Top Python Interview Questions and Answers (PDF) for 2023

By Steve Campbell Updated January 14, 2023

Here are Python interview questions and answers for fresher as well as experienced candidates to get their dream job.

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 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 is Python 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 is memory 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 the interpreter takes care of this Python private heap.

The allocation of Python heap space for Python objects is done by the Python memory manager. The core API gives access to some tools for the programmer to code.

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.

6) What are the tools that help to find bugs or perform the 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 example., 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 reference values are according to the functions. Therefore, 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 built-in type does python provides?

Python provides two built-in types: 1) Mutable and 2) Immutable.

Mutable built-in types are:

- List
- Sets
- Dictionaries
- Immutable built-in types
- Strings
- Tuples
- Numbers

Immutable built-in types are:

- Strings
- Tuples
- Numbers

12) Explain namespace in Python

In Python, every name introduced has a place where it lives and can be hooked for. This is known as a 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 the 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 do 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) Explain pass in Python

Pass means no-operation Python statement, or in other words, it is a place holder in a 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 a list.

17) What is the 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) Explain slicing in Python?

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 a `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 the negative index, (-1) is the last index, and (-2) is the second last index, and so forth.

23) How can you 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 a 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) What are the rules for local and global variables in Python?

Here 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) 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) What is the use of // operator in Python?

It is a Floor Divisionoperator, which is used for dividing two operands with the result as a 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

Here are the 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 Flask and 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 a little dependency to update and less security bugs.

36) What is the difference between Django, Pyramid, and Flask?

Flask is a “microframework” primarily build for a small application with simpler requirements. In a flask, you don’t have to use external libraries. Flask is ready to use.

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

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

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

- Integration with WTFforms
- 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 that 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 applications (RDBS). Such a 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():** It is called before a request and pass no arguments
- **after_request():** It is called after a request and pass the response that will be sent to the client
- **teardown_request():** It is called in a situation when exception is raised, and response is not guaranteed. They are called after the response has been constructed. They are not allowed to modify the request, and their values are ignored.

42) If you have multiple Memcache servers, and one of them fails that contain data, will it try to get them?

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 a larger load on the database server when lost data is reloaded as the 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 machine's 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 a semaphore lock. In this system, when the value expires, the first process acquires the lock and starts generating a new value.

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

Here are the ways you should not use memcached 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.

46) What is Python If Statement?

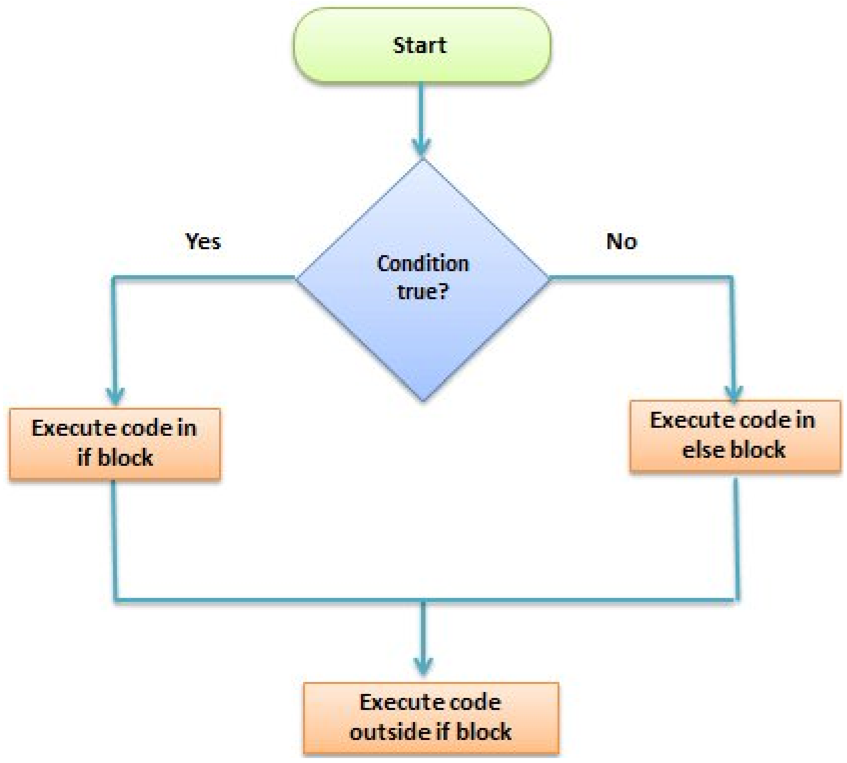
Python if Statement is used for decision-making operations. It contains a body of code that runs only when the condition given in the if statement is true. If the condition is false, then the optional else statement runs, which contains some code for the else condition.

