1) What is the best way to debug a Python program?

This command can be used to debug a Python program:

```
python -m pdb Python-script.py
```

2) What does the Python keyword imply?

In Python, we can use the yield keyword to convert any Python function into a Python generator. yield functions similarly to a conventional return keyword. However, it will always return a generator object. A function can also use the yield keyword multiple times.

```
def creating_gen(index):
 months = ['jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', 'oct', 'nov', 'dec']
 yield months[index]
 yield months[index+2]
next_month = creating_gen(3)
print(next(next_month), next(next_month))
```

3) How can I make a tuple out of a list?

We can transform a list into a tuple using the Python tuple() method. Since a tuple is immutable, we can't update the list after it has been converted to a tuple.

```
month = ['jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', 'oct', 'nov', 'dec']
converting_list = tuple(month)
print(converting_list)
print(type(converting_list))
```

4) What exactly is a NumPy array?

NumPy arrays are much more versatile than Python lists. Reading and writing objects are quicker and more efficient using NumPy arrays.

5) In Python, in what ways can you make an empty NumPy array?

In Python, there are two ways to build an empty array:

```
import numpy
# Method 1
array_1 = numpy.array([])
print(array_1)
# Method 2
array_2 = numpy.empty(shape=(3,3))
```

```
print(array_2)
```

6) In Python, what is a negative index?

In Arrays and Lists, Python contains a unique feature called negative indexing. Python starts indexing from the beginning of an array or list with a positive integer but reads items from the end of an array or list with a negative index.

7) Tell the output of the following code:

```
import array
a = [4, 6, 8, 3, 1, 7]
print(a[-3])
print(a[-5])
print(a[-1])
```

Output:

```
3
6
7
```

8) What is the Python data type SET, and how can I use it?

Set is a Python data type which is a sort of collection. Since Python 2.4, it has been a part of the language. A set is a collection of distinct and immutable items that are not in any particular sequence.

9) In Python, how do you create random numbers?

We can create random data in Python using several functions. They are as follows:

- random() - This function gives a floating-point value ranging from 0 to 1.
- uniform(X, Y) - This function gives a floating-point value in the range of X and Y.
- randint(X, Y) - This function gives a random integer between X and Y values.

10) How do you print the summation of all the numbers from 1 to 101?

Using this program, we can display the summation of all numbers from 1 to 101:

```
print(sum(range(1, 102)))
```

Output:

```
5151
```

11) In a function, how do you create a global variable?

We can create a global variable by designating it as global within every function that assigns to it. We can utilize it in other functions.

```
global_var = 0
def modify_global_var():
 global global_var # Setting global_var as a global variable
 global_var = 10
def printing_global_var():
 print(global_var) # There is no need to declare global variable
modify_global_var()
printing_global_var() # Prints 10
```

12) Is it possible to construct a Python program that calculates the mean of numbers in a list?

Calculating the average of numbers in Python:

```
n = int(input("Number of Elements to take the average of: "))
l = []
for i in range(1, n + 1):
 element = int(input("Enter the element: "))
 l.append(element)
average = sum(l) / n
print("Average of the elements in the list:", round(average, 2))
```

13) Is it possible to build a Python program that reverses a number?

Python program to reverse a number:

```
n = int(input("Enter number: "))
reverse = 0
while n > 0:
 digit = n % 10
 reverse = reverse * 10 + digit
 n = n // 10
print("The reverse of the number:", reverse)
```

14) In Python, what is the distinction between a list and a tuple?

List:

- Lists are editable, which means that we can change them.

- Lists are comparatively slower.
- Syntax: list1 = [100, 'Itika', 200]

Tuple:

- Tuples (which are just lists that we cannot alter) are immutable.
- Tuples are more efficient than lists.
- Syntax: tup1 = (100, 'Itika', 200)

15) Is Python a programming language or a scripting language?

Although we can use Python to write scripts, it is primarily used as a general-purpose programming language.

16) Python is an interpreted programming language. Explain.

Any scripting language not in machine code before execution is called an interpreted language. Python is thus an interpreted language.

17) What is the meaning of PEP 8?

PEP (Python Enhancement Proposal) is the acronym for Python Enhancement Proposal. It's a collection of guidelines for formatting Python code for better readability.

18) What are the advantages of Python?

