Python scripting questionaries’

1. Have u developed any framework? any automated the scripts?
2. where do u write your code? built in or eclipse or pytest tool?
3. Diff data structures in python -> list, tuples, arrays
4. Difference bw list and tuple? where do we use lists and tuples? any scenarios?
5. Difference bw append and extend? list --> append function--> it will b appended in this list.
6. suppose we want to delete elements at some indexes in the list. answer is pop () remove the elements repeatedly
7. Suppose in a dictonary, i dic with 3 key value pairs, one pain has 40 value, i want to update 40 with 50
d = {1:10, 2:20, 4:40}
ans required is = {1:10, 2:20, 4:50}
d.update(d[2][50])

8. i = {1,2,2,3,3,1} Remove duplicates from List
list(dict.fromkeys(i))

9.How to read file

10.String questions
11.Recursion code
12.How to find the pair of elements?
13.The list has lower letters, Numbers and duplicates letters and numbers. Required result is change lower case letters to upper case, remove numbers and remove the duplicate letters. Then reverse this list or array or xyz

Basic Python screening questions and answers

1) What are decorators in Python?

Decorators in Python are essentially functions that add functionality to an existing function in Python without changing the structure of the function itself. They are represented by the @decorator_name in Python and are called in bottom-up fashion.

2)What are lists and tuples? What is the key difference between the two?

Lists and Tuples are both sequence data types that can store a collection of objects in Python. The objects stored in both sequences can have different data types. Lists are represented with square brackets ['sara', 6, 0.19], while tuples are represented with parantheses ('ansh', 5, 0.97).
But what is the real difference between the two? The key difference between the two is that while lists are mutable, tuples on the other hand are immutable objects. This means that lists can be modified, appended or sliced on-the-go but tuples remain constant and cannot

be modified in any manner. You can run the following example on Python IDLE to confirm the difference:

3)What are Dict and List comprehensions?

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:

4)What is pass in Python?

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.

5)How do you copy an object in Python?

In Python, the assignment statement (= operator) does not copy objects. Instead, it creates a binding between the existing object and the target variable name. To create copies of an object in Python, we need to use the copy module. Moreover, there are two ways of creating copies for the given object using the copy module -

- Shallow Copy is a bit-wise copy of an object. The copied object created has an exact copy of the values in the original object. If either of the values are references to other objects, just the reference addresses for the same are copied.
- Deep Copy copies all values recursively from source to target object, i.e. it even duplicates the objects referenced by the source object.

```
from copy import copy, deepcopy

copy.copy () or copy.deepcopy()
```

6)What is the difference between xrange and range in Python?

xrange() and range() are quite similar in terms of functionality. They both generate a sequence of integers, with the only difference that range() returns a Python list, whereas, xrange() returns an xrange object.

So how does that make a difference? It sure does, because unlike range(), xrange() doesn't generate a static list, it creates the value on the go. This technique is commonly used with an object type generators and has been termed as "yielding".

Yielding is crucial in applications where memory is a constraint. Creating a static list as in range() can lead to a Memory Error in such conditions, while, xrange() can handle it optimally by using just enough memory for the generator (significantly less in comparison).

7) What is self in Python?

Self is a keyword in Python used to define an instance or an object of a class. In Python, it is explicity used as the first paramter, unlike in Java where it is optional. It helps in disinguishing between the methods and attributes of a class from its local variables.

8) What is __init__?

__init__ is a contructor method in Python and is automatically called to allocate memory when a new object/instance is created. All classes have a __init__ method associated with them. It helps in distinguishing methods and attributes of a class from local variables.

9) What are generators in Python?

Generators are functions that return an iterable collection of items, one at a time, in a set manner. Generators, in general, are used to create iterators with a different approach. They employ the use of yield keyword rather than return to return a generator object.

10) What is PYTHONPATH in Python?

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.

11) What is the difference between .py and .pyc files?

- .py files contain the source code of a program. Whereas, .pyc file contains the bytecode of your program. We get bytecode after compilation of .py file (source code). .pyc files are not created for all the files that you run. It is only created for the files that you import.
- Before executing a python program python interpreter checks for the compiled files. If the file is present, the virtual machine executes it. If not found, it checks for .py file. If found, compiles it to .pyc file and then python virtual machine executes it.
- Having .pyc file saves you the compilation time.

