**Basic Python Interview Questions**

**Q1. What is the difference between list and tuples in Python?**

|  |  |
| --- | --- |
| **LIST vs TUPLES** | |
| **LIST** | **TUPLES** |
| Lists are mutable i.e they can be edited. | Tuples are immutable (tuples are lists which can’t be edited). |
| Lists are slower than tuples. | Tuples are faster than list. |
| Syntax: list\_1 = [10, ‘Chelsea’, 20] | Syntax: tup\_1 = (10, ‘Chelsea’ , 20) |

**Q2. What are the key features of Python?**

* Python is an **interpreted** language. That means that, unlike languages like *C* and its variants, Python does not need to be compiled before it is run. Other interpreted languages include *PHP* and *Ruby*.
* Python is **dynamically typed**, this means that you don’t need to state the types of variables when you declare them or anything like that. You can do things like x=111 and then x="I'm a string" without error
* Python is well suited to [**object orientated programming**](https://www.edureka.co/blog/python-class/) in that it allows the definition of classes along with composition and inheritance. Python does not have access specifiers (like C++’s public, private).
* In Python, [**functions**](https://www.edureka.co/blog/python-functions) are**first-class objects**. This means that they can be assigned to variables, returned from other functions and passed into functions. Classes are also first class objects
* **Writing Python code is quick** but running it is often slower than compiled languages. Fortunately，Python allows the inclusion of C based extensions so bottlenecks can be optimized away and often are. The [numpy](https://www.edureka.co/blog/python-numpy-tutorial/) package is a good example of this, it’s really quite quick because a lot of the number crunching it does isn’t actually done by Python
* Python finds **use in many spheres** – web applications, automation, scientific modeling, big data applications and many more. It’s also often used as “glue” code to get other languages and components to play nice.

**Q3. What type of language is python? Programming or scripting?**

***Ans:***Python is capable of scripting, but in general sense, it is considered as a general-purpose programming language. To know more about Scripting, you can refer to the [Python Scripting Tutorial](https://youtu.be/9F6zAuYtuFw).

**Q4.How is Python an interpreted language?**

***Ans:***An interpreted language is any programming language which is not in machine level code before runtime. Therefore, Python is an interpreted language.

**Q5.What is pep 8?**

***Ans:***PEP stands for **Python Enhancement Proposal.**It is a set of rules that specify how to format Python code for maximum readability.

**Q6. How is memory managed in Python?**

**Ans:**

1. Memory management in python is managed by ***Python private heap space***. All Python objects and data structures are located in a private heap. The programmer does not have access to this private heap. The python interpreter takes care of this instead.
2. The allocation of heap space for Python objects is done by Python’s memory manager. The core API gives access to some tools for the programmer to code.
3. Python also has an inbuilt garbage collector, which recycles all the unused memory and so that it can be made available to the heap space.

**Q7. What is namespace in Python?**

***Ans:***A namespace is a naming system used to make sure that names are unique to avoid naming conflicts.

**Q8. What is PYTHONPATH?**

***Ans:***It is an environment variable which is used when a module is imported. Whenever a module is imported, PYTHONPATH is also looked up to check for the presence of the imported modules in various directories. The interpreter uses it to determine which module to load.

**Q9. What are python modules? Name some commonly used built-in modules in Python?**

***Ans:***Python modules are files containing Python code. This code can either be functions classes or variables. A Python module is a .py file containing executable code.

Some of the commonly used built-in modules are:

* os
* sys
* math
* random
* data time
* JSON

**Q10.What are local variables and global variables in Python?**

**Global Variables:**

Variables declared outside a function or in global space are called global variables. These variables can be accessed by any function in the program.

**Local Variables:**

Any variable declared inside a function is known as a local variable. This variable is present in the local space and not in the global space.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | a=2  def add():  b=3  c=a+b  print(c)  add() |

**Output:**5

When you try to access the local variable outside the function add(), it will throw an error.

**Q11. Is python case sensitive?**

***Ans:***Yes. Python is a case sensitive language.

**Q12.What is type conversion in Python?**

***Ans:***Type conversion refers to the conversion of one data type iinto another.

**int()** – converts any data type into integer type

**float()** – converts any data type into float type

**ord()** – converts characters into integer

**hex(**) – converts integers to hexadecimal

**oct()** – converts integer to octal

**tuple() –**This function is used to convert to a tuple.

**set() –**This function returns the type after converting to set.

**list() –**This function is used to convert any data type to a list type.

**dict() –**This function is used to convert a tuple of order (key,value) into a dictionary.

**str() –**Used to convert integer into a string.

**complex(real,imag) –** This functionconverts real numbers to complex(real,imag) number.

**Q13. How to install Python on Windows and set path variable?**

***Ans:***To install Python on Windows, follow the below steps:

* Install python from this link: <https://www.python.org/downloads/>
* After this, install it on your PC. Look for the location where PYTHON has been installed on your PC using the following command on your command prompt: cmd python.
* Then go to advanced system settings and add a new variable and name it as PYTHON\_NAME and paste the copied path.
* Look for the path variable, select its value and select ‘edit’.
* Add a semicolon towards the end of the value if it’s not present and then type %PYTHON\_HOME%

**Q14. Is indentation required in python?**

***Ans:***Indentation is necessary for Python. It specifies a block of code. All code within loops, classes, functions, etc is specified within an indented block. It is usually done using four space characters. If your code is not indented necessarily, it will not execute accurately and will throw errors as well.

**Q15. What is the difference between Python Arrays and lists?**

***Ans:***Arrays and lists, in Python, have the same way of storing data. But, arrays can hold only a single data type elements whereas lists can hold any data type elements.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | import array as arr  My\_Array=arr.array('i',[1,2,3,4])  My\_list=[1,'abc',1.20]  print(My\_Array)  print(My\_list) |

**Output:**

array(‘i’, [1, 2, 3, 4]) [1, ‘abc’, 1.2]

**Q16. What are functions in Python?**

***Ans:***A function is a block of code which is executed only when it is called. To define a [Python function](https://www.edureka.co/blog/python-functions), the **def** keyword is used.

**Example:**

|  |  |
| --- | --- |
| 1  2  3 | def Newfunc():  print("Hi, Welcome to Edureka")  Newfunc(); #calling the function |

**Output:**Hi, Welcome to Edureka

**Q17.What is \_\_init\_\_?**

***Ans:***\_\_init\_\_ is a method or constructor in [Python](https://www.edureka.co/blog/python-programming-language). This method is automatically called to allocate memory when a new object/ instance of a class is created. All classes have the \_\_init\_\_ method.

Here is an example of how to use it.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | class Employee:  def \_\_init\_\_(self, name, age,salary):  self.name = name  self.age = age  self.salary = 20000  E1 = Employee("XYZ", 23, 20000)  # E1 is the instance of class Employee.  #\_\_init\_\_ allocates memory for E1.  print(E1.name)  print(E1.age)  print(E1.salary) |

**Output:**

XYZ

23

20000

**Q18.What is a lambda function?**

***Ans:***An anonymous function is known as a lambda function. This function can have any number of parameters but, can have just one statement.

**Example:**

|  |  |
| --- | --- |
| 1  2 | a = lambda x,y : x+y  print(a(5, 6)) |

**Output:**11

**Q19. What is self in Python?**

***Ans:***Self is an instance or an object of a class. In Python, this is explicitly included as the first parameter. However, this is not the case in Java where it’s optional.  It helps to differentiate between the methods and attributes of a class with local variables.

The self variable in the init method refers to the newly created object while in other methods, it refers to the object whose method was called.

**Q20.** **How does break, continue and pass work?**

|  |  |
| --- | --- |
| Break | Allows loop termination when some condition is met and the control is transferred to the next statement. |
| Continue | Allows skipping some part of a loop when some specific condition is met and the control is transferred to the beginning of the loop |
| Pass | Used when you need some block of code syntactically, but you want to skip its execution. This is basically a null operation. Nothing happens when this is executed. |

**Q21. What does [::-1} do?**

***Ans:*** [::-1] is used to reverse the order of an array or a sequence.

*For example:*

|  |  |
| --- | --- |
| 1  2  3 | import array as arr  My\_Array=arr.array('i',[1,2,3,4,5])  My\_Array[::-1] |

**Output**: array(‘i’, [5, 4, 3, 2, 1])

[::-1] reprints a reversed copy of ordered data structures such as an array or a list. the original array or list remains unchanged.

**Q22. How can you randomize the items of a list in place in Python?**

**Ans:** Consider the example shown below:

|  |  |
| --- | --- |
| 1  2  3  4 | from random import shuffle  x = ['Keep', 'The', 'Blue', 'Flag', 'Flying', 'High']  shuffle(x)  print(x) |

The output of the following code is as below.

['Flying', 'Keep', 'Blue', 'High', 'The', 'Flag']

**Q23. What are python iterators?**

***Ans:***Iterators are objects which can be traversed though or iterated upon.

**Q24. How can you generate random numbers in Python?**

**Ans:** Random module is the standard module that is used to generate a random number. The method is defined as:

|  |  |
| --- | --- |
| 1  2 | 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 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 a different instance of individual threads. The other random generators that are used in this are:

1. 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.
2. uniform(a, b): it chooses a floating point number that is defined in the range of [a,b).Iyt returns the floating point number
3. 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.
4. The Random class that is used and instantiated creates an independent multiple random number generators.

**Q25. What is the difference between range & xrange?**

**Ans:** For the most part, xrange and range are the exact same in terms of functionality. They both provide a way to generate a list of integers for you to use, however you please. The only difference is that range returns a Python list object and x range returns an xrange object.

This means that xrange doesn’t actually generate a static list at run-time like range does. It creates the values as you need them with a special technique called yielding. This technique is used with a type of object known as generators. That means that if you have a really gigantic range you’d like to generate a list for, say one billion, xrange is the function to use.

This is especially true if you have a really memory sensitive system such as a cell phone that you are working with, as range will use as much memory as it can to create your array of integers, which can result in a Memory Error and crash your program. It’s a memory hungry beast.

**Q26. How do you write comments in python?**

***Ans:***Comments in Python start with a # character. However, alternatively at times, commenting is done using docstrings(strings enclosed within triple quotes).

**Example:**

#Comments in Python start like this

print("Comments in Python start with a #")

**Output:**Comments in Python start with a #

**Q27. What is pickling and unpickling?**

**Ans:** Pickle module accepts any Python object and converts it into a string representation and dumps it into a file by using dump function, this process is called pickling. While the process of retrieving original Python objects from the stored string representation is called unpickling.

**Q28. What are the generators in python?**

***Ans:***Functions that return an iterable set of items are called generators.

**Q29. How will you capitalize the first letter of string?**

***Ans:***In Python, the capitalize() method capitalizes the first letter of a string. If the string already consists of a capital letter at the beginning, then, it returns the original string.

**Q30. How will you convert a string to all lowercase?**

***Ans:***To convert a string to lowercase, lower() function can be used.

**Example:**

|  |  |
| --- | --- |
| 1  2 | stg='ABCD'  print(stg.lower()) |

**Output:**abcd

**Q31. How to comment multiple lines in python?**

***Ans:***Multi-line comments appear in more than one line. All the lines to be commented are to be prefixed by a #. You can also a very good **shortcut method to comment multiple lines**. All you need to do is hold the ctrl key and **left click** in every place wherever you want to include a # character and type a # just once. This will comment all the lines where you introduced your cursor.

**Q32.What are docstrings in Python?**

***Ans:***Docstrings are not actually comments, but, they are ***documentation strings***. These docstrings are within triple quotes. They are not assigned to any variable and therefore, at times, serve the purpose of comments as well.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | """  Using docstring as a comment.  This code divides 2 numbers  """  x=8  y=4  z=x/y  print(z) |

**Output:**2.0

**Q33. What is the purpose of is, not and in operators?**

***Ans:***Operators are special functions. They take one or more values and produce a corresponding result.

is: returns true when 2 operands are true  (Example: “a” is ‘a’)

not: returns the inverse of the boolean value

in: checks if some element is present in some sequence

**Q34. What is the usage of help() and dir() function in Python?**

**Ans:** Help() and dir() both functions are accessible from the Python interpreter and used for viewing a consolidated dump of built-in functions.

1. Help() function: The help() function is used to display the documentation string and also facilitates you to see the help related to modules, keywords, attributes, etc.
2. Dir() function: The dir() function is used to display the defined symbols.

**Q35. Whenever Python exits, why isn’t all the memory de-allocated?**

**Ans:**

1. Whenever Python exits, especially those Python modules which are having circular references to other objects or the objects that are referenced from the global namespaces are not always de-allocated or freed.
2. It is impossible to de-allocate those portions of memory that are reserved by the C library.
3. On exit, because of having its own efficient clean up mechanism, Python would try to de-allocate/destroy every other object.

**Q36. What is a dictionary in Python?**

**Ans:** The built-in datatypes in Python is called dictionary. It defines one-to-one relationship between keys and values. Dictionaries contain pair of keys and their corresponding values. Dictionaries are indexed by keys.

Let’s take an example:

The following example contains some keys. Country, Capital & PM. Their corresponding values are India, Delhi and Modi respectively.

|  |  |
| --- | --- |
| 1 | dict={'Country':'India','Capital':'Delhi','PM':'Modi'} |
| 1 | print dict[Country] |

India

|  |  |
| --- | --- |
| 1 | print dict[Capital] |

Delhi

|  |  |
| --- | --- |
| 1 | print dict[PM] |

Modi

**Q37. How can the ternary operators be used in python?**

**Ans:** 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.

**Syntax**:

The Ternary operator will be given as:  
[on\_true] if [expression] else [on\_false]x, y = 25, 50big = x if x < y else y

**Example:**

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.

**Q38. What does this mean: \*args, \*\*kwargs? And why would we use it?**

**Ans:** We use \*args when we aren’t sure how many arguments are going to be passed to a function, or if we want to pass a stored list or tuple of arguments to a function. \*\*kwargs is used when we don’t know how many keyword arguments will be passed to a function, or it can be used to pass the values of a dictionary as keyword arguments. The identifiers args and kwargs are a convention, you could also use \*bob and \*\*billy but that would not be wise.

**Q39. What does len() do?**

***Ans:***It is used to determine the length of a string, a list, an array, etc.

**Example:**

|  |  |
| --- | --- |
| 1  2 | stg='ABCD'  len(stg) |

**Q40. Explain split(), sub(), subn() methods of “re” module in Python.**

**Ans:** To modify the strings, Python’s “re” module is providing 3 methods. They are:

* split() – uses a regex pattern to “split” a given string into a list.
* sub() – finds all substrings where the regex pattern matches and then replace them with a different string
* subn() – it is similar to sub() and also returns the new string along with the no. of replacements.

**Q41. What are negative indexes and why are they used?**

**Ans:** 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.

**Q42.** **What are Python packages?**

***Ans:***Python packages are namespaces containing multiple modules.

**Q43.How can files be deleted in Python?**

***Ans:***To delete a file in Python, you need to import the OS Module. After that, you need to use the os.remove() function.

**Example:**

|  |  |
| --- | --- |
| 1  2 | import os  os.remove("xyz.txt") |

**Q44. What are the built-in types of python?**

***Ans:***Built-in types in Python are as follows –

* Integers
* Floating-point
* Complex numbers
* Strings
* Boolean
* Built-in functions

**Q45. What advantages do NumPy arrays offer over (nested) Python lists?**

**Ans:**

1. Python’s lists are efficient general-purpose containers. They support (fairly) efficient insertion, deletion, appending, and concatenation, and Python’s list comprehensions make them easy to construct and manipulate.
2. They have certain limitations: they don’t support “vectorized” operations like elementwise addition and multiplication, and the fact that they can contain objects of differing types mean that Python must store type information for every element, and must execute type dispatching code when operating on each element.
3. [NumPy](https://www.edureka.co/blog/python-numpy-tutorial/) is not just more efficient; it is also more convenient. You get a lot of vector and matrix operations for free, which sometimes allow one to avoid unnecessary work. And they are also efficiently implemented.
4. NumPy array is faster and You get a lot built in with NumPy, FFTs, convolutions, fast searching, basic statistics, linear algebra, [histograms](https://www.edureka.co/blog/python-matplotlib-tutorial/#Histogram), etc.

**Q46.** **How to add values to a python array?**

***Ans:***Elements can be added to an array using the **append()**, **extend()** and the **insert (i,x)** functions.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | a=arr.array('d', [1.1 , 2.1 ,3.1] )  a.append(3.4)  print(a)  a.extend([4.5,6.3,6.8])  print(a)  a.insert(2,3.8)  print(a) |

**Output:**

array(‘d’, [1.1, 2.1, 3.1, 3.4])

array(‘d’, [1.1, 2.1, 3.1, 3.4, 4.5, 6.3, 6.8])

array(‘d’, [1.1, 2.1, 3.8, 3.1, 3.4, 4.5, 6.3, 6.8])

**Q47. How to remove values to a python array?**

***Ans:***Array elements can be removed using **pop()** or **remove()** method. The difference between these two functions is that the former returns the deleted value whereas the latter does not.

|  |  |
| --- | --- |
| 1  2  3  4  5 | a=arr.array('d', [1.1, 2.2, 3.8, 3.1, 3.7, 1.2, 4.6])  print(a.pop())  print(a.pop(3))  a.remove(1.1)  print(a) |

**Output:**

4.6

3.1

array(‘d’, [2.2, 3.8, 3.7, 1.2])

**Q48.** **Does Python have OOps concepts?**

***Ans:***Python is an object-oriented programming language. This means that any program can be solved in python by creating an object model. However, Python can be treated as procedural as well as structural language.

