**Python**

**What is Python?**

Python is an object-oriented language, which means that it can model real-world objects. It is also dynamically-typed because it carries out type-checking at runtime.

The distinctive feature about Python is that it is an interpreted language. The Python IDLE (Integrated DeveLopment Environment) executes instructions a line at a time.

**1. Why python is Interpreted language**.

**Ans :** Python programs runs directly from source code each type python programs

are executed code is required. python converts source code to interpreted code

and again converted into machine language that code is executed. So python is

interpreted language.

**Difference between interpreter and compiler:**

|  |  |
| --- | --- |
| Interpreter | Compiler |
| Translates program one statement at a time.  It takes a single line of code or instruction at a time. | Scans the entire program and translates it as a whole into machine code.  It takes an entire program at a time. |
| It takes less amount of time to analyze the source code but the overall execution time is slower. | It takes large amount of time to analyze the source code but the overall execution time is comparatively faster. |
| No intermediate object code is generated, hence are memory efficient. | Generates intermediate object code which further requires linking, hence requires more memory. |
| Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy. | It generates the error message only after scanning the whole program. Hence debugging is comparatively hard. |
| Programming language like Python, Ruby use interpreters. | Programming language like C, C++ use compilers. |

**What is a middleware**

Middlewares are hooks to modify Django request or response object. Putting the definition of middleware from Django docs.

Middleware is a framework of hooks into Django’s request/response processing. It’s a light, low-level “plugin” system for globally altering Django’s input or output.

**2**.**What is the difference b/w list tuple and dictionary**.

**Ans :** List is mutable type meaning that list can be modified after they have been

created. A tuple is similar to list except it is immutable .Tuples have structure

and list have order,Tuples are heterogeneous data structure,while list are

homogeneous sequence. A dictionary is a key-value store. It is not ordered and

it requires that the keys are hashable. It is fast for loop by key.

**3. Slicing a List**

Sometimes, you **may not want** an entire list or a single item, but a number of items from it. Here, the slicing operator [:] comes into play.

Suppose we want items second through fourth from list ‘list1’. We write the following code for this.

>>> **list**1=[1,[2,3],(4,5),False,'No']

>>> **list**1[1:4]

[[2, 3], (4, 5), False]

**3**.**Can we use tuple or list as a key in dictionary.?**

**Ans :** Only immutable elements can be used as dictionary keys, and hence only tuples and

not lists can be used as keys.

**4**.**Can we create dictionary of dictionary in python.?Ex= {{}}**

**Ans:** No

**5.Slicing in python.**

**Ans:** Slicing operator allows to retrieve a part of sequence.

### **Definition of Methods**

A method differs from a function only in two aspects:

it belongs to a class and it is defined within a class

the first parameter in the definition of a method has to be a reference "self" to the instance of the class

a method is called without this parameter "self"

We extend our class by defining some methods. The body of these methods is still not specified:

class Account(object):

def transfer(self, target, amount):

pass

def deposit(self, amount):

pass

def withdraw(self, amount):

pass

def balance(self):

pass

**6**.**Indexing in python.**

**Ans:** Indexing operation allows to access a particular items in sequence directly from list.

**Accessing characters in strings by index in Python:**

Typically it's more useful to access the individual characters of a string by using Python's array-like indexing syntax. Here, as with all sequences, it's important to remember that indexing is zero-based; that is, the first item in the sequence is number 0.

>>> s = 'Don Quijote'

>>> s[4] # Get the 5th character

'Q'

>>> s[-1]

'e

**7. Functional programming in python ? List Comprehension ,Lambda,join,yield,filter,map and decorators.**

**Ans :Functional programming** is all about expressions. We may say that the Functional

programming is an expression oriented programming.

Expression oriented functions of Python provides are:

· map(aFunction, aSequence)

· filter(aFunction, aSequence)

· reduce(aFunction, aSequence)

· lambda

· list comprehension

**[i]**List comprehensions are a tool for transforming one list (any iterable actually) into another list. During this transformation, elements can be conditionally included in the new list and each element can be transformed as needed. List comprehension always returns a list object. Syntax of list comprehension is .

Syntax: **[expression for item in list if conditional]**

**[ii]**Lambda generates a tiny anonymous function can be created without used the return keyword.

[**iii**]Map : The map(aFunction, aSequence) function applies a passed-in function to each item in an iterable object and returns a list containing all the function call results.

[**iv] Filter : filter** extracts each element in the sequence for which the function returns True.

This function reduces a list to a single value by combining elements via a supplied

function.

**[v] Join :** The method **join()** returns a string in which the string elements of sequence have been joined by *str* separator.

[Vi] **yield :**The **yield** enables a function to **suspend** and **resume** while it turns in a value at the time of the suspension of the execution. It returns is a **generator object.**

**[Vii] next() :** The **next()** function takes a generator object and returns its next value**.**

**8**.**What is negative indexing in python.**

**Ans :** python array and list items can be accessed with both positive or negative numbers known as index. For instance our array/list is of size n, then positive index 0 is the first index 1, second last index will be (n-1). For negative index, -n is the first index ,-(-n-1) is the second last negative index will be.