When you want to justify one condition while the other condition is not true, then you use Python if-else statement.

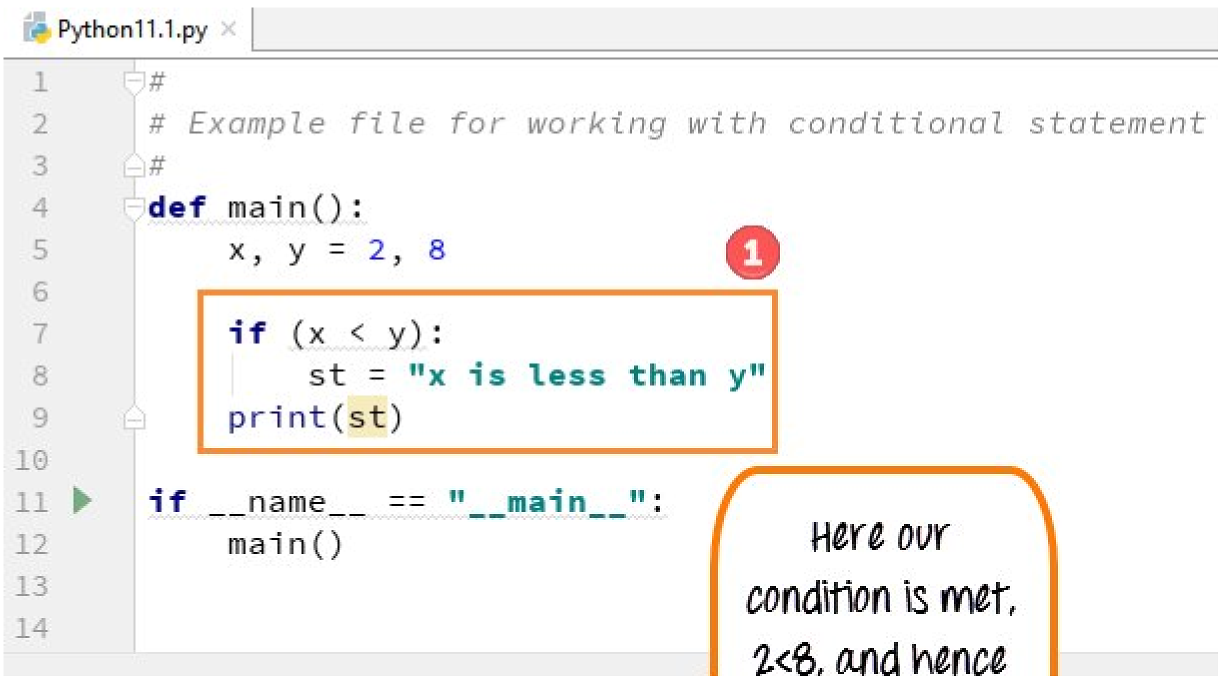
Python if Statement Syntax:

```
if expression
Statement
else
Statement
```

Python if…else Flowchart



Let's see an example of Python if else Statement:


Python11.1.py

```
#
# Example file for working with conditional statement
#
def main():
    x, y = 2, 8

    if (x < y):
        st = "x is less than y"
    print(st)

if __name__ == "__main__":
    main()
```

```
"C:\Users\DK\Deskto
x is less than y
```

Let’s see an example of Python if else Statement:

```
def main():
    x,y =2,8

    if(x < y):
        st= "x is less than y"
    print(st)

if __name__ == "__main__":
    main()
```

---

47) Explain While loop in Python with example

While loop does the exact same thing what “if statement” does, but instead of running the code block once, they jump back to the point where it began the code and repeat the whole process again.

