**INTRODUCTION TO PYTHON:**

🡪 Python is a popular, versatile, and easy-to-learn programming language.

🡪 That is widely used in various fields such as web development, data analysis, artificial intelligence, machine learning, scientific computing, and automation.

🡪It was created by **Guido van Rossum in 1991** and is known for its simple syntax, readability, and extensive standard library.

**Key Features of python:**

1.Easy to learn and use.

2.Interpreted Language.

3. Dynamically Typed.

4. Platform-Independent.

**Datatypes:**

🡪A data type is a classification that specifies the kind of value a variable holds. 🡪Python is a dynamically typed language, means you don’t need to declare the type of a variable explicitly.

**Categories of Data Types:**

1. Numeric Types:(integers, floats, and complex numbers)

2. Sequence Types:(strings, lists, and tuples)

3. Set Types:(set and frozenset)

4. Mapping Types: (dict)

5. Boolean Type:(True or False)

6. Binary Types:(bytes, bytearray, memoryview).

7. None Type:(None).

**1. Numeric Types**

These types deal with numbers.

* **int (Integer)**: Represents whole numbers.

Ex: age = 23

* **float (Floating Point)**: Represents decimal numbers.

Ex: price = 89.99

* **complex (Complex Numbers)**: Represents complex numbers with a real and imaginary part.

Ex: c = 5 + 3j

**2. Sequence Types**

These types hold ordered collections of items.

* **str (String)**: A sequence of characters, enclosed in single, double, or triple quotes.

Ex: message = "Hello, Python!"

* **list**: A mutable (changeable) ordered collection of items, which can be of different types. They allow you to store and manipulate multiple items in a single variable.

Ex: fruits = ["apple", "banana", "cherry"]

* **tuple**: An immutable (unchangeable) ordered collection of items. Tuples can hold items of different data types, such as integers, strings, floats, or even other tuples.

Ex: person = ("Ram", 25, "Engineer")

**3. Set Types**

These types hold unordered collections of unique items.

* **set**: A mutable collection of unique items. Sets automatically remove duplicate values, ensuring that each item is unique.

Ex: numbers = {1, 2, 3, 3, 4} # {1, 2, 3, 4}

**1.Union(|):**The union combines all the elements from set1 and set2, but it removes duplicates (since sets only store unique elements).

**Syntax**: set1 | set2 or set1.union(set2)

1. **Intersection (**&**)**:The intersection finds the common elements between set1 and set2

**Syntax**: set1 & set2 or set1.intersection(set2)

1. **Difference(-)**:The difference operation removes the elements of set2 from set1 and returns the result.

**Syntax**: set1 - set2 or set1.difference(set2)

1. **Symmetric Difference (**^**)**:The symmetric difference removes the common elements (3 and 4 in this case) and returns the elements that are unique to each set.

**Syntax**: set1 ^ set2 or set1.symmetric\_difference(set2)

* **frozenset**: An immutable version of a set. Supports set operations like union, intersection, and difference.

Ex: chars = frozenset("hello")  
 print(chars) # frozenset({'h', 'e', 'o', 'l'})

**4. Mapping Type**

* **dict (Dictionary)**: A mutable collection of key-value pairs.

Ex: student = {"Id": 99, "name": "laxmi"}

**5. Boolean Type**

* **bool**: Represents True or False.
* (True is equivalent to 1 and False is equivalent to 0).

Ex: x = 10

y = 20

print (x > y) # Output: False

print (x < y) # Output: True

print (x == 10) # Output: True

**6. Binary Types**

Used for handling binary data.

* **bytes**: Immutable sequence of bytes. Each byte is an integer in the range 0-255.Using b or B as prefix.

Ex: data = b"Hello"  
 print(data) # Output:b'Hello'

* **bytearray**: Mutable sequence of bytes. A bytearray object is similar to bytes. Represented with bytearray(b'...').

Ex: data=bytearray(b”Hello)

Print(data) #Output:datatype(b’Hello’)

* **memoryview**: A memory view object of a bytes-like object.

Ex: data = b”hello”

mv=memoryview(data)

Print(mv[0]) # 104 (ASCII value of h)

**7. None Type**

* **None**: Represents the absence of a value or a null value.

Ex: result = None

Print(result) # Output:None