* **Container:**
  + Container is an entity that contains multiple data items.
  + Also known as a collection or a compound data type
* **Container Data Types:**
  + **Lists**
  + **Tuples**
  + **Sets**
  + **Dictionaries**
* **Dictionaries**
  + **Dictionary is a collection of key-value pairs.**
  + Also known as maps or associative arrays.
  + **A Dictionary contains comma separated key: value pairs enclosed within { }.**
    - d1 = { } 🡨 Empty Dictionary
    - d2 = { ‘a101’ : ‘Aditya’ , ‘r101’ : ‘Rishabh’ }

here, ‘a101’, ‘r101’ are keys, whereas ‘Aditya’, ‘Rishabh’ are values.

* + Different keys may have same values.
    - d = { 10 : ‘A’, 20 : ‘A’, 30 : ‘Z’ }
  + Keys must be unique. If keys are same but values are different, latest key value pair gets stored.
    - d = { 10 : ‘A’, 20 : ‘B’, 10 : ‘Z’ } 🡨 will store { 10 : ‘Z’, 20 : ‘B’}
  + If key value pairs are repeated, then only one pair get stored.
    - d = { 10 : ‘A’, 20 : ‘B’, 10 : ‘A’ } 🡨 will store { 10 : ‘A’, 20 : ‘B’}
  + Keys are immutable but values can be changed. Number, String and Tuple can become keys.
  + Keys must be consisting of one element.
* **Accessing Dictionary Elements**
  + print (num) 🡨 entire Dictionary can be printed by just using the name of the Dictionary.
  + print (d[10]) 🡨 ‘A’ 🡨 Dictionary preserves insertion order. However, elements are

not accessed using the position but using the key. Thus, elements are not position indexed, but key indexed.

* + Dictionaries can’t be sliced.
* **Looping in Dictionaries**
  + Using **for** only**.**
  + #iterate over key-value pairs # iterate over keys

for k, v in d.items( ): for k in d.keys( ):

print (k, v) print (k)

* + # iterate over keys – shorter way # iterate over values  
    for k in d: for v in d.values( ):

print (k) print (v)

* + #Use buit-in enumerate( ) function to keep track of index of the key-value pairs that is being referred to

for i, (k, v) in enumerate(d.items( )): # 🡨 ( ) around k, v are necessary.

print (i,k)

* **Basic Dictionary Operations**
  + **Dictionaries are mutable (changeable).** So we can perform add/delete/modify operations on it.
    - d[30] = ‘C’ 🡨 add new key-value pair. Insertion will take place at the end

of the existing dictionary, since dictionary preserves the

insertion order.

* + - d[10] = ‘D’ 🡨 modify value for a key.
    - del(d[30]) 🡨 delete a key-value pair.
    - del(d) 🡨 delete dictionary object.
  + **Dictionary keys can’t be changed in place.**
  + **One Dictionary can’t be concatenated (appended) at the end of another Dictionary.**
  + **Two Dictionaries can’t be merged to create a new Dictionary.**
  + **Two Dictionary objects can’t be compared using < , >.**
  + **Conversion:** 
    - **A string/list/set can be converted into a Dictionary using the dict() conversion function.**
    - Dict = dict({1: 'Microsoft', 2: 'Google', 3:'Facebook'})
  + **Aliasing:** 
    - **On assigning one Dictionary to another, both refer to the same Dictionary. Changing one changes the other.**
    - **Also known as shallow copy or aliasing.**
      * d1 = { 15: ‘Fifteen’ , 8: ‘Eight’ , 6 : ‘Six’}
      * d2 = d1 🡨 doesn’t copy Dictionary. d2 refers to the same Dictionary as d1.
      * print (d1 is d2) 🡨 True
  + **Cloning is allowed.**
  + **Searching is allowed.**
  + **Identity checking is allowed.**
    - **Use (is) operator to check whether the two variables are referring to the same Dictionary.**
      * d1 = { 15: ‘Fifteen’ , 8: ‘Eight’ , 6 : ‘Six’}
      * d2 = d1 🡨 doesn’t copy Dictionary. d2 refers to the same Dictionary as d1.
      * print (d1 is d2) 🡨 True
  + **Emptiness:** 
    - **We can check if a Dictionary is empty using not operator.**
      * d = { }
      * if not d:

print (“Empty Dictionary.”)