The syntax of while loop is as follows:

```
while expression
Statement
```

The example of while loop is as follows:

```
x=0
#define a while loop
while(x <4):
        print(x)
        x = x+1
```

48) What is enumerate() in Python?

Enumerate() in Python is a built-in function used for assigning an index to each item of the iterable object. It adds a loop on the iterable objects while keeping track of the current item and returns the object in an enumerable form. This object can be used in a for loop to convert it into a list by using list() method.

Example of enumerate() is as follows:

Suppose we want to do numbering for our month (Jan, Feb, Marc, ….June), so we declare the variable i that enumerate the numbers while m will print the number of month in list.

```
#use a for loop over a collection
Months = ["Jan","Feb","Mar","April","May","June"]
for i, m in enumerate (Months):
        print(i,m)

# use the break and continue statements

        #for x in range (10,20):
        #if (x == 15): break
        #if (x % 5 == 0) : continue
        #print x
```

49) How can you use for loop to repeat the same statement over and again?

You can use for loop for even repeating the same statement over and again. Here in the example, we have printed out the word “guru99” three times.

Example:

To repeat the same statement a number of times, we have declared the number in variable i (i in 123). So when you run the code as shown below, it prints the statement (guru99) that many times the number declared for our the variable in (i in 123).

```
for i in '123':
 print ("guru99",i,)
```

50) What is Tuple Matching in Python?

Tuple Matching in Python is a method of grouping the tuples by matching the second element in the tuples. It is achieved by using a dictionary by checking the second element in each tuple in python programming. However, we can make new tuples by taking portions of existing tuples.

Syntax:
`Tup = ('Jan','feb','march')`

To write an empty tuple, you need to write as two parentheses containing nothing-
`tup1 = ();`

51) Explain Dictionary in Python with example

A Dictionary in Python is the unordered and changeable collection of data values that holds key-value pairs. Each key-value pair in the dictionary maps the key to its associated value making it more optimized. A Dictionary in python is declared by enclosing a comma-separated list of key-value pairs using curly braces({}). Python Dictionary is classified into two elements: Keys and Values.

Syntax for Python Dictionary:
`Dict = { ' Tim': 18, xyz,.. }`

Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print((Dict['Tiffany']))
```

52) How can you copy the entire dictionary to a new dictionary?

You can also copy the entire dictionary to a new dictionary. For example, here we have copied our original dictionary to the new dictionary name “Boys” and “Girls”.

Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
studentX=Boys.copy()
studentY=Girls.copy()
print(studentX)
print(studentY)
```

53) How can you Update Python Dictionary?

You can update a dictionary by adding a new entry or a key-value pair to an existing entry or by deleting an existing entry. Here in the example, we will add another name, “Sarah” to our existing dictionary.

Example

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Dict.update({"Sarah":9})
print(Dict)
```

54) Give example of dictionary items() method

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print("Students Name: %s" % list(Dict.items()))
```

55) How can you sort elements in Python dictionary?

In the dictionary, you can easily sort the elements. For example, if we want to print the name of the elements of our dictionary alphabetically, we have to use for loop. It will sort each element of the dictionary accordingly.

