.

To set Path:

- Right click on my computer
- Click on properties
- Click on Advanced system setting
- Click on advanced
- Click on environment variables
- Go to system variables, select 'path'
- Click on 'edit'
- Copy the installation folder location of python software in the begging of the variable value
- Click on 'OK'

Now path setting is secured.

Q46. What Are The Implementation In Python Program?

Python program can be implemented by two ways

1. **Interactive Mode** (Submit statement by statement explicitly)
2. **Batch Mode** (Writing all statements and submit all statements)

In Interactive mode python command shell is required. It is available in installation of python cell.
In Interactive mode is not suitable for developing the projects & Applications
Interactive mode is used for predefined function and programs.
Example:

X=1000
Y=2000
X+Y
3000
QuitX+Y
X, Y is not found.

Interactive mode is unfit for looping purpose.

Interactive Mode:

- The concept of submitting one by one python statements explicitly in the python interpreter is known as "Interactive Mode"
- In Order to submit the one by one python statements explicitly to the python interpreter, we use python command line shell.
- Python command line shell is present in python software
- We can open the python command line shell by executing python command on command prompt or terminal

Example:

```
c:/users/mindmajix>python
>>>3+4
       7
>>> 'mindmajix'*3
'mindmajix mindmajix mindmajix'
>>> x=1000
>>>y=2000
>>>x+y
      3000
>>>Quit
>>>x+y
c:/users/sailu>python
```

Error: name 'X' not defined

Batch Mode:

In the concept of writing the group of python statements in a file, save the file with extension .py and submit that entire file to the python interpreter is known as Batch Mode.

- In Order to develop the python files we use editors or IDE's
- Different editors are notepad, notepad++, edit+,nano, VI, gedil and so on.

Open the notepad and write the following code

Example:

```
X=1000
Y=2000
Print(x+y, x-y, x*y)
```

Save the file in D drive mindmajix python folder with the demo.py

Open command prompt and execute following commands

```
Python D:/mindmajix python/Demo.py
3000
-1000
2000.000
```

Save another method if we correctly set the path

```
D:
D:/>d mindmajix python
D:/mindmajix Python>python Demo.py
3000
-1000
2000.000
```

Q47. What Are The Data Types Supports in Python Language?

- Python Int
- Python Float
- Python Complex
- Python Boolean
- Python Dictionary
- Python List
- Python Tuple
- Python Strings
- Python Set

Every data type in python language is internally implemented as a class
Python language data types are categorized into two types.
They are::

- Fundamental Types
- Collection Types

Q48. What Are The Types of Objects Support in Python Language?

Python supports are two types are of objects. They are

- Immutable Objects
- Mutable Objects

Immutable Objects

- The objects which doesn't allow to modify the contents of those objects are known as 'Immutable Objects'
- Before creating immutable objects with some content python interpreter verifies is already any object is available. In memory location with same content or not.
- If already object is not available then python interpreter creates new objects with that content and store that object address two reference variable.
- If already object is present in memory location with the same content creating new objects already existing object address will be given to the reference variable.

```
i=1000
print(i)
print(type(i))
print(id(i))
j=2000
print(j)
print(type(j))
print(id(j))
x=3000
print(x)
print(type(x))
print(id(x))
y=3000
print(y)
print(type(y))
print(id(y))
```

Output:

```
1000
<class>
36785936
2000
<class>
37520144
3000
<class>
37520192
3000
<class>
37520192
>>>
>>></class></class></class></class>
```

- Int, float, complex, bool, str, tuple are immutable objects
- Immutable objects performance is high
- Applying iterations on Immutable objects takes less time
- All fundamentals types represented classes objects and tuple class objects are immutable objects.

Mutable Objects:

1. The Objects which allows modifying the contents of those objects are known as 'Mutable Objects'

2. We can create two different mutable objects with the same content

Program:

```
x=[10,20,30]
print(x)
print(type(x))
print(id(x))
y=[10,20,30]
print(y)
print(type(y))
print(id(y))
```

Output:

```
[10,20,30]
<class>
37561608
[10,20,30]
<class>
37117704
>>>
>>></class></class>
```

- List, set, dict classes objects are mutable objects

- Mutable objects performance is low when compared to immutable objects
- Applying Iterations mutable objects takes huge time

Q49. Control flow statements?

