Frozenset () → It is an inbuilt () which uses to change mutable values into immutable values i.e → The elements of sets is not added or remove once created

Syn:- frozenset (iterable_object_name)

Ex:-

```
nu = {1, 2, 3, 4, 5, 6, 7, 8, 9}
fnum = frozenset(nu)
print ("frozenset object is :", fnum)
```

op

frozenset object is : frozenset {1, 2, 3, 4, 5, 6, 7, 8, 9}

Dictionary:—

It is a datatype which is uses to store data values in key: value pairs.

> A dictionary is a collection of data which is ordered, changeable and do not allow duplicates.

Creating dictionary:—

Dict_name = { key : value }

Operation performed in dictionary:—

Adding an item

→ Dict_name [key] = value

Changing value of an item

→ Dict_name [existing_key] = new_value

- To get number of items in a dictionary

→ len (Dict_name)

- Print only keys :—

→ Dict_name . keys ()

- Print only values

→ Dict_name . values ()

Removing from dictionary :—

For removing we use following funch.

1) Pop — This is use to remove the item with specified key name.

ex :—

```
# use of pop()
dog1 = {"name": "jack", "age": 15, "colour": "black", "height": 6, "weight": 65}
print (dog1)
dog1.pop ('colour')
print (dog1)
```

O/P

{"name": "jack", "age": 15, "colour": "black", "height": 6, "weight": 65}

{"name": "jack", ---- (Colour) not ((

2) popitem = This funch is uses to remove the last inserted item.

ex :—

```
dog1.popitem ()
print (dog1)
```

O/P

{"name": "jack", "age": 15, "height": 6}

Del → This keyword deletes the item with specific key name.

Ex :—

```
del dog1["age"]
print(dog1)
print(dog1)
```

O/P

{'name': 'jack', 'height': 6}

clear() → This funch is uses to clears all the items in the dictionary.

Ex :- dog1.clear()
print(dog1) Ex :- {}.

difference b/w Del and clear

del	clear
del is a keyword	i) clear() is the own funch of dictionary.
ii) del can also dele te the whole dict- ionary	ii) It will only clear the values stored in dictionary.

Ex :-
del dog1
print(dog1)
↳ O/p
error because dog1 dictionary is not present

Ex . dog1.clear()
print(dog1)
O/P
{}
↳ Only values were deleted

Page :
Date : / /

Program flow control :—

Condition Statement :—

i) if
ii) if - else
iii) if - elif
iv) Nested if

i) **if :—** It check the condition if it is true than it will execute the statement.

→ The statement will be given with an indentation (space)

Ex :—

```
print ("enter age")
a = int( input() )

if a > 18:
    print ("Can vote")
```

O/P

```
enter age : 19
19
Can vote.
```

ii) **if - else :—** If the condition is true than statement inside the if will be executed. Otherwise statement inside else will be executed.

→ if and else should have ":"

Ex :-

```
print ("enter age")
a = int (input ())

if a>=18:
    print ("can vote")
else:
    print ("can not vote")
```

iii) elif :- This statement is used when you have more than one condition has to be checked.

Ex :-

```
print ("enter a number:")
a = int (input ())
if a>0:
    print ("positive")
elif a<0:
    print ("Negative")
else:
    print ("The number is zero").
```

→ And , OR :-

'And' or 'OR' statement is used when we have to check two or more statement together.

→ And → if all the condition are satisfied it will return 'True' else return 'False'.

→ OR → It will return True if any condition is satisfied

Page :
Date : / /

Nested if :- If inside another if is known as nested if.

```
x = 41
if x > 10:
    print ("Above ten")
    if x > 20:
        print ("and also above 20!")
else:
    print ("but not above 20.")
```