1. What is PEP 8 and why is it useful?

- PEP is an acronym for Python Enhancement Proposal.
- It is an official design document that contains a set of rules specifying how to format Python code and helps to achieve maximum readability.
- PEP 8 is useful because it documents all style guides for Python code.
- This is because contributing to the Python open-source community requires you to adhere to these style guidelines strictly

2. What is scope in Python?

- A scope in Python is a block of code in which a Python code lives.
- Global scope: It refers to the top-most scope in a program. These are global variables available throughout the code execution since their inception in the main body of the Python code.
- Local scope: It refers to the local variables available inside a current function and can only be used locally.

3. Is Python considered a programming or a scripting language?

- Python is generally considered to be a general-purpose programming language, but we can also use it for scripting.
- A scripting language is also a programming language that is used for automating a repeated task that involves similar types of steps while executing a program.
- Filename extensions for Python scripting language are of different types, such as `.py, .pyc, .pyd, .pyo, .pyw, and .pyz.`

4. What are keywords in Python?

- Keywords in Python are reserved words with a special meaning.
- They are generally used to define different types of variables.
- We cannot use keywords in place of variable or function names.

- There are 35 reserved keywords in Python

5. What is init ?

- In Python, init is a method or constructor used to automatically allocate memory when a new object or instance of a class is created.
- All classes have the `__init__` method included.

6. What are iterators in Python?

- An iterator in Python is an object used to iterate over a finite number of elements in data structures like lists, tuples, dicts, and sets.
- Iterators allow us to traverse through all the elements of a collection and return a single element at a time.
- The iterator object is initialized via the `iter()` method and uses the `next()` method for iteration.

7. What is a Python decorator?

- A decorator in Python is a function that allows a user to add a new piece of functionality to an existing object without modifying its structure.
- You usually call decorators before the definition of the function you want to decorate.

8. What are pickling and unpickling?

- **Pickling** is a process in Python where an object hierarchy is transformed into a byte stream. Pickling is also known as “serialization” or “marshaling”
- **Unpickling**, in contrast, is the opposite of pickling. It happens when a byte stream is converted back into an object hierarchy.
- Pickling uses the pickle module in Python. This module has the method `pickle.dump()` to dump Python objects to disks to achieve pickling.
- Unpickling uses the method `pickle.load()` to retrieve the data as Python objects.

9.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.

10.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.

11. What is the use of the pass keyword in Python?

- Pass is a null statement that does nothing.
- It is often used as a placeholder where a statement is required syntactically, but no action needs to be taken.
- For example, if you want to define a function or a class but haven't yet decided what it should do, you can use the pass as a placeholder.

12. What is the use of the continue keyword in Python?

- Continue is used in a loop to skip over the current iteration and move on to the next one.
- When continue is encountered, the current iteration of the loop is terminated, and the next one begins.

13.What is the use of try and except block in Python?

- The try and except blocks in Python are used to handle exceptions. An exception is an error that occurs during the execution of a program.
- The try block contains code that might cause an exception to be raised. The except block contains code that is executed if an exception is raised during the execution of the try block.
- Using a try-except block will save the code from an error to occur and can be executed with a message or output we want in the except block.

14.What are Python functions, and how do they help in code optimization?

- In Python, a function is a block of code that can be called by other parts of your program. Functions are useful because they allow you to reuse code and divide your code into logical blocks that can be tested and maintained separately.
- To call a function in Python, you simply use the function name followed by a pair of parentheses and any necessary arguments. The function may or may not return a value that depends on the usage of the turn statement.

Functions can also help in code optimization:

- Code reuse: Functions allow you to reuse code by encapsulating it in a single place and calling it multiple times from different parts of your program. This can help to reduce redundancy and make your code more concise and easier to maintain.
- Improved readability: By dividing your code into logical blocks, functions can make your code more readable and easier to understand. This can make it easier to identify bugs and make changes to your code.
- Easier testing: Functions allow you to test individual blocks of code separately, which can make it easier to find and fix bugs.
- Improved performance: Functions can also help to improve the performance of your code by allowing you to use optimized code

libraries or by allowing the Python interpreter to optimize the code more effectively.