The advantages of using Python are as follows:

- Simple to understand and use - Python is a powerful programming language that is easy to learn, read, and write.
- Interpreted language - Python is an interpreted language, which means it runs the program line by line and pauses if any line contains an error.
- Dynamically typed - When coding, the programmer does not set data types to variables. During execution, they are automatically assigned.
- Python is free and open-source to use and share. It's free and open source.
- Extensive library support - Python has a large library of functions that can perform practically any task. It also allows you to use Python Package Manager to import additional packages (pip).
- Python applications are portable and can run on any system without modification.
- Python's data structures are easy to understand.
- It allows for additional functionality while requiring less coding.

19) In Python, what are decorators?

Decorators are used to add certain layout patterns to a method without affecting the structure of the function. Typically, decorators are defined before the function they will be enhancing. We should first define a decorator function before using it. The function to which we will apply the decorator function is

then written, and the decorator function is simply positioned above it. In this instance, the @ symbol comes preceding the decorator.

20) How is a Python Dictionary different from List comprehensions?

Dictionary and list comprehensions are yet another means of defining dictionaries and lists in a simple manner.

This is an example of list comprehension:

```
list_comp = [i for i in range(4)]
print(list_comp)
```

This is an example of dictionary comprehension:

```
dictt = {i : i+2 for i in range(10)}
print(dictt)
```

21) What is the most prevalent Python built-in data types?

The most prevalent built-in data types in Python are:

- Numbers - Integers, complex numbers, and floating points are the most prevalent built-in data types in Python. For example, 1, 8.1, 3+6i.
- List - A list is a collection of objects that are arranged in a specific order. The components of a list can be of multiple data types. For example, [10, 'itika', 7, 4]
- Tuple - It's also a set of items in a specific order. Tuples, unlike lists, are immutable, meaning we cannot modify them. For example, (7, 'itika', 2)
- String - A string is a collection of characters. Single or double quotations are used to declare them. "Itika", "She is learning coding through Javatpoint" are examples of strings.
- Dictionary - A dictionary is an unordered collection of data values that are stored in a key-value pair format. For example, {1: 'itika', 2: 'lavanya', 3: 'jenifer'}
- Sets - A set is an unordered collection of data items. Sets do not permit the use of duplicate values. For example, {'apple', 'banana', 'cherry'}
- Boolean - Boolean data types are either True or False.
- None - The None keyword represents a null value or no value at all.

22) What's the distinction between .py and .pyc files?

The Python code we save is contained in the .py files. The .pyc files are created when the program is compiled from the .py files. These files contain the bytecode for the Python files that we imported. The interpreter reduces processing time if we transform the source files in .py format to .pyc files.

23) How is a local variable different from a global variable?

Global Variables: Global variables are declared outside of a function and have scope throughout the program. Any function within the program has access to these variables.

Local Variables: Local variables are declared inside a function and have a limited scope within that function. They are not accessible outside of the function.

24) What is the distinction between Python Arrays and Python Lists?

In Python, arrays and lists both store data, but they have some differences:

Arrays:

- Arrays can only have elements of the same data type.
- Arrays are more efficient in terms of storage and performance for numerical computations.
- Arrays are defined and manipulated using the NumPy library in Python.

Lists:

- Lists can have elements of different data types.
- Lists are more versatile and provide more built-in functions for manipulation.
- Lists are part of the core Python language.

25) What exactly is __init__?

In Python, __init__ is a special method (also known as a constructor) that gets automatically called when a new object/instance of a class is created. It is used to initialize the attributes of the object and reserve memory for it. The __init__ method is available in all classes and can be used to perform any necessary setup or initialization.

26) What is a lambda function, and how does it work?

A lambda function is a small anonymous function in Python that can have any number of parameters but can only have a single expression. It is defined using the lambda keyword, followed by the parameters and a colon, and then the expression. Lambda functions are often used as one-liners or as arguments to higher-order functions.

27) In Python, what is the self?

In Python, self is a conventionally used name for the first parameter of a method in a class. It represents the instance of the class that the method is being called on. By using the self parameter, we can access the attributes and methods of the object within the class.

