Chapter 2 – Data Type & Basics

Strings:

- String is shown with single or double quotation marks.
- Eg: “Shivra” or ‘Shivrar’.
- String as array:
  - a=“Shivrar”
  - Print(a[1])
  - Print(a[3])

  Looping Through a String:
  - For x in “banana”:

        print(x)

  String Length:

  s=“Hello,Class”

  Print(len(a))

- Check String:
  - msg = “Very good morning students "
    print(“good" in msg)
  - msg = “Very good morning students"
    if “good" in msg:
      print("Yes, ‘good' is present.")
- Check if not String:
  - msg = " Very good morning students
    print(“animations” in msg)
  - msg = “Very good morning students"
    if “animations" in msg:
      print("Yes, ‘animations' is present.")

- Concatenation,Repitation,Slicing:

  Concatenation:

  “python” + “Tutorial” = pythontutorial

  Repitation:

  Python*2

  Slicing(range):

```
b = "Hello, World!“
print(b[2:5])
b = "Hello, World!"
print(b[:5])
 b = "Hello, World!"
print(b[2:])
b = "Hello, World!"
print(b[-5:-2])
b = "Tutorial"
print(b[-5:-2])
```

Modification:

Replace:

```
a = "Hello, Students”
print(a.replace(“e", "J"))
```

Split:

```
a = "Hello, World!"
b = a.split(",")
print(b)
```

Formatting:

Use the format() method to insert numbers into strings:

```
age = 36
txt = "My name is John, and I am {}"
print(txt.format(age))
```

The format() method takes unlimited number of arguments, and are placed into the respective placeholders:

```
quantity = 3
itemno = 567
price = 49.95
myorder = "I want {} pieces of item {} for {} dollars."
print(myorder.format(quantity, itemno, price))
```

Type Specific Method:

```
find():
    str=‘edurekha’
    str.find(‘rekha’)
Replace():
    str=‘Pythons’
    str.replace(‘ns’,‘n’)
max():
    str=‘missisippi’
    max(str)
min():
    str=‘missisippi’
    min(str)
```

Lists:

- Lists are used to store multiple items in a single variable.
- List items are ordered, changeable, and allow duplicate values.
- List items are indexed, the first item has index [0], the second item has index [1] etc.
- It can allow duplicates.
  - thislist = ["apple", "banana", "cherry", "apple", "cherry"]
    print(thislist)
  - thislist = ["apple", "banana", "cherry"]
    print(len(thislist))

  The List() constructor:
  - thislist = list(("apple", "banana", "cherry")) # note the double round-brackets
    print(thislist)

Access Items:

```
thislist = ["apple", "banana", "cherry"]
print(thislist[1])
```

Range of index:

```
list = ["apple", "banana", "cherry"]
print(list[1:])

 list = ["apple", "banana", "cherry, “mango"]
print(list[:4])
```

Change Item Value

```
list = ["apple", "banana", "cherry"]
list[1] = "blackcurrant"
print(list)

 list = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]
list[1:3] = ["blackcurrant", "watermelon"]
print(list)
```

Insert Items:

```
list = ["apple", "banana", "cherry"]
list.insert(2, "watermelon")
print(list)
```

Append Items:

```
list = ["apple", "banana", "cherry"]
list.append("orange")
print(list)
```

Insert Items:

```
list = ["apple", "banana", "cherry"]
list.insert(1, "orange")
print(list)
```

Extend List:

```
list = ["apple", "banana", "cherry"]
tropical = ["mango", "pineapple", "papaya"]
list.extend(tropical)
print(list)
```

Remove List:

```
list = ["apple", "banana", "cherry"]
list.remove("banana")
print(list)
```

Remove Specified index:

```
 list = ["apple", "banana", "cherry"]
list.pop(1)
print(list)
```

Sort List Ascending:

```
 list = ["orange", "mango", "kiwi", "pineapple", "banana"]
list.sort()
print(list)
```

Sort in Descending:

```
list = ["orange", "mango", "kiwi", "pineapple", "banana"]
list.sort(reverse = True)
print(list)
```

Copy List:

```
thislist = ["apple", "banana", "cherry"]
mylist = thislist.copy()
print(mylist)
```

Join List:

```
list1 = ["a", "b", "c"]
list2 = [1, 2, 3]
list3 = list1 + list2
print(list3)

list1 = ["a", "b" , "c"]
list2 = [1, 2, 3]
for x in list2:
    list1.append(x)
print(list1)
```

Tupples:

- A tupples is a sequence of immutable python objects like: floating number , String, literals etc.
- The tupples can’t be changed unlike lists.
- Tuples are defined using curve brackets.
  - Mytuples=(‘edurekha’, 2.4, 5, ‘python’)

  Concatenation:

  tup(‘a’, ‘b’, ‘c’)

  tup+(‘d’, ‘e’)

