Q 1. What is List? How will you reverse a list?

Ans. In programming, a list is a data structure that allows you to store a collection of items in a specific order. A list is typically represented within square brackets [] and the items in the list are separated by commas. Lists are mutable, meaning you can modify their elements.

To reverse a list, we can use the reverse() method or slicing technique.

The reverse() method reverses the order of elements in the original list in-place.

This slicing technique creates a new list with elements in reverse order by specifying a step value

of -1.

Q2. How will you remove last object from a list? Suppose list1 is [2, 33, 222, 14, and 25], what is list1 [-1]?

Ans. To remove the last object from a list in Python, we can use the "pop()" method or slice the list. Here's how to achieve this

The "pop()" method removes and returns the last element from the list. In this case, the "pop()" method is called without any index specified, so it removes the last element from the list "list1", which is 25.

"list1[-1]", it accesses the last element of the list "list1". In this case, "list1[-1]" would be equal to 25, which is the last element in the list.

Q3. Differentiate between append () and extend () methods?

Ans. Both the “append()” and “extend()” methods are used to add elements to a list in Python, but they differ in how they handle the elements being added

1. “append()” method:

The “append()” method is used to add a single element to the end of a list. It modifies the original list by appending the new element as a single item.

2. “extend()” method:

The “extend()” method is used to add multiple elements to the end of a list. It takes an iterable as an argument (such as a list or tuple) and appends each element individually to the original list.

Q4. How will you compare two lists?

Ans. To compare two lists in Python, we can use the comparison operators (“==“, “!=“, “<“, “>“, “<=“, “>=“) or the “==“ operator with the “all()” function.

Q5. What is tuple? Difference between list and tuple.

Ans. A tuple in Python is an ordered, immutable collection of elements enclosed in parentheses “()”. Tuples are similar to lists, but they have a few key differences

1. Mutability: Tuples are immutable, which means their elements cannot be modified or reassigned after the tuple is created. In contrast, lists are mutable, and their elements can be modified or reassigned.

2. Syntax: Tuples are defined using parentheses “()” or without any enclosing brackets. For example, “(1, 2, 3)” or “1, 2, 3” both create tuples. Lists, on the other hand, are defined using square brackets “[]”.

3. Size: Tuples are typically used to store a fixed collection of elements with different data types. The size of a tuple is fixed when it is created and cannot be changed. In contrast, lists can grow or shrink dynamically as elements are added or removed.

4. Usage: Tuples are commonly used for grouping related values together, such as coordinates “(x, y)” or date/time values “(year, month, day, hour, minute)”. They are also useful when you want to ensure that the elements remain unchanged. Lists are more versatile and commonly used for storing and manipulating collections of items.

Q 6. How will you create a dictionary using tuples in python?

Ans. you can use the dict() constructor or directly assign key-value pairs using curly braces {}.

Ex. 1st. Dict(tuplelist), 2nd { (key, value) , (key, value) , (key, value) }

Q 7. How Do You Traverse Through A Dictionary Object In Python ?

Ans. There are four ways to traverse through a dictionary object in Python

- **Iterating through keys directly**: This is the simplest way to iterate through a dictionary. You can use a for loop to iterate over the dictionary's keys, and then access the corresponding values using the dictionary's get() method.

- **Iterating through .items():** This method returns a list of key-value pairs, which you can then iterate over in a for loop.

- **Iterating through .keys():** This method returns a list of all the keys in the dictionary. You can then iterate over this list in a for loop.

- **Iterating through .values():** This method returns a list of all the values in the dictionary. You can then iterate over this list in a for loop.

Q 8. How Do You Check The Presence Of A Key In A Dictionary?

Ans. There are several ways to check the presence of a key in a dictionary in Python.

Here are some of the most common methods:

- **Using the in operator:** The in operator can be used to check if a value is present in a sequence, such as a list or a string. It can also be used to check if a key is present in a dictionary.

- **Using the get() method:** The get() method can be used to retrieve the value associated with a key in a dictionary. If the key is not present in the dictionary, the get() method will return None.

Q 9. Why Do You Use the Zip () Method in Python?

Ans. The zip() method in Python is used to combine two or more iterables into a single iterable, where elements from corresponding positions are paired together. The resulting iterable contains tuples, where the first element from each iterable is paired together, the second element from each iterable is paired together, and so on.