Example:

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
Students = list(Dict.keys())
Students.sort()
for S in Students:
      print(":".join((S,str(Dict[S]))))
```

56) Give an example of Dictionary len() and Python List cmp() method

Dictionary len() Example:

```
Dict = {'Tim': 18,'Charlie':12,'Tiffany':22,'Robert':25}
print("Length : %d" % len (Dict))
```

cmp() Example:

```
Boys = {'Tim': 18,'Charlie':12,'Robert':25}
Girls = {'Tiffany':22}
print cmp(Girls, Boys)
```

57) What are all dictionary methods:

Here is the list of dictionary methods:

- copy()
- update()

- items()
- sort()
- len()
- cmp()
- Str()

---

58) Explain Arithmetic operators with example

Arithmetic Operators perform various arithmetic calculations like addition, subtraction, multiplication, division, %modulus, exponent, etc. There are various methods for arithmetic calculation in Python, like you can use the eval function, declare variable & calculate, or call functions.

Example: For arithmetic operators, we will take a simple example of addition where we will add two-digit 4+5=9

```
x= 4
y= 5
print(x + y)
```

---

59) Give example of logical operators

Example of logical operators:

```
a = True
b = False
print(('a and b is',a and b))
print(('a or b is',a or b))
print(('not a is',not a))
```

---

60) Explain membership operators with example

These operators test for membership in a sequence such as lists, strings, or tuples. Two membership operators are used in Python. (in, not in). It gives the result based on the variable present in a specified sequence or string.

Example:

For example here, we check whether the value of x=4 and value of y=8 is available in list or not by using in and not in operators.

```
x = 4
y = 8
list = [1, 2, 3, 4, 5 ];
if ( x in list ):
   print("Line 1 - x is available in the given list")
else:
   print("Line 1 - x is not available in the given list")
if ( y not in list ):
   print("Line 2 - y is not available in the given list")
else:
   print("Line 2 - y is available in the given list")
```

61) Write code to demonstrate operator precedence in Python:

```
v = 4
w = 5
x = 8
y = 2
z = 0
z = (v+w) * x / y;
print("Value of (v+w) * x/ y is ",  z)
```

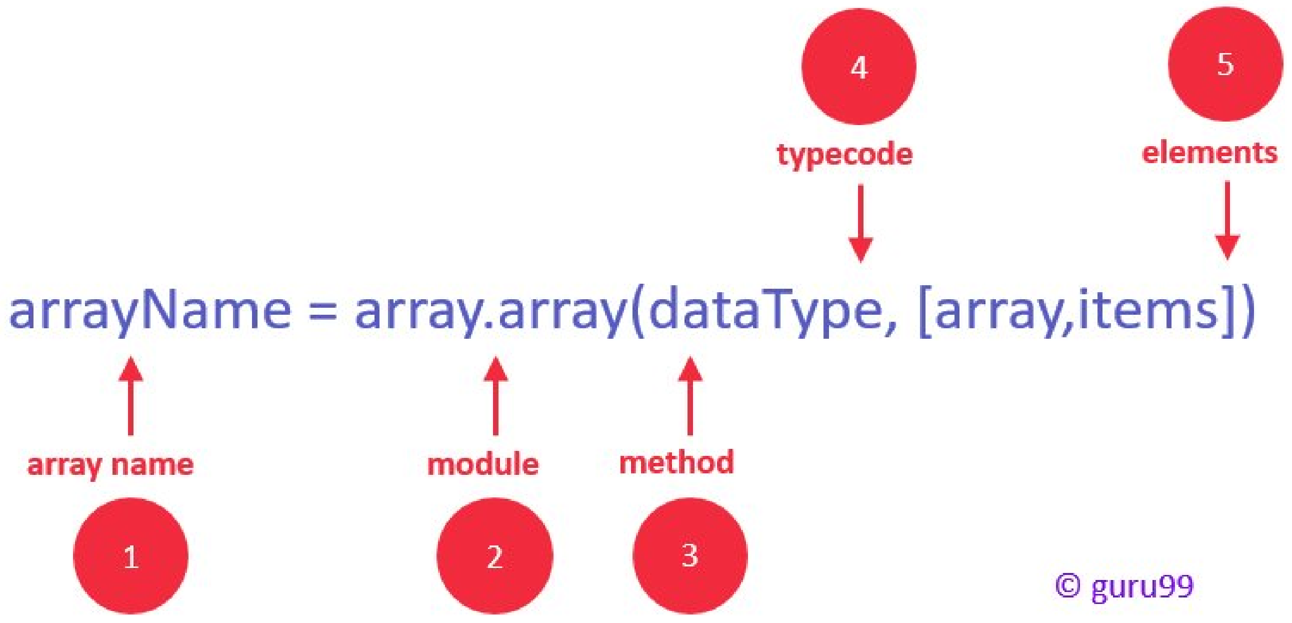
62) Explain arrays in Pythons with example

A Python Array is a collection of a common type of data structures having elements with the same data type. It is used to store collections of data. In Python programming, arrays are handled by the “array” module. If you create arrays using the array module, elements of the array must be of the same numeric type.

Syntax to Create an Array in Python

You can declare an array in Python while initializing it using the following syntax.
`arrayName = array.array(type code for data type, [array,items])`

The following image explains the syntax.



Array Syntax

1. **Identifier**: specify a name like usually, you do for variables
2. **Module**: Python has a special module for creating array in Python, called "array" – you must import it before using it
3. **Method**: the array module has a method for initializing the array. It takes two arguments, type code, and elements.
4. **Type Code**: specify the data type using the type codes available (see list below)
5. **Elements**: specify the array elements within the square brackets, for example [130,450,103]

Example

