PYTHON LITERALS

**Python Literals:-**

Python Literals can be defined as data that is given in a variable or constant.

Python supports the following literals:

**1. String literals:**

String literals can be formed by enclosing a text in the quotes. We can use both single as well as double quotes to create a string.

**Example:**

**"Aman" , '12345'**

**Types of Strings:**

There are two types of Strings supported in Python:

**a) Single-line String**- Strings that are terminated within a single-line are known as Single line Strings.

**Example:**

**text1='hello'**

**b) Multi-line String -** A piece of text that is written in multiple lines is known as multiple lines string.

There are two ways to create multiline strings:

**1) Adding black slash at the end of each line.**

**Example:**

**text1='hello\**

**user'**

**print(text1)**

**Output**

**hellouser**

**2) Using triple quotation marks:-**

**Example:**

**str2='''welcome**

**to**

**SSSIT'''**

**print(str2)**

**Output:**

**welcome**

**to**

**SSSIT**

**2. Numeric literals:**

Numeric Literals are immutable. Numeric literals can belong to following four different numerical types.

|  |  |  |  |
| --- | --- | --- | --- |
| **Int(signed integers)** | **Long(long integers)** | **float(floating point)** | **Complex(complex)** |
| Numbers( can be both positive and negative) with no fractional part.eg: 100 | Integers of unlimited size followed by lowercase or uppercase L eg: 87032845L | Real numbers with both integer and fractional part eg: -26.2 | In the form of a+bj where a forms the real part and b forms the imaginary part of the complex number. eg: 3.14j |

**Example - Numeric Literals**

**x = 0b10100 #Binary Literals**

**y = 100 #Decimal Literal**

**z = 0o215 #Octal Literal**

**u = 0x12d #Hexadecimal Literal**

**#Float Literal**

**float\_1 = 100.5**

**float\_2 = 1.5e2**

**#Complex Literal**

**a = 5+3.14j**

**print(x, y, z, u)**

**print(float\_1, float\_2)**

**print(a, a.imag, a.real)**

**Output:**

**20 100 141 301**

**100.5 150.0**

**(5+3.14j) 3.14 5.0**

**3. Boolean literals:**

A Boolean literal can have any of the two values: True or False.

**Example - Boolean Literals**

**x = (1 == True)**

**y = (2 == False)**

**z = (3 == True)**

**a = True + 10**

**b = False + 10**

**print("x is", x)**

**print("y is", y)**

**print("z is", z)**

**print("a:", a)**

**print("b:", b)**

**Output:**

**x is True**

**y is False**

**z is False**

**a: 11**

**b: 10**

**4. Special literals.**

Python contains one special literal i.e., **None.**

None is used to specify to that field that is not created. It is also used for the end of lists in Python.

**Example - Special Literals**

**val1=10**

**val2=None**

**print(val1)**

**print(val2)**

**Output:**

**10**

**None**

**5. Literal Collections.**

Python provides the four types of literal collection such as List literals, Tuple literals, Dict literals, and Set literals.

**List:**

* List contains items of different data types. Lists are mutable i.e., modifiable.
* The values stored in List are separated by comma(,) and enclosed within square brackets([]). We can store different types of data in a List.

**Example - List literals**

**list=['John',678,20.4,'Peter']**

**list1=[456,'Andrew']**

**print(list)**

**print(list + list1)**

**Output:**

**['John', 678, 20.4, 'Peter']**

**['John', 678, 20.4, 'Peter', 456, 'Andrew']**

**Dictionary:**

* Python dictionary stores the data in the key-value pair.
* It is enclosed by curly-braces {} and each pair is separated by the commas(,).

**Example**

**dict = {'name': 'Pater', 'Age':18,'Roll\_nu':101}**

**print(dict)**

**Output:**

**{'name': 'Pater', 'Age': 18, 'Roll\_nu': 101}**

**Tuple:**

* Python tuple is a collection of different data-type. It is immutable which means it cannot be modified after creation.
* It is enclosed by the parentheses () and each element is separated by the comma(,).

**Example**

**tup = (10,20,"Dev",[2,3,4])**

**print(tup)**

**Output:**

**(10, 20, 'Dev', [2, 3, 4])**

**Set:**

* Python set is the collection of the unordered dataset.
* It is enclosed by the {} and each element is separated by the comma(,).

**Example: - Set Literals**

**set = {'apple','grapes','guava','papaya'}**

**print(set)**

**Output:**

**{'guava', 'apple', 'papaya', 'grapes'}**