Python FSD

Python Type Casting

TYPE CASTING

- We can convert one type to another type. This conversion is called type casting.
- Python supports a wide variety of functions or methods like: int(), float(), str(), hex(), oct(), tuple(), set(), list(), dict(), etc. for the type casting in python.
- There are two varieties of typecasting in python namely - Explicit Conversion(Explicit type casting in python), and Implicit Conversion(Implicit type casting in python).

PYTHON IMPLICIT TYPE CONVERSION

- In certain situations, Python automatically converts one data type to another. This is known as implicit type conversion

```
integer_number = 123
float_number = 1.23

new_number = integer_number + float_number

# display new value and resulting data type
print("Value:",new_number)
print("Data Type:",type(new_number))
```

EXPLICIT TYPE CASTING

- We get TypeError, if we try to add str and int. For example, '12' + 23. Python is not able to use Implicit Conversion in such conditions.
- Python has a solution for these types of situations which is known as Explicit Conversion.
- In Explicit Type Conversion, users convert the data type of an object to required data type.

ADDITION OF STRING AND INTEGER USING EXPLICIT CONVERSION

- We can add a string and an integer in python. If we convert a string into an integer, we can add both integers to get the desired output.

```
string = "56"
number = 44

# Converting the string into an integer number.
string_number = int(string)

sum_of_numbers = number + string_number
print("The Sum of both the numbers is: ", sum_of_numbers)
```

TYPE CASTING INT TO STRING

- We can use the str() function in python to convert a specified value or an object into a string object.

```
number = 55
string = str(number)
print("Converted string is:",string)
```

CONVERTING STRING TO TUPLE

- Tuple() : this function is use to convert string to tuple.

```
#string to tuple
input_string ="Welcome to python training"
print(tuple(input_string));
```

CONVERTING STRING TO SET

- The set() function is used to convert any iterable into sets. A set is a mutable collection that can only store distinct values. By mutable, it means that the set can be changed or modified.

```
#string to set
x = 'success'
print(set(x))
```