1. What's python? Provide some key features of python.

Ans:

Python is a high-level, interpreted programming language known for its simplicity and readability.

Key features:

* Easy to learn & use: Simple syntax similar to English.
* Interpreted: Executes code line by line
* Dynamic typed: No need to declare variables types
* Object oriented & Functional: Supports multiple programming paradigms.
* Extensive Libraries: Rich standard Libraries like NumPy, Pandas, etc.

1. Explain the differences between python 2 and python3.

Ans:-

Python 3 introduced significant improvements over Python 2, making it more efficient, readable, and future-proof. Some key differences include:

* **Print Statement:** In Python 2, print is used without parentheses (print "Hello"), whereas in Python 3, it is a function (print("Hello")).
* **Integer Division:** In Python 2, division between two integers returns an integer (5 / 2 = 2), while in Python 3, it returns a float (5 / 2 = 2.5). For integer division in Python 3, // must be used.
* **Unicode Handling:** Python 3 uses Unicode by default for strings, while Python 2 treats them as ASCII unless explicitly specified with u"string".
* **End of Support:** Python 2 reached its end of life in 2020, and Python 3 is now the standard for all development.

1. How do you comment out multiple lines of code in python?

Ans:

There are two ways to comment multiple lines in Python:

**Using # on each line:**# This is a comment

# spanning multiple lines

**Using triple quotes (''' or """) as a multi-line string:** While primarily used for docstrings, this can also be used as a temporary multi-line comment.  
"""

This is a multi-line comment

using triple quotes.

"""

1. Describe the difference between == and is in python.

Ans:

* == checks if two variables have the same **value**.
* is checks if two variables refer to the **same object in memory**.  
  Example:

a = [1, 2, 3]

b = a

c = [1, 2, 3]

print(a == c) # True (same values)

print(a is c) # False (different objects in memory)

print(a is b) # True (both reference the same object)

1. What is PEP8? Why is it important?

Ans:

PEP 8 (Python Enhancement Proposal 8) is a set of guidelines for writing Python code in a clean, readable, and maintainable way. It covers aspects like naming conventions, indentation, line length, whitespace usage, and import ordering. Adhering to PEP 8 makes code more consistent, easier to understand, and enhances collaboration among developers.

1. Explain the concept of dynamic typing in python.

Ans:  
Python is a dynamically typed language, meaning variables do not have fixed types and can be reassigned to different types at runtime. This provides flexibility but also requires careful handling to avoid unintended behavior.

x = 10 # x is an integer x = "Hello" # Now x is a string

Unlike statically typed languages (e.g., Java, C++), Python does not require explicit type declarations. However, dynamic typing can sometimes lead to type-related errors that are harder to debug.

1. What are tuples in python? How are they different from lists?

Ans:

Tuples and lists are both used to store collections of items, but they differ in mutability:

* **Tuples (tuple)** are immutable, meaning their elements cannot be changed after creation. They are defined using parentheses ().
* **Lists (list)** are mutable, allowing modification of elements. They are defined using square brackets [].  
  Example:  
  my\_tuple = (1, 2, 3)

my\_list = [1, 2, 3]

my\_list[0] = 10 # Allowed

# my\_tuple[0] = 10 # TypeError: 'tuple' object does not support item assignment

Tuples are generally used for fixed data, while lists are preferred for dynamic collections.

1. How do you create a function in Python? Provide an example.

Ans:

A function in Python is defined using the def keyword and can take parameters to perform specific tasks.  
Example:

python

CopyEdit

def greet(name):

return f"Hello, {name}!"

print(greet("Alice")) # Output: Hello, Alice!

1. Explain the difference between local and global variables in python.

Ans:  
**Local Variables:** Declared inside a function and can only be accessed within that function.

* **Global Variables:** Declared outside any function and can be accessed throughout the program.

Example:

global\_var = "I am global" # Global variable

def my\_function():

local\_var = "I am local" # Local variable

print(local\_var) # Accessible only inside this function

my\_function()

print(global\_var) # Accessible anywhere in the script

Modifying a global variable inside a function requires using the global keyword, but it is generally recommended to avoid modifying global variables directly for better code maintainability.

1. How do you handle exception in Python.

Ans:

In Python, exceptions are handled using try-except blocks. The try block contains the code that may raise an exception, while the except block handles the error. Optionally, finally is used for cleanup operations, and else runs if no exception occurs.

Example :

try:

x = 10 / 0

except ZeroDivisionError:

print("Cannot divide by zero.")

finally:

print("Execution completed.")

This ensures the program doesn't crash and handles errors gracefully.