**9**.**Range and Xrange in python.**

**Ans:** Range returns a list object Xrange returns an xrange object. Xrange is faster than range bcs it consume less memory.

**10**.**Difference b/w extend and append.**

**Ans:** Append is used for append single elements item into the list while extend is used for append a list item into another list.

**11**.**Generator in python.**

**Ans : Generator are iterables, but you can only iterate over them once. Its because they do not all values in memory they generate the values on the fly.**

**Ex; mygenerator=(x\*\*x form x in range(10))**

**print(mygenerator)**

**12**.**How to memory manage in python**.

**Ans:** Python also has a garbage collector I.e gc module, gc module manage the memory management in python.

**gc.enable()**

**gc.disable()**

**13**.**Different b/w compiled and interpreted language.**

**Ans:** Interpreted language is executed at the run time according to the instructions like in shell scripting and compiled language is one which is compiled (changed into Assembly language, which CPU can understand ) and then executed like in c++.

|  |
| --- |
|  |

**14**.**Thread and multi-threading in python** .

Ans:**Python multi-threading** uses **threads** to do parallel **processing**. This is the most common way to do parallel work in many programming languages

Multi threading and Multiprocessing both ways are to achieve multitasking.

ii) **Difference between multi-threading and multi-processing.**

Thread are light weight .Thread share the same address space with each thread. Each process have their own address space.

**15.** **Modules and packages.**

**Ans :** A modules is a single file in this file we declare a variables , function and classes. While a packages is a collection modules in directories that gives packages hierarchy.

**16.How to pass argument in python pass by reference or pass by value.**

**Ans :** Pass By reference.

**17**.**How to pass multiple argument in python.**

**Ans:** If we are not sure how many argument are going to be passed then use the \*args.

**18 .What is the difference between \*args and \*\*kwargs.**

**Ans :**\*args is used to send a non key-worded variable length argument list to the function.

If we want to pass a stored list or tuple of argument to a function,then used the \*\*kwargs.

It can be used to pass the values of dictionary as keyword arguments.

**18**.**Global and Local variable in python**

**Ans :**A variable declare inside the function it is called local variable,where as variable declare outside

function it is said to be global variable.

**19**.**Shallow and deep copy in python.**

**Ans :** A shallow copy constructs a new compound objects and then inserts references into it to the objects found in the original. A deep copy constructs a new compound object and then recursively inserts copies into it the objects found in the original.

**20**.**Exception handling in python**

**Ans :** Error detected during execution of a program is called exception. Exception can be handled using the try....except statement.

**21**. **Pickling and unpickling in python**

**Ans:**A Pickle is a standard module which serialized or deserialized theobject structure .pickle module accepts any python objects converts into a string and dumps into a file, which can be used later ,process is called pickling.

Unpickling is a process of retrieving original python objects from the stored string for use later.

**22**.**What is anonymous function in python.**

**Ans:** Anonymous function contains (Lambda,filter,reduce and map) function ,these function are called anonymous function.

**23**.**Delegation in python**

**24**.**Why we used \_\_init\_\_ == \_\_main\_\_ in python**.

**Ans :** When the Python interpreter reads a source file, it executes all of the code found in it. Before executing the code, it will define a few special variables. For example, if the python interpreter is running that module (the source file) as the main program, it sets the special **\_\_name\_\_** variable to have a value **\_\_main\_\_**. If this file is being imported from another module, **\_\_name\_\_** will be set to the module's name.

**25 .Unicode**

**Ans :**Unicode is a system to represent characters from all the word's different language.

**26 . Scope in python ?**

A *scope* defines the visibility of a name within a block. If a local variable is defined in a block, its scope includes that block. If the definition occurs in a function block, the scope extends to any blocks contained within the defining one, unless a contained block introduces a different binding for the name.

**27.How to find the missing number from list.**

a=[1,2,3,4,5,7,8,9,10]

a[-1]\*(a[-1] + a[0]) / 2 - sum(a)

**28.How to get largest number from list without using built().**

**Ans :** def maximum(items, default=None):

iterator = iter(items)

m = next(iterator)

for item in iterator:

if item > m:

m = item

return m

status=maximum([9,10,1,0,2,50,3,4,5,6,8,])

**ii) How to sort dictionary by Keys**

mydict = {'carl':40, 'alan':2, 'bob':1, 'danny':3}

for key in sorted(mydict. Iterkeys()):

print "%s: %s" % (key, mydict[key])

**29.How to get smallest number from list in python.**

**Ans :** def maximum(items, default=None):

iterator = iter(items)

m = next(iterator)

for item in iterator:

if item < m:

m = item

return m

status=maximum([9,10,1,0,2,50,3,4,5,6,8,])

**30 . Stack and Queue in python ?**

Ans : **Stack** is a LIFO (last in first out) data structure. The associated link to wikipedia contains detailed description and examples. **Queue** is a FIFO (first in first out) data structure.