**Q49. What is the difference between deep and shallow copy?**

***Ans:****Shallow copy* is used when a new instance type gets created and it keeps the values that are copied in the new instance. 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. Shallow copy allows faster execution of the program and it depends on the size of the data that is used.

*Deep copy* is used to store the values that are already copied. Deep copy doesn’t copy the reference pointers to the objects. It 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. Deep copy makes execution of the program slower due to making certain copies for each object that is been called.

**Q50. How is Multithreading achieved in Python?**

**Ans:**

1. Python has a multi-threading package but if you want to multi-thread to speed your code up, then it’s usually not a good idea to use it.
2. Python has a construct called the Global Interpreter Lock (GIL). The GIL makes sure that only one of your ‘threads’ can execute at any one time. A thread acquires the GIL, does a little work, then passes the GIL onto the next thread.
3. This happens very quickly so to the human eye it may seem like your threads are executing in parallel, but they are really just taking turns using the same CPU core.
4. All this GIL passing adds overhead to execution. This means that if you want to make your code run faster then using the threading package often isn’t a good idea.

**Q51.** **What is the process of compilation and linking in python?**

**Ans:** 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 are required in this as:

1. Create a file with any name and in any language that is supported by the compiler of your system. For example file.c or file.cpp
2. Place this file in the Modules/ directory of the distribution which is getting used.
3. Add a line in the file Setup.local that is present in the Modules/ directory.
4. Run the file using spam file.o
5. After a successful run of this rebuild the interpreter by using the make command on the top-level directory.
6. If the file is changed then run rebuildMakefile by using the command as ‘make Makefile’.

**Q52.** **What are Python libraries? Name a few of them.**

Python libraries are a collection of Python packages. Some of the majorly used python libraries are – [Numpy](https://www.edureka.co/blog/python-numpy-tutorial/), [Pandas](https://www.edureka.co/blog/python-pandas-tutorial/), [Matplotlib](https://www.edureka.co/blog/python-matplotlib-tutorial/), [Scikit-learn](https://www.edureka.co/blog/scikit-learn-machine-learning/) and many more.

**Q53. What is split used for?**

The split() method is used to separate a given string in Python.

**Example:**

|  |  |
| --- | --- |
| 1  2 | a="edureka python"  print(a.split()) |

**Output:**[‘edureka’, ‘python’]

**Q54. How to import modules in python?**

Modules can be imported using the **import**keyword.  You can import modules in three ways-

**Example:**

|  |  |
| --- | --- |
| 1  2  3 | import array           #importing using the original module name  import array as arr    # importing using an alias name  from array import \*    #imports everything present in the array module |

**OOPS Interview Questions**

**Q55. Explain Inheritance in Python with an example.**

**Ans:** Inheritance allows One class to gain all the members(say attributes and methods) of another class. Inheritance provides code reusability, makes it easier to create and maintain an application. The class from which we are inheriting is called super-class and the class that is inherited is called a derived / child class.

They are different types of inheritance supported by Python:

1. Single Inheritance – where a derived class acquires the members of a single super class.
2. Multi-level inheritance – a derived class d1 in inherited from base class base1, and d2 are inherited from base2.
3. Hierarchical inheritance – from one base class you can inherit any number of child classes
4. Multiple inheritance – a derived class is inherited from more than one base class.

**Q56. How are classes created in Python?**

**Ans:**Class in Python is created using the **class**keyword.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5 | class Employee:  def \_\_init\_\_(self, name):  self.name = name  E1=Employee("abc")  print(E1.name) |

**Output:**abc

**Q57. What is monkey patching in Python?**

**Ans:** In Python, the term monkey patch only refers to dynamic modifications of a class or module at run-time.

Consider the below example:

|  |  |
| --- | --- |
| 1  2  3  4 | # m.py  class MyClass:  def f(self):  print "f()" |

We can then run the monkey-patch testing like this:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | import m  def monkey\_f(self):  print "monkey\_f()"    m.MyClass.f = monkey\_f  obj = m.MyClass()  obj.f() |

The output will be as below:

monkey\_f()

As we can see, we did make some changes in the behavior of*f()* in *MyClass*using the function we defined, *monkey\_f()*, outside of the module *m*.

**Q58. Does python support multiple inheritance?**

**Ans:**Multiple inheritance means that a class can be derived from more than one parent classes. Python does support multiple inheritance, unlike Java.

**Q59. What is Polymorphism in Python?**

**Ans:**Polymorphism means the ability to take multiple forms. So, for instance, if the parent class has a method named ABC then the child class also can have a method with the same name ABC having its own parameters and variables. Python allows polymorphism.

**Q60. Define encapsulation in Python?**

**Ans:**Encapsulation means binding the code and the data together. A Python class in an example of encapsulation.

**Q61. How do you do data abstraction in Python?**

**Ans:**Data Abstraction is providing only the required details and hiding the implementation from the world. It can be achieved in Python by using interfaces and abstract classes.

**Q62.Does python make use of access specifiers?**

**Ans:**Python does not deprive access to an instance variable or function. Python lays down the concept of prefixing the name of the variable, function or method with a single or double underscore to imitate the behavior of protected and private access specifiers.

**Q63. How to create an empty class in Python?**

**Ans:**An empty class is a class that does not have any code defined within its block. It can be created using the *pass*keyword. However, you can create objects of this class outside the class itself. IN PYTHON THE PASS command does nothing when its executed. it’s a null statement.

**For example-**

|  |  |
| --- | --- |
| 1  2  3  4  5 | class a:    &nbsp; pass  obj=a()  obj.name="xyz"  print("Name = ",obj.name) |

**Output:**

Name = xyz

**Q64. What does an object() do?**

**Ans:**It returns a featureless object that is a base for all classes. Also, it does not take any parameters.

**Basic Python Programs**

**Q65. Write a program in Python to execute the Bubble sort algorithm.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | def bs(a):&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;# a = name of list    &nbsp; b=len(a)-1&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;# minus 1 because we always compare 2 adjacent values    &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;    &nbsp; for x in range(b):    &nbsp; &nbsp; &nbsp; for y in range(b-x):    &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; if a[y]>a[y+1]:    &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; a[y],a[y+1]=a[y+1],a[y]    &nbsp; return a  a=[32,5,3,6,7,54,87]  bs(a) |

**Output:**[3, 5, 6, 7, 32, 54, 87]

**Q66. Write a program in Python to produce Star triangle.**

|  |  |
| --- | --- |
| 1  2  3  4 | def pyfunc(r):      for x in range(r):          print(' '\*(r-x-1)+'\*'\*(2\*x+1))  pyfunc(9) |

**Output:**

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | # Enter number of terms needed&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;#0,1,1,2,3,5....  a=int(input("Enter the terms"))  f=0&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;#first element of series  s=1&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;#second element of series  if a<=0:    &nbsp; print("The requested series is  ",f)  else:    &nbsp; print(f,s,end=" ")    &nbsp; for x in range(2,a):    &nbsp; &nbsp; &nbsp; next=f+s&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;    &nbsp; &nbsp; &nbsp; print(next,end=" ")    &nbsp; &nbsp; &nbsp; f=s    &nbsp; &nbsp; &nbsp; s=next</pre> |

**Q67. Write a program to produce Fibonacci series in Python.**

**Output:** Enter the terms 5 0 1 1 2 3

**Q68. Write a program in Python to check if a number is prime.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | a=int(input("enter number"))&nbsp; &nbsp; &nbsp;  if a>1:    &nbsp; for x in range(2,a):    &nbsp; &nbsp; &nbsp; if(a%x)==0:    &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; print("not prime")    &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; break    &nbsp; else:    &nbsp; &nbsp; &nbsp; print("Prime")  else:    &nbsp; print("not prime") |

**Output:**

enter number 3

Prime

**Q69. Write a program in Python to check if a sequence is a Palindrome.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | a=input("enter sequence")  b=a[::-1]  if a==b:    &nbsp; print("palindrome")  else:    &nbsp; print("Not a Palindrome") |

**Output:**

enter sequence 323 palindrome

**Q70. Write a one-liner that will count the number of capital letters in a file. Your code should work even if the file is too big to fit in memory.**

**Ans:**  Let us first write a multiple line solution and then convert it to one-liner code.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | with open(SOME\_LARGE\_FILE) as fh:  count = 0  text = fh.read()  for character in text:      if character.isupper():  count += 1 |

We will now try to transform this into a single line.

|  |  |
| --- | --- |
| 1 | count sum(1 for line in fh for character in line if character.isupper()) |

**Q71. Write a sorting algorithm for a numerical dataset in Python.**

**Ans:** The following code can be used to sort a list in Python:

|  |  |
| --- | --- |
| 1  2  3  4 | list = ["1", "4", "0", "6", "9"]  list = [int(i) for i in list]  list.sort()  print (list) |

**Q72. Looking at the below code, write down the final values of A0, A1, …An.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | A0 = dict(zip(('a','b','c','d','e'),(1,2,3,4,5)))  A1 = range(10)A2 = sorted([i for i in A1 if i in A0])  A3 = sorted([A0[s] for s in A0])  A4 = [i for i in A1 if i in A3]  A5 = {i:i\*i for i in A1}  A6 = [[i,i\*i] for i in A1]  print(A0,A1,A2,A3,A4,A5,A6) |

**Ans:** The following will be the final outputs of A0, A1, … A6

A0 = {'a': 1, 'c': 3, 'b': 2, 'e': 5, 'd': 4} # the order may vary

A1 = range(0, 10)

A2 = []

A3 = [1, 2, 3, 4, 5]

A4 = [1, 2, 3, 4, 5]

A5 = {0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}

A6 = [[0, 0], [1, 1], [2, 4], [3, 9], [4, 16], [5, 25], [6, 36], [7, 49], [8, 64], [9, 81]]

**Python Libraries Interview Questions**

**Q73. Explain what Flask is and its benefits?**

**Ans:** Flask is a web microframework for Python based on “Werkzeug, Jinja2 and good intentions” BSD license. Werkzeug and Jinja2 are two of its dependencies. This means it will have little to no dependencies on external libraries.  It makes the framework light while there is a little dependency to update and fewer security bugs.

A session basically allows you to remember information from one request to another. In a flask, a session uses a signed cookie so the user can look at the session contents and modify. The user can modify the session if only it has the secret key Flask.secret\_key.

**Q74. Is Django better than Flask?**

**Ans:** Django and Flask map the URL’s or addresses typed in the web browsers to functions in Python.

Flask is much simpler compared to Django but, Flask does not do a lot for you meaning you will need to specify the details, whereas Django does a lot for you wherein you would not need to do much work. [Django](https://www.edureka.co/blog/django-tutorial/) consists of prewritten code, which the user will need to analyze whereas Flask gives the users to create their own code, therefore, making it simpler to understand the code. Technically both are equally good and both contain their own pros and cons.

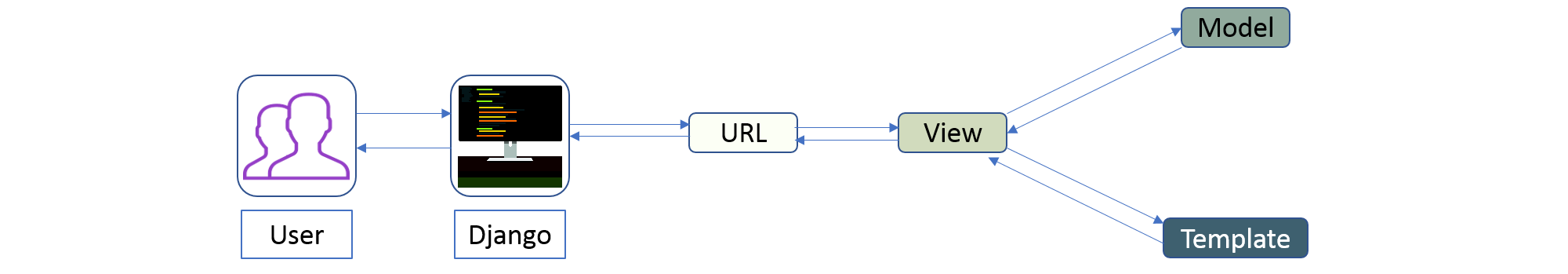
**Q75. Mention the differences between Django, Pyramid and Flask.**

**Ans:**

* Flask is a “microframework” primarily build for a small application with simpler requirements. In flask, you have to use external libraries. Flask is ready to use.
* Pyramid is built for larger applications. It provides flexibility and lets the developer use the right tools for their project. The developer can choose the database, URL structure, templating style and more. Pyramid is heavy configurable.
* Django can also be used for larger applications just like Pyramid. It includes an ORM.

**Q76. Discuss Django architecture.**

**Ans:** Django MVT Pattern:



The developer provides the Model, the view and the template then just maps it to a URL and Django does the magic to serve it to the user.

**Q77. Explain how you can set up the Database in Django.**

**Ans:** You can use the command edit mysite/setting.py, it is a normal python module with module level representing Django settings.

Django uses SQLite by default; it is easy for Django users as such it won’t require any other type of installation. In the case your database choice is different that you have to the following keys in the DATABASE ‘default’ item to match your database connection settings.

* **Engines**: you can change the database by using ‘django.db.backends.sqlite3’ , ‘django.db.backeneds.mysql’, ‘django.db.backends.postgresql\_psycopg2’, ‘django.db.backends.oracle’ and so on
* **Name**: The name of your database. In the case if you are using SQLite as your database, in that case, database will be a file on your computer, Name should be a full absolute path, including the file name of that file.
* If you are not choosing SQLite as your database then settings like Password, Host, User, etc. must be added.

Django uses SQLite as a default database, it stores data as a single file in the filesystem. If you do have a database server—PostgreSQL, MySQL, Oracle, MSSQL—and want to use it rather than SQLite, then use your database’s administration tools to create a new database for your Django project. Either way, with your (empty) database in place, all that remains is to tell Django how to use it. This is where your project’s settings.py file comes in.

We will add the following lines of code to the *setting.py* file:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | DATABASES = {       'default': {            'ENGINE' : 'django.db.backends.sqlite3',            'NAME' : os.path.join(BASE\_DIR, 'db.sqlite3'),       }  } |

**Q78. Give an example how you can write a VIEW in Django?**

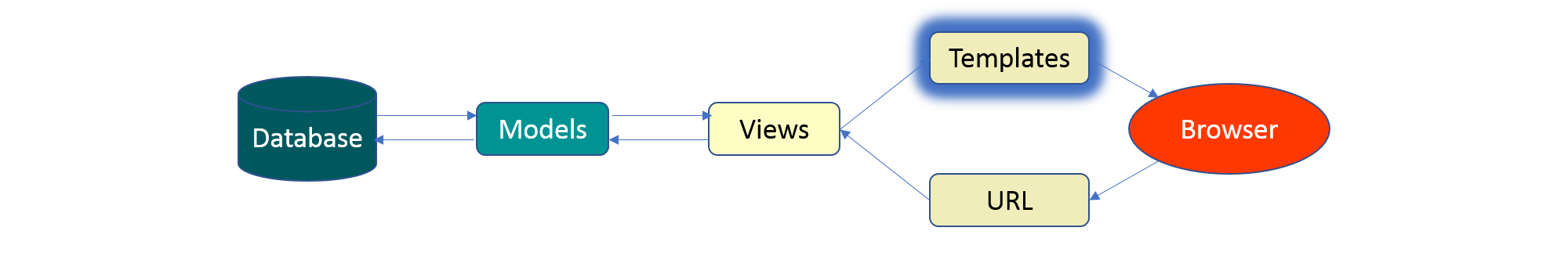
**Ans:** This is how we can use write a view in Django:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | from django.http import HttpResponse  import datetime    def Current\_datetime(request):       now = datetime.datetime.now()       html = "<html><body>It is now %s</body></html> % now       return HttpResponse(html) |

*Returns the current date and time, as an HTML document*

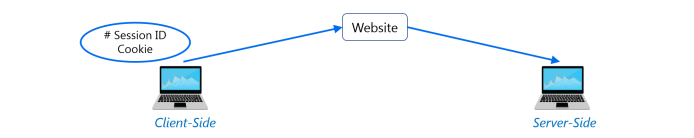
**Q79. Mention what the Django templates consist of.**

**Ans:** The template is a simple text file.  It can create any text-based format like XML, CSV, HTML, etc.  A template contains variables that get replaced with values when the template is evaluated and tags (% tag %) that control the logic of the template.

