Python-Lists,Sets,Tuple

Lists	Sets	Tuple
Def: List is used to store the different types of data like int,float,string,bool	**Def**:Set is a collection which is unordered(Not in order) and Unindexed.	**Def:**Tuples are used to store the values/items in a single variable.
Mutable(Which can modify the data)	Mutable(We can change the data)	Immutable(Can't be changed)
Lists: Represented by square brackets [ ].	Represented by curly braces{ }.	Represented by Round brackets ( ).
Allow Duplicates	Allow Duplicates	Allow Duplicates.
Eg: values=[1,8,23,11,4,2]	Eg: values={5,9,11,4,2}	Eg:values=(3, 8,'2', [3,9,12])
Syntax: variable.method()	Syntax: variable.method()	Syntax: variable.method()
In lists ,we can access the values through index values value[0]	Sets cannot be referred to by Index (or) Key.	Sets can be accessed by index values.
Slicing can be Possible	Slicing is not Possible.	Slicing is possible.
Methods: append(),copy(),clear(), count(),extend(),index(), insert(),insert(),pop(),remove(),reverse(),sort()	**Methods:** add(),clear(),pop(),update(),remove(),copy() **Operations**: union(),difference(),intersection(), isdisjoint(),issubset(),issuperset(), symmetric_difference()	Methods: len(),max(),min(),sum()

Python- Dictionary,Strings,Frozenset

Dictionary	Strings	Frozenset
Def: Dictionary is to store the data and enclosing a comma-separated list by Key-value pairs	**Def**: String is a sequence of characters. Eg: a-z characters.	**Def**:
Here Keys are Immutable, Mutable(Which can be modify)	Strings are Immutable(Can't be changed)	
Dictionary: Represented by Curly brackets { }.	String represented by single,double,triple quotes→' ' , " ", "" ""	
Keys won't allow Duplicates		
Eg: values={'1': 'Apple', '2':'Redmi', '3': Samsung}	Eg: string='Python class' string="Python class" string='''Python class'''	
Syntax: variable.method()	—------	
In the Dictionary ,we can't access the index values.Just we can access the value by key only	—------	
Slicing can't be Possible	—------	
Methods: copy(),clear(), get(),items(),key(),pop(), update(),values()	Methods: count(),endswith(),find(),format(), index(),isalnum(),join(),lower(), lstrip(),replace(),rstrip(),split(), startswith(),strip(),title(),upper(), isalpha()	

Functions and Advance Functions

Functions:

A Function is a block of code which only runs when it is called.

Syntax:

```
def  first_function():    # function definition
        print("This is my python")    #function body
first_function()    #function calling
```

Example:

```
def my_sum(a,b):
        return a+b
my_sum(10,34)
```

Output: 44

- With in the function anything we can implement

Example:

```
        def  karun():
                print("Python")
        divya()
```

Output: Python

Example:

```
        def karun(a)"    # Here 'a' refers the single parameter
                print("python", a)
        karun("Program")  # 'Program' refers the Argument
```

Example:

```
        def karun(*a):  # * refers the Arbitrary Parameter
                print(a)
        karun(1,6,3,8,92)
```

Output: Here o/p data will be printed as Tuple because of *

Example:

```
        def karun(**a):   # **a refers the Keyword Argument
                print(a)
        karun(a=20, b=30, c=199)
```

Output: {'a'=1, 'b'=30. 'c'=199}   #Here ** will print the output as a Dictionary

File Handling and Error Handling

File Handling:

File Operations such as operating a file, reading from it, writing, losing, remaining a file, deleting a file and various file methods.

- By using File handling , we can run these operations using “Python Code”.

Modes of Files:

r → read operation : only read
w → write operation : write,create new file, truncate
r+ → read and write operation : read,write
w+ → write and read operation : read, write, create new file, truncate
a → append operation : write, create new file, truncate.

Syntax: Read: It is used to open the file

```
f= open(‘demo.txt’, mode=’r’)
        c=f.read()
        print(c)
        f.close()
```

- File should be closed after opening the file.
- read() → To read the data
- readline() → To get only single line
- readlines() → To get total data
- read(4) → To get that particular data.

Syntax:Write→ Data loss

```
f=open(‘demo.txt’, mode=’w’)
        f.write(“This is Python”)
```

Syntax: Append → Data will be inserted

```
f=open(‘demo.txt’, mode=’a’)
        c= f.write(“Python programming”)
        f.close()
```