In [ ]:

In [ ]:

Welocme Python

*#Encapsulation :*