Dictionary Comprehension
Dictionary comprehension allows to create dictionary using for loop and if statement

Syntax-1: {key:value for variable in iterable}
Syntax-2: {key:value for variable in iterable if test}

Example:

```
# create student dictionary with rollno as key and name as value

n=int(input("enter how many students?"))

#without comprehension
stud_dict={}
for i in range(n):
    rollno=int(input("enter rollno"))
    name=input("enter name")
    stud_dict[rollno]=name

print(stud_dict)

#with comprehension

stud={int(input("enter rollno")):input("enter name") for i in range(n)}
print(stud)
```

Output:

```
enter how many students?2
enter rollno101
enter namenaresh
enter rollno102
enter namesuresh
{101: 'naresh', 102: 'suresh'}
enter rollno101
enter namenaresh
enter rollno102
enter namesuresh
{101: 'naresh', 102: 'suresh'}
```

Example:

```
>>> dict1={num:num**2 for num in range(1,11)}
>>> print(dict1)
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100}
import math
>>> dict2={num:math.sqrt(num) for num in range(1,11)}
>>> print(dict2)
{1: 1.0, 2: 1.4142135623730951, 3: 1.7320508075688772, 4: 2.0, 5:
2.23606797749979, 6: 2.449489742783178, 7: 2.6457513110645907, 8:
2.8284271247461903, 9: 3.0, 10: 3.1622776601683795}
>>> dict3={num:chr(num) for num in range(65,91)}
>>> print(dict3)
{65: 'A', 66: 'B', 67: 'C', 68: 'D', 69: 'E', 70: 'F', 71: 'G', 72: 'H', 73: 'I', 74: 'J',
75: 'K', 76: 'L', 77: 'M', 78: 'N', 79: 'O', 80: 'P', 81: 'Q', 82: 'R', 83: 'S', 84: 'T',
85: 'U', 86: 'V', 87: 'W', 88: 'X', 89: 'Y', 90: 'Z'}
>>> dict3[65]
'A'
```

Example:

```
# create student dictionary with rollno as key and
# name,2 subject marks as values

n=int(input("enter how many students?"))
stud_dict={int(input("enter rollno")):[input("enter name"),int(input("enter
sub1")),int(input("enter sub2"))] for i in range(n)}
for name,l in stud_dict.items():
    print(name,l)
```

Output:

```
enter how many students?2
enter rollno101
enter namenaresh
enter sub150
enter sub260
enter rollno102
enter namesuresh
enter sub190
enter sub298
101 ['naresh', 50, 60]
102 ['suresh', 90, 98]
```

Example:

```
>>> student_grade={'naresh':'A',
            'suresh':'B',
            'kishore':'A',
            'ramesh':'B',
            'rajesh':'C'}
>>> student_dictA={name:grade for name,grade in student_grade.items() if
grade=='A'}
>>> print(student_dictA)
{'naresh': 'A', 'kishore': 'A'}
>>> student_dictB={name:grade for name,grade in student_grade.items() if
grade=='B'}
>>> print(student_dictB)
{'suresh': 'B', 'ramesh': 'B'}
>>> sales_dict={2010:45000,
...          2011:55000,
...          2012:35000,
...          2013:65000,
...          2014:76000,
...          2015:78000,
...          2016:45000}
>>> sales_dict1={year:sales for year,sales in sales_dict.items() if
sales<=50000}
>>> print(sales_dict1)
{2010: 45000, 2012: 35000, 2016: 45000}
```

Nested dictionary

Nested dictionaries are allowed.

In order to represent nested dictionary, inner dictionary must be defined as value.

Syntax: {key:{key:value,key:value,…},
key:{key:value,key:value,…}}

```
>>> sales={2010:[1000,2000,3000,4000,5400],
    2011:[6000,3000,2000,3500,6500]}
```

```
>>> print(sales)
{2010: [1000, 2000, 3000, 4000, 5400], 2011: [6000, 3000, 2000, 3500,
6500]}
>>> sales[2010][0]
1000
>>> sales[2010][2]
3000
>>> sales[2011][-1]
6500
>>> sales={2010:{'jan':1000,'feb':2000,'mar':5000},
...        2011:{'jan':6000,'feb':4500,'mar':9000}}
>>> sales[2010]['mar']
5000
>>> sales[2010]['jan']
1000
>>> sales[2011]['feb']
4500
```

What is difference between list,set and dictionary?

List	**Set**	**Dictionary**
List is ordered collection	Set is unordered collection	Dictionary is mapping collection
In list data is organized in sequential order	In set data is organized using hashing data structure	In dictionary data is organized as key and value pair
List allows duplicates	Set does not allows duplicates	Dictionary allows duplicate values but does not allows duplicate keys
List in sequence which support indexing and slicing	Does not support index	This support key and does not support indexing and slicing
Lit is created using []	Set is created with {value,…}	Dictionary is created using {key:value,…}

String

String is an immutable sequence data type.
String is collection of characters; these characters can be alphabets, digits or special characters.

The string contains only alphabets is called alphabetic string
The string contains alphabets and digits is called alpha numeric string

“str” class or data type represents string object

We can represent string in 3 ways

1. Within single quotes
2. Within double quotes
3. Within triple single quotes or double quotes

Example:

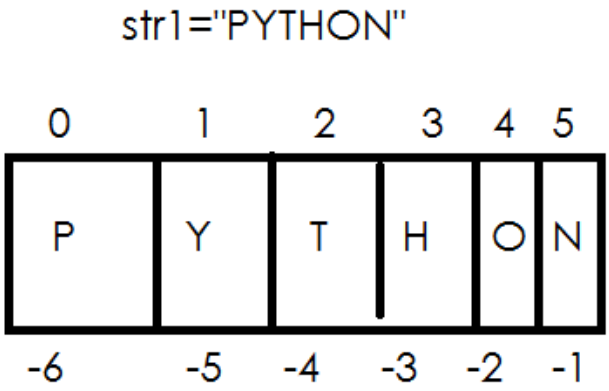
```
>>> str1='PYTHON'
>>> str2="PYTHON"
>>> str3='''PYTHON'''
>>> str4="""PYTHON"""
>>> print(str1,str2,str3,str4,sep="\n")
PYTHON
PYTHON
PYTHON
PYTHON
>>> s1='Python is a "simple" language'
>>> s2="python is a 'simple' language"
>>> s3='''python is a
general purpose
programming
language'''
>>> s4="""python is a
general purpose
programming
language"""
>>> print(s1,s2,s3,s4,sep="\n")
Python is a "simple" language
python is a 'simple' language
python is a
general purpose
programming
language
python is a
general purpose
```

programming
language
>>>

String is a sequence data type and it allows reading characters in different ways

1. Using index
2. Using slicing
3. Using for loop
4. Using iterator
5. Using enumerate

Using index




```
>>> str1="PYTHON"
>>> str1[0]="p"
Traceback (most recent call last):
  File "<pyshell#56>", line 1, in <module>
    str1[0]="p"
TypeError: 'str' object does not support item assignment
```