28) How do the following commands work: break, pass, and continue?

break: The break statement is used to exit or terminate a loop (for loop or while loop) prematurely. When encountered, it immediately ends the loop and transfers control to the next statement outside the loop.

pass: The pass statement is a null statement in Python. It is used when a statement is required syntactically but no action is needed. It is commonly used as a placeholder or a stub for future code implementation.

continue: The continue statement is used to skip the rest of the current iteration of a loop and move to the next iteration. It allows you to skip certain iterations based on a condition without terminating the loop.

29) How can you randomize the elements of a list while it's running?

To randomize the elements of a list while it's running, you can use the random.shuffle() function from the random module in Python. This function shuffles the elements of the list in a random order.

30) What is the difference between pickling and unpickling?

Pickling and unpickling are processes used for serialization and deserialization of Python objects:

Pickling: Pickling is the process of converting a Python object into a binary representation (a byte stream) that can be stored or transmitted. The pickle module in Python is used for pickling objects.

Unpickling: Unpickling is the reverse process of pickling. It is the process of converting the binary representation (byte stream) back into a Python object. The pickle module is also used for unpickling objects.

31) How do you convert all the characters of a string to lowercase?

To convert all the characters of a string to lowercase in Python, you can use the lower() method. It returns a new string with all the characters converted to lowercase.

32) How do you comment on multiple lines at once in Python?

To comment on multiple lines at once in Python, you can use triple quotes (''' or """). Any text enclosed within triple quotes is treated as a comment and can span multiple lines.

33) What are docstrings in Python?

Docstrings (documentation strings) in Python are used to provide documentation or comments about modules, classes, functions, or methods. They are enclosed within triple quotes (''' or """) and can span multiple lines. Docstrings can be accessed using the __doc__ attribute.

34) Explain the split(), sub(), and subn() methods of the Python "re" module.

The Python "re" module provides various methods for working with regular expressions:

split(): The split() method is used to split a string into a list of substrings based on a specified pattern (regular expression).

sub(): The sub() method is used to replace occurrences of a pattern in a string with a specified replacement string.

subn(): The subn() method is similar to sub(), but it also returns the number of substitutions made.

35) What is the best way to add items to a Python array?

To add items to a Python array, you can use the following methods:

- append(): The append() method is used to add an element to the end of the array.
- extend(): The extend() method is used to add multiple elements to the end of the array by extending it with another iterable.
- insert(): The insert() method is used to insert an element at a specified position in the array.

36) What is the best way to remove values from a Python array?

To remove values from a Python array, you can use the following methods:

- pop(): The pop() method removes and returns the last element of the array. If an index is provided, it removes and returns the element at that index.
- remove(): The remove() method removes the first occurrence of a specified value from the array.

37) In Python, what is monkey patching?

Monkey patching in Python refers to the dynamic modification of a class or module at runtime. It allows you to add, modify, or replace methods or attributes of an existing class or module without directly modifying its source code. Monkey patching can be useful in situations where you want to extend or modify the behavior of an existing class or module without subclassing or modifying the original implementation.

38) How do you create an empty class in Python?

An empty class in Python can be created using the pass statement. The pass statement is a null statement that does nothing. It is used as a placeholder when a statement is syntactically required but no action is needed. By creating an empty class, you can later add attributes and methods to it.

39) Write a Python script to implement the Bubble sort algorithm.

Below is an example of a Python script that implements the Bubble sort algorithm:

```
# Python program to implement the Bubble sort algorithm
def bubble_sort(array):
```

```
 n = len(array)
 for i in range(n - 1):
 for j in range(0, n - i - 1):
 if array[j] > array[j + 1]:
 array[j], array[j + 1] = array[j + 1], array[j]
array = [23, 14, 64, 13, 64, 23, 86]
bubble_sort(array)
print("Sorted array:", array)
```

40) In Python, create a program that generates a Fibonacci sequence.

Below is an example of a Python program that generates a Fibonacci sequence:

```
# Python program to generate a Fibonacci sequence
n = int(input("Enter the number of terms to generate: "))
fibonacci_sequence = [0, 1]
if n <= 0:
 print("Invalid input. Please enter a positive integer.")
elif n == 1:
 print("Fibonacci sequence:", fibonacci_sequence[0])
else:
 while len(fibonacci_sequence) < n:
 next_number = fibonacci_sequence[-1] + fibonacci_sequence[-2]
 fibonacci_sequence.append(next_number)
 print("Fibonacci sequence:", fibonacci_sequence)
```

41) Make a Python program that checks if a given number is prime.

