<u>Python Interview Questions:</u>

Q1. 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.

Q2. 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 element whereas lists can hold any data type elements.

Q3. 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.

Q4. 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:
a = lambda x,y : x+y
print(a(5, 6))

Q5. 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.

Q6. What are python iterators?
Ans: Iterators are objects which can be traversed through or iterated upon.

Q7.What is __init__?
Ans: __init__ is a method or constructor in Python. This method is automatically called to allocate memory when a new object/ instance of a class is created. All classes have the __init__ method.

```
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
```

Q8. Python dictionary vs list, which is faster?
Ans: The dict is based on hashing, so it is much quicker than linear list search. In Python, the average time complexity of a dictionary key lookup is O(1), since they are implemented as hash tables. The time complexity of lookup in a list is O(n) on average.

Q9. 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.

Q10. What are the generators in python?
Ans: Functions that return an iterable set of items are called generators.

Q11. 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.

Q12. 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.

Q13. 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’.

Q14. 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:

```
# m.py
class MyClass:
def f(self):
print "f()"
```

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

```
 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.

Q15. What are Decorators?
Ans: Decorators are a very powerful and useful tool in Python as they are the specific change that we make in Python syntax to alter functions easily.

Q16. What is the maximum possible length of an identifier?
Ans: Identifiers in Python can be of any length.

Q17. 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.

```
dict={'Country':'India','Capital':'Delhi','PM':'Modi'}
print dict[Country]
Output:India
print dict[Capital]
Output:Delhi
print dict[PM]
Output:Modi
```

Q18. Differentiate between List and Tuple?
Ans: Let's analyze the differences between List and Tuple:
List

- Lists are Mutable datatype.
- Lists consume more memory
- The list is better for performing operations, such as insertion and deletion.
- Implication of iterations is Time-consuming

Tuple

- Tuples are Immutable datatype.
- Tuple consume less memory as compared to the list
- Tuple data type is appropriate for accessing the elements
- Implication of iterations is comparatively Faster

Q19. 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.

Q20. 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.

Q21.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.

Q22. 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.

Q23. 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

Q24. 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.

Q25. 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.

Q26. What does len() do?
Ans: It is used to determine the length of a string, a list, an array, etc.
Example:

```
stg='ABCD'
len(stg)
Output:4
```

Q27. 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 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, etc.

Q28. 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.

Q29. 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.

Q30. 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.

Q31. 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

Q32. What is the difference between Mutable datatype and Immutable datatype?
Ans: Mutable data types can be edited i.e., they can change at runtime. Eg – List, Dictionary, etc.
Immutable data types can not be edited i.e., they can not change at runtime. Eg – String, Tuple, etc.

Q33. What is the difference between Set and Dictionary?
Ans: Set is an unordered collection of data type that is iterable, mutable, and has no duplicate elements.
Dictionary in Python is an unordered collection of data values, used to store data values like a map.

Q34. 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.

Q35. Define encapsulation in Python?
Ans: Encapsulation means binding the code and the data together. A Python class in an example of encapsulation.

Q36. 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.

Q37. 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.

Q38. Write a program in Python to check if a number is prime.
Ans:

```
a=int(input("enter number"))
if a=1:
  for x in range(2,a):
      if(a%x)==0:
       print("not prime")
  break
  else:
    print("Prime")
else:
  print("not prime")
```

Q39. What is difference between / and // in Python?
Ans: // represents floor division whereas / represents precised division. For Example:
5//2 = 2
5/2 = 2.5

Q40. Define encapsulation in Python?
Ans: Encapsulation means binding the code and the data together. A Python class is an example of encapsulation.

Q41. How is Exceptional handling done in Python?
Ans: There are 3 main keywords i.e. try, except, and finally which are used to catch exceptions and handle the recovering mechanism accordingly. Try is the block of a code which is monitored for errors. Except block gets executed when an error occurs.
The beauty of the final block is to execute the code after trying for error. This block gets executed irrespective of whether an error occurred or not. Finally block is used to do the required cleanup activities of objects/variables.

Q42. 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.

Q43. What is PIP?
Ans: PIP is an acronym for Python Installer Package which provides a seamless interface to install various Python modules. It is a command-line tool that can search for packages over the internet and install them without any user interaction.

Q44. Is Python a compiled language or an interpreted language?
Ans: Actually, Python is a partially compiled language and partially interpreted language. The compilation part is done first when we execute our code and this will generate byte code and internally this byte code gets converted by the python virtual machine(p.v.m) according to the underlying platform(machine+operating system).

Q45. How memory management is done in Python?
Ans: Python uses its private heap space to manage the memory. Basically, all the objects and data structures are stored in the private heap space. Even the programmer can not access this p[rivate space as the interpreter takes care of this space. Python also has an inbuilt garbage collector, which recycles all the unused memory and frees the memory and makes it available to the heap space.

Q46. What are Python namespaces? Why are they used?
Ans: A namespace in Python ensures that object names in a program are unique and can be used without any conflict. Python implements these namespaces as dictionaries with 'name as key' mapped to a corresponding 'object as value'. This allows for multiple namespaces to use the same name and map it to a separate object. A few examples of namespaces are as follows:
Local Namespace includes local names inside a function. the namespace is temporarily created for a function call and gets cleared when the function returns.
Global Namespace includes names from various imported packages/ modules that are being used in the current project. This namespace is created when the package is imported in the script and lasts until the execution of the script.
Built-in Namespace includes built-in functions of core Python and built-in names for various types of exceptions.
The lifecycle of a namespace depends upon the scope of objects they are mapped to. If the scope of an object ends, the lifecycle of that namespace comes to an end. Hence, it isn't possible to access inner namespace objects from an outer namespace.

Q47. What are Dict and List comprehensions?
Ans: Python comprehensions, like decorators, are syntactic sugar constructs that help build altered and filtered lists, dictionaries, or sets from a given list, dictionary, or set. Using comprehensions saves a lot of time and code that might be considerably more verbose (containing more lines of code). Let's check out some examples, where comprehensions can be truly beneficial:
Performing mathematical operations on the entire list
```
my_list = [2, 3, 5, 7, 11]
squared_list = [x**2 for x in my_list]    # list comprehension
# output => [4 , 9 , 25 , 49 , 121]
squared_dict = {x:x**2 for x in my_list}    # dict comprehension
# output => {11: 121, 2: 4 , 3: 9 , 5: 25 , 7: 49}
```
Performing conditional filtering operations on the entire list
```
my_list = [2, 3, 5, 7, 11]
squared_list = [x**2 for x in my_list if x%2 != 0]    # list comprehension
# output => [9 , 25 , 49 , 121]
squared_dict = {x:x**2 for x in my_list if x%2 != 0}    # dict comprehension
# output => {11: 121, 3: 9 , 5: 25 , 7: 49}
```
Combining multiple lists into one
Comprehensions allow for multiple iterators and hence, can be used to combine multiple lists into one.
```
a = [1, 2, 3]
b = [7, 8, 9]
[(x + y) for (x,y) in zip(a,b)]  # parallel iterators
```

```
# output => [8, 10, 12]
[(x,y) for x in a for y in b]    # nested iterators
# output => [(1, 7), (1, 8), (1, 9), (2, 7), (2, 8), (2, 9), (3, 7), (3, 8), (3, 9)]
```

Flattening a multi-dimensional list

A similar approach of nested iterators (as above) can be applied to flatten a multi-dimensional list or work upon its inner elements.

```
my_list = [[10,20,30],[40,50,60],[70,80,90]]
flattened = [x for temp in my_list for x in temp]
# output => [10, 20, 30, 40, 50, 60, 70, 80, 90]
```

Note: List comprehensions have the same effect as the map method in other languages. They follow the mathematical set builder notation rather than map and filter functions in Python.

Q48. Is it possible to call parent class without its instance creation?
Ans: Yes, it is possible if the base class is instantiated by other child classes or if the base class is a static method.

Q49. What is pass in Python?
Ans: The pass keyword represents a null operation in Python. It is generally used for the purpose of filling up empty blocks of code which may execute during runtime but has yet to be written. Without the pass statement in the following code, we may run into some errors during code execution.