**Figure:***Python Interview Questions – Django Template*

**Q80. Explain the use of session in Django framework?**

**Ans:** Django provides a session that lets you store and retrieve data on a per-site-visitor basis. Django abstracts the process of sending and receiving cookies, by placing a session ID cookie on the client side, and storing all the related data on the server side.



[BOOK A SLOT](https://www.edureka.co/mymock-interview-service" \t "_blank)

So the data itself is not stored client side. This is nice from a security perspective.

**Q81.  List out the inheritance styles in Django.**

**Ans:** In Django, there are three possible inheritance styles:

1. Abstract Base Classes: This style is used when you only want parent’s class to hold information that you don’t want to type out for each child model.
2. Multi-table Inheritance: This style is used If you are sub-classing an existing model and need each model to have its own database table.
3. Proxy models: You can use this model, If you only want to modify the Python level behavior of the model, without changing the model’s fields.

**Web Scraping – Python Interview Questions**

**Q82. How To Save An Image Locally Using Python Whose URL Address I Already Know?**

**Ans:** We will use the following code to save an image locally from an URL address

|  |  |
| --- | --- |
| 1  2 | import urllib.request  urllib.request.urlretrieve("URL", "local-filename.jpg") |

**Q83. How can you Get the Google cache age of any URL or web page?**

**Ans:** Use the following URL format:

http://webcache.googleusercontent.com/search?q=cache:URLGOESHERE

Be sure to replace “URLGOESHERE” with the proper web address of the page or site whose cache you want to retrieve and see the time for. For example, to check the Google Webcache age of edureka.co you’d use the following URL:

http://webcache.googleusercontent.com/search?q=cache:edureka.co

**Q84. You are required to scrap data from IMDb top 250 movies page. It should only have fields movie name, year, and rating.**

**Ans:** We will use the following lines of code:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | from bs4 import BeautifulSoup    import requests  import sys    url = '<http://www.imdb.com/chart/top>'  response = requests.get(url)  soup = BeautifulSoup(response.text)  tr = soup.findChildren("tr")  tr = iter(tr)  next(tr)    for movie in tr:  title = movie.find('td', {'class': 'titleColumn'} ).find('a').contents[0]  year = movie.find('td', {'class': 'titleColumn'} ).find('span', {'class': 'secondaryInfo'}).contents[0]  rating = movie.find('td', {'class': 'ratingColumn imdbRating'} ).find('strong').contents[0]  row = title + ' - ' + year + ' ' + ' ' + rating    print(row) |

The above code will help scrap data from IMDb’s top 250 list

**Data Analysis – Python Interview Questions**

**Q85. What is map function in Python?**

**Ans:** *map* function executes the function given as the first argument on all the elements of the iterable given as the second argument. If the function given takes in more than 1 arguments, then many iterables are given. #Follow the link to know more similar functions.

**Q86. Is python numpy better than lists?**

**Ans:** We use python numpy array instead of a list because of the below three reasons:

1. Less Memory
2. Fast
3. Convenient

For more information on these parameters, you can refer to this section – [Numpy Vs List](https://www.edureka.co/blog/python-numpy-tutorial/#NumpyVsList).

**Q87. How to get indices of N maximum values in a NumPy array?**

**Ans:** We can get the indices of N maximum values in a NumPy array using the below code:

|  |  |
| --- | --- |
| 1  2  3 | import numpy as np  arr = np.array([1, 3, 2, 4, 5])  print(arr.argsort()[-3:][::-1]) |

Output

[ 4 3 1 ]

**Q88. How do you calculate percentiles with Python/ NumPy?**

**Ans:** We can calculate percentiles with the following code

|  |  |
| --- | --- |
| 1  2  3  4 | import numpy as np  a = np.array([1,2,3,4,5])  p = np.percentile(a, 50) #Returns 50th percentile, e.g. median  print(p) |

Output

3

**Q89. What is the difference between NumPy and SciPy?**

**Ans:**

1. In an ideal world, NumPy would contain nothing but the array data type and the most basic operations: indexing, sorting, reshaping, basic elementwise functions, et cetera.
2. All numerical code would reside in SciPy. However, one of NumPy’s important goals is compatibility, so NumPy tries to retain all features supported by either of its predecessors.
3. Thus NumPy contains some linear algebra functions, even though these more properly belong in SciPy. In any case, SciPy contains more fully-featured versions of the linear algebra modules, as well as many other numerical algorithms.
4. If you are doing scientific computing with python, you should probably install both NumPy and SciPy. Most new features belong in SciPy rather than NumPy.

**Q90. How do you make 3D plots/visualizations using NumPy/SciPy?**

**Ans:** Like 2D plotting, 3D graphics is beyond the scope of NumPy and SciPy, but just as in the 2D case, packages exist that integrate with NumPy. Matplotlib provides basic 3D plotting in the mplot3d subpackage, whereas Mayavi provides a wide range of high-quality 3D visualization features, utilizing the powerful VTK engine.

**Multiple Choice Questions (MCQ)**

**Q91. Which of the following statements create a dictionary? (Multiple Correct Answers Possible)**

a) d = {}  
b) d = {“john”:40, “peter”:45}  
c) d = {40:”john”, 45:”peter”}  
d) d = (40:”john”, 45:”50”)

**Answer:**b, c & d.

Dictionaries are created by specifying keys and values.

**Q92. Which one of these is floor division?**

a) /  
b) //  
c) %  
d) None of the mentioned

**Answer:**b) //

When both of the operands are integer then python chops out the fraction part and gives you the round off value, to get the accurate answer use floor division. For ex, 5/2 = 2.5 but both of the operands are integer so answer of this expression in python is 2. To get the 2.5 as the answer, use floor division using //. So, 5//2 = 2.5

**Q93. What is the maximum possible length of an identifier?**

a) 31 characters  
b) 63 characters  
c) 79 characters  
d) None of the above

**Answer:**d) None of the above

Identifiers can be of any length.

**Q94. Why are local variable names beginning with an underscore discouraged?**

a) they are used to indicate a private variables of a class  
b) they confuse the interpreter  
c) they are used to indicate global variables  
d) they slow down execution

**Answer:**a) they are used to indicate a private variable of a class

As Python has no concept of private variables, leading underscores are used to indicate variables that must not be accessed from outside the class.

**Q95. Which of the following is an invalid statement?**

a) abc = 1,000,000  
b) a b c = 1000 2000 3000  
c) a,b,c = 1000, 2000, 3000  
d) a\_b\_c = 1,000,000

**Answer:**b) a b c = 1000 2000 3000

Spaces are not allowed in variable names.

**Q96. What is the output of the following?**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | try:      if '1' != 1:          raise "someError"      else:          print("someError has not occured")  except "someError":      print ("someError has occured") |

a) someError has occured  
b) someError has not occured  
c) invalid code  
d) none of the above

**Answer:**c) invalid code

A new exception class must inherit from a BaseException. There is no such inheritance here.

**Q97. Suppose list1 is [2, 33, 222, 14, 25], What is list1[-1] ?**

a) Error  
b) None  
c) 25  
d) 2

**Answer:**c) 25

The index -1 corresponds to the last index in the list.

**Q98. To open a file c:scores.txt for writing, we use**

a) outfile = open(“c:scores.txt”, “r”)  
b) outfile = open(“c:scores.txt”, “w”)  
c) outfile = open(file = “c:scores.txt”, “r”)  
d) outfile = open(file = “c:scores.txt”, “o”)

**Answer:**b) The location contains double slashes ( ) and w is used to indicate that file is being written to.

**Q99. What is the output of the following?**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | f = None    for i in range (5):      with open("data.txt", "w") as f:          if i > 2:              break    print f.closed |

a) True  
b) False  
c) None  
d) Error

**Answer:**a) True

The WITH statement when used with open file guarantees that the file object is closed when the with block exits.

**Q100. When will the else part of try-except-else be executed?**

a) always  
b) when an exception occurs  
c) when no exception occurs  
d) when an exception occurs into except block

**Answer:**c) when no exception occurs

**1) What is Python? What are the benefits of using Python?**

Python is a programming language with objects, modules, threads, exceptions and automatic memory management. The benefits of pythons are that it is simple and easy, portable, extensible, build-in data structure and it is an open source.

**2) What is PEP 8?**

PEP 8 is a coding convention, a set of recommendation, about how to write your Python code more readable.

**3) What is pickling and unpickling?**

Pickle module accepts any Python object and converts it into a string representation and dumps it into a file by using dump function, this process is called pickling. While the process of retrieving original Python objects from the stored string representation is called unpickling.

**4) How Python is interpreted?**

Python language is an interpreted language. Python program runs directly from the source code. It converts the source code that is written by the programmer into an intermediate language, which is again translated into machine language that has to be executed.

**5) How memory is managed in Python?**

* Python memory is managed by Python private heap space. All Python objects and data structures are located in a private heap. The programmer does not have an access to this private heap and interpreter takes care of this Python private heap.
* The allocation of Python heap space for Python objects is done by Python memory manager. The core API gives access to some tools for the programmer to code.
* Python also have an inbuilt garbage collector, which recycle all the unused memory and frees the memory and makes it available to the heap space.

**6) What are the tools that help to find bugs or perform static analysis?**

PyChecker is a static analysis tool that detects the bugs in Python source code and warns about the style and complexity of the bug. Pylint is another tool that verifies whether the module meets the coding standard.

**7) What are Python decorators?**

A Python decorator is a specific change that we make in Python syntax to alter functions easily.

**8) What is the difference between list and tuple?**

The difference between list and tuple is that list is mutable while tuple is not. Tuple can be hashed for e.g as a key for dictionaries.

**9) How are arguments passed by value or by reference?**

Everything in Python is an object and all variables hold references to the objects. The references values are according to the functions; as a result you cannot change the value of the references. However, you can change the objects if it is mutable.

**10) What is Dict and List comprehensions are?**

They are syntax constructions to ease the creation of a Dictionary or List based on existing iterable.

**11) What are the built-in type does python provides?**

There are mutable and Immutable types of Pythons built in types Mutable built-in types

* List
* Sets
* Dictionaries

Immutable built-in types

* Strings
* Tuples
* Numbers

**12) What is namespace in Python?**

In Python, every name introduced has a place where it lives and can be hooked for. This is known as namespace. It is like a box where a variable name is mapped to the object placed. Whenever the variable is searched out, this box will be searched, to get corresponding object.

**13) What is lambda in Python?**

It is a single expression anonymous function often used as inline function.

**14) Why lambda forms in python does not have statements?**

A lambda form in python does not have statements as it is used to make new function object and then return them at runtime.

**15) What is pass in Python?**

Pass means, no-operation Python statement, or in other words it is a place holder in compound statement, where there should be a blank left and nothing has to be written there.

**16) In Python what are iterators?**

In Python, iterators are used to iterate a group of elements, containers like list.

**17) What is unittest in Python?**

A unit testing framework in Python is known as unittest. It supports sharing of setups, automation testing, shutdown code for tests, aggregation of tests into collections etc.

**18) In Python what is slicing?**

A mechanism to select a range of items from sequence types like list, tuple, strings etc. is known as slicing.

**19) What are generators in Python?**

The way of implementing iterators are known as generators. It is a normal function except that it yields expression in the function.

**20) What is docstring in Python?**

A Python documentation string is known as docstring, it is a way of documenting Python functions, modules and classes.

**21) How can you copy an object in Python?**

To copy an object in Python, you can try copy.copy () or copy.deepcopy() for the general case. You cannot copy all objects but most of them.

**22) What is negative index in Python?**

Python sequences can be index in positive and negative numbers. For positive index, 0 is the first index, 1 is the second index and so forth. For negative index, (-1) is the last index and (-2) is the second last index and so forth.

**23) How you can convert a number to a string?**

In order to convert a number into a string, use the inbuilt function str(). If you want a octal or hexadecimal representation, use the inbuilt function oct() or hex().

**24) What is the difference between Xrange and range?**

Xrange returns the xrange object while range returns the list, and uses the same memory and no matter what the range size is.

**25) What is module and package in Python?**

In Python, module is the way to structure program. Each Python program file is a module, which imports other modules like objects and attributes.

The folder of Python program is a package of modules. A package can have modules or subfolders.

**26) Mention what are the rules for local and global variables in Python?**

**Local variables**: If a variable is assigned a new value anywhere within the function's body, it's assumed to be local.

**Global variables**: Those variables that are only referenced inside a function are implicitly global.

**27) How can you share global variables across modules?**

To share global variables across modules within a single program, create a special module. Import the config module in all modules of your application. The module will be available as a global variable across modules.

**28) Explain how can you make a Python Script executable on Unix?**

To make a Python Script executable on Unix, you need to do two things,

* Script file's mode must be executable and
* the first line must begin with # ( #!/usr/local/bin/python)

**29) Explain how to delete a file in Python?**

By using a command os.remove (filename) or os.unlink(filename)

**30) Explain how can you generate random numbers in Python?**

To generate random numbers in Python, you need to import command as

import random

random.random()

This returns a random floating point number in the range [0,1)

**31) Explain how can you access a module written in Python from C?**

You can access a module written in Python from C by following method,

Module = =PyImport\_ImportModule("<modulename>");

**32) Mention the use of // operator in Python?**

It is a Floor Divisionoperator , which is used for dividing two operands with the result as quotient showing only digits before the decimal point. For instance, 10//5 = 2 and 10.0//5.0 = 2.0.

**33) Mention five benefits of using Python?**

* Python comprises of a huge standard library for most Internet platforms like Email, HTML, etc.
* Python does not require explicit memory management as the interpreter itself allocates the memory to new variables and free them automatically
* Provide easy readability due to use of square brackets
* Easy-to-learn for beginners
* Having the built-in data types saves programming time and effort from declaring variables

**34) Mention the use of the split function** **in Python**?

The use of the split function in Python is that it breaks a string into shorter strings using the defined separator. It gives a list of all words present in the string.

**35) Explain what is Flask & its benefits**?

Flask is a web micro framework for Python based on "Werkzeug, Jinja 2 and good intentions" BSD licensed. Werkzeug and jingja are two of its dependencies.

Flask is part of the micro-framework. Which means it will have little to no dependencies on external libraries. It makes the framework light while there is little dependency to update and less security bugs.

**36) Mention what is the difference between Django, Pyramid, and Flask?**

Flask is a "microframework" primarily build for a small application with simpler requirements. In flask, you have to use external libraries. Flask is ready to use.

Pyramid are build for larger applications. It provides flexibility and lets the developer use the right tools for their project. The developer can choose the database, URL structure, templating style and more. Pyramid is heavy configurable.

Like Pyramid, Django can also used for larger applications. It includes an ORM.

**37) Mention what is Flask-WTF and what are their features?**

Flask-WTF offers simple integration with WTForms. Features include for Flask WTF are

* Integration with wtforms
* Secure form with csrf token
* Global csrf protection
* Internationalization integration
* Recaptcha supporting
* File upload that works with Flask Uploads

**38) Explain what is the common way for the Flask script to work?**

The common way for the flask script to work is

* Either it should be the import path for your application
* Or the path to a Python file

**39) Explain how you can access sessions in Flask?**

A session basically allows you to remember information from one request to another. In a flask, it uses a signed cookie so the user can look at the session contents and modify. The user can modify the session if only it has the secret key Flask.secret\_key.

**40) Is Flask an MVC model and if yes give an example showing MVC pattern for your application?**

Basically, Flask is a minimalistic framework which behaves same as MVC framework. So MVC is a perfect fit for Flask, and the pattern for MVC we will consider for the following example

|  |  |
| --- | --- |
| from flask import Flask  app = Flask(\_name\_)  @app.route("/")  Def hello():  return "Hello World"  app.run(debug = True) | In this code your,   * Configuration part will be   from flask import Flask  app = Flask(\_name\_)   * View part will be   @app.route("/")  Def hello():  return "Hello World"   * While you model or main part will be   app.run(debug = True) |

**41) Explain database connection in Python Flask?**

Flask supports database powered application (RDBS). Such system requires creating a schema, which requires piping the shema.sql file into a sqlite3 command. So you need to install sqlite3 command in order to create or initiate the database in Flask.

Flask allows to request database in three ways

* before\_request() : They are called before a request and pass no arguments
* after\_request() : They are called after a request and pass the response that will be sent to the client
* teardown\_request(): They are called in situation when exception is raised, and response are not guaranteed. They are called after the response been constructed. They are not allowed to modify the request, and their values are ignored.

**42) You are having multiple Memcache servers running Python, in which one of the memcacher server fails, and it has your data, will it ever try to get key data from that one failed server?**

The data in the failed server won't get removed, but there is a provision for auto-failure, which you can configure for multiple nodes. Fail-over can be triggered during any kind of socket or Memcached server level errors and not during normal client errors like adding an existing key, etc.