By default, python program execution starts from the first line, execute each and every statement only once and transactions the program if the last statement of the program execution is over.
Control flow statements are used to disturb the normal flow of the execution of the program.

Q50. What is a Tuple?

- Tuple Objects can be created by using parenthesis or by calling tuple function or by assigning multiple values to a single variable
- Tuple objects are immutable objects
- Incision order is preserved
- Duplicate elements are allowed
- Heterogeneous elements are allowed
- Tuple supports both positive and negative indexing
- The elements of the tuple can be mutable or immutable

Example:

```
x=()
print(x)
print(type(x))
print(len(x))
y-tuple()
print(y)
print(type(y))
print(len(y))
z=10,20
print(z)
print(type(z))
print(len(z))
p=(10,20,30,40,50,10,20,10) Insertion & duplicate
print(p)
q=(100, 123.123, True, “mindmajix”) Heterogeneous
print(q)
```

Output:

```
()
<class>
0
()
<class>
0
(10,20)
<class>
2
(10,20,30,40,50,10,20,10)
(100, 123.123, True, "mindmajix")
>>>

</class></class></class>
```

Q51. What is the difference Between List And Tuple?

List	Tuple
List objects are mutable objects	Tuple objects are immutable Objects
Applying Iterations on list objects takes longer time	Applying iterations on tuple Objects takes less time

If the frequent operation is the insertion or deletion of the elements then it is recommended to use a list	If the frequent operation is the retrieval of the elements then it is recommended to use a tuple
List can't be used as a 'key' for the dictionary	Tuple can be used as a key for the dictionary if the tuple is storing only immutable elements

Q52. What is the Dictionary?

- Dictionary objects can be created by using curly braces {} or by calling dictionary function
- Dictionary objects are mutable objects
- Dictionary represents key value base
- Each key value pair of Dictionary is known as a item
- Dictionary keys must be immutable
- Dictionary values can be mutable or immutable
- Duplicate keys are not allowed but values can be duplicate
- Insertion order is not preserved
- Heterogeneous keys and heterogeneous values are allowed

Q53. What is Anonymous Function or Lambda Function?

A function which doesn't contain any name is known as an anonymous function lambda function

Syntax:
Lambda arguments:expression
Lambda function we can assign to the variable & we can call the lambda function through the variable
Example:

```
myfunction=lambda x:x*x
a=myfunction(10)
print(a)
```

Output: 100
>>>

Q54. Modules Search Path?

By default python interpreter search for the imported modules in the following locations:

- Current directory (main module location)
- Environment variable path
- Installation-dependent directory
- If the imported module is not found in any one of the above locations. Then python interpreter gives error.

Built-in attributes of a module:

- By default for each and every python module some properties are added internally and we call those properties as a built-in-attribute of a module

Q55. What are the Packages?

A package is nothing but a folder or dictionary which represents a collection of modules
A package can also contain sub packages

- We can import the modules of the package by using package name.module name
- We can import the modules of the package by using package name.subpackage name.module name

Q56. What is File Handling?

- File is a named location on the disk, which stores the data in permanent manner.
- Python language provides various functions and methods to provide the communication between python programs and files.
- Python programs can open the file, perform the read or write operations on the file and close the file
- We can open the files by calling open function of built-in-modules
- At the time of opening the file, we have to specify the mode of the file
- Mode of the file indicates for what purpose the file is going to be opened(r,w,a,b)

Q57. What are the Runtime Errors?

The errors which occur after starting the execution of the programs are known as runtime errors.
Runtime errors can occur because of

- Invalid Input
- Invalid Logic
- Memory issues
- Hardware failures and so on

With respect to every reason which causes to runtime error corresponding runtime error representation class is available

- Runtime error representation classes technically we call as an exception class.
- While executing the program if any runtime error occurs corresponding runtime error representation class object is created
- Creating runtime error representation class object is technically known as a rising exception
- While executing the program if an exception is raised, then internally python interpreter verify any code is implemented to handle the raised exception or not
- If a code is not implemented to handle raised exception then the program will be terminated abnormally

Q58. What is Abnormal Termination?

The concept of terminating the program in the middle of its execution without executing the last statement of the main module is known as an abnormal termination
Abnormal termination is an undesirable situation in programming languages.