12) What are unittests in Python?

- unittest is a unit testing framework of Python.
- Unit testing means testing different components of software separately. Can you think why unit testing is important? Imagine a scenario, you are building software which uses three components namely A, B, and C. Now, suppose your software breaks at a point time. How will you find which component was responsible for breaking the software? Maybe it was component A that failed, which in turn failed component B, and this actually failed the software. There can be many such combinations.
- This is why it is necessary to test each and every component properly so that we know which component might be highly responsible for the failure of the software.

13) How are arguments passed by value or by reference in python?

- Pass by value: Copy of the actual object is passed. Changing the value of the copy of the object will not change the value of the original object.
- Pass by reference: Reference to the actual object is passed. Changing the value of the new object will change the value of the original object.
  In Python, arguments are passed by reference, i.e., reference to the actual object is passed.

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

- Script file must begin with #!/usr/bin/env python

15) Explain how to delete a file in Python?

- Use command os.remove(file_name)

```
import os
os.remove("ChangedFile.csv")
print("File Removed!")
```

16) What is the difference between Python Arrays and lists?

- Arrays in python can only contain elements of same data types i.e., data type of array should be homogeneous. It is a thin wrapper around C language arrays and consumes far less memory than lists.
- Lists in python can contain elements of different data types i.e., data type of lists can be heterogeneous. It has the disadvantage of consuming large memory.

17) 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

18) Define “module” and “package”.

Ans: Each Python program file is a “module”, which imports other modules like “objects” and “attributes”.

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

19) How are data types defined in Python and how much bytes do integer and decimal data types hold?

Answer: In Python, there is no need to define a variable’s data type explicitly.
Based on the value assigned to a variable, Python stores the appropriate data type. In the case of numbers such as Integer, Float, etc, the length of data is unlimited.

20) What are the different environment variables identified by Python?

Answer:

- PYTHONPATH: This environment variable helps the interpreter as to where to locate the module files imported in the program.
- PYTHONSTARTUP: This environment variable contains the path of the Initialization file containing source code.
- PYTHONCASEOK: This variable is used to find the first case-insensitive match in the import statement

21) What does stringVar.strip() does?

Answer: This is one of the string methods which removes leading/trailing white space.

22) Write the command to get all keys from the dictionary.

Answer: *print dict.keys()*

23) How to create and import a package in Python?

Python - Packages

We organize a large number of files in different folders and subfolders based on some criteria, so that we can find and manage them easily. In the same way, a package in Python takes the concept of the modular approach to next logical level. As you know, a module can contain multiple objects, such as classes, functions, etc. A package can contain one or more relevant modules. Physically, a package is actually a folder containing one or more module files.

Let's create a package named mypackage, using the following steps:

- Create a new folder named D:\MyApp.
- Inside MyApp, create a subfolder with the name 'mypackage'.
- Create an empty __init__.py file in the mypackage folder.
- Using a Python-aware editor like IDLE, create modules greet.py and functions.py with following code:

greet.py
 Copy

```
def SayHello(name):
    print("Hello " + name)
    return
```