```
import array as myarray
abc = myarray.array('d', [2.5, 4.9, 6.7])
```

63) How can you access array elements?

You can access any array item by using its index.

The syntax is

`arrayName[indexNum]`

Example

```
import array
balance = array.array('i', [300,200,100])
print(balance[1])
```

---

64) How can you insert elements in array?

Python array insert operation enables you to insert one or more items into an array at the beginning, end, or any given index of the array. This method expects two arguments index and value.

The syntax is

`arrayName.insert(index, value)`

Example

Let us add a new value right after the second item of the array. Currently, our balance array has three items: 300, 200, and 100. Consider the second array item with a value of 200 and index 1.

In order to insert the new value right "after" index 1, you need to reference index 2 in your insert method, as shown in the below Python array example:

```
import array
balance = array.array('i', [300,200,100])
balance.insert(2, 150)
print(balance)
```

65) How can you delete elements in array?

With this operation, you can delete one item from an array by value. This method accepts only one argument, value. After running this method, the array items are re-arranged, and indices are re-assigned.

The syntax is

`arrayName.remove(value)`

Example

Let's remove the value of "3" from the array

```
import array as myarray
first = myarray.array('b', [2, 3, 4])
first.remove(3)
print(first)
```

66) How can you search and get the index of a value in an array?

With this operation, you can search for an item in an array based on its value. This method accepts only one argument, value. It is a non-destructive method, which means it does not affect the array values.

The syntax is

`arrayName.index(value)`

Example

Let's find the value of "3" in the array. This method returns the index of the searched value.

```
import array as myarray
```

```
number = myarray.array('b', [2, 3, 4, 5, 6])
print(number.index(3))
```

67) How can you reverse array in Python?

You can use reverse() to reverse array in Python.

Example:

```
import array as myarray
number = myarray.array('b', [1,2, 3])
number.reverse()
print(number)
```

68) Give example to convert array to Unicode

The Example to convert array to Unicode is:

```
from array import array
p = array('u',
[u'\u0050',u'\u0059',u'\u0054',u'\u0048',u'\u004F',u'\u004E'])
print(p)
q = p.tounicode()
print(q)
```

69) Give an example of a class in Python

Example of class in Python

```
                # Example file for working with classes
class myClass():
  def method1(self):
      print("Guru99")
  def method2(self,someString):
      print("Software Testing:" + someString)

   def main():
  # exercise the class methods
  c = myClass ()
  c.method1()
  c.method2(" Testing is fun")

if __name__== "__main__":
  main()
```

---

70) Explain Inheritance with example

Inheritance is a feature used in object-oriented programming; it refers to defining a new class with less or no modification to an existing class. The new class is called the derived class, and from one which it inherits is called the base. Python supports inheritance; it also supports multiple inheritances. A class can inherit attributes and behavior methods from another class called subclass or heir class.

Example of inheritance:

```
# Example file for working with classes
class myClass():
  def method1(self):
      print("Guru99")

 class childClass(myClass):
  #def method1(self):
        #myClass.method1(self);
        #print ("childClass Method1")

  def method2(self):
        print("childClass method2")

def main():
  # exercise the class methods
  c2 = childClass()
  c2.method1()
  #c2.method2()

if __name__== "__main__":
  main()
```

71) Give example of Python constructors

Example of Python Constructors

```
class User:
    name = ""

    def __init__(self, name):
        self.name = name

    def sayHello(self):
        print("Welcome to Guru99, " + self.name)

User1 = User("Alex")
User1.sayHello()
```

72) How can you access values in string?

Python does not support a character type, these are treated as strings of length one, also considered as a substring.

You can use square brackets for slicing along with the index or indices to obtain a substring.

```
var1 = "Guru99!"
var2 = "Software Testing"
print ("var1[0]:",var1[0])
print ("var2[1:5]:",var2[1:5])
```

73) Explain all string operators with example

String operators with example:

Operator	Description	Example
[]	Slice- it gives the letter from the given index	a[1] will give “u” from the word Guru as such (0=G, 1=u, 2=r and 3=u)
[:]	Range slice-it gives the characters from the given range	x [1:3] it will give “ur” from the word Guru. Remember it will not consider 0, which is G, it will consider word after that is ur.
in	Membership-returns true if a letter exists in the given string	u is present in word Guru, and hence it will give 1 (True)
not in	Membership-returns true if a letter exists is not in the given string	l not present in word Guru and hence it will give 1
r/R	Raw string suppresses the actual meaning of escape characters.	Print r’\n’ prints \n and print R’/n’ prints \n
% – Used for string format	%r – It insert the canonical string representation of the object (i.e., repr(o)) %s- It inserts the presentation string representation of the object (i.e., str(o)) %d- it will format a number for display	The output of this code will be “guru 99”.
+	It concatenates 2 strings	It concatenates strings and gives the result
*	Repeat	It prints the character twice.

74) Give example of sleep() function in Python

Example of sleep() function in Python

```
import time
print("Welcome to guru99 Python Tutorials")
time.sleep(5)
print("This message will be printed after a wait of 5 seconds")
```

---

75) What is timmer method in Python?

Timer is a method available with Threading, and it helps to get the same functionality as Python time sleep.

```
from threading import Timer

print('Code Execution Started')

def display():
    print('Welcome to Guru99 Tutorials')

t = Timer(5, display)
t.start()
```

---

76) Give example of calendar class

Example of calendar class

```
import calendar
# Create a plain text calendar
c = calendar.TextCalendar(calendar.THURSDAY)
str = c.formatmonth(2025, 1, 0, 0)
print(str)

# Create an HTML formatted calendar
```

```
hc = calendar.HTMLCalendar(calendar.THURSDAY)
str = hc.formatmonth(2025, 1)
print(str)
# loop over the days of a month
# zeroes indicate that the day of the week is in a next month or
overlapping month
for i in c.itermonthdays(2025, 4):
    print(i)

    # The calendar can give info based on local such a names of days
and months (full and abbreviated forms)
    for name in calendar.month_name:
        print(name)
    for day in calendar.day_name:
        print(day)
    # calculate days based on a rule: For instance an audit day on
the second Monday of every month
    # Figure out what days that would be for each month, we can use
the script as shown here
    for month in range(1, 13):
        # It retrieves a list of weeks that represent the month
        mycal = calendar.monthcalendar(2025, month)
        # The first MONDAY has to be within the first two weeks
        week1 = mycal[0]
        week2 = mycal[1]
        if week1[calendar.MONDAY] != 0:
            auditday = week1[calendar.MONDAY]
        else:
        # if the first MONDAY isn't in the first week, it must be in
the second week
            auditday = week2[calendar.MONDAY]
print("%10s %2d" % (calendar.month_name[month], auditday))
```

77) Explain Python ZIP file with example

Python allows you to quickly create zip/tar archives.

Following command will zip the entire directory
`shutil.make_archive(output_filename, 'zip', dir_name)`

Following command gives you control on the files you want to archive

`ZipFile.write(filename)`

Example of Python ZIP file