Q59. What is Try Block?

A block which is preceded by the try keyword is known as a try block
Syntax:

```
try{
   //statements that may cause an exception
}
```

The statements which cause to runtime errors and other statements which depends on the execution of runtime errors statements are recommended to represent in a try block

While executing try block statement if any exception is raised then immediately try block identifies that exception, receive that exception and forward that exception to except block without executing remaining statements to try block.

Q60. What is the Difference Between Methods & Constructors?

Methods	Constructor
Method name can be any name	Constructor name should be
Method will be executed whenever we call a method	Constructor will be executed automatically whenever we create a object
With respect to one object one method can be called for 'n' members of lines	With respect to one object one constructor can be executed only once
Methods are used to represent business logic to perform the operations	Constructors are used to define & initialize the non static variable

Q61. What is the Encapsulation?

The concept of binding or grouping related data members along with its related functionalities is known as an Encapsulation.

Q62. What is Garbage Collection?

The concept of removing unused or unreferenced objects from the memory location is known as a Garbage Collection.

- While executing the program, if garbage collection takes place then more memory space is available for the program and rest of the program execution becomes faster.
- Garbage collector is a predefined program, which removes the unused or unreferenced objects from the memory location
- Any object reference count becomes zero then we call that object as a unused or unreferenced object
- Then no.of reference variables which are pointing the object is known as a reference count of the object.
- While executing the python program if any object reference count becomes zero, then internally python interpreter calls the garbage collector and garbage collector will remove that object from memory location.

Q63. Executing DML Commands Through Python Programs?

DML Commands are used to modify the data of the database objects

- Whenever we execute DML Commands the records are going to be modified temporarily
- Whenever we run "rollback" command the modified records will come back to its original state
- To modify the records of the database objects permanently we use "commit" command
- After executing the commit command even though we execute "rollback" command, the modified records will not come back to its original state
- Create the emp1 table in the database by using following command
- Create table emp1 as select * from emp;

Whenever we run the DML commands through the python program, then the no.of records which are modified because of that command will be stored into the rowcount attribute of cursor object

After executing the DML Command through the python program we have to call commit method of cursor object.

Q64. What is Multithreading?

Thread Is a functionality or logic which can execute simultaneously along with the other part of the program

- Thread is a lightweight process
- Any program which is under execution is known as process
- We can define the threads in python by overwriting run method of thread class
- Thread class is a predefined class which is defined in threading module
- Thread in module is a predefined module
- If we call the run method directly the logic of the run method will be executed as a normal method logic
- In order to execute the logic of the run method as we use start method of thread class.

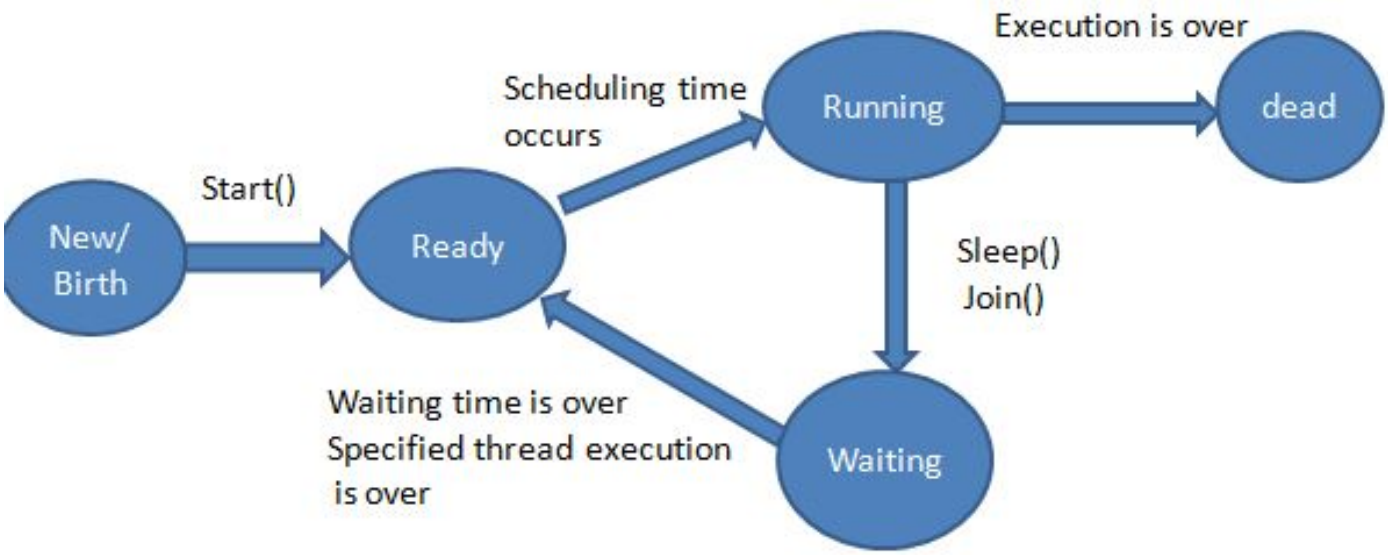
Example

```
Import threading
Class x(threading.Thread):
      Def run(self):
         For p in range(1, 101):
              print(p)
Class y(threading.Thread):
      Def run(self):
           For q in range(1, 101):
              print(q)
x1=x()
y1=y()
x1.start()
y1.start()
```