The zip() method is useful for a variety of tasks, such as,

- **Iterating over multiple iterables at the same time. :** For example, you could use zip() to iterate over a list of names and a list of ages, and print out the name and age of each person.

- **Creating dictionaries :** You can use zip() to create a dictionary from two lists, where the first list is the keys and the second list is the values. For example, you could use zip() to create a dictionary that maps names to ages.

- **Parallelizing tasks :** You can use zip() to parallelize tasks by iterating over multiple iterables at the same time. For example, you could use zip() to download files from multiple websites at the same time.

Q 10. How Many Basic Types Of Functions Are Available In Python?

Ans. There are two basic types of functions available in Python, built-in functions and user-defined functions.

- **Built-in functions :** are predefined functions that are available in the Python language. They are pre-written and can be used without any additional code. Some examples of built-in functions are print(), len(), sum(), min(), max(), etc.

- **User-defined functions :** are functions that are created by the user to perform a specific task. They are defined using the “def” keyword. User-defined functions can take any number of arguments and can return any type of value.

Q 11. How can you pick a random item from a list or tuple?

Ans. There are two ways to pick a random item from a list or tuple in Python

- **Using the random.choice() function**

The random.choice() function returns a random element from a sequence. To pick a random item from a list or tuple, you can simply pass the list or tuple to the random.choice()

- **Using the random.sample() function**

The random.sample() function returns a random sample of elements from a sequence. The sample is taken without replacement, which means that an element cannot be selected more than once. To pick a random item from a list or tuple, you can pass the list or tuple to the random.sample() function and specify the number of items you want to select.

Which method you use to pick a random item from a list or tuple depends on your specific needs. If you only need to pick a single random item, then the random.choice() function is a good option. If you need to

pick multiple random items, then the random.sample() function is a better option.

Q 12. How can you pick a random item from a range?

Ans. There are two ways to pick a random item from a range in Python

- **Using the "random.randrange()" function**

The "random.randrange()" function returns a random integer from a range. The range can be specified by passing two arguments to the function, the start and end of the range.

- **Using the "random.randint()" function**

The "random.randint()" function is a newer function that returns a random integer from a range. The range can be specified by passing two arguments to the function, the start and end of the range.

Q 13. How can you get a random number in python?

Ans. There are two ways to get a random number in Python

- **Using the random.randint() function**

The random.randint() function returns a random integer from a range. The range can be specified by passing two arguments to the function, the start and end of the range.

- **Using the random.random() function**

The random.random() function returns a random floating-point number between 0 and 1.

Which method you use to get a random number in Python depends on your specific needs. If you need a random integer, then the random.randint() function is a good option. If you need a random floating-point number, then the random.random() function is a good option.

Q 14. How will you set the starting value in generating random numbers?

Ans. To set the starting value in generating random numbers, you can make use of the “random.seed()” function from the “random” module in Python. The “random.seed()” function initializes the random number generator with a seed value.

Here's an example of setting the starting value for generating random numbers,

-------- code ----------

```
import random

# Set the seed value
seed_value = 42
random.seed(seed_value)

# Generate random numbers
random_number_1 = random.random()
random_number_2 = random.randint(1, 100)

print("Random Number 1:", random_number_1)
print("Random Number 2:", random_number_2)
```

In this example, the seed value is set to “42” using “random.seed(seed_value)”. This ensures that the subsequent random number generation will always produce the same sequence of random numbers when the program is run with the same seed value.

The program then generates two random numbers: “random_number_1” using “random.random()” to generate a float between 0 and 1 (exclusive), and “random_number_2” using “random.randint(1, 100)” to generate an integer between 1 and 100 (inclusive).

By setting the seed value, you can control the starting point of the random number sequence. This can be useful when you need to reproduce the same set of random numbers for testing or when you want to ensure consistent behavior in random number generation.

Q 15. How will you randomizes the items of a list in place?

Ans. There are two ways to randomize the items of a list in place in Python

- **Using the random.shuffle() function**

The random.shuffle() function takes a sequence as an argument and shuffles the order of the items in the sequence. The shuffle is done in place, which means that the original list is modified.

- **Using the random.sample() function**

The random.sample() function takes a sequence as an argument and returns a random sample of items from the sequence. The sample is taken without replacement, which means that an item cannot be selected more than once.

Which method you use to randomize the items of a list in place depends on your specific needs. If you need to shuffle the entire list, then the random.shuffle() function is a good option. If you need to take a random sample of items from the list, then the random.sample() function is a good option.