**43) Explain how you can minimize the Memcached server outages in your Python Development?**

* When one instance fails, several of them goes down, this will put larger load on the database server when lost data is reloaded as client make a request. To avoid this, if your code has been written to minimize cache stampedes then it will leave a minimal impact
* Another way is to bring up an instance of Memcached on a new machine using the lost machines IP address
* Code is another option to minimize server outages as it gives you the liberty to change the Memcached server list with minimal work
* Setting timeout value is another option that some Memcached clients implement for Memcached server outage. When your Memcached server goes down, the client will keep trying to send a request till the time-out limit is reached

**44) Explain what is Dogpile effect? How can you prevent this effect?**

Dogpile effect is referred to the event when cache expires, and websites are hit by the multiple requests made by the client at the same time. This effect can be prevented by using semaphore lock. In this system when value expires, first process acquires the lock and starts generating new value.

**45) Explain how Memcached should not be used in your Python project?**

* Memcached common misuse is to use it as a data store, and not as a cache
* Never use Memcached as the only source of the information you need to run your application. Data should always be available through another source as well
* Memcached is just a key or value store and cannot perform query over the data or iterate over the contents to extract information
* Memcached does not offer any form of security either in encryption or authentication

**Top Answers to Python Interview Questions**

**1. Compare between Java and Python.**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Java** | **Python** |
| Ease of use | Good | Very Good |
| Speed of coding | Average | Excellent |
| Data types | Static typed | Dynamically typed |
| Data Science & machine learning applications | Average | Very Good |

**2. What is Python?**

[Python](https://intellipaat.com/blog/python-certification/) is a high-level, interpreted, interactive, and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently, whereas other languages use punctuation, and it has fewer syntactical constructions than other languages.

**3. What are the key features of Python?**

* Python is an interpreted language, so it doesn’t need to be compiled before execution unlike languages like C.
* Python is dynamically typed, so there is no need to declare a variable with the data type. Python Interpreter will identify the data type on the basis of the value of the variable.

For example, if you run the following code line, it will run without any error:

a = 100

a = "Intellipaat"

* Python follows object-oriented programming paradigm with an exception of having access specifiers. Other than access specifiers (public and private keywords), Python has classes, inheritance, and all other usual OOPs concepts.
* Python is a cross-platform language which means that a Python program written on a Windows system will also run on a Linux system with little or no modifications at all.
* Python is literally a general-purpose language which means that Python finds its way in various domains such as Web Application Development, Automation, Data Science, Machine Learning, and more.

**Check out this video on Python Tutorial**

Learn for free ! Subscribe to our youtube Channel.

**4. What is the purpose of PYTHONPATH environment variable?**

PYTHONPATH has a role similar to PATH. This variable tells Python Interpreter where to locate the module files imported into a program. It should include Python source library directory and the directories containing Python source code. PYTHONPATH is sometimes preset by Python Installer.

**5. What is the purpose of PYTHONSTARTUP, PYTHONCASEOK, and PYTHONHOME environment variables?**

* PYTHONSTARTUP: It contains the path of an initialization file containing Python source code. It is executed every time you start the interpreter. It is named as .pythonrc.py in Unix, and it contains commands that load utilities or modify PYTHONPATH.
* PYTHONCASEOK: It is used in Windows to instruct Python to find the first case-insensitive match in an import statement. Set this variable with any value to activate it.
* PYTHONHOME: It is an alternative module search path. It is usually embedded in PYTHONSTARTUP or PYTHONPATH directories to make switching of module libraries easy.

***Go through the***[***Python Course in London***](https://intellipaat.com/python-certification-training-online-london/)***to get clear understanding of Python.***

**6. What are the supported data types in Python?**

Python has five standard data types:

* Numbers
* Strings
* Lists
* Tuples
* Dictionaries

**7. What is the difference between lists and tuples?**

|  |  |
| --- | --- |
| **Lists** | **Tuples** |
| Lists are mutable, i.e., they can be edited. | Tuples are immutable (Tuples are lists which cannot be edited). |
| Lists are usually slower than tuples. | Tuples are faster than lists. |
| Syntax: list\_1 = [10, ‘Intellipaat’, 20] | Syntax: tup\_1 = (10, ‘Intellipaat’ , 20) |

**8. How is memory managed in Python?**

* Memory in Python is managed by Python private heap space. All Python objects and data structures are located in a private heap. This private heap is taken care of by Python Interpreter itself, and a programmer doesn’t have any access to this private heap.
* Python memory manager takes care of the allocation of Python private heap space.
* Memory for Python private heap space is made available by Python’s inbuilt garbage collector which recycles and frees up all the unused memory.

**9. Explain Inheritance in Python with an example.**

As Python follows an object-oriented programming paradigm, classes in Python have the ability of inheriting the properties of another class. This process is known as inheritance. Inheritance provides the code reusability feature. The class that is being inherited is called a super-class and the class that inherits the other class is called a derived or child class. Following types of inheritance are supported in Python:

1. Single inheritance: When a class inherits only one super class
2. Multiple inheritance: When a class inherits multiple super classes
3. Multilevel inheritance: When a class inherits a super class and then another class inherits this derived class forming a ‘parent, child, and grandchild’ class structure
4. Hierarchical inheritance: When one super class is inherited by multiple derived classes

***Want to become master in Python programming check out this***[***Python training***](https://intellipaat.com/python-for-data-science-training/)***?***

**10. What is a dictionary in Python?**

Python dictionary is one of the supported [data types in Python](https://intellipaat.com/tutorial/python-tutorial/datatypes-in-python/). It is an unordered collection of elements. The elements in dictionaries are stored as key–value pairs. Dictionaries are indexed by keys.

For example, we have a dictionary named ‘dict’. It contains some keys: Country, Capital along with their corresponding values: India and New Delhi.

dict={‘Country’:’India’,’Capital’:’New Delhi’, }

**11. Can you write an efficient code to count the number of capital letters in a file?**

The normal solution for this problem statement would be as follows:

with open(SOME\_LARGE\_FILE) as countletter:

count = 0

text = countletter.read()

for character in text:

if character.isupper():

count += 1

To make this code more efficient, the whole code block can be converted into a one-liner code using generator expression. Then, the equivalent code line for the above code block would be:

count sum(1 for line in countletter for character in line if character.isupper())

**12. Write a code to sort a numerical list in Python.**

The following code can be used to sort a numerical list in Python:

list = [“2”, “5”, “7”, “8”, “1”]

list = [int(i) for i in list]

list.sort()

print (list)

**13. How will you reverse a list in Python?**

list.reverse(): This function reverses objects of list.

**14. How will you remove the last object from a list?**

list.pop(obj=list[-1]):

Here, ‘−1’ represents the last element of the list. The pop function removes and returns the last object (or obj) from the list.

**15. What are negative indexes and why are they used?**

To access an element from ordered sequences, we simply use the index of the element, which is the number of the position of that element. The index usually starts from 0, meaning that the first element has the index 0 and the second has 1, and so on.

When we use the index to access elements from the end of a list, it’s called reverse indexing. In reverse indexing, the indexing of elements start from the last element with the index number being ‘−1’. The second last element has the index ‘−2’, and so on. These indexes used in reverse indexing are called negative indexes.

**16. What are split(), sub(), and subn() methods in Python?**

These methods belong to Python RegEx ‘re’ module and are used to modify strings.

* split(): This method is used to split a given string into a list.
* sub(): This method is used to find a substring where a regex pattern matches and then it replaces that matched substring with a different string.
* subn(): This method is similar to the sub() method, but it also returns the new string, along with the number of replacements.

***Learn more about Python in this***[***Python training in New York***](https://intellipaat.com/python-certification-training-online-new-york/)***to get ahead in your career!***

**17. How are range and xrange different from one another?**

Functions in Python, range() and xrange() are used to iterate a fixed number of times in a for loop. Functionality-wise, both of these functions are the same. Difference comes when talking about Python version support for these functions and their return values.

|  |  |
| --- | --- |
| **The range() Method** | **The xrange() Method** |
| In Python 3, xrange() is not supported; instead the range() function is used to iterate in for loops. | The xrange() function is used in Python 2 to iterate in for loops. |
| It returns a list. | It returns a generator object as it doesn’t really generate a static list at the run time. |
| It takes more memory as it keeps the entire list of iterating numbers in memory. | It takes less memory as it keeps only one number at a time in memory. |

**18. Define pickling and unpickling in Python.**

Pickling is the process of converting Python objects such as lists, dicts, etc. into a character stream. This is done using a module named pickle, hence the name pickling.

The process of retrieving the original Python object from the stored string representation, which is the reverse of the pickling process, is called unpickling.

**19. What is a map function in Python?**

The map() function in Python has two parameters, function and iterable. The map() function takes a function as an argument and then applies that function to all the elements of an iterable, passed to it as another argument. It returns an object list of results.

For example:

def calculateSq(n):

return n\*n

numbers = (2, 3, 4, 5)

result = map( calculateSq, numbers)

print(result)

**20. Write a code to get indices of N maximum values in a NumPy array.**

We can get the indices of N maximum values in a NumPy array using the below code:

import numpy as np

ar = np.array([1, 3, 2, 4, 5, 6])

print(ar.argsort()[-3:][::-1])

***Interested in learning Python? Click here to learn more in this***[***Python Training in Sydney***](https://intellipaat.com/python-certification-training-online-sydney/)***!***

**21. What is a Python module?**

Modules are independent Python scripts with .py extension that can be reused in other Python codes or scripts using the import statement. A module can consist of functions, classes, and variables, or some runnable code. Modules not only help in keeping Python codes organized but also in making codes less complex and more efficient. The syntax to import modules in Python codes is as follows:

import module\_name   # include this code line on top of the script

**Watch this Python Projects Video for Beginners:**

**22. What do file-related modules in Python do? Can you name some file-related modules in Python?**

Python comes with some file-related modules that have functions to manipulate text files and binary files on a file system. These modules can be used to create text or binary files, update their content, copy, delete, and more.

Some file-related modules are os, os.path, and shutil.os. The os.path module has functions to access the file system, while the shutil.os module can be used to copy or delete files.

**23. Explain the use of with statement and its syntax.**

In Python, using the ‘with’ statement, we can open a file and close it as soon as the block of code, where ‘with’ is used, exits, without having to use the close() method.

with open("filename", "mode") as file\_var:

**24. Explain all file processing modes supported in Python.**

Python has various file processing modes.

For opening files, there are three modes:

* read-only mode (r)
* write-only mode (w)
* read–write mode (rw)

To open a text file using these modes, we will have to append ‘t’ with them as follows:

* read-only mode (rt)
* write-only mode (wt)
* read–write mode (rwt)

Similarly, a binary file can be opened by appending ‘b’ with them as follows:

* read-only mode (rb)
* write-only mode (wb)
* read–write mode (rwb)

To append content in files, we can use the append mode (a). Again, for text files, the mode would be ‘at’, and for binary files it would be ‘ab’.

***Become Master of Python by going through this online***[***Python course in Toronto***](https://intellipaat.com/python-certification-training-online-toronto/)***.***

**25. Is indentation optional in Python?**

Indentation in Python is compulsory and is part of its syntax. All programming languages have some way of defining the scope and extent of the block of codes; in Python, it is indentation. Indentation provides better readability to the code, which is probably why Python has made it compulsory.

**26. How are Python arrays and Python lists different from each other?**

In the case of Python, when people say arrays, they are usually talking about lists. It is because lists are fundamental to Python just as arrays are fundamental to most of the low-level languages. But, there is indeed a module named array in Python which is used or mentioned very rarely. Following are some of the differences between arrays and lists.

|  |  |
| --- | --- |
| **Arrays** | **Lists** |
| Arrays can only store homogeneous data (data of the same type). | Lists can store heterogenous and arbitrary data. |
| Since only one type of data can be stored, arrays use memory for only one type of objects. Thus, mostly, arrays use lesser memory than lists. | Lists can store data of multiple data types and thus require more memory than arrays. |
| Length of an array is pre-fixed while creating it, so more elements cannot be added. | Since the length of a list is not fixed, appending items in it is possible. |

**27. Write a code to display the contents of a file in reverse.**

for line in reversed(list(open(filename.txt))):

print(line.rstrip())

**28. Differentiate between NumPy and SciPy.**

|  |  |
| --- | --- |
| **NumPy** | **SciPy** |
| NumPy stands for Numerical Python. | SciPy stands for Scientific Python. |
| It is used for efficient and general numeric computations on numerical data saved in arrays. For example, sorting, indexing, reshaping, and more. | This module is a collection of tools in Python to perform operations such as integration, differentiation, and more. |
| Except for the general computing, there are some linear algebraic functions available in this module, but they are not fully fledged. | Fully fledged algebraic functions are available in SciPy for algebraic computations. |

**29. Which of the following is an invalid statement?**

1. a) xyz = 1,000,000
2. b) x y z = 1000 2000 3000
3. c) x,y,z = 1000, 2000, 3000
4. d) x\_y\_z = 1,000,000

Answer: b

**30. Can we make multi-line comments in Python?**

Python does not have a specific syntax for including [multi-line comments](https://intellipaat.com/community/2791/way-to-create-multiline-comments-in-python?show=2791#q2791) like other programming languages, but programmers can use triple-quoted strings (docstrings) for making multi-line comments, as when docstring is not being used as the first statement inside a method, it gets ignored by Python parser.

**31. What would be the output if I run the following code block?**

list1 = [2, 33, 222, 14, 25]  
print(list1[-2])

(A)          14  
(B)          33  
(C)          25  
(D)          Error

Ans: 14

**32. Write a command to open the file c:\hello.txt for writing.**

f= open(“hello.txt”, “wt”)

**33. What is \_\_init\_\_ in Python?**

\_\_init\_\_ is a reserved method in Python classes which is equivalent to constructors in OOP terminology. The \_\_init\_\_ method is called automatically whenever a new object is initiated. The \_\_init\_\_ method allocates memory to the new object as soon as it is created. This method can also be used to initialize variables.

**Video**

**Check out this video on Python for Data Science**

Learn for free ! Subscribe to our youtube Channel.

**34. What do you understand by Tkinter?**

Tkinter is an inbuilt Python module that is used to create GUI applications. It’s Python’s standard toolkit for GUI development. Tkinter comes with Python, so there is no installation needed. We can start using it by importing it in our script.

**35. Is Python fully object oriented?**

Python does follow an object-oriented programming paradigm and has all the [basic OOPs concepts](https://intellipaat.com/tutorial/java-tutorial/oops-concepts-in-java/) such as inheritance, polymorphism, and more, with the exception of access specifiers. Python doesn’t support strong encapsulation (adding private keyword before data members) although it does have a convention that can be used for data hiding, i.e., prefixing data member with two underscores.

**36. What is lambda function in Python?**

Lambda function is an anonymous function (functions that don’t have names) in Python. To define anonymous functions, we use the lambda keyword instead of the def keyword, hence the name ‘lambda function’. Lambda functions can have any number of arguments but can have only one statement.

**37. What is self-keyword in Python?**

Self-keyword is used as the first parameter of a function inside a class that represents the instance of the class. The object or the instance of the class is automatically passed to the method that it belongs to and is received in the ‘self-keyword’. Users can use another name for the first parameter of the function that catches the object of the class, but it is recommended to use the self-keyword only as it is more of a Python convention.

**38. What are control flow statements in Python?**

[Control flow statements](https://intellipaat.com/tutorial/python-tutorial/python-control-flow-statements/) are used to manipulate or change the execution flow of a program. Generally, the flow of execution of a program runs from top to bottom, but certain statements in Python can break this top to bottom order of execution. Control flow statements include decision-making, looping, and more.

**39. What is the difference between append() and extend() methods?**

Both append() and extend() methods are methods used for lists. These methods are used to add elements at the end of a list.

append(element): Adds the given element at the end of the list which called this method

extend(another-list): Adds the elements of another-list at the end of the list which called the extend method

**40. What are loop interruption statements in Python?**

There are two types of loop interruption statements in Python that let users terminate a loop iteration prematurely, i.e., without letting the loop run its full iterations.

Following are the two loop interruption statements:

* Python break statement: This statement immediately terminates the loop entirely, and the control flow of the program is shifted directly to the outside of the loop.
* Python continue statement: Continue statement terminates the current loop iteration and moves the control flow of the program to the next iteration of the loop, letting the user skip only the current iteration.

**41. What is docstring in Python?**

Python lets the user include a description or quick notes for their methods using documentation strings or docstrings. Docstrings are different from regular comments in Python as rather than being completely ignored by Python Interpreter like in the case of comments Python strings can actually be accessed at the run time using the dot operator when docstring is the first statement in a method or function.

***Are you interested in learning***[***Python course in Bangalore***](https://intellipaat.com/python-certification-training-online-bangalore/)***from Experts?***

**42. What is the output of the following?**

x = [‘ab’, ‘cd’]

print(len(list(map(list, x))))

Output: [[‘a’, ‘b’], [‘c’, ‘d’]].

