```
1   from collections import OrderedDict
2   from collections import defaultdict
3
4   # Different ways of constructing a dictionary.
5   d = {}
6   d = dict()
7   d = dict(Bangalore=25, Chennai=35, Delhi=30)
8   d = dict([("Bangalore", 25), ("Chennai", 35), ("Delhi", 30)])
9   d = dict(zip(["Bangalore", "Chennai", "Delhi"], [25, 35, 30]))
10  d = dict({'Bangalore': 25, "Chennai": 35, "Delhi": 30})
11
12  print(len(d))    # Prints the length of the dictionary
13
14  # Accessing elements of a dictionary
15  print(d['Bangalore'])
16  print(d.get('Bangalore'))
17
18  # list inside the dictionary as values.
19  location = {'country': 'India', 'states': ['Karnataka', 'Andra', 'Kerala']}
20
21  # points
22  points = {'a': 1, 'b': 2, 'c': 3}
23
24  # Nested Dictionary
25  prices = {'IBM': {'current': 90.1, 'low': 88.3, 'high': 92.7}, 'HP': {"curren
26
27  # accessing inner dictionary
```

```
print(prices['IBM']['current'])
print(prices['IBM']['high'])

# Accessing a key that does not exist
# print(employee['age'])      # Throws exception KeyError: 'age'
print(d.get('Noida'))      # get() method Does not throw an exception, but re
print(d.get('Noida', 'The Key not found in the dictionary'))   # Throws excep

# Adding / Updating the dictionary
d['Mysore'] = 26.5  # Upadting the dictionary key with new value
d['Bangalore'] = 27.0

# Appending items to the list which is value of the key 'states'
location['states'].append("Gujrat")
location['states'].append("Maharastra")

# Incrementing value of key 'a'
points['a'] = points['a'] + 1
points['a'] += 1

# Adding a new key value pair
points['d'] = 1
points['d'] = points['d'] + 1

# Looping through Key's and Value's of the Dictionary
print(d.items())     # Returns a tuple of key,value pairs

for item in d:      # Prints only key's of the dictionary
    print(item)

for item in d:
    print(d[item])   # Prints Values of the dictionary

for key, value in d.items():     # Tuple un-packing
    print(key, value)

for key in d.keys():
    print(key)

for value in d.values():
    print(value)

for index, items in enumerate(d.items()):
    print(index, items)

```

```
# Count number of words in a sentence
sentence = 'hello world hello world welcome to python'
words = sentence.split()

word_count = {}
for word in words:
    if word in word_count:
        word_count[word] += 1
    else:
        word_count[word] = 1

print(word_count)

# Using get method
word_count = {}
for word in words:
    word_count[word] = word_count.get(word, 0) + 1

"""
1. This creates a new dictionary using our colors as keys, with all values se
2. This allows us to increment each key without worrying whether it has been
"""
for name in names:
    _count[name] += 1

# using defaultDict
word_count = defaultdict(int)
for word in words:
    word_count[word] += 1

# Counting number of characters in a string
s = 'abracadabraca'
char_count = {}
for c in s:
    if c in char_count:
        char_count[c] += 1
    else:
        char_count[c] = 1

# Counting number of vowels in a string
s = 'hello world welcome to python'
vowels = {}
for c in s:
    if c in 'aeiou':
        if c in vowels:
```

```
            vowels[c] +=1
        else:
            vowels[c] = 1
print(vowels)

# defaultDict
# 1. Creats a key if the key does not exist
# 2. Initialise the value to Zero in case of defaultdict of int's
# 3. Returns the value which is zero

# Counting occurances of word in the string
sentence = "hello world welcome to python hello hi hello hello"
word_count = defaultdict(int)
words = sentence.split()
for word in words:
    word_count[word] += 1

# Counting occurances of each character in the string
s = 'abracadabraca'
chr_count = defaultdict(int)
for c in s:
    chr_count[c] += 1

# 1. Creats a key if the key does not exist
# 2. Initialise the value to empty list in case of defaultdict of list
# 3. Returns the empty list

profile = defaultdict(list)     # One to Many Mapping
profile['language'].append('Java')
profile['language'].append('Python')

cities = [('India', 'Bangalore'),
          ('India', 'Chennai'),
          ('India', 'Delhi'),
          ('India', 'Kolkata'),
          ('USA', 'Dallas'),
          ('USA', 'New York'),
          ('USA', 'Chicago'),
          ('China', 'Bejing'),
          ('China', 'Shaingai')
          ]

dd = defaultdict(list)
for country, city in cities:
    dd[country].append(city)
```

```
163
164 # Composite Keys
165 # Dictionary key must be of Immutable Type. e.g
166 # Dict keys should always be Hashable. (All immutable objects are Hashable)
167 holidays = {
168     (26, 1): 'Republic Day',
169     (15, 8): 'Independance Day',
170     (25, 6): 'Yoga Day'
171 }
172
173 # Deleting the key and value
174 d.popitem()      # Returns and deletes the last key/value pair in the diction
175 print(d.pop('age'))    # Returns and Deletes the mentioned key from the dicti
176 # del employee['age']     # Deletes the Key 'age' and its value
177
178
179 # Merging Dictionaries
180 d1 = {'fname': 'steve', 'lname': 'jobs'}
181 d2 = {'age': 56, 'company': 'apple'}
182
183 d3 = {**d1, **d2}
184
185
186 # Using fromkeys method
187 names = ['apple', 'google', 'yahoo', 'gmail', 'google', 'apple']
188 _count = dict.fromkeys(names, 0)
189
190 # OrderedDict
191 # Ordered Dictonary Maintains Order
192 d = OrderedDict()
193 d['apple'] = 'A'
194 d['google'] = 'G'
195 d['yahoo'] = 'Y'
196
197 for key, value in d.items():
198     print(key, value)
```