15. What is the difference between return and yield keywords?

- Return is used to exit a function and return a value to the caller. When a return statement is encountered, the function terminates immediately, and the value of the expression following the return statement is returned to the caller.
- Yield, on the other hand, is used to define a ***generator*** function. A generator function is a special kind of function that produces a sequence of values one at a time instead of returning a single value. When a yield statement is encountered, the generator function produces a value and suspends its execution, saving its state for later.

16.What is the use of the ‘assert’ keyword in Python?

- In Python, the assert statement is used to test a condition. If the condition is True, then the program continues to execute. If the condition is False, then the program raises an AssertionError exception.
- The assert statement is often used to check the internal consistency of a program. For example, you might use an assert statement to check that a list is sorted before performing a binary search on the list.
- It’s important to note that the assert statement is used for debugging purposes and is not intended to be used as a way to handle runtime errors. In production code, you should use try and except blocks to handle exceptions that might be raised at runtime.

17.Where can we use a tuple instead of a list?

- We can use tuples as dictionary keys as they are hashable. Since tuples are immutable, it is safer to use if we don’t want values to change.
- Tuples are faster and have less memory, so we can use tuples to access only the elements.

18. What is inheritance, and how is it useful?

- With inheritance, we can use the attributes and methods of the parent class in the child class.
- We can also modify the parent class methods to suit the child class.

They are different types of inheritance supported by Python:

- Single Inheritance – where a derived class acquires the members of a single superclass.
- Multi-level inheritance – a derived class d1 is inherited from base class base1, and d2 is inherited from base2.
- Hierarchical inheritance – from one base class you can inherit any number of child classes
- Multiple inheritances – a derived class is inherited from more than one base class.

19.Is indentation required in python?

- 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.

20.What is the difference between Python Arrays and lists?

- Arrays and lists, in Python, have the same way of storing data. But, arrays can hold only a single data type elements whereas lists can hold any data type elements.

21.What are Python libraries? Name a few of them.

Python libraries are a collection of Python packages. Some of the majorly used python libraries are – Numpy, Pandas, Matplotlib, Scikit-learn and many more.

22.What is Polymorphism in Python?

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.

23.Define encapsulation in Python?

Encapsulation means binding the code and the data together. A Python class in an example of encapsulation

24.What is Pythonpath?

- A Pythonpath tells the Python interpreter to locate the module files that can be imported into the program.
- It includes the Python source library directory and source code directory. You can preset Pythonpath as a Python installer

25.What is the maximum possible length of an identifier in Python?

- The maximum possible length of an identifier is the maximum length of 79 characters.

26.How is memory managed in Python?

- Memory management in Python is handled by the Python Memory Manager. The memory allocated by the manager is in form of a private heap space dedicated to Python. All Python objects are stored in this heap and being private, it is inaccessible to the programmer. Though, python does provide some core API functions to work upon the private heap space.
- Additionally, Python has an in-built garbage collection to recycle the unused memory for the private heap space.

27.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

28. Write a Program to add two integers >0 without using the plus operator.

```
def add_nums(num1, num2):
  while num2 != 0:
      data = num1 & num2
      num1 = num1 ^ num2
      num2 = data << 1
  return num1
print(add_nums(2, 10))#12
```

29.Write a program to check and return the pairs of a given array A whose sum value is equal to a target value N.

```
def print_pairs(arr, N):
  # hash set
  hash_set = set()#we can use [ ] also

  for i in range(0, len(arr)):
      val = N-arr[i]
      if (val in hash_set):    #check if N-x is there in set, print the pair
          print("Pairs " + str(arr[i]) + ", " + str(val))
      hash_set.add(arr[i])

# driver code
arr = [1, 2, 40, 3, 9, 4]
N = 3
print_pairs(arr, N)
```

30.Write a program) which takes a sequence of numbers and check if all numbers are unique.

```
def check_distinct(data_list):
 if len(data_list) == len(set(data_list)):
   return True
 else:
   return False;
print(check_distinct([1,6,5,8]))     #Prints True
print(check_distinct([2,2,5,5,7,8])) #Prints False
```