Explanation: Each element of x is converted into list

**43. Which one of the following is not the correct syntax for creating a set?**

1. (a) set([[1,2],[3,4],[4,5]])(b) set([1,2,2,3,4,5])(c) {1,2,3,4}(d) set((1,2,3,4))Answer: (a)Explanation: The argument given for the set must be an iterable.

**44. What is functional programming? Does Python follow a functional programming style? If yes, list a few methods to implement functionally oriented programming in Python.**

Functional programming is a coding style where the main source of logic in a program comes from functions.

Incorporating functional programming in our codes means writing pure functions.

Pure functions are functions that cause little or no changes outside the scope of the function. These changes are referred to as side effects. To reduce side effects, pure functions are used, which makes the code easy to follow, test, or debug.

Python does follow a functional programming style. Following are some of the examples of functional programming in Python.

* 1. filter(): Filter lets us filter some values based on a conditional logic.

>>> list(filter(lambda x:x>6,range(9))) [7, 8]

* 1. map(): Map applies a function to every element in an iterable.

>>> list(map(lambda x:x\*\*2,range(5))) [0, 1, 4, 9, 16, 25]

* 1. reduce(): Reduce repeatedly reduces a sequence pair-wise until it reaches a single value.

>>> from functools import reduce >>> reduce(lambda x,y:x-y,[1,2,3,4,5]) -13

**45. How does Python Flask handle database requests?**

Flask supports a database-powered application (RDBS). Such a system requires creating a schema, which needs piping the schema.sql file into a sqlite3 command. So, you need to install the sqlite3 command in order to create or initiate the database in Flask.

Flask allows to request for database in three ways:

* before\_request(): They are called before a request and pass no arguments.
* after\_request(): They are called after a request and pass the response that will be sent to the client.
* teardown\_request(): They are called in a situation when an exception is raised, and responses are not guaranteed. They are called after the response has been constructed. They are not allowed to modify the request, and their values are ignored.

***Get certified from top***[***Python course in Singapore.***](https://intellipaat.com/python-certification-training-online-singapore/)***Now***

**46. Write a Python program to check whether a given string is a palindrome or not, without using an iterative method. Note: A palindrome is a word, phrase, or sequence that reads the same backward as forward, e.g., madam, nurses run, etc.**

def fun(string):

s1 = string

s = string[::-1]

if(s1 == s):

return true

else:

return false

print(fun(“madam”))

**47. Write a Python program to calculate the sum of a list of numbers.**

def sum(num):

if len(num) == 1:

return num[0]               #with only one element in the list, sum result will be equal to the element.

else:

return num[0] + sum(num[1:])

print(sum([2, 4, 5, 6, 7]))

Sample Output:

24

**48. Do we need to declare variables with data types in Python?**

No. Python is a dynamically typed language, which means that Python Interpreter automatically identifies the data type of a variable based on the type of value assigned to the variable.

***Learn Complete***[***Python training at Hyderabad***](https://intellipaat.com/python-certification-training-online-hyderabad/)***in 24 Hrs.***

**49. How will you read a random line in a file?**

We can read a random line in a file using a module named ‘random’.

For example:

import random

def read\_random(fname):

lines = open(fname).read().splitlines()

return random.choice(lines)

print(read\_random (‘hello.txt’))

**50. Write a Python program to count the total number of lines in a text file.**

def file\_count(fname):

with open(fname) as f:

for i, 1 in enumerate(f):

paas

return i+1

print(“Total number of lines in the text file: ”, file\_count(“file.txt”))

**INTRODUCTION TO OBJECT-ORIENTED PROGRAMMING IN PYTHON**

Object Oriented Programming is a way of computer programming using the idea of “[objects](https://www.edureka.co/blog/python-class/#Objects)” to represents data and methods. It is also, an approach used for creating neat and reusable code instead of a redundant one. the program is divided into self-contained objects or several mini-programs. Every Individual object represents a different part of the application having its own logic and data to communicate within themselves.

Now, to get a more clear picture of why we use oops instead of pop, I have listed down the differences below.

**Difference between Object-Oriented and Procedural Oriented Programming**

|  |  |
| --- | --- |
| **Object-Oriented Programming (OOP)** | **Procedural-Oriented Programming (Pop)** |
| It is a bottom-up approach | It is a top-down approach |
| Program is divided into objects | Program is divided into functions |
| Makes use of *Access modifiers*  ‘public’, private’, protected’ | Doesn’t use *Access modifiers* |
| It is more secure | It is less secure |
| Object can move freely within member functions | Data can move freely from function to function within programs |
| It supports inheritance | It does not support inheritance |

That was all about the differences, moving ahead let’s get an idea of classes and objects.

**What are Classes and Objects?**

A class is a collection of objects or you can say it is a blueprint of objects defining the common attributes and behavior. Now the question arises, how do you do that?

Well, it logically groups the data in such a way that code reusability becomes easy. I can give you a real-life example- think of an office going ’employee’ as a class and all the attributes related to it like ’emp\_name’, ’emp\_age’, ’emp\_salary’, ’emp\_id’ as the objects in [Python](https://www.edureka.co/blog/python-programming-language). Let us see from the coding perspective that how do you instantiate a class and an object.

Class is defined under a “Class” Keyword.  
**Example:**

|  |  |
| --- | --- |
| 1 | class class1(): // class 1 is the name of the class |

**Note:**Python is not case-sensitive.

**Objects:**

Objects are an instance of a class. It is an entity that has state and behavior. In a nutshell, it is an instance of a class that can access the data.

**Syntax:**obj = class1()

Here obj is the “object “ of class1.

**Creating an Object and Class in python:**

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | class employee():      def \_\_init\_\_(self,name,age,id,salary):   //creating a function          self.name = name // self is an instance of a class          self.age = age          self.salary = salary          self.id = id    emp1 = employee("harshit",22,1000,1234) //creating objects  emp2 = employee("arjun",23,2000,2234)  print(emp1.\_\_dict\_\_)//Prints dictionary |

**Explanation:**’emp1′ and ’emp2′ are the objects that are instantiated against the class ’employee’.Here, the word (\_\_dict\_\_) is a “dictionary” which prints all the values of object ‘emp1’ against the given parameter (name, age, salary).(\_\_init\_\_) acts like a constructor that is invoked whenever an object is created.

I hope now you guys won’t face any problem while dealing with ‘classes’ and ‘objects’ in the future.

With this, let me take you through a ride of [Object Oriented Programming](https://www.edureka.co/blog/python-class/) methodologies:

**Object-Oriented Programming methodologies:**

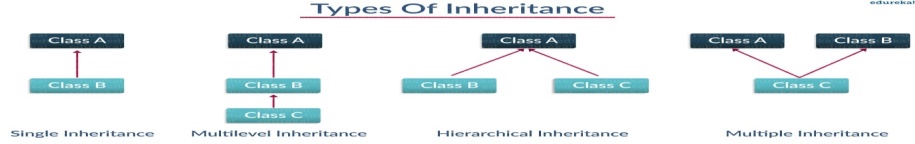
Object-Oriented Programming methodologies deal with the following concepts.

* Inheritance
* Polymorphism
* Encapsulation
* Abstraction

Let us understand the first concept of inheritance in detail.

**Inheritance:**

Ever heard of this dialogue from relatives “you look exactly like your father/mother” the reason behind this is called ‘[inheritance](https://www.edureka.co/blog/python-class/#Inheritance)’. From the Programming aspect, It generally means “inheriting or transfer of characteristics from parent to child class without any modification”. The new class is called the **derived/child**class and the one from which it is derived is called a **parent/base**class.

**[Python Certification Training for Data Science](https://www.edureka.co/data-science-python-certification-course" \t "_blank)**[](https://www.edureka.co/data-science-python-certification-course" \t "_blank)

* *[Instructor-led Live Sessions](https://www.edureka.co/data-science-python-certification-course" \t "_blank)*
* *[Real-life Case Studies](https://www.edureka.co/data-science-python-certification-course" \t "_blank)*
* *[Assignments](https://www.edureka.co/data-science-python-certification-course" \t "_blank)*
* *[Lifetime Access](https://www.edureka.co/data-science-python-certification-course" \t "_blank)*

[Explore Curriculum](https://www.edureka.co/data-science-python-certification-course" \t "_blank)

Let us understand each of the subtopics in detail.

**Single Inheritance:**

Single level inheritance enables a derived class to inherit characteristics from a single parent class.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | class employee1()://This is a parent class  def \_\_init\_\_(self, name, age, salary):  self.name = name  self.age = age  self.salary = salary    class childemployee(employee1)://This is a child class  def \_\_init\_\_(self, name, age, salary,id):  self.name = name  self.age = age  self.salary = salary  self.id = id  emp1 = employee1('harshit',22,1000)    print(emp1.age) |

**Example:**

**Output**: 22

**Explanation:**

* I am taking the parent class and created a constructor (\_\_init\_\_),  class itself is initializing the attributes with parameters(‘name’, ‘age’ and ‘salary’).
* Created a child class ‘childemployee’ which is inheriting the properties from a parent class and finally instantiated objects ’emp1′ and ’emp2′ against the parameters.
* Finally, I have printed the age of emp1. Well, you can do a hell lot of things like print the whole dictionary or name or salary.

**Multilevel Inheritance:**

Multi-level inheritance enables a derived class to inherit properties from an immediate parent class which in turn inherits properties from his parent class.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | class employee()://Super class  def \_\_init\_\_(self,name,age,salary):  self.name = name  self.age = age  self.salary = salary  class childemployee1(employee)://First child class  def \_\_init\_\_(self,name,age,salary):  self.name = name  self.age = age  self.salary = salary    class childemployee2(childemployee1)://Second child class  def \_\_init\_\_(self, name, age, salary):  self.name = name  self.age = age  self.salary = salary  emp1 = employee('harshit',22,1000)  emp2 = childemployee1('arjun',23,2000)    print(emp1.age)  print(emp2.age) |

Output: 22,23

**Explanation:**

* It is clearly explained in the code written above, Here I have defined the superclass as employee and child class as childemployee1. Now, childemployee1 acts as a parent for childemployee2.
* I have instantiated two objects ’emp1′ and ’emp2′ where I am  passing the parameters “name”, “age”, “salary” for emp1 from superclass  “employee” and “name”, “age, “salary” and “id” from the parent class “childemployee1”

**Hierarchical Inheritance:**

Hierarchical level inheritance enables more than one derived class to inherit properties from a parent class.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | class employee():  def \_\_init\_\_(self, name, age, salary):     //Hierarchical Inheritance  self.name = name  self.age = age  self.salary = salary    class childemployee1(employee):  def \_\_init\_\_(self,name,age,salary):  self.name = name  self.age = age  self.salary = salary    class childemployee2(employee):  def \_\_init\_\_(self, name, age, salary):  self.name = name  self.age = age  self.salary = salary  emp1 = employee('harshit',22,1000)  emp2 = employee('arjun',23,2000)    print(emp1.age)  print(emp2.age) |

Output: 22,23

**Explanation:**

* In the above example, you can clearly see there are two child class “childemployee1” and “childemployee2”. They are inheriting functionalities from a common parent class that is “employee”.
* Objects ’emp1′ and ’emp2′ are instantiated against the parameters ‘name’, ‘age’, ‘salary’.

**Multiple Inheritance:**

Multiple level inheritance enables one derived class to inherit properties from more than one base class.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | class employee1()://Parent class      def \_\_init\_\_(self, name, age, salary):          self.name = name          self.age = age          self.salary = salary    class employee2()://Parent class      def \_\_init\_\_(self,name,age,salary,id):       self.name = name       self.age = age       self.salary = salary       self.id = id    class childemployee(employee1,employee2):      def \_\_init\_\_(self, name, age, salary,id):       self.name = name       self.age = age       self.salary = salary       self.id = id  emp1 = employee1('harshit',22,1000)  emp2 = employee2('arjun',23,2000,1234)    print(emp1.age)  print(emp2.id) |

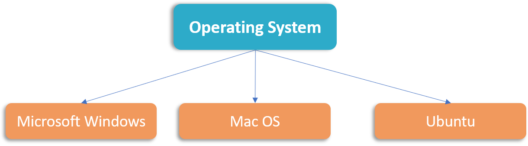
Output: 22,1234

**Explanation:**In the above example, I have taken two parent class “employee1” and “employee2”.And a child class “childemployee”, which is inheriting both parent class by instantiating the objects ’emp1′ and ’emp2′ against the parameters of parent classes.

This was all about inheritance, moving ahead in Object-Oriented Programming Python, let’s take a deep dive in ‘[polymorphism](https://www.edureka.co/blog/python-class/#Polymorphism)‘.

**Polymorphism:**

You all must have used GPS for navigating the route, Isn’t it amazing how many different routes you come across for the same destination depending on the traffic, from a programming point of view this is called ‘polymorphism’. It is one such OOP methodology where one task can be performed in several different ways. To put it in simple words,*it is a property of an object which allows it to take multiple forms*.



Polymorphism is of two types:

**Compile-time Polymorphism:**

A compile-time polymorphism also called as static polymorphism which gets resolved during the compilation time of the program. One common example is “method overloading”. Let me show you a quick example of the same.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29 | class employee1():  def name(self):  print("Harshit is his name")  def salary(self):  print("3000 is his salary")    def age(self):  print("22 is his age")    class employee2():  def name(self):  print("Rahul is his name")    def salary(self):  print("4000 is his salary")    def age(self):  print("23 is his age")    def func(obj)://Method Overloading  obj.name()  obj.salary()  obj.age()    obj\_emp1 = employee1()  obj\_emp2 = employee2()    func(obj\_emp1)  func(obj\_emp2) |

**Output:**

Harshit is his name  
3000 is his salary  
22 is his age  
Rahul is his name  
4000 is his salary  
23 is his age

**Explanation:**

* In the above Program, I have created two classes ’employee1′ and ’employee2′ and created functions for both ‘name’, ‘salary’ and  ‘age’ and printed the value of the same without taking it from the user.
* Now, welcome to the main part where I have created a function with ‘obj’ as the parameter and calling all the three functions i.e. ‘name’, ‘age’ and ‘salary’.
* Later, instantiated objects emp\_1 and emp\_2 against the two classes and simply called the [function](https://www.edureka.co/blog/python-functions). Such type is called method overloading which allows a class to have more than one method under the same name.

**Run-time Polymorphism:**

A run-time Polymorphism is also, called as dynamic polymorphism where it gets resolved into the run time. One common example of Run-time polymorphism is “method overriding”. Let me show you through an example for a better understanding.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | class employee():     def \_\_init\_\_(self,name,age,id,salary):         self.name = name         self.age = age         self.salary = salary         self.id = id  def earn(self):          pass    class childemployee1(employee):       def earn(self)://Run-time polymorphism        print("no money")    class childemployee2(employee):       def earn(self):         print("has money")    c = childemployee1  c.earn(employee)  d = childemployee2  d.earn(employee) |

**Output:** no money, has money

**Explanation:**In the above example, I have created two classes ‘childemployee1’ and ‘childemployee2’ which are derived from the same base class ‘employee’.Here’s the catch one did not receive money whereas the other one gets. Now the real question is how did this happen? Well, here if you look closely I created an empty function and used *Pass*( a statement which is used when you do not want to execute any command or code). Now, Under the two derived classes, I used the same empty function and made use of the print statement as ‘no money’ and ‘has money’.Lastly, created two objects and called the function.

Moving on to the next Object-Oriented Programming Python methodology, I’ll talk about encapsulation.

**Encapsulation:**

In a raw form, encapsulation basically means binding up of data in a single class. Python does not have any private keyword, unlike [Java](https://www.edureka.co/blog/access-modifiers-in-java/). A class shouldn’t be directly accessed but be prefixed in an underscore.

Let me show you an example for a better understanding.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | class employee(object):  def \_\_init\_\_(self):  self.name = 1234  self.\_age = 1234  self.\_\_salary = 1234    object1 = employee()  print(object1.name)  print(object1.\_age)  print(object1.\_\_salary) |

**Output:**1234  
Traceback (most recent call last):  
1234  
File “C:/Users/Harshit\_Kant/PycharmProjects/test1/venv/encapsu.py”, line 10, in  
print(object1.\_\_salary)  
**AttributeError:** ’employee’ object has no attribute ‘\_\_salary’

**Explanation:**You will get this question what is the underscore and error? Well, python class treats the private variables as(\_\_salary) which can not be accessed directly.

So, I have made use of the setter method which provides indirect access to them in my next example.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | class employee():  def \_\_init\_\_(self):  self.\_\_maxearn = 1000000  def earn(self):  print("earning is:{}".format(self.\_\_maxearn))    def setmaxearn(self,earn)://setter method used for accesing private class  self.\_\_maxearn = earn    emp1 = employee()  emp1.earn()    emp1.\_\_maxearn = 10000  emp1.earn()    emp1.setmaxearn(10000)  emp1.earn() |

**Output:**

earning is:1000000,earning is:1000000,earning is:10000

**Explanation:**Making Use of the **setter method** provides *indirect access to the private class method*. Here I have defined a class employee and used a (\_\_maxearn) which is the setter method used here to store the maximum earning of the employee, and a setter function setmaxearn() which is taking price as the parameter.

