https://www.edureka.co/blog/interview-questions/python-interview-questions/

O2.What is Python?

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language.
Like Perl, Python source code is also available under the GNU General Public License (GPL).

Q3.What is "#!/usr/bin/python3" code?
It is called as a **"Shebang line"** and Its purpose is to define the location of the interpreter. By adding the line #!/usr/bin/python3 on the top of the script, we can run the file.py on Unix system and automatically will understand that this is a python script. Alternative, you could run the script as "python3 file.py" .

Q4.The __future__ module
Python 3.x introduced some Python 2-incompatible keywords and features that can be imported via the in-built __future__ module in Python 2. It is recommended to use __future__ imports, if you are planning Python 3.x support for your code. For example, if we want Python 3.x's integer division behavior in Python 2, add the following import statement.
from __future__ import division

Q5.What is pep 8?

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

Q6.What are the benefits of using Python?

- Free and open-source– Python is free to use and distribute. It is open source.
- Easy to use– Python is a high-level programming language that is easy to use, read, write and learn.
- Dynamically typed– the developer does not assign data types to variables at the time of coding. It automatically gets assigned during execution.
- Portable– Python programs can run on any platform without requiring any change.

Q7.What are Python namespaces?

A **namespace** in python refers to the name which is assigned to each object in python. The objects are variables & functions. As each object is created, its name along with space(the address of the outer function in which the object is), gets created. The namespaces are maintained in python like a dictionary where the key is the namespace and value is the address of the object. There 4 types of namespace in python-

1. Built-in namespace– contain all the built-in objects in python and are available whenever python is running.
2. Global namespace– for all the objects created at the level of the main program.
3. Enclosing namespaces– at the higher level or outer function.
4. Local namespaces– at the local or inner function.

Q8.What are decorators in Python?

Decorators are used to add some design patterns to a function without changing its structure. Decorators generally are defined before the function they are enhancing. To apply a decorator we first define the decorator function. Then we write the function it is applied to and simply add the decorator function above the function it has to be applied to. For this, we use the @ symbol before the decorator.

Q9.What are Dict and List comprehensions?

Ans: Dictionary and list comprehensions are just another concise way to define dictionaries and lists.

Q14.What are Literals in Python and explain about different Literals

Ans: A literal in python source code represents a fixed value for primitive data types. There are 5 types of literals in python-

1. String literals– A string literal is created by assigning some text enclosed in single or double quotes to a variable. To create multiline literals, assign the multiline text enclosed in triple quotes. Eg.`name="Tanya"`
2. A character literal– It is created by assigning a single character enclosed in double quotes. Eg. `a='t'`
3. Numeric literals include numeric values that can be either integer, floating point value, or a complex number. Eg. `a=50`
4. Boolean literals– These can be 2 values- either True or False.
5. Literal Collections– These are of 4 types-

a) List collections-Eg. `a=[1,2,3,'Amit']`

b) Tuple literals- Eg. `a=(5,6,7,8)`

c) Dictionary literals- Eg. `dict={1: 'apple', 2: 'mango, 3: 'banana`'}`

d) Set literals- Eg. `{"Tanya", "Rohit", "Mohan"}`

6. Special literal- Python has 1 special literal None which is used to return a null variable.

Q15.How to combine dataframes in pandas?

Ans: The dataframes in python can be combined in the following ways-

1. Concatenating them by stacking the 2 dataframes vertically.
2. Concatenating them by stacking the 2 dataframes horizontally.
3. Combining them on a common column. This is referred to as joining.

The concat() function is used to concatenate two dataframes. Its syntax is- `pd.concat([dataframe1, dataframe2])`.

Dataframes are joined together on a common column called a key. When we combine all the rows in dataframe it is union and the join used is outer join. While, when we combine the common rows or intersection, the join used is the inner join. Its syntax is-

```
pd.concat([dataframe1, dataframe2], axis='axis', join='type_of_join)
```

Q16.What are the new features added in Python 3.9.0.0 version?

Ans: The new features in Python 3.9.0.0 version are-

- New Dictionary functions Merge(|) and Update(|=)
- New String Methods to Remove Prefixes and Suffixes
- Type Hinting Generics in Standard Collections
- New Parser based on PEG rather than LL1
- New modules like zoneinfo and graphlib
- Deprecated functions and commands such as deprecated parser and symbol modules, deprecated functions, etc.
- Removal of erroneous methods, functions, etc.

Q17. How is memory managed in Python?

Ans: Memory is managed in Python in the following ways:

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

Q18. What is namespace in Python?

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

Q19. What is PYTHONPATH?

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

Q20. 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

Q21.What are local variables and global variables in Python?

Global Variables:

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

Local Variables:

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

Example of list comprehension is-

```
x=[i for i in range(5)]
```

The above code creates a list as below-

```
4
[0,1,2,3,4]
```

Example of dictionary comprehension is-

```
x=[i : i+2 for i in range(5)]
```

The above code creates a list as below-

```
[0: 2, 1: 3, 2: 4, 3: 5, 4: 6]
```

Q10.What are the common built-in data types in Python?

Numbers– They include integers, floating-point numbers, and complex numbers. eg. `1, 7.9,3+4i`

List– An ordered sequence of items is called a list. The elements may belong to different data types. Eg. `[5,'market',2.4]`

Tuple– It is also an ordered sequence of elements. Unlike lists , tuples are immutable, which means they can't be changed. Eg. `(3,'tool',1)`

String– A sequence of characters is called a string. They are declared within single or double-quotes. Eg. `"Sana"`, `'She is going to the market'`, etc.

Set– Sets are a collection of unique items that are not in order. Eg. `{7,6,8}`

Dictionary– A dictionary stores values in key : value pairs where each value can be accessed through its key. The order of items is not important. Eg. `{1:'apple',2:'mango}`

Boolean– There are 2 boolean values- True and False.

Q11.What is the difference between .py and .pyc files?

The .py files are the python source code files. While the .pyc files contain the bytecode of the python files. .pyc files are created when the code is imported from some other source. The interpreter converts the source .py files to .pyc files which helps by saving time.

Q12.What is slicing in Python?

Slicing is used to access parts of sequences like lists, tuples, and strings. The syntax of slicing is-`[start:end:step]`. The step can be omitted as well. When we write `[start:end]` this returns all the elements of the sequence from the start (inclusive) till the end-1 element. If the start or end element is negative i, it means the ith element from the end. The step indicates the jump or how many elements have to be skipped. Eg. if there is a list- `[1,2,3,4,5,6,7,8]`. Then `[-1:2:2]` will return elements starting from the last element till the third element by printing every second element.i.e. `[8,6,4]`.

Q13.What are Keywords in Python?

Keywords in python are reserved words that have special meaning.They are generally used to define type of variables. Keywords cannot be used for variable or function names. There are following 33 keywords in python

Q28.What is __init__?

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.

Q29.What is a lambda function?

An anonymous function This function can have any number of parameters but, can have just one statement.

Q30. What is self in Python?

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

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

Q32. What does [::-1} do?

Ans: [::-1] is used to reverse the order of an array or a sequence. [::-1] reprints a reversed copy of ordered data structures such as an array or a list. the original array or list remains unchanged.

Q34. What are python iterators?

Iterators are objects which are traversed through or iterated upon.

Q38. 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.

Q39. What are the generators in python?

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

Q43.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.

Q44. 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

Q45. 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.

Q49. 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.