Below is an example of a Python program that checks if a given number is prime:

```
# Python program to check if a number is prime
num = int(input("Enter a number: "))
is_prime = True
if num <= 1:
 is_prime = False
else:
 for i in range(2, int(num ** 0.5) + 1):
 if num % i == 0:
 is_prime = False
 break
if is_prime:
 print(num, "is a prime number")
else:
 print(num, "is not a prime number")
```

42) What is the purpose of the "__name__" variable in Python?

The __name__ variable in Python is a built-in variable that represents the name of the current module. It allows you to determine whether the current module is being run as the main program or being imported as a module.

When a Python module is run as the main program, the __name__ variable is set to "__main__". If the module is imported as a module into another program, the __name__ variable is set to the name of the module.

43) In a NumPy array, how do I extract the indices of the N maximum values?

To extract the indices of the N maximum values in a NumPy array, you can use the argsort() method. The argsort() method returns the indices that would sort the array in ascending order. By using the [-N:] slice, you can select the last N indices, which correspond to the N maximum values.

```
import numpy as np
array = np.array([4, 8, 4, 9, 2])
indices = array.argsort()[-N:][::-1]
print("Indices of the N maximum values:", indices)
```

44) Using Python/NumPy, write code to compute percentiles.

To compute percentiles using Python and NumPy, you can use the percentile() function from the NumPy library. The percentile() function takes two arguments: the array and the percentile value. It returns the value below which a given percentage of observations falls.

```
import numpy as np
array = np.array([3, 6, 1, 6, 5, 2])
percentile = np.percentile(array, percentile_value)
print("Percentile:", percentile)
```

45) Write a Python program to determine whether a number is binary.

To determine whether a number is binary or not, you can convert it to a string and check if it contains only '0' and '1' characters. If it contains any other characters, it is not a binary number.

```
def is_binary(number):
 binary_characters = set('01')
 return all(char in binary_characters for char in str(number))
number = 1101001
if is_binary(number):
 print("The number is binary.")
```

```
else:
 print("The number is not binary.")
```

46) Using the iterative technique, calculate the factorial in Python.

To calculate the factorial of a number using the iterative technique in Python, you can use a loop to multiply the numbers from 1 to the given number.

```
def factorial_iterative(n):
 factorial = 1
 for i in range(1, n + 1):
 factorial *= i
 return factorial
number = 12
result = factorial_iterative(number)
print("Factorial of", number, "is", result)
```

47) Compute the LCM of two given numbers using Python code.

To compute the least common multiple (LCM) of two given numbers using Python, you can use the following approach:

```
def compute_lcm(num1, num2):
 if num1 > num2:
 greater_num = num1
 else:
 greater_num = num2
 while True:
 if greater_num % num1 == 0 and greater_num % num2 == 0:
 lcm = greater_num
 break
 greater_num += 1
 return lcm
num1 = 24
num2 = 92
lcm = compute_lcm(num1, num2)
print("LCM of", num1, "and", num2, "is", lcm)
```

48) Write a Python program that removes vowels from a string.

To remove vowels from a string in Python, you can iterate over each character in the string and check if it is a vowel. If it is not a vowel, append it to a new string. The resulting string will be the original string with vowels removed.

```
def remove_vowels(string):
```

```
 vowels = 'aeiouAEIOU'
 result = ''
 for char in string:
 if char not in vowels:
 result += char
 return result
string = 'Javatpoint'
result = remove_vowels(string)
print("String without vowels:", result)
```

49) Write a Python program that reverses a given list.

To reverse a given list in Python, you can use list slicing with a step of -1. This will create a new list with elements in reverse order.

```
list_ = [23, 12, 5, 24, 23, 76, 86, 24, 86, 24, 75]
reversed_list = list_[::-1]
print("Reversed list:", reversed_list)
```

50) Write a Python program that rotates an array by two positions to the right.

To rotate an array by two positions to the right in Python, you can use list slicing and assignment. First, store the last two elements of the array in a temporary variable. Then, shift the remaining elements to the right. Finally, assign the temporary variable to the first two elements of the array.

```
list_ = [23, 12, 5, 24, 23, 76, 86, 24, 86, 24, 75]
temp = list_[-2:]
list_[:2], list_[2:] = temp, list_[:-2]
print("Array after right rotation:", list_)
```