This is a clear example of encapsulation where we are restricting the access to private class method and then use the setter method to grant access.

Next up in object-oriented programming python methodology talks about one of the key concepts called [abstraction](https://www.edureka.co/blog/python-class/#Abstraction).

**Abstraction:**

Suppose you booked a movie ticket from bookmyshow using net banking or any other process. You don’t know the procedure of how the pin is generated or how the verification is done. This is called ‘abstraction’ from the programming aspect, it basically means you only show the implementation details of a particular process and hide the details from the user. It is used to simplify complex problems by modeling classes appropriate to the problem.

An abstract class cannot be instantiated which simply means you cannot create objects for this type of [class](https://www.edureka.co/blog/python-class/#WhatIsPythonClass). It can only be used for inheriting the functionalities.

**Example:**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | from abc import ABC,abstractmethod  class employee(ABC):  def emp\_id(self,id,name,age,salary):    //Abstraction  pass    class childemployee1(employee):  def emp\_id(self,id):  print("emp\_id is 12345")    emp1 = childemployee1()  emp1.emp\_id(id) |

**Output**: emp\_id is 12345 **Explanation:**As you can see in the above example, we have imported an abstract method and the rest of the program has a parent and a derived class. An object is instantiated for the ‘childemployee’ base class and functionality of abstract is being used.

**Basic OOPs Interview Questions:**

**1. What is the difference between OOP and SOP?**

|  |  |
| --- | --- |
| Object-Oriented Programming | Structural Programming |
| Object-Oriented Programming is a type of programming which is based on objects rather than just functions and procedures | Provides logical structure to a program where programs are divided functions |
| Bottom-up approach | Top-down approach |
| Provides data hiding | Does not provide data hiding |
| Can solve problems of any complexity | Can solve moderate problems |
| Code can be reused thereby reducing redundancy | Does not support code reusability |

**2. What is OOPs?**

OOPs (Object-Oriented Programming) is a type of programming which is based on objects rather than just functions and procedures. Individual objects are grouped into classes. OOPs implements real-world entities like inheritance, polymorphism, hiding, etc into programming. It also allows binding data and code together.

**3. Why use OOPs?**

* OOPs allows clarity in programming thereby allowing simplicity in solving complex problems
* Code can be reused through inheritance thereby reducing redundancy
* Data and code are bound together by encapsulation
* OOPs allows data hiding, therefore, private data is kept confidential
* Problems can be divided into different parts making it simple to solve
* The concept of polymorphism gives flexibility to the program by allowing the entities to have multiple forms

**4. What are the main features of OOPs?**

* Inheritance
* Encapsulation
* Polymorphism
* Data Abstraction

**Classes and Objects OOPs Interview Questions:**

**5. What is an object?**

An object is a real-world entity which is the basic unit of OOPs for example chair, cat, dog, etc. Different objects have different states or attributes, and behaviors.

**6. What is a class?**

A class is a prototype that consists of objects in different states and with different behaviors. It has a number of methods that are common the objects present within that class.

**7. What is the difference between a class and a structure?**

**Class:**User-defined blueprint from which objects are created. It consists of methods or set of instructions that are to be performed on the objects.

**Structure:**A structure is basically a user-defined collection of variables which are of different data types.

**8. Can you call the base class method without creating an instance?**

Yes, you can call the base class without instantiating it if:

* It is a static method
* The base class is inherited by some other subclass

**9. What is the difference between a class and an object?**

|  |  |
| --- | --- |
| Object | Class |
| A real-world entity which is an instance of a class | A class is basically a template or a blueprint within which objects can be created |
| An object acts like a variable of the class | Binds methods and data together into a single unit |
| An object is a physical entity | A class is a logical entity |
| Objects take memory space when they are created | A class does not take memory space when created |
| Objects can be declared as and when required | Classes are declared just once |

**Interview Questions on Features of OOPs**

**Inheritance:**

**10. What is inheritance?**

Inheritance is a feature of OOPs which allows classes inherit common properties from other classes. For example, if there is a class such as ‘vehicle’, other classes like ‘car’, ‘bike’, etc can inherit common properties from the vehicle class. This property helps you get rid of redundant code thereby reducing the overall size of the code.

**11. What are the different types of inheritance?**

* Single inheritance
* Multiple inheritance
* Multilevel inheritance
* Hierarchical inheritance
* Hybrid inheritance

**12. What is the difference between multiple and multilevel inheritance?**

|  |  |
| --- | --- |
| Multiple Inheritance | Multilevel Inheritance |
| Multiple inheritance comes into picture when a class inherits more than one base class | Multilevel inheritance means a class inherits from another class which itself is a subclass of some other base class |
| Example: A class defining a child inherits from two base classes Mother and Father | Example: A class describing a sports car will inherit from a base class Car which inturn inherits another class Vehicle |

**13. What is hybrid inheritance?**

Hybrid inheritance is a combination of multiple and multi-level inheritance.

**14. What is hierarchical inheritance?**

Hierarchical inheritance refers to inheritance where one base class has more than one subclasses. For example, the vehicle class can have ‘car’, ‘bike’, etc as its subclasses.

**15. What are the limitations of inheritance?**

* Increases the time and effort required to execute a program as it requires jumping back and forth between different classes
* The parent class and the child class get tightly coupled
* Any modifications to the program would require changes both in the parent as well as the child class
* Needs careful implementation else would lead to incorrect results

**16. What is a superclass?**

A superclass or base class is a class that acts as a parent to some other class or classes. For example, the Vehicle class is a superclass of class Car.

**17. What is a subclass?**

A class that inherits from another class is called the subclass. For example, the class Car is a subclass or a derived of Vehicle class.

**Polymorphism**

**18. What is polymorphism?**

Polymorphism refers to the ability to exist in multiple forms. Multiple definitions can be given to a single interface. For example, if you have a class named Vehicle, it can have a method named speed but you cannot define it because different vehicles have different speed. This method will be defined in the subclasses with different definitions for different vehicles.

**19. What is static polymorphism?**

Static polymorphism (static binding) is a kind of polymorphism that occurs at compile time. An example of compile-time polymorphism is method overloading.

**20. What is dynamic polymorphism?**

Runtime polymorphism or dynamic polymorphism (dynamic binding) is a type of polymorphism which is resolved during runtime. An example of runtime polymorphism is method overriding.

**21. What is method overloading?**

Method overloading is a feature of OOPs which makes it possible to give the same name to more than one methods within a class if the arguments passed differ.

**22. What is method overriding?**

Method overriding is a feature of OOPs by which the child class or the subclass can redefine methods present in the base class or parent class. Here, the method that is overridden has the same name as well as the signature meaning the arguments passed and the return type.

**23. What is operator overloading?**

Operator overloading refers to implementing operators using user-defined types based on the arguments passed along with it.

**24. Differentiate between overloading and overriding.**

|  |  |
| --- | --- |
| Overloading | Overriding |
| Two or more methods having the same name but different parameters or signature | Child class redefining methods present in the base class with the same parameters/ signature |
| Resolved during compile-time | Resolved during runtime |

**Encapsulation**

**25. What is encapsulation?**

Encapsulation refers to binding the data and the code that works on that together in a single unit. For example, a class. Encapsulation also allows data-hiding as the data specified in one class is hidden from other classes.

**26. What are ‘access specifiers’?**

Access specifiers or access modifiers are keywords that determine the accessibility of methods, classes, etc in OOPs. These access specifiers allow the implementation of encapsulation. The most common access specifiers are public, private and protected. However, there are a few more which are specific to the programming languages.

**27. What is the difference between public, private and protected access modifiers?**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Accessibility from own class | Accessibility from derived class | Accessibility from world |
| Public | Yes | Yes | Yes |
| Private | Yes | No | No |
| Protected | Yes | Yes | No |

**Data abstraction**

**28. What is data abstraction?**

Data abstraction is a very important feature of OOPs that allows displaying only the important information and hiding the implementation details. For example, while riding a bike, you know that if you raise the accelerator, the speed will increase, but you don’t know how it actually happens. This is data abstraction as the implementation details are hidden from the rider.

**29. How to achieve data abstraction?**

Data abstraction can be achieved through:

* Abstract class
* Abstract method

**30. What is an abstract class?**

An abstract class is a class that consists of abstract methods. These methods are basically declared but not defined. If these methods are to be used in some subclass, they need to be exclusively defined in the subclass.

Next

**31.** **Can you create an instance of an abstract class?**

No. Instances of an abstract class cannot be created because it does not have a complete implementation. However, instances of subclass inheriting the abstract class can be created.

**32. What is an interface?**

It is a concept of OOPs that allows you to declare methods without defining them. Interfaces, unlike classes, are not blueprints because they do not contain detailed instructions or actions to be performed. Any class that implements an interface defines the methods of the interface.

**33. Differentiate between data abstraction and encapsulation.**

|  |  |
| --- | --- |
| Data abstraction | Encapsulation |
| Solves the problem at the design level | Solves the problem at the implementation level |
| Allows showing important aspects while hiding implementation details | Binds code and data together into a single unit and hides it from the world |

**Methods and Functions OOPs interview questions**

**34. What are virtual functions?**

Virtual functions are functions that are present in the parent class and are overridden by the subclass. These functions are used to achieve runtime polymorphism.

**35. What are pure virtual functions?**

Pure virtual functions or abstract functions are functions that are only declared in the base class. This means that they do not contain any definition in the base class and need to be redefined in the subclass.

**36. What is a constructor?**

A constructor is a special type of method that has the same name as the class and is used to initialize objects of that class.

**37. What is a destructor?**

A destructor is a method that is automatically invoked when an object is destroyed. The destructor also recovers the heap space that was allocated to the destroyed object, closes the files and database connections of the object, etc.

**38. Types of constructors**

Types of constructors differ from language to language. However, all the possible constructors are:

* Default constructor
* Parameterized constructor
* Copy constructor
* Static constructor
* Private constructor

**39. What is a copy constructor?**

A copy constructor creates objects by copying variables from another object of the same class. The main aim of a copy constructor is to create a new object from an existing one.

**40. What is the use of ‘finalize’?**

Finalize as an object method used to free up unmanaged resources and cleanup before Garbage Collection(GC). It performs memory management tasks.

**41. What is Garbage Collection(GC)?**

GC is an implementation of automatic memory management. The Garbage collector frees up space occupied by objects that are no longer in existence.

**42. Differentiate between a class and a method.**

|  |  |
| --- | --- |
| Class | Method |
| A class is basically a template that binds the code and data together into a single unit. Classes consist of methods, variables, etc | Callable set of instructions also called a procedure or function that are to be performed on the given data |

**43. Differentiate between an abstract class and an interface?**

|  |  |  |
| --- | --- | --- |
| Basis for comparison | Abstract Class | Interface |
| Methods | Can have abstract as well as other methods | Only abstract methods |
| Final Variables | May contain final and non-final variables | Variables declared are final by default |
| Accessibility of Data Members | Can be private, public, etc | Public by default |
| Implementation | Can provide the implementation of an interface | Cannot provide the implementation of an abstract class |

**44. What is a final variable?**

A variable whose value does not change. It always refers to the same object by the property of non-transversity.

**Exception Handling**

**45. What is an exception?**

An exception is a kind of notification that interrupts the normal execution of a program. Exceptions provide a pattern to the error and transfer the error to the exception handler to resolve it. The state of the program is saved as soon as an exception is raised.

**46. What is exception handling?**

Exception handling in Object-Oriented Programming is a very important concept that is used to manage errors. An exception handler allows errors to be thrown and caught and implements a centralized mechanism to resolve them.

**47. What is the difference between an error and an exception?**

|  |  |
| --- | --- |
| Error | Exception |
| Errors are problems that should not be encountered by applications | Conditions that an application might try to catch |

**48. What is a try/ catch block?**

A try/ catch block is used to handle exceptions. The try block defines a set of statements that may lead to an error. The catch block basically catches the exception.

**49. What is a finally block?**

A finally block consists of code that is used to execute important code such as closing a connection, etc. This block executes when the try block exits. It also makes sure that finally block executes even in case some unexpected exception is encountered.

**Limitations of OOPs**

**50. What are the limitations of OOPs?**

* Usually not suitable for small problems
* Requires intensive testing
* Takes more time to solve the problem
* Requires proper planning
* The programmer should think of solving a problem in terms of objects

Now, you can begin reading the questions/answers given in the below section. All of these are related to the application of Python and would test your scripting skills of the language.

* [**What is the function to randomize the items of a list in-place**](https://www.techbeamers.com/python-interview-questions-experienced/#q1)**?**
* [**What is the best way to split a string in Python?**](https://www.techbeamers.com/python-interview-questions-experienced/#q2)
* [**What is the right way to transform a Python string into a list?**](https://www.techbeamers.com/python-interview-questions-experienced/#q3)
* [**How does exception handling in Python differ from Java? Also, list the optional clauses for a “try-except” block in Python?**](https://www.techbeamers.com/python-interview-questions-experienced/#q4)
* [**What do you know about the “list” and “dict” comprehensions? Explain with an example.**](https://www.techbeamers.com/python-interview-questions-experienced/#q5)
* [**What are the methods you know to copy an object in Python?**](https://www.techbeamers.com/python-interview-questions-experienced/#q6)
* [**Can you write code to determine the name of an object in Python?**](https://www.techbeamers.com/python-interview-questions-experienced/#q7)
* [**Can you write code to check whether the given object belongs to a class or its subclass?**](https://www.techbeamers.com/python-interview-questions-experienced/#q8)
* [**What is the result of the following Python program?**](https://www.techbeamers.com/python-interview-questions-experienced/#q9)
* [**What is the result of the below lines of code?**](https://www.techbeamers.com/python-interview-questions-experienced/#q10)
* [**What is the result of the below Python code?**](https://www.techbeamers.com/python-interview-questions-experienced/#q11)
* [**How would you produce a list with unique elements from a list with duplicate elements?**](https://www.techbeamers.com/python-interview-questions-experienced/#q12)
* [**Can you iterate over a list of words and use a dictionary to keep track of the frequency(count) of each word? Consider the below example.**](https://www.techbeamers.com/python-interview-questions-experienced/#q13)
* [**What is the result of the following Python code?**](https://www.techbeamers.com/python-interview-questions-experienced/#q14)
* [**Can you describe what’s wrong with the below code?**](https://www.techbeamers.com/python-interview-questions-experienced/#q15)

Top 15 Python Questions And Answers For Experienced.

**Python Questions and Answers for Experienced**

💡 **Fact – Python assumes a TAB equal to 8 Spaces.**

Q-1. What Is The Function To Randomize The Items Of A List In-Place?

**Ans.** Python has a built-in module called as <random>. It exports a public method <shuffle(<list>)> which can randomize any input sequence.

import random

list = [2, 18, 8, 4]

print "Prior Shuffling - 0", list

random.shuffle(list)

print "After Shuffling - 1", list

random.shuffle(list)

print "After Shuffling - 2", list

Q-2. What Is The Best Way To Split A String In Python?

**Ans.** We can use Python <split()> function to break a string into substrings based on the defined separator. It returns the list of all words present in the input string.

test = "I am learning Python."

print test.split(" ")

Program Output.

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

['I', 'am', 'learning', 'Python.']

Q-3. What Is The Right Way To Transform A Python String Into A List?

**Ans.** In Python, strings are just like lists. And it is easy to convert a string into the list. Simply by passing the string as an argument to the list would result in a string-to-list conversion.

list("I am learning Python.")

Program Output.

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

=> ['I', ' ', 'a', 'm', ' ', 'l', 'e', 'a', 'r', 'n', 'i', 'n', 'g', ' ', 'P', 'y', 't', 'h', 'o', 'n', '.']

Q-4. How Does Exception Handling In Python Differ From Java? Also, List The Optional Clauses For A <Try-Except> Block In Python?

**Ans.** Unlike Java, Python implements exception handling in a bit different way. It provides an option of using a <try-except> block where the programmer can see the error details without terminating the program. Sometimes, along with the problem, this <try-except> statement offers a solution to deal with the error.

There are following clauses available in Python language.

**1.** try-except-finally  
**2.** try-except-else

💡 **Must Read –**[**30 Most Important Python Interview Questions and Answers**](https://www.techbeamers.com/python-interview-questions-programmers/)**.**

Q-5. What Do You Know About The <List> And <Dict> Comprehensions? Explain With An Example.

**Ans.** The <List/Dict> comprehensions provide an easier way to create the corresponding object using the existing iterable. As per official Python documents, the list comprehensions are usually faster than the standard loops. But it’s something that may change between releases.

The <List/Dict> Comprehensions Examples.