```
import os
import shutil
from zipfile import ZipFile
from os import path
from shutil import make_archive

    # Check if file exists
       if path.exists("guru99.txt"):
    # get the path to the file in the current directory
        src = path.realpath("guru99.txt");
    # rename the original file
        os.rename("career.guru99.txt","guru99.txt")
    # now put things into a ZIP archive
        root_dir,tail = path.split(src)
        shutil.make_archive("guru99 archive","zip",root_dir)
    # more fine-grained control over ZIP files
        with ZipFile("testguru99.zip", "w") as newzip:
            newzip.write("guru99.txt")
            newzip.write("guru99.txt.bak")
```

78) What are the common examples of exceptions in Python?

The common examples of exceptions in Python are:

- Division by Zero
- Accessing a file that does not exist.
- Addition of two incompatible types
- Trying to access a nonexistent index of a sequence
- Removing the table from the disconnected database server.
- ATM withdrawal of more than the available amount

79) Explain important Python errors

The important Python errors are 1) ArithmeticError, 2) ImportError, and 3) IndexError.

- **ArithmeticError:** ArithmeticError act as a base class for all arithmetic exceptions. It is raised for errors in arithmetic operations.
- **ImportError:** ImportError is raised when you are trying to import a module which does not present. This kind of exception occurs if you have made a typing mistake in the module name or the module which is not present in the standard path.
- **IndexError:** An IndexError is raised when you try to refer a sequence which is out of range.

---

80) Explain JSON dumps() in Python with example

json.dumps() in Python is a method that converts dictionary objects of Python into JSON string data format. It is useful when the objects are required to be in string format for the operations like parsing, printing, etc.

Example:

```
import json

x = {
  "name": "Ken",
  "age": 45,
  "married": True,
  "children": ("Alice","Bob"),
  "pets": ['Dog'],
  "cars": [
    {"model": "Audi A1", "mpg": 15.1},
    {"model": "Zeep Compass", "mpg": 18.1}
  ]
}
# sorting result in asscending order by keys:
sorted_string = json.dumps(x, indent=4, sort_keys=True)
print(sorted_string)
```

---

81) Explain in detail JSON to Python (Decoding) with example

JSON string decoding is done with the help of inbuilt method `json.loads()` & `json.load()` of JSON library in Python.

Here translation table show example of JSON objects to Python objects which are helpful to perform decoding in Python of JSON string.

JSON	Python
Object	Dict
Array	List
String	Unicode
number – int	Number – int, long
number – real	Float
True	True
False	False
Null	None

The basic JSON to Python example of decoding with the help of json.loads function:

```
import json  # json library imported
# json data string
person_data = '{  "person":  { "name":  "Kenn",  "sex":  "male",
"age":  28}}'
# Decoding or converting JSON format in dictionary using loads()
dict_obj = json.loads(person_data)
print(dict_obj)
# check type of dict_obj
print("Type of dict_obj", type(dict_obj))
# get human object details
print("Person......",  dict_obj.get('person'))
```

82) Write code for encode() method

Code for encode() method:

```
# import JSONEncoder class from json
from json.encoder import JSONEncoder
colour_dict = { "colour": ["red", "yellow", "green" ]}
# directly called encode method of JSON
JSONEncoder().encode(colour_dict)
```

83) Write a Python code for array in numpy to create Python Matrix

Code for array in numpy to create Python Matrix

```
import numpy as np
M1 = np.array([[5, -10, 15], [3, -6, 9], [-4, 8, 12]])
print(M1)
```

84) Write a Phyhon code for matrix subtraction

Phyhon code for matrix subtraction