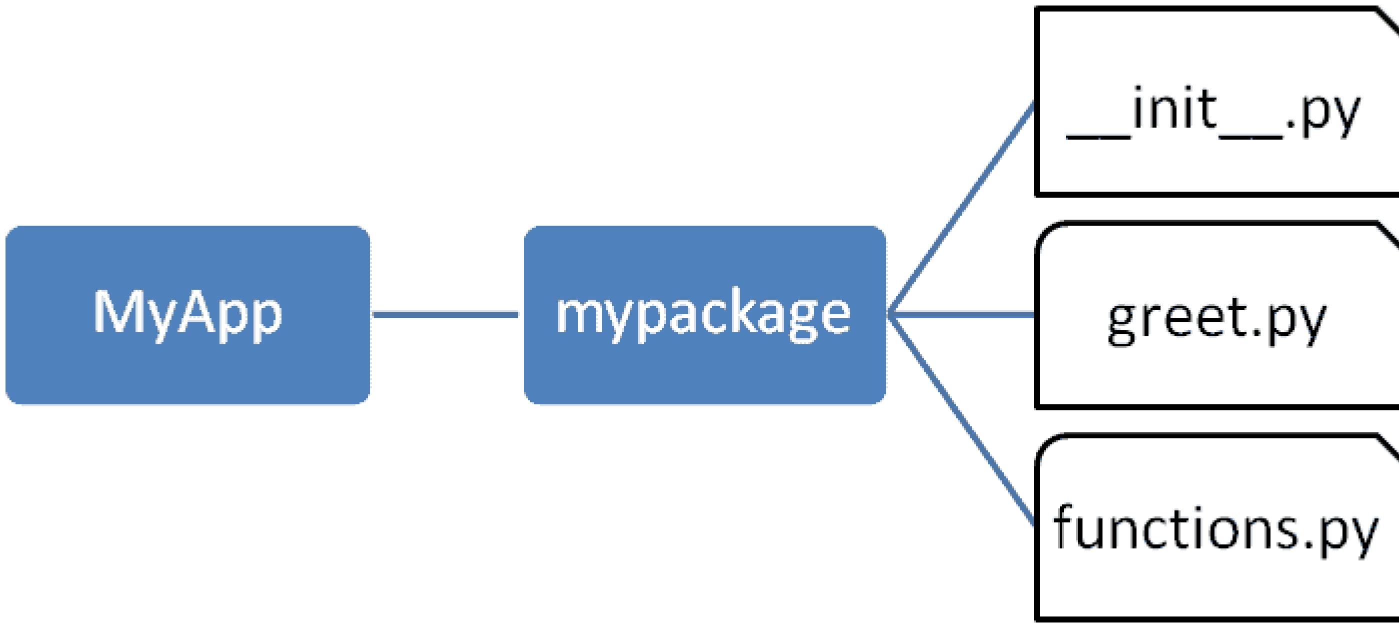
functions.py

 Copy

```
def sum(x,y):
    return x+y

def average(x,y):
    return (x+y)/2
def power(x,y):
    return x**y
```

That's it. We have created our package called mypackage. The following is a folder structure:



Package Folder Structure

Importing a Module from a Package

Now, to test our package, invoke the Python prompt from the MyApp folder.

```
D:\MyApp>python
```

Import the functions module from the mypackage package and call its power() function.

```
>>> from mypackage import functions
>>> functions.power(3,2)
9
```

It is also possible to import specific functions from a module in the package

```
>>> from mypackage.functions import sum
>>> sum(10,20)
30
>>> average(10,12)
Traceback (most recent call last):
File "<pyshell#13>", line 1, in <module>
NameError: name 'average' is not defined
```

__init__.py

The package folder contains a special file called __init__.py, which stores the package's content. It serves two purposes:

1. The Python interpreter recognizes a folder as the package if it contains __init__.py file.

2. __init__.py exposes specified resources from its modules to be imported.

An empty __init__.py file makes all functions from above modules available when this package is imported. Note that __init__.py is essential for the folder to be recognized by Python as a package. You can optionally define functions from individual modules to be made available.

Note:

We shall also create another Python script in the MyApp folder and import the mypackage package in it. It should be at the same level of the package to be imported.

The __init__.py file is normally kept empty. However, it can also be used to choose specific functions from modules in the package folder and make them available for import. Modify __init__.py as below:

__init__.py

Copy

```
from .functions import average, power
from .greet import SayHello
```

The specified functions can now be imported in the interpreter session or another executable script.

Create test.py in the MyApp folder to test mypackage.

test.py

Copy

```
from mypackage import power, average, SayHello
SayHello()
x=power(3,2)
print("power(3,2) : ", x)
```

Note that functions power() and SayHello() are imported from the package and not from their respective modules, as done earlier. The output of above script is:

```
D:\MyApp>python test.py
Hello world
power(3,2) : 9
```

Linux Questions
How to copy the file?
List the last 10 lines in the file

Automation questions
Zip related questions