* + - * print (bool(d)) 🡨 False
  + **Built-in functions on Dictionaries**
    - len(d) : return number of key-value pairs.
    - max(d) : return maximum key in dictionary d.
    - min(d) : return minimum key in dictionary d.
    - sum(d) : return sum of all keys if keys are numbers.
    - any(d) : returns True if any key of dictionary d is True.
    - all(d) : returns True if all keys of dictionary d are True.
    - reversed(d) : can be used for reversing dict/keys/ values.
    - Examples of reversed( )
      * Use of reversed function to reverse a dictionary by keys:
        + d1 = { 15: ‘Fifteen’ , 8: ‘Eight’ , 6 : ‘Six’}

for k, v in reversed(d1.items()):

print (k, v)

* + **Dictionary Methods**
    - **Any Dictionary is an object of type Dictionary.**
    - **Its methods can be accessed using the syntax dic.method( )**
      * d = {72: ‘Darshit’ , 60: ‘Amita’ , 58 : ‘Ashit’ , 52: ‘Parul’, 54 : ‘Devang’, 56 : ‘Nina’}
      * n = { 74: 'Ragi', 2006 : 'Aashna' }
      * d.update(n) 🡨 one dictionary (n) can be merged with another one (d).
      * print (d.popitem()) 🡨 removes and returns item in LIFO order.
      * print (d.pop(54)) 🡨 removes key and returns its value.
      * d.clear() 🡨 Makes d an empty dictionary.
  + **Dictionary Varieties**
    - **Keys in a dictionary must be unique and immutable. Numbers, strings or tuples can be used as keys. If tuple is used as a key it should not contain any mutable element like list.**
      * d = { (1, 5) : ‘ME126’, (3, 2) : ‘ME102’}
    - **Dictionaries can be nested.** 
      * contacts = { 'Darshit' : { 'DoB' : '18/02/72', 'BG' : 'O+ve' },

'Ragi' : { 'DoB': '26/04/74', 'BG' : 'A+ve' } }

* + - * print (contacts)
      * Output:

{'Darshit': {'DoB': '18/02/72', 'BG': 'O+ve'}, 'Ragi': {'DoB': '26/04/74', 'BG': 'A+ve'}}

* + - * print (contacts['Darshit'])
      * Output:

{'DoB': '18/02/72', 'BG': 'O+ve'}

* + - **Two dictionaries can be merged to create a third dictionary by unpacking the two dictionaries using \*\*. If we use \* only keys will be unpacked.**
      * animals = { 'Tiger' : 141, 'Lion' : 663, ' leopard' : 110 }
      * birds = { 'Eagle' : 38, 'Sparrow' : 450, 'Parrot' : 200 }
      * combined = { \*\* animals, \*\* birds }
      * print (combined)
      * Output:

{'Tiger': 141, 'Lion': 663, ' leopard': 110, 'Eagle': 38, 'Sparrow': 450, 'Parrot': 200}

* + - **Sorting Dictionaries:**
      * wildAnimals = sorted(animals.items ( ) ) 🡨 Based on key.
      * mybirds = sorted(birds.items(), reverse = True) 🡨 Based on Reverse order of key.
      * **import operator**
      * c1 = sorted(combined.items(), key = operator.itemgetter(1)) 🡨 Sorting by value.
      * c1 = sorted(combined.items(), key = operator.itemgetter(1), reverse = True) 🡨 Sorting by value in reverse order.
* **Write following programs considering Dictionary in mind:**

1. Write a program to create three dictionaries and concatenate them to create fourth dictionary.
2. Write a program to check whether a dictionary is empty or not.
3. Create a dictionary with dept no, employee roll no. and salary. Find out department wise min and maximum of salary.
4. Write a program that reads a string from the keyboard and creates dictionary containing frequency of each character occurring in the string.
5. Create two dictionaries – one containing grocery items and their prices and another containing grocery items and quantity purchased. By using the values from these two dictionaries compute the total bill.