```
import numpy as np
M1 = np.array([[3, 6, 9], [5, -10, 15], [-7, 14, 21]])
M2 = np.array([[9, -18, 27], [11, 22, 33], [13, -26, 39]])
M3 = M1 - M2
print(M3)
```

85) Write code for matrix multiplication

Code for matrix multiplication

import numpy as np

```
M1 = np.array([[3, 6], [5, -10]])
M2 = np.array([[9, -18], [11, 22]])
M3 = M1.dot(M2)
print(M3)
```

86) Explain slicing of matrix with example

Slicing will return you the elements from the matrix based on the start /end index given.

The syntax for slicing is:

`[start:end]`

- If the start index is not given, it is considered as 0. For example [:5], it means as [0:5].
- If the end is not passed, it will take as the length of the array.
- If the start/end has negative values, it will the slicing will be done from the end of the array.

Before we work on slicing on a matrix, let us first understand how to apply slice on a simple array.

```
import numpy as np

arr = np.array([2,4,6,8,10,12,14,16])
print(arr[3:6]) # will print the elements from 3 to 5
print(arr[:5]) # will print the elements from 0 to 4
print(arr[2:]) # will print the elements from 2 to length of the
array.
print(arr[-5:-1]) # will print from the end i.e. -5 to -2
print(arr[:-1]) # will print from end i.e. 0 to -2
```

87) Write Python code to find average via loop

Here is a code to find average via loop:

```
def cal_average(num):
    sum_num = 0
    for t in num:
        sum_num = sum_num + t

    avg = sum_num / len(num)
    return avg

print("The average is", cal_average([18,25,3,41,5]))
```

88) Write a code for list count

Here, is a Code for list count:

```
list1 = ['red', 'green', 'blue', 'orange', 'green', 'gray', 'green']
color_count = list1.count('green')
print('The count of color: green is ', color_count)
```

89) How can you count duplicate elements in a given list?

count duplicate elements in a given list

```
list1 = [2,3,4,3,10,3,5,6,3]
elm_count = list1.count(3)
print('The count of element: 3 is ', elm_count)
```

90) Write a code to get index of an element in a list using for loop

Code to get index of an element in a list using for loop:

```
my_list = ['Guru', 'Siya', 'Tiya', 'Guru', 'Daksh', 'Riya', 'Guru']
all_indexes = []
for i in range(0, len(my_list)) :
    if my_list[i] == 'Guru' :
        all_indexes.append(i)
print("Originallist ", my_list)
print("Indexes for element Guru : ", all_indexes)
```

91) Give an example of Python type()

```
str_list = "Welcome to Guru99"
age = 50
pi = 3.14
c_num = 3j+10
my_list = ["A", "B", "C", "D"]
my_tuple = ("A", "B", "C", "D")
my_dict = {"A":"a", "B":"b", "C":"c", "D":"d"}
my_set = {'A', 'B', 'C', 'D'}

print("The type is : ",type(str_list))
print("The type is : ",type(age))
print("The type is : ",type(pi))
print("The type is : ",type(c_num))
print("The type is : ",type(my_list))
print("The type is : ",type(my_tuple))
print("The type is : ",type(my_dict))
print("The type is : ",type(my_set))
```

92) How can you print without a newline in Python?

From Python 3+, there is an additional parameter introduced for print() called end=. This parameter takes care of removing the newline that is added by default in print().

In the Python 3 print without newline example below, we want the strings to print on the same line in Python. To get that working, just add end=”” inside print() as shown in the example below:

```
print("Hello World ", end="")
print("Welcome to Guru99 Tutorials")
```

93) How to print the star(*) pattern without newline and space?

Code to print the star(*) pattern without newline and space:

```
for i in range(0, 20):
    print('*', end="")
```

These interview questions will also help in your viva(orals)

You Might Like:

- Python vs Ruby – Difference Between Them
- Python String format() Explain with EXAMPLES
- BEST Python Certification Exam in 2023
- Python map() function with EXAMPLES
- Python remove Duplicates from a List

⟵ Prev

Report a Bug