#Simple Iteration

item = []

for n in range(10):

item.append(n\*2)

print item

#List Comprehension

item = [n\*2 for n in range(10)]

print item

Both the above example would yield the same output.

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

[0, 2, 4, 6, 8, 10, 12, 14, 16, 18]

#Dict Comprehension

item = {n: n\*2 for n in range(10)}

print item

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

{0: 0, 1: 2, 2: 4, 3: 6, 4: 8, 5: 10, 6: 12, 7: 14, 8: 16, 9: 18}

💡 **Fact – In interactive mode, the last printed expression is assigned to the variable \_ (underscore).**

Q-6. What Are The Methods You Know To Copy An Object In Python?

**Ans.** Commonly, we use <copy.copy()> or <copy.deepcopy()> to perform copy operation on objects. Though not all objects support these methods but most do.

But some objects are easier to copy. Like the dictionary objects provide a <copy()> method.

Example.

item = {n: n\*2 for n in range(10)}

newdict = item.copy()

print newdict

Q-7. Can You Write Code To Determine The Name Of An Object In Python?

**Ans.** No objects in Python have any associated names. So there is no way of getting the one for an object. The assignment is only the means of binding a name to the value. The name then can only refer to access the value. The most we can do is to find the reference name of the object.

Example.

class Test:

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

self.cards = []

self.name = name

def \_\_str\_\_(self):

return '{} holds ...'.format(self.name)

obj1 = Test('obj1')

print obj1

obj2 = Test('obj2')

print obj2

Q-8. Can You Write Code To Check Whether The Given Object Belongs To A Class Or Its Subclass?

**Ans.** Python has a built-in method to list the instances of an object that may consist of many classes. It returns in the form of a table containing tuples instead of the individual classes. Its syntax is as follows.

<isinstance(obj, (class1, class2, ...))>

The above method checks the presence of an object 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))>.

Also, it’s not recommended to use the built-in classes. Create an user-defined class instead.

We can take the following example to determine the object of a particular class.

Example.

def lookUp(obj):

if isinstance(obj, Mailbox):

print "Look for a mailbox"

elif isinstance(obj, Document):

print "Look for a document"

else:

print "Unidentified object"

Q-9. What Is The Result Of The Following Python Program?

**Ans.** The example code is as follows.

def multiplexers ():

return [lambda n: index \* n for index in range (4)]

print [m (2) for m in multiplexers ()]

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

[6, 6, 6, 6]

The output of the above code is <[6, 6, 6, 6]>. It’s because of the late binding as the value of the variable <index> gets looked up after a call to any of multiplexers functions.

💡 **Also Read – [20 Python Programming Interview Questions for Practice](https://www.techbeamers.com/python-programming-interview-questions-with-answers/" \t "_blank).**

Q-10. What Is The Result Of The Below Lines Of Code?

Here is the example code.

def fast (items= []):

items.append (1)

return items

print fast ()

print fast ()

**Ans.** The above code will give the following result.

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

[1]

[1, 1]

The function <fast> evaluates its arguments only once after the function gets defined. However, since <items> is a list, so it’ll get modified by appending a <1> to it.

💡 **Fact – You can inspect objects in Python by using dir().**

Q-11. What Is The Result Of The Below Python Code?

keyword = 'aeioubcdfg'

print keyword [:3] + keyword [3:]

**Ans.** The above code will produce the following result.

<'aeioubcdfg'>

In Python, while performing string slicing, whenever the indices of both the slices collide, a <+> operator get applied to concatenates them.

Q-12. How Would You Produce A List With Unique Elements From A List With Duplicate Elements?

**Ans.** Iterating the list is not a desirable solution. The right answer should look like this.

duplicates = ['a','b','c','d','d','d','e','a','b','f','g','g','h']

uniqueItems = list(set(duplicates))

print sorted(uniqueItems)

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']

Q-13. Can You Iterate Over A List Of Words And Use A Dictionary To Keep Track Of The Frequency(Count) Of Each Word? Consider The Below Example.

{'Number':Frequency, '2':2, '3':2}

**Ans.** Please find out the below code.

def dic(words):

wordList = {}

for index in words:

try:

wordList[index] += 1

except KeyError:

wordList[index] = 1

return wordList

wordList='1,3,2,4,5,3,2,1,4,3,2'.split(',')

print wordList

print dic(wordList)

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

['1', '3', '2', '4', '5', '3', '2', '1', '4', '3', '2']

{'1': 2, '3': 3, '2': 3, '5': 1, '4': 2}

Q-14. What Is The Result Of The Following Python Code?

class Test(object):

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

self.x = 1

t = Test()

print t.x

print t.x

print t.x

print t.x

**Ans.** All print statement will display <1>. It’s because the value of object’s attribute(x) is never changing.

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[GCC 4.8.2] on linux

1

1

1

1

Also, <x> becomes a part of the public members of class Test.

Hence, it can be accessed directly.

💡 **More Questions – [Top 10 Python Questions Every Developer Should Know](https://www.techbeamers.com/10-python-interview-questions/).**

Q-15. Can You Describe What’s Wrong With The Below Code?

testProc([1, 2, 3]) # Explicitly passing in a list

testProc() # Using a default empty list

def testProc(n = []):

# Do something with n

print n

**Ans.** The above code would throw a <NameError>.

The variable n is local to the function <testProc> and can’t be accessed outside.

So, printing it won’t be possible.

**Top Answers to Python Interview Questions**

**1. Compare between Java and Python.**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Java** | **Python** |
| Ease of use | Good | Very Good |
| Speed of coding | Average | Excellent |
| Data types | Static typed | Dynamically typed |
| Data Science & machine learning applications | Average | Very Good |

**2. What is Python?**

[Python](https://intellipaat.com/blog/python-certification/) is a high-level, interpreted, interactive, and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently, whereas other languages use punctuation, and it has fewer syntactical constructions than other languages.

**3. What are the key features of Python?**

* Python is an interpreted language, so it doesn’t need to be compiled before execution unlike languages like C.
* Python is dynamically typed, so there is no need to declare a variable with the data type. Python Interpreter will identify the data type on the basis of the value of the variable.

For example, if you run the following code line, it will run without any error:

a = 100

a = "Intellipaat"

* Python follows object-oriented programming paradigm with an exception of having access specifiers. Other than access specifiers (public and private keywords), Python has classes, inheritance, and all other usual OOPs concepts.
* Python is a cross-platform language which means that a Python program written on a Windows system will also run on a Linux system with little or no modifications at all.
* Python is literally a general-purpose language which means that Python finds its way in various domains such as Web Application Development, Automation, Data Science, Machine Learning, and more.

**Check out this video on Python Tutorial for Beginners:**

Learn for free ! Subscribe to our youtube Channel.

**4. What is the purpose of PYTHONPATH environment variable?**

PYTHONPATH has a role similar to PATH. This variable tells Python Interpreter where to locate the module files imported into a program. It should include Python source library directory and the directories containing Python source code. PYTHONPATH is sometimes preset by Python Installer.

**5. What is the purpose of PYTHONSTARTUP, PYTHONCASEOK, and PYTHONHOME environment variables?**

* PYTHONSTARTUP: It contains the path of an initialization file containing Python source code. It is executed every time you start the interpreter. It is named as .pythonrc.py in Unix, and it contains commands that load utilities or modify PYTHONPATH.
* PYTHONCASEOK: It is used in Windows to instruct Python to find the first case-insensitive match in an import statement. Set this variable with any value to activate it.
* PYTHONHOME: It is an alternative module search path. It is usually embedded in PYTHONSTARTUP or PYTHONPATH directories to make switching of module libraries easy.

***Go through the***[***Python Course in London***](https://intellipaat.com/python-certification-training-online-london/)***to get clear understanding of Python.***

**6. What are the supported data types in Python?**

Python has five standard data types:

* Numbers
* Strings
* Lists
* Tuples
* Dictionaries

**7. What is the difference between lists and tuples?**

|  |  |
| --- | --- |
| **Lists** | **Tuples** |
| Lists are mutable, i.e., they can be edited. | Tuples are immutable (Tuples are lists which cannot be edited). |
| Lists are usually slower than tuples. | Tuples are faster than lists. |
| Syntax: list\_1 = [10, ‘Intellipaat’, 20] | Syntax: tup\_1 = (10, ‘Intellipaat’ , 20) |

**8. How is memory managed in Python?**

* Memory in Python is managed by Python private heap space. All Python objects and data structures are located in a private heap. This private heap is taken care of by Python Interpreter itself, and a programmer doesn’t have any access to this private heap.
* Python memory manager takes care of the allocation of Python private heap space.
* Memory for Python private heap space is made available by Python’s inbuilt garbage collector which recycles and frees up all the unused memory.

**9. Explain Inheritance in Python with an example.**

As Python follows an object-oriented programming paradigm, classes in Python have the ability of inheriting the properties of another class. This process is known as inheritance. Inheritance provides the code reusability feature. The class that is being inherited is called a super-class and the class that inherits the other class is called a derived or child class. Following types of inheritance are supported in Python:

1. Single inheritance: When a class inherits only one super class
2. Multiple inheritance: When a class inherits multiple super classes
3. Multilevel inheritance: When a class inherits a super class and then another class inherits this derived class forming a ‘parent, child, and grandchild’ class structure
4. Hierarchical inheritance: When one super class is inherited by multiple derived classes

***Want to become master in Python programming check out this***[***Python training***](https://intellipaat.com/python-for-data-science-training/)***?***

**10. What is a dictionary in Python?**

Python dictionary is one of the supported [data types in Python](https://intellipaat.com/tutorial/python-tutorial/datatypes-in-python/). It is an unordered collection of elements. The elements in dictionaries are stored as key–value pairs. Dictionaries are indexed by keys.

For example, we have a dictionary named ‘dict’. It contains some keys: Country, Capital along with their corresponding values: India and New Delhi.

dict={‘Country’:’India’,’Capital’:’New Delhi’, }

**Check out this video on Python Tutorial**

Learn for free ! Subscribe to our youtube Channel.

**11. Can you write an efficient code to count the number of capital letters in a file?**

The normal solution for this problem statement would be as follows:

with open(SOME\_LARGE\_FILE) as countletter:

count = 0

text = countletter.read()

for character in text:

if character.isupper():

count += 1

To make this code more efficient, the whole code block can be converted into a one-liner code using generator expression. Then, the equivalent code line for the above code block would be:

count sum(1 for line in countletter for character in line if character.isupper())

**12. Write a code to sort a numerical list in Python.**

The following code can be used to sort a numerical list in Python:

list = [“2”, “5”, “7”, “8”, “1”]

list = [int(i) for i in list]

list.sort()

print (list)

**13. How will you reverse a list in Python?**

list.reverse(): This function reverses objects of list.

**14. How will you remove the last object from a list?**

list.pop(obj=list[-1]):

Here, ‘−1’ represents the last element of the list. The pop function removes and returns the last object (or obj) from the list.

**15. What are negative indexes and why are they used?**

To access an element from ordered sequences, we simply use the index of the element, which is the number of the position of that element. The index usually starts from 0, meaning that the first element has the index 0 and the second has 1, and so on.

When we use the index to access elements from the end of a list, it’s called reverse indexing. In reverse indexing, the indexing of elements start from the last element with the index number being ‘−1’. The second last element has the index ‘−2’, and so on. These indexes used in reverse indexing are called negative indexes.

**16. What are split(), sub(), and subn() methods in Python?**

These methods belong to Python RegEx ‘re’ module and are used to modify strings.

* split(): This method is used to split a given string into a list.
* sub(): This method is used to find a substring where a regex pattern matches and then it replaces that matched substring with a different string.
* subn(): This method is similar to the sub() method, but it also returns the new string, along with the number of replacements.

***Learn more about Python in this***[***Python training in New York***](https://intellipaat.com/python-certification-training-online-new-york/)***to get ahead in your career!***

**17. How are range and xrange different from one another?**

Functions in Python, range() and xrange() are used to iterate a fixed number of times in a for loop. Functionality-wise, both of these functions are the same. Difference comes when talking about Python version support for these functions and their return values.

|  |  |
| --- | --- |
| **The range() Method** | **The xrange() Method** |
| In Python 3, xrange() is not supported; instead the range() function is used to iterate in for loops. | The xrange() function is used in Python 2 to iterate in for loops. |
| It returns a list. | It returns a generator object as it doesn’t really generate a static list at the run time. |
| It takes more memory as it keeps the entire list of iterating numbers in memory. | It takes less memory as it keeps only one number at a time in memory. |

**18. Define pickling and unpickling in Python.**

Pickling is the process of converting Python objects such as lists, dicts, etc. into a character stream. This is done using a module named pickle, hence the name pickling.

The process of retrieving the original Python object from the stored string representation, which is the reverse of the pickling process, is called unpickling.

**19. What is a map function in Python?**

The map() function in Python has two parameters, function and iterable. The map() function takes a function as an argument and then applies that function to all the elements of an iterable, passed to it as another argument. It returns an object list of results.

For example:

def calculateSq(n):

return n\*n

numbers = (2, 3, 4, 5)

result = map( calculateSq, numbers)

print(result)

**20. Write a code to get indices of N maximum values in a NumPy array.**

We can get the indices of N maximum values in a NumPy array using the below code:

import numpy as np

ar = np.array([1, 3, 2, 4, 5, 6])

print(ar.argsort()[-3:][::-1])

***Interested in learning Python? Click here to learn more in this***[***Python Training in Sydney***](https://intellipaat.com/python-certification-training-online-sydney/)***!***

**21. What is a Python module?**

Modules are independent Python scripts with .py extension that can be reused in other Python codes or scripts using the import statement. A module can consist of functions, classes, and variables, or some runnable code. Modules not only help in keeping Python codes organized but also in making codes less complex and more efficient. The syntax to import modules in Python codes is as follows:

import module\_name   # include this code line on top of the script

**Watch this Python Projects Video for Beginners:**

**22. What do file-related modules in Python do? Can you name some file-related modules in Python?**

Python comes with some file-related modules that have functions to manipulate text files and binary files on a file system. These modules can be used to create text or binary files, update their content, copy, delete, and more.

Some file-related modules are os, os.path, and shutil.os. The os.path module has functions to access the file system, while the shutil.os module can be used to copy or delete files.

**23. Explain the use of with statement and its syntax.**

In Python, using the ‘with’ statement, we can open a file and close it as soon as the block of code, where ‘with’ is used, exits, without having to use the close() method.

with open("filename", "mode") as file\_var:

**24. Explain all file processing modes supported in Python.**

Python has various file processing modes.

For opening files, there are three modes:

* read-only mode (r)
* write-only mode (w)
* read–write mode (rw)

To open a text file using these modes, we will have to append ‘t’ with them as follows:

* read-only mode (rt)
* write-only mode (wt)
* read–write mode (rwt)

Similarly, a binary file can be opened by appending ‘b’ with them as follows:

* read-only mode (rb)
* write-only mode (wb)
* read–write mode (rwb)

To append content in files, we can use the append mode (a). Again, for text files, the mode would be ‘at’, and for binary files it would be ‘ab’.

***Become Master of Python by going through this online***[***Python course in Toronto***](https://intellipaat.com/python-certification-training-online-toronto/)***.***

**25. Is indentation optional in Python?**

Indentation in Python is compulsory and is part of its syntax. All programming languages have some way of defining the scope and extent of the block of codes; in Python, it is indentation. Indentation provides better readability to the code, which is probably why Python has made it compulsory.

**26. How are Python arrays and Python lists different from each other?**

In the case of Python, when people say arrays, they are usually talking about lists. It is because lists are fundamental to Python just as arrays are fundamental to most of the low-level languages. But, there is indeed a module named array in Python which is used or mentioned very rarely. Following are some of the differences between arrays and lists.

|  |  |
| --- | --- |
| **Arrays** | **Lists** |
| Arrays can only store homogeneous data (data of the same type). | Lists can store heterogenous and arbitrary data. |
| Since only one type of data can be stored, arrays use memory for only one type of objects. Thus, mostly, arrays use lesser memory than lists. | Lists can store data of multiple data types and thus require more memory than arrays. |
| Length of an array is pre-fixed while creating it, so more elements cannot be added. | Since the length of a list is not fixed, appending items in it is possible. |

**27. Write a code to display the contents of a file in reverse.**

for line in reversed(list(open(filename.txt))):

print(line.rstrip())

**28. Differentiate between NumPy and SciPy.**

|  |  |
| --- | --- |
| **NumPy** | **SciPy** |
| NumPy stands for Numerical Python. | SciPy stands for Scientific Python. |
| It is used for efficient and general numeric computations on numerical data saved in arrays. For example, sorting, indexing, reshaping, and more. | This module is a collection of tools in Python to perform operations such as integration, differentiation, and more. |
| Except for the general computing, there are some linear algebraic functions available in this module, but they are not fully fledged. | Fully fledged algebraic functions are available in SciPy for algebraic computations. |

**29. Which of the following is an invalid statement?**

1. a) xyz = 1,000,000
2. b) x y z = 1000 2000 3000
3. c) x,y,z = 1000, 2000, 3000
4. d) x\_y\_z = 1,000,000

Answer: b

**30. Can we make multi-line comments in Python?**

Python does not have a specific syntax for including [multi-line comments](https://intellipaat.com/community/2791/way-to-create-multiline-comments-in-python?show=2791#q2791) like other programming languages, but programmers can use triple-quoted strings (docstrings) for making multi-line comments, as when docstring is not being used as the first statement inside a method, it gets ignored by Python parser.