For example check below url :

<https://welcomedata.wordpress.com/2015/07/03/using-lists-as-stacks-and-queues-in-python/>

**31.What is Queue() is python.?**

**32.Difference between pass by reference and pass by value ?**

When you pass something by reference what you doing is passing the object it self and when you passing the value doing is creating a copy of that object and variable the passing and passing the copy to the function .

When you pass something by reference any change you make to the variable inside the function those changes are reflected outside value as well and when you pass something by value the changes that you make to the function copy of the variable those changes are not reflected back to the calling function.

**33.Difference between Django and Flask ?**

**Django** is a full blown web development framework, and almost everything is included in the standard package. Admin, ORM,... you'll find it all in **Django**. ... Compare the **Django** tutorial with the **Flask** (A Python Microframework) tutorial and you'll see the **difference** in learning curve

**34.what is middleware in django ?**

Middleware is a framework of hooks into Django’s request/response processing. It’s a light, low-level “plugin” system for globally altering Django’s input or output.

**35.what is managers in django ?**

A Manager is the interface through which database query operations are provided to Django models. At least one Manager exists for every model in a Django application.

The way Manager classes work is documented in [Making queries](https://docs.djangoproject.com/en/1.10/topics/db/queries/); this document specifically touches on model options that customize Manager behavior.

**36.How to sort the dictionary on the basis of keys basis.**

**By Keys:**

mydict = {'carl':40,

'alan':2,

'bob':1,

'danny':3}

for key in sorted(mydict.iterkeys()):

print "%s: %s" % (key, mydict[key])

**By values:**

for key, value in sorted(mydict.iteritems(), key=lambda (k,v): (v,k)):

print "%s: %s" % (key, value)

**37. Static Variable ?**

Variables declared inside the class definition, but not inside a method are class or static variables.

**38. Django**

**Please follow below url for django Interview question.**

[**http://career.guru99.com/top-16-django-interview-questions/**](http://career.guru99.com/top-16-django-interview-questions/)

**31.Pandas :** Pandas is a python package providing fast flexible and expressive data structure designed to make working with 'relational' or both easy initiative.

**32.Numpy :** Numy is used for array processing for number string record and object.

OR

Numpy is array processing package designed to efficiently manipulate large dimensional array of arbitrary records without sacrificing too much speed for multi dimensional array.

Ex: Suppose I have two python list:

a) list1 = [1.5,2.5,3.5,4.5]

b) list2 =[1.5,2.5,3.5,4.5]

print(list1\*list2)-------Non int type will produce

then resolve this through numpy

import numpy

list1 =numpy.array([1.5,2.5,3.5,4.5])

list2 =numpy.array([1.5,2.5,3.5,4.5])

print(list1\*list2)------ Result Success

**Python oop:** There are three widely used programming paradigms there. Procedural programming, functional programming and object-oriented programming. Python supports both procedural and object-oriented programming. There is some limited support for functional programming too.

**Objects:**Everything in Python is an object. Objects are basic building blocks of a Python OOP program.

**Class :** The class is a blueprint that defines a nature of a future object. From classes we construct instances. Technically, we can say that attributes are variables and methods are functions defined inside a class.

All the below value returns object. Therefore we can say that all are object in python.

print type(1)

print type("")

print type([])

print type({})

print type(())

print type(object)

print type(function)

print type(sys)

**Attribute :** An *attribute* is a characteristic of an object.

**Method :** Methods are functions defined inside the body of a class. They are used to perform operations with the attributes of our objects.

**Inheritance :** Inherit the properties from base class to derived class this is called inheritance. Important benefits of inheritance are code reuse and reduction of complexity of a program.

**Polymorphism** : The polymorphism is the process of using an operator or function in different ways for different data input.

In another word we can say that to take more than one form.

Eg :

In practical terms, polymorphism means that if class B inherits from class A, it doesn’t have to inherit everything about class A; it can do some of the things that class A does differently.

**Encapsulation** : The wrapping of function and data into a single logical unit is known as encapsulation.

**Operator Overloading :** same operator to have different meaning according to the context is called operator overloading.

**Function Overloading** : The assignment of more than one behavior to a particular function,

Eg : Same function but different argument.

**Constructor :** Constructor is always written as a function called \_\_init\_\_().It always takes as first argument a reference to the instance being constructed .This is typically called self.

**Self** : Self is a reference variable.

….....End....................

**Logical Question:**

If you are given an array of numbers as input and you can use a function which returns a random number between 0 and 1, can you make a function which returns elements from the array randomly without repeating.

**Fullstack:** FullStack is mainly based on three tier of Architecture.

i)Presentation layer : This involves with the UI related,like HTML,CSS and JavaScript.

ii) Business Logic Layer: This contains the application programming.

iii)Data Access layer : Data persistence & data access through API(we will discuss of database section ).