```
def myEmptyFunc():
   # do nothing
   pass
myEmptyFunc()    # nothing happens
## Without the pass keyword
# File "<stdin>", line 3
# IndentationError: expected an indented block
```

Q50. What is PYTHONPATH in Python?
Ans: PYTHONPATH is an environment variable which you can set to add additional directories where Python will look for modules and packages. This is especially useful in maintaining Python libraries that you do not wish to install in the global default location.

Q51. What is the use of help() and dir() functions?
Ans: help() function in Python is used to display the documentation of modules, classes, functions, keywords, etc. If no parameter is passed to the help() function, then an interactive help utility is launched on the console.
dir() function tries to return a valid list of attributes and methods of the object it is called upon. It behaves differently with different objects, as it aims to produce the most relevant data, rather than the complete information.

For Modules/Library objects, it returns a list of all attributes, contained in that module.
For Class Objects, it returns a list of all valid attributes and base attributes.
With no arguments passed, it returns a list of attributes in the current scope.

Q52. What is PEP 8 and why is it important?
Ans: PEP stands for Python Enhancement Proposal. A PEP is an official design document providing information to the Python community, or describing a new feature for Python or its processes. PEP 8 is especially important since it documents the style guidelines for Python Code. Apparently contributing to the Python open-source community requires you to follow these style guidelines sincerely and strictly.

Q53. Show me three different ways of fetching every third item in the list.
Ans: There are three different ways to do this: via a yield function, a list comprehension, or a for loop.

Q54. Using list comprehension, print the odd numbers between 0 and 100.
Ans: List comprehensions are a feature in Python that allows us to work with algorithms within the default list data structure in Python. Here, we're looking for odd numbers.

```
[x for x in range(100) if x%2]
```

A list comprehension allows us to simplify our range and filtering algorithm so we can pack it into one line of code—return every element in the range between 0 and 100, if it divides perfectly by 2.
Be prepared to describe the list comprehension in detail with the right notation if you need to write it down on paper, or recite it over the phone as coding interviews sometimes demand.

Q55. what will be the output of this? print(min(max(False,-3,-4), 2,7))
Ans:
1. False==0 and true==1 (Bool Operator), max(False,-3,-4) = False.
2. print(min(False,2,7))
3. print(False)
4. Result --> False

Q56. What would be the output if I run the following code block?

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

Ans: 14, Negative Indexing is supported in Python's List, it will output second last element of the list.

Q57. What arithmetic operators cannot be used with strings? '+', '-', '*' or all of them?
Ans: '-' can't be used with strings. '+' and '*' are used usually used. Ex: 'abc' + 'def' = 'abcdef' or 'aa'*3 = 'aaaaaa'.

Q58. Solve the following:

```
c,d = 1.8, 120000
c,d = d, c
print(d)
```

Ans: 1.8. Simple swap in python.

Q59. Solve the following:

```
Tuple= (2,3,4)
Tuple.append(4)
print(Tuple)
```

Ans: Tuple is immutable so we can not append here it will give attribute error.

Q60. Solve the following:
set = {1, 2, 3}
set.add(1)
print(set)
Ans: A Set has the attribute of arranging the data in ascending order and removing all the duplicates. In this case, it appended the data which is already in the set hence the resultant is {1,2,3}.

Q61. program to check given number representation is in binary or not
Ans:
```
num = int(input("please give a number : "))
while(num>0):
   j=num%10
   if j!=0 and j!=1:
      print("num is not binary")
      break
   num=num//10
   if num==0:
      print("num is binary")
```

Q62. Python program for palindrome number using recursive method
Ans:
```
n = int(input("please give a number : "))
def reverse(num):
   if num<10:
     return num
   else:
     return int(str(num%10) + str(reverse(num//10)))
def isPalindrome(num):
   if num == reverse(num):
      return 1
   return 0
if isPalindrome(n) == 1:
   print("Given number is a palindrome")
else:
   print("Given number is a not palindrome")
```

Q63. Python program to reverse a number
Ans:
```
n = int(input("please give a number : "))
print("before reverse your numeber is : %d" %n)
reverse = 0
while n!=0:
   reverse = reverse*10 + n%10
   n = (n//10)
print("After reverse : %d" %reverse)
```

Q64. What does [::-1] do?
Ans: [::-1], this is an example of slice notation and helps to reverse the sequence with the help of indexing.
[Start,stop,step count]
Let's understand with an example of an array:

```
import array as arr
Array_d=arr.array('i',[1,2,3,4,5])
Array_d[::-1]               #reverse the array or sequence
Output: 5,4,3,2,1
```

Q65. Program to check a number is Armstrong or not in python programming language
Ans:

```
i=0
result=0
n = int(input("please give a number : "))
number1 = n
temp = n
while n!=0:
   n = (n//10)
   i=i+1;
while number1!=0:
   n=number1%10
   result=result+pow(n,i)
   number1=number1//10
if temp==result:
   print("number is armstrong")
else:
   print("number is not armstrong")
```