10 ) What is :

1. Negative indexing in python- Negative index is used in python to index starting from the last element of the list, tuple or any other container class which supports indexing. -1 refers to the last element, -2 refers to the second last element and so on. For example, in mylist=[1,2,3,4,5], mylist[-1] gives 5 as the output.

1. Packing and UnPacking- Packing is putting multiple values into a single variable. On the other hand, unpacking puts each value from a list,tuple or dict into a new variable. For example,

Example of unpacking

x= (1,2)

a, b=x

a=1

b=2

Example of packing- Here, we are packing multiple values into a single variable.

def sum(\*args):

print(args)

sum(1,2,3,4,5)

Output is a tuple

(1,2,3,4,5)

1. Mutable and Immutable- Some of the objects like lists and dictionaries are **mutable**, meaning you can change their content without changing their identity. Other objects like integers, floats, strings and tuples are **immutable** objects that cannot be changed. Strings are immutable in Python, which means you cannot change an existing string. The best you can do is create a new string that is a variation on the original.
2. Append and Extend- **Append** adds its argument as a single element to the end of a list. The length of the list itself will increase by one. For example,

x=[1,2,3]

x.append([4,5])

print(x)

gives you [1,2,3,[4,5]]

**Extend** iterates over its argument adding each element to the list, extending the list. The length of the list will increase by however many elements were in the iterable argument. For example,

x.extend([4,5]) will give [1,2,3,4,5] output.

1. Pickling and Unpickling- **Pickling** is the process whereby a Python object hierarchy is converted into a byte stream, that can be stored on disk or sent over a network. **Unpickling** is the inverse operation, whereby a byte stream is converted back into an object hierarchy.

13) What is split function in python?

**split()** method returns a list of strings after breaking the given string by the specified separator.

**Syntax :**

**str.split(separator,maxsplit)**

**Parameters :**

**separator**- The is a delimiter. The string splits at this specified separator. If is not provided then any white space is a separator.

maxsplit-It is a number, which tells us to split the string into maximum of provided number of times. If it is not provided, then there is no limit.

19) What is memory Management in Python?

Python divides the RAM into two parts called Run-time stack and the Heap. Memory management in Python involves a private heap containing all Python objects and data structures. At the lowest level, a raw memory allocator ensures that there is enough room in the private heap for storing all Python-related data by interacting with the memory manager of the operating system. It is important to understand that the management of the Python heap is performed by the interpreter itself and that the user has no control over it.

The activation record is a "chunk of memory" (a bunch of buckets) which contains all the information necessary to keep track of a "function call". The run-time stack is a stack of Activation Records. When a function is called, a block is reserved on the top of the stack for local variables and some bookkeeping data. When that function returns, the block becomes unused and can be used the next time a function is called. The stack is always reserved in a LIFO (last in first out) order; the most recently reserved block is always the next block to be freed. This makes it simple to keep track of the stack; freeing a block from the stack is nothing more than adjusting one pointer.

Python allocates memory transparently, manages objects using a reference count system, and frees memory when an object’s reference count falls to zero.

20) What is the difference between range and xrange function

Both range and xrange in python 2 perform the same function i.e. they generate a range of numbers. But the only difference is that range() gives list object and xrange() give xrange object.

List object means that a list of numbers is created and stored in memory when the range() is given. So, a big list occupies a lot of memory. All operations of list can be applied on list object returned by range function.

Xrange object means that all numbers in the range are not generated and stored at once in the memory. They are created when you need them. So, this saves a lot of memory. It is good to use xrange in for-loop. Its implementation is faster than range(). Operations of list cannot be performed on xrange object.

Python 3 does not have xrange function. Range function in python 3 does what xrange does in python 2.