**31. What would be the output if I run the following code block?**

list1 = [2, 33, 222, 14, 25]  
print(list1[-2])

(A)          14  
(B)          33  
(C)          25  
(D)          Error

Ans: 14

**32. Write a command to open the file c:\hello.txt for writing.**

f= open(“hello.txt”, “wt”)

**33. What is \_\_init\_\_ in Python?**

\_\_init\_\_ is a reserved method in Python classes which is equivalent to constructors in OOP terminology. The \_\_init\_\_ method is called automatically whenever a new object is initiated. The \_\_init\_\_ method allocates memory to the new object as soon as it is created. This method can also be used to initialize variables.

**Video**

**Check out this video on Python for Data Science**

Learn for free ! Subscribe to our youtube Channel.

**34. What do you understand by Tkinter?**

Tkinter is an inbuilt Python module that is used to create GUI applications. It’s Python’s standard toolkit for GUI development. Tkinter comes with Python, so there is no installation needed. We can start using it by importing it in our script.

**35. Is Python fully object oriented?**

Python does follow an object-oriented programming paradigm and has all the [basic OOPs concepts](https://intellipaat.com/tutorial/java-tutorial/oops-concepts-in-java/) such as inheritance, polymorphism, and more, with the exception of access specifiers. Python doesn’t support strong encapsulation (adding private keyword before data members) although it does have a convention that can be used for data hiding, i.e., prefixing data member with two underscores.

**36. What is lambda function in Python?**

Lambda function is an anonymous function (functions that don’t have names) in Python. To define anonymous functions, we use the lambda keyword instead of the def keyword, hence the name ‘lambda function’. Lambda functions can have any number of arguments but can have only one statement.

**37. What is self-keyword in Python?**

Self-keyword is used as the first parameter of a function inside a class that represents the instance of the class. The object or the instance of the class is automatically passed to the method that it belongs to and is received in the ‘self-keyword’. Users can use another name for the first parameter of the function that catches the object of the class, but it is recommended to use the self-keyword only as it is more of a Python convention.

**38. What are control flow statements in Python?**

[Control flow statements](https://intellipaat.com/tutorial/python-tutorial/python-control-flow-statements/) are used to manipulate or change the execution flow of a program. Generally, the flow of execution of a program runs from top to bottom, but certain statements in Python can break this top to bottom order of execution. Control flow statements include decision-making, looping, and more.

**39. What is the difference between append() and extend() methods?**

Both append() and extend() methods are methods used for lists. These methods are used to add elements at the end of a list.

append(element): Adds the given element at the end of the list which called this method

extend(another-list): Adds the elements of another-list at the end of the list which called the extend method

**40. What are loop interruption statements in Python?**

There are two types of loop interruption statements in Python that let users terminate a loop iteration prematurely, i.e., without letting the loop run its full iterations.

Following are the two loop interruption statements:

* Python break statement: This statement immediately terminates the loop entirely, and the control flow of the program is shifted directly to the outside of the loop.
* Python continue statement: Continue statement terminates the current loop iteration and moves the control flow of the program to the next iteration of the loop, letting the user skip only the current iteration.

**41. What is docstring in Python?**

Python lets the user include a description or quick notes for their methods using documentation strings or docstrings. Docstrings are different from regular comments in Python as rather than being completely ignored by Python Interpreter like in the case of comments Python strings can actually be accessed at the run time using the dot operator when docstring is the first statement in a method or function.

***Are you interested in learning***[***Python course in Bangalore***](https://intellipaat.com/python-certification-training-online-bangalore/)***from Experts?***

**42. What is the output of the following?**

x = [‘ab’, ‘cd’]

print(len(list(map(list, x))))

Output: [[‘a’, ‘b’], [‘c’, ‘d’]].

Explanation: Each element of x is converted into list

**43. Which one of the following is not the correct syntax for creating a set?**

1. (a) set([[1,2],[3,4],[4,5]])(b) set([1,2,2,3,4,5])(c) {1,2,3,4}(d) set((1,2,3,4))Answer: (a)Explanation: The argument given for the set must be an iterable.

**44. What is functional programming? Does Python follow a functional programming style? If yes, list a few methods to implement functionally oriented programming in Python.**

Functional programming is a coding style where the main source of logic in a program comes from functions.

Incorporating functional programming in our codes means writing pure functions.

Pure functions are functions that cause little or no changes outside the scope of the function. These changes are referred to as side effects. To reduce side effects, pure functions are used, which makes the code easy to follow, test, or debug.

Python does follow a functional programming style. Following are some of the examples of functional programming in Python.

* 1. filter(): Filter lets us filter some values based on a conditional logic.

>>> list(filter(lambda x:x>6,range(9))) [7, 8]

* 1. map(): Map applies a function to every element in an iterable.

 >>> list(map(lambda x:x\*\*2,range(5))) [0, 1, 4, 9, 16, 25]

reduce(): Reduce repeatedly reduces a sequence pair-wise until it reaches a single value.

 >>> from functools import reduce >>> reduce(lambda x,y:x-y,[1,2,3,4,5]) -13

**45. How does Python Flask handle database requests?**

Flask supports a database-powered application (RDBS). Such a system requires creating a schema, which needs piping the schema.sql file into a sqlite3 command. So, you need to install the sqlite3 command in order to create or initiate the database in Flask.

Flask allows to request for database in three ways:

* before\_request(): They are called before a request and pass no arguments.
* after\_request(): They are called after a request and pass the response that will be sent to the client.
* teardown\_request(): They are called in a situation when an exception is raised, and responses are not guaranteed. They are called after the response has been constructed. They are not allowed to modify the request, and their values are ignored.

***Get certified from top***[***Python course in Singapore.***](https://intellipaat.com/python-certification-training-online-singapore/)***Now***

**46. Write a Python program to check whether a given string is a palindrome or not, without using an iterative method. Note: A palindrome is a word, phrase, or sequence that reads the same backward as forward, e.g., madam, nurses run, etc.**

def fun(string):

s1 = string

s = string[::-1]

if(s1 == s):

return true

else:

return false

print(fun(“madam”))

**47. Write a Python program to calculate the sum of a list of numbers.**

def sum(num):

if len(num) == 1:

return num[0]               #with only one element in the list, sum result will be equal to the element.

else:

return num[0] + sum(num[1:])

print(sum([2, 4, 5, 6, 7]))

Sample Output:

24

**48. Do we need to declare variables with data types in Python?**

No. Python is a dynamically typed language, which means that Python Interpreter automatically identifies the data type of a variable based on the type of value assigned to the variable.

***Learn Complete***[***Python training at Hyderabad***](https://intellipaat.com/python-certification-training-online-hyderabad/)***in 24 Hrs.***

**49. How will you read a random line in a file?**

We can read a random line in a file using a module named ‘random’.

For example:

import random

def read\_random(fname):

lines = open(fname).read().splitlines()

return random.choice(lines)

print(read\_random (‘hello.txt’))

**50. Write a Python program to count the total number of lines in a text file.**

def file\_count(fname):

with open(fname) as f:

for i, 1 in enumerate(f):

paas

return i+1

print(“Total number of lines in the text file: ”, file\_count(“file.txt”))

https://career.guru99.com/

**Top 50 OOPs Interview Questions & Answers**

**1. What is OOPS?**

OOPS is abbreviated as Object Oriented Programming system in which programs are considered as a

collection of objects. Each object is nothing but an instance of a class.

**2. Write basic concepts of OOPS?**

Following are the concepts of OOPS and are as follows:

1. Abstraction.

2. Encapsulation.

3. Inheritance.

4. Polymorphism.

**3. What is a class?**

A class is simply a representation of a type of object. It is the blueprint/ plan/ template that describes

the details of an object.

**4. What is an object?**

An object is an instance of a class. It has its own state, behavior, and identity.

**5. What is Encapsulation?**

Encapsulation is an attribute of an object, and it contains all data which is hidden. That hidden data

can be restricted to the members of that class.

Levels are Public, Protected, Private, Internal and Protected Internal.

**6. What is Polymorphism?**

Polymorphism is nothing but assigning behavior or value in a subclass to something that was already

declared in the main class. Simply, polymorphism takes more than one form.

**7. What is Inheritance?**

Inheritance is a concept where one class shares the structure and behavior defined in another class. If

inheritance applied on one class is called Single Inheritance, and if it depends on multiple classes,

then it is called multiple Inheritance.

**8. What are manipulators?**

Manipulators are the functions which can be used in conjunction with the insertion (<<) and

extraction (>>) operators on an object. Examples are endl and setw.

**9. Define a constructor?**

A constructor is a method used to initialize the state of an object, and it gets invoked at the time of

object creation. Rules forconstructor are:

Constructor Name should be same as class name.

A constructor must have no return type.

**10. Define Destructor?**

A destructor is a method which is automatically called when the object is made of scope or destroyed.

Destructor name is also same as class name but with the tilde symbol before the name.

**11. What is an Inline function?**

An inline function is a technique used by the compilers and instructs to insert complete body of the

function wherever that function is used in the program source code.

**12. What is a virtual function?**

A virtual function is a member function of a class, and its functionality can be overridden in its derived

class. This function can be implemented by using a keyword called virtual, and it can be given during

function declaration.

A virtual function can A token in C++, and it can be achieved in C Language by using function

pointers or pointers to function.

**13. What is a friend function?**

A friend function is a friend of a class that is allowed to access to Public, private or protected data in

that same class. If the function is defined outside the class cannot access such information.

Friend can be declared anywhere in the class declaration, and it cannot be affected by access control

keywords like private, public or protected.

**14. What is function overloading?**

Function overloading an as a normal function, but it can perform different tasks. It allows the creation

of several methods with the same name which differ from each other by the type of input and output

of the function.

Example

void add(int& a, int& b);

void add(double& a, double& b);

void add(struct bob& a, struct bob& b);

**15. What is operator overloading?**

Operator overloading is a function where different operators are applied and depends on the

arguments. Operator,-,\* can be used to pass through the function, and it has their own precedence to

execute

**16. What is an abstract class?**

An abstract class is a class which cannot be instantiated. Creation of an object is not possible with an

abstract class, but it can be inherited. An abstract class can contain only Abstract method. Java allows

only abstract method in abstract class while for other languages allow non-abstract method as well.

**17. What is a ternary operator?**

The ternary operator is said to be an operator which takes three arguments. Arguments and results

are of different data types, and it depends on the function. The ternary operator is also called a

conditional operator.

**18. What is the use of finalize method?**

Finalize method helps to perform cleanup operations on the resources which are not currently used.

Finalize method is protected, and it is accessible only through this class or by a derived class.

**19. What are different types of arguments?**

A parameter is a variable used during the declaration of the function or subroutine and arguments are

passed to the an, and it should match with the parameter defined. There are two types of Arguments.

Call by Value – Value passed will get modified only inside the function, and it returns the same

value whatever it is passed it into the function.

Call by Reference – Value passed will get modified in both inside and outside the functions and

it returns the same or different value.

**20. What is the super keyword?**

Super keyword is used to invoke the overridden method which overrides one of its superclass

methods. This keyword allows to access overridden methods and also to access hidden members of

the superclass.

It also forwards a call from a constructor to a constructor in the superclass.

**21. What is method overriding?**

Method overriding is a feature that allows a subclass to provide the implementation of a method that

overrides in the main class. This will overrides the implementation in the superclass by providing the

same method name, same parameter and same return type.

**22. What is an interface?**

An interface is a collection of an abstract method. If the class implements an inheritance, and then

thereby inherits all the abstract methods of an interface.

**23. What is exception handling?**

An exception is an event that occurs during the execution of a program. Exceptions can be of any

type – Runtime exception, Error exceptions. Those exceptions are adequately handled through

exception handling mechanism like try, catch and throw keywords.

**24. What are tokens?**

The token is recognized by a compiler, and it cannot be broken down into component elements.

Keywords, identifiers, constants, string literals and operators are examples of tokens.

Even punctuation characters are also considered as tokens – Brackets, Commas, Braces and

Parentheses.

**25. Difference between overloading and overriding?**

Overloading is static binding whereas Overriding is dynamic binding. Overloading is nothing but the

same method with different arguments, and it may or may not return the same value in the same

class itself.

Overriding is the same method names with same arguments and return types associated with the

class and its child class.

**26. Difference between class and an object?**

An object is an instance of a class. Objects hold multiple information, but classes don’t have any

information. Definition of properties and functions can be done in class and can be used by the object.

A class can have sub-classes, and an object doesn’t have sub-objects.

**27. What is an abstraction?**

Abstraction is a good feature of OOPS, and it shows only the necessary details to the client of an

object. Means, it shows only required details for an object, not the inner constructors, of an object.

Example – When you want to switch On television, it not necessary to show all the functions of TV.

Whatever is required to switch on TV will be showed by using abstract class.

**28. What are access modifiers?**

Access modifiers determine the scope of the method or variables that can be accessed from other

various objects or classes. There are 5 types of access modifiers, and they are as follows:

Private.

Protected.

Public.

Friend.

Protected Friend.

**29. What are sealed modifiers?**

Sealed modifiers are the access modifiers where it cannot be inherited by the methods. Sealed

modifiers can also be applied to properties, events, and methods. This modifier cannot be applied to

static members.

**30. How can we call the base method without creating an instance?**

Yes, it is possible to call the base method without creating an instance. And that method should be

“Static method”.

Doing inheritance from that class.-Use Base Keyword from a derived class.

**31. What is the difference between new and override?**

The new modifier instructs the compiler to use the new implementation instead of the base class

function. Whereas, Override modifier helps to override the base class function.

**32. What are the various types of constructors?**

There are three various types of constructors, and they are as follows:

– Default Constructor – With no parameters.

– Parametric Constructor – With Parameters. Create a new instance of a class and also passing

arguments simultaneously.

– Copy Constructor – Which creates a new object as a copy of an existing object.

**33. What is early and late binding?**

Early binding refers to the assignment of values to variables during design time whereas late binding

refers to the assignment of values to variables during run time.

**34. What is ‘this’ pointer?**

THIS pointer refers to the current object of a class. THIS keyword is used as a pointer which

differentiates between the current object with the global object. Basically, it refers to the current

object.

**35. What is the difference between structure and a class?**

Structure default access type is public , but class access type is private. A structure is used for

grouping data whereas class can be used for grouping data and methods. Structures are exclusively

used for data, and it doesn’t require strict validation , but classes are used to encapsulates and inherit

data which requires strict validation.

**36. What is the default access modifier in a class?**

The default access modifier of a class is Private by default.

**37. What is a pure virtual function?**

A pure virtual function is a function which can be overridden in the derived class but cannot be

defined. A virtual function can be declared as Pure by using the operator =0.

Example -.

[crayon-5b8545082b39c785950919/]

**38. What are all the operators that cannot be overloaded?**

Following are the operators that cannot be overloaded -.

1. Scope Resolution (:: )

2. Member Selection (.)

3. Member selection through a pointer to function (.\*)

**39. What is dynamic or run time polymorphism?**

Dynamic or Run time polymorphism is also known as method overriding in which call to an overridden

function is resolved during run time, not at the compile time. It means having two or more methods

with the same name, same signature but with different implementation.

**40. Do we require a parameter for constructors?**

No, we do not require a parameter for constructors.

**41. What is a copy constructor?**

This is a special constructor for creating a new object as a copy of an existing object. There will

always be only one copy constructor that can be either defined by the user or the system.

**42. What does the keyword virtual represented in the method definition?**

It means, we can override the method.

**43. Whether static method can use nonstatic members?**

False.

**44. What is a base class, sub class, and super class?**

The base class is the most generalized class, and it is said to be a root class.

A Sub class is a class that inherits from one or more base classes.

The superclass is the parent class from which another class inherits.

**45. What is static and dynamic binding?**

Binding is nothing but the association of a name with the class. Static binding is a binding in which

name can be associated with the class during compilation time, and it is also called as early Binding.

Dynamic binding is a binding in which name can be associated with the class during execution time,

and it is also called as Late Binding.

**46. How many instances can be created for an abstract class?**

Zero instances will be created for an abstract class.

**47. Which keyword can be used for overloading?**

Operator keyword is used for overloading.

**48. What is the default access specifier in a class definition?**

Private access specifier is used in a class definition.

**49. Which OOPS concept is used as reuse mechanism?**

Inheritance is the OOPS concept that can be used as reuse mechanism.

**50. Which OOPS concept exposes only necessary information to the calling functions?**