  Repetetion:

  tup(‘a’, ‘b’, ‘c’)

  tup*2

Accessing:

```
    thistuple = ("apple", "banana", "cherry")
    print(thistuple[1])
```

Concatenation(Joining):

```
    tuple1 = ("a", "b" , "c")
    tuple2 = (1, 2, 3)
    tuple3 = tuple1 + tuple2
    print(tuple3)
```

Slicing:

```
    tup=(‘a’, ‘b’, ‘c’)
    tup[1:2]
```

Indexing:

```
    tup=(‘a’, ‘b’, ‘c’)
    tup[0]
```

Length:

```
    tup=(‘please’, ‘keep’, ‘quit’)
    print(len(tup))
```

```
tuple1 = ("abc", 34, True, 40, "male")
```

Sets:

- Sets are used to store multiple items in a single variable.
- A set is a collection which is both unordered and unindexed.
- Sets are written with curly brackets.
- Every element is unique (no duplicate).
- Set items are unordered, unchangeable, and do not allow duplicate values.
    - thisset = {"apple", "banana", "cherry", "apple"}
      print(thisset)

  Set Constructor()

    - thisset = set(("apple", "banana", "cherry")) # note the double round-brackets
      print(thisset)

Access Items:

```
thisset = {"apple", "banana", "cherry"}
for x in thisset:
    print(x)
thisset = {"apple", "banana", "cherry"}
print("banana" in thisset)
```

Add Items:

```
thisset = {"apple", "banana", "cherry"}
thisset.add("orange")
print(thisset)
```

Add Sets:

```
thisset = {"apple", "banana", "cherry"}
tropical = {"pineapple", "mango", "papaya"}
thisset.update(tropical)
print(thisset)
```

Update:

```
thisset = {"apple", "banana", "cherry"}
mylist = ["kiwi", "orange"]
thisset.update(mylist)
print(thisset)
```

Remove Items:

```
thisset = {"apple", "banana", "cherry"}
thisset.remove("banana")
print(thisset)

thisset = {"apple", "banana", "cherry"}
thisset.discard("banana")
print(thisset)
```

Empty the set:

```
thisset = {"apple", "banana", "cherry"}
thisset.clear()
print(thisset)
```

Join Sets:

```
set1 = {"a", "b" , "c"}
set2 = {1, 2, 3}
set3 = set1.union(set2)
print(set3)
```

Insert the elements of set2 into set1:

```
set1 = {"a", "b" , "c"}
set2 = {1, 2, 3}
set1.update(set2)
print(set1)
```

Keep Only the duplicates:

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
x.intersection_update(y)
print(x)
```

The symmetric_difference() method will return a new set, that contains only the elements that are NOT present in both sets

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}
z = x.symmetric_difference(y)
print(z)
```

Dictionaries:

- Dictionaries are used to store data values in key:value pairs.
- A dictionary is a collection which is ordered*, changeable and does not allow duplicates.
- Dictionary items are ordered, changeable, and does not allow duplicates.
- Dictionary items are presented in key:value pairs, and can be referred to by using the key name.
- Dictionary Length:
    - print(len(thisdict))

Accessing Items:

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
x = thisdict["model"]
```

Get the value of key:

```
x = thisdict.get("model")
```

List of the keys:

```
x = thisdict.keys()
```

Add A new Item:

```
    car = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
    }
    x = car.keys()
    print(x) #before the change
    car["color"] = "white"
    print(x) #after the change
x = thisdict.values()
```

Updatation in original Dictionary:

```
car = {
"brand": "Ford",
"model": "Mustang",
"year": 1964
}
x = car.values()
print(x) #before the change
car["year"] = 2020
print(x) #after the change
```

Updation in values of Dictionary:

```
car["year"] = 2020
print(x) #after the change
```

Check If Key Exists:

```
if "model" in thisdict:
print("Yes, 'model' is one of the keys in the   thisdict dictionary")
```

Add New Items:

```
thisdict = {
        "brand": "Ford",
        "model": "Mustang",
        "year": 1964
}
thisdict["color"] = "red"
print(thisdict)
```

Remove the Items:

```
 thisdict = {
"brand": "Ford",
 "model": "Mustang",
"year": 1964
}
thisdict.pop("model")
print(thisdict)
```

The del keyword removes the item with the specified key name:

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
del thisdict["model"]
print(thisdict)
```

Copy the dictionary:

```
thisdict = {
    "brand": "Ford",
    "model": "Mustang",
    "year": 1964
}
mydict = thisdict.copy()
print(mydict)
```

Differences:

- List is a collection which is ordered and changeable. Allows duplicate members.
- **Tuple** is a collection which is ordered and unchangeable. Allows duplicate members.
- Set is a collection which is unordered and unindexed. No duplicate members.
- **Dictionary** is a collection which is ordered* and changeable. No duplicate members.