Q65. What is scheduling?

Among multiple threads which thread as to start the execution first, how much time the thread as to execute after allocated time is over, which thread, as to continue the execution next this, comes under scheduling
Scheduling is highly dynamic

Q66. What is Threads Life Cycle?



- Creating the object of a class which is overwriting run method of thread class is known as a creating thread
- Whenever thread is created then we call thread is in new state or birth state thread.
- Whenever we call the start method on the new state threads then those threads will be forwarded for scheduling
- The threads which are forwarded for scheduling are known as ready state threads
- Whenever scheduling time occurs, ready state thread starts execution
- The threads which are executing are known as running state threads
- Whenever sleep fun or join methods are called on the running state threads then immediately those threads will wait.
- The threads which are waiting are known as waiting state threads
- Whenever waiting time is over or specified thread execution is over then immediately waiting state threads are forwarded for scheduling.
- If running state threads execution is over then immediately those threads execution will be terminated
- The threads which execution is terminated are known as dead state threads.

Q67. For loop is implemented in python language as follows?

For an element in iterable:

```
iter-obj=iter(iterable)
While true;
Try:
element=next(iter_obj)
except(slop iteration)
Break
```

For loop takes the given object, convert that object in the form of iterable object & gets the one by one element form the iterable object.

While getting the one by value element from the iterable object if stop iteration exception is raised then for loop internally handle that exception

Q68. OS Module

OS Module is a predefined module and which provides various functions and methods to perform the operating system related activities, such as creating the files, removing the files, creating the directories removing the directories, executing the operating system related commands, etc.

Example:

```
Import os
cwd=os.getwd()
print(“1”, cwd)
os.chdir(“samples”)
print(“2”, os.getcwd())
os.chdir(os.pardir)
print(“3”,os.getcwd())
```

Output:

```
D:pythonstudmatosmodule
D:pythonstudmatosmodulesample
D:pythonstudmatosmodule
```

Q69. What is Hierarchical Inheritance?

The concept of inheriting the properties from one class into multiple classes separately is known as hierarchical inheritance.

Example:

```
Class x(object):
Def m1(self):
print(“in m1 of x”)
Class y(x):
Def m2(self):
print(“in m2 of y”)
Class z(x):
Def m3(self):
print(“in m3 of z”)
y1=y()
y1.m1()
y1.m2()
a=y1.--hash--()
print(a)
z1=z()
z1.m1()
z1.m3()
b=z1.hash--()
print(b)
```

Output:

```
M m1 of X
In m2 of Y
2337815
In m1 of X
In m3 of Z
2099735
>>>
```

Q70. What Are Applications of Python?

Applications of Python		
Applications of Python	**Java**	**.Net**
Automation App	NO	NO
Data Analytics	NO	NO
Scientific App	NO	NO

Web App	Yes	Yes
Web Scrapping	NO	NO
Test Cases	Yes	Yes
Network with JOT	Yes	NO
Admin Script	NO	NO
GUI	Yes	Yes
Gaming	Yes	Yes
Animation	NO	NO

Subscribe For Free Demo

Phone *

E-mail Address *

SUBSCRIBE

Free Demo for Corporate & Online Trainings.

