1. What is Python? What are the benefits of using Python?

Python is a high-level programming language used for web development, data analysis, artificial intelligence, and other applications. The benefits of using Python include its simplicity, readability, portability, and vast collection of libraries and frameworks.

1. What are the main data types in Python? Explain each one.

The main data types in Python include integers, floats, booleans, strings, lists, tuples, sets, and dictionaries. Integers are whole numbers, floats are decimal numbers, booleans are true or false values, strings are sequences of characters, lists are ordered collections of items, tuples are ordered and immutable collections of items, sets are unordered collections of unique items, and dictionaries are unordered collections of key-value pairs.

1. What are functions in Python? How do you define and call a function? What are the different types of functions in Python?

Functions in Python are reusable blocks of code that perform a specific task. To define a function, use the **def** keyword followed by the function name, parameter(s), and colon. To call a function, simply type the function name with any arguments inside parentheses. There are several types of functions in Python, including built-in functions, user-defined functions, lambda functions, and recursive functions.

1. What is object-oriented programming in Python?

Object-oriented programming (OOP) is a programming paradigm that uses objects to represent real-world entities.

1. Explain the concepts of classes, objects, and inheritance in Python.

In Python, classes are templates for creating objects, objects are instances of a class, and inheritance allows for the creation of new classes based on existing ones.

1. What is the difference between a shallow copy and a deep copy in Python?

A shallow copy creates a new object with a new memory address, but the elements inside the object still reference the original elements. A deep copy creates a new object with a new memory address and new elements that are completely independent of the original elements

1. What is a decorator in Python? How do you use it?

A decorator in Python is a function that takes another function as input and returns a new function as output, usually with some additional functionality. You can use decorators to add functionality to existing functions without modifying their code

1. What are the different types of exceptions in Python? How do you handle exceptions in your code?

There are many types of exceptions in Python, including NameError, TypeError, and ValueError.

**What is \_\_init\_\_?**

\_\_init\_\_ is a contructor method in Python and is automatically called to allocate memory when a new object/instance is created. All classes have a **\_\_init\_\_** method associated with them. It helps in distinguishing methods and attributes of a class from local variables.

var={10,20,10,10,30,30,20}

print(len(var)) #output is 3 because we are using set method so set not allowed duplicate value

--------------------------------------------------------------------------------------------------------

lst=[1,0,2,0,4,0,6]

for i in lst:

if i==0:

lst.remove(i)

lst.append(i)

print(lst) #using list method

-------------------------------------------------------------------------------------

def func(x,y):

x=1

y=1

return x+y

print(func(3,4)) #output is 2 because

-----------------------------------------------------------------

var='hello'

print(var[-0]) #output is h . because -0 consiter as 0 so

----------------------------------------------------------------

def f():

pass

print(type(f())) #output is <class 'NoneType'> cause we did not any value there if we pass the value and we return

----------------------------------------------------------

atuple=1,2,3

a,b,c=atuple

print(a) #outputb is 1 Top of Form