About The Author

Ravindra Savaram is a Content Lead at Mindmajix.com (https://mindmajix.com). His passion lies in writing articles on the most popular IT platforms including Machine learning, DevOps, Data Science, Artificial Intelligence, RPA, Deep Learning, and so on. You can stay up to date on all these technologies by following him on LinkedIn (https://www.linkedin.com/in/savaram-ravindra-48064641/) and Twitter (https://twitter.com/s11ravindra).

Social Share

(https://www.facebook.com/sharer.php?u=https://mindmajix.com/python-interview-questions)
(https://twitter.com/intent/tweet?status=https://mindmajix.com/python-interview-questions)
(https://plus.google.com/share?url=https://mindmajix.com/python-interview-questions)
(https://www.linkedin.com/shareArticle?mini=true&url=https://mindmajix.com/python-interview-questions&summary=&source=)

PREVIOUS (HTTPS://MINDMAJIX.COM/ORACLE-GOLDENGATE-INTERVI NEXT (HTTPS://MINDMAJIX.COM/SAP-SUCCESSFACTORS-INTERVIEV

DROP US A QUERY

Full Name

E-mail Address

Course Name

US ▾ +1 - Phone Number *

Message

I'm not a robot reCAPTCHA Privacy - Terms

CONTACT US

Categories

- BOXI (https://mindmajix.com/boxi/)
- SAP BPC (https://mindmajix.com/sap-bpc/)
- Continuous Integration (https://mindmajix.com/continuous-integration/)
- SSIS (https://mindmajix.com/ssis/)
- ArcSight (https://mindmajix.com/arcsight/)
- STATA (https://mindmajix.com/stata/)
- Advanced SAS (https://mindmajix.com/advanced-sas/)
- Ab Initio (https://mindmajix.com/ab-initio/)
- SAP PP (https://mindmajix.com/sap-pp/)
- Apache Cassandra (https://mindmajix.com/apache-cassandra/)

VIEW MORE (HTTPS://MINDMAJIX.COM/BLOG-CATEGORY)

Popular Courses in 2019

Salesforce Lightning Training (https://mindmajix.com/salesforce-lightning-training)

★★★★★

☺ 1458 Learners

AngularJS Training (https://mindmajix.com/angularjs-training)

★★★★★

☺ 3766 Learners

Majix

Mindmajix - Online global training platform connecting individuals with the best trainers around the globe. With the diverse range of courses, Training Materials, Resume formats and On Job Support, we have it all covered to get into IT Career. Instructor Led Training - Made easy.

(https://www.facebook.com/MindMajixTechnologies) (https://twitter.com/mindmajix) (https://plus.google.com/+MindmajixTechnologies)

(https://www.linkedin.com/company/mindmajix-technologies-pvt-ltd-) (https://www.youtube.com/channel/UCkKemMaRnFPlNLHZ0zOYfpA)

Company

Home (https://mindmajix.com/)

About Us (https://mindmajix.com/about)

Courses (https://mindmajix.com/all-courses)

Sample Resumes (https://mindmajix.com/sample-resumes)

Blog (https://mindmajix.com/blog)

Contact Us (https://mindmajix.com/contact)

Reviews (https://mindmajix.com/reviews)

Write for us (https://mindmajix.com/write-for-us)

Trending Courses

Tableau Training (https://mindmajix.com/tableau-training)

Oracle DBA training (https://mindmajix.com/oracle-dba-training)

Qlikview Training (https://mindmajix.com/qlikview-training)

Docker Training (https://mindmajix.com/docker-training)

JBoss Training (https://mindmajix.com/jboss-training)

Informatica Training (https://mindmajix.com/informatica-training)

Cassandra Training (https://mindmajix.com/apache-cassandra-training)

Blockchain Training (https://mindmajix.com/blockchain-training)

Contact info

244 Fifth Avenue, Suite 1222 New York(NY) United States (US) - 10001

Call us on

Drop us a Query

USA : +1 917 456 8403

info@mindmajix.com (mailto:info@mindmajix.com)

#811, 10th A Main, Suite No.506 1st Floor, Indira Nagar Bangalore, India - 560038

India : +91 905 240 3388

info@mindmajix.com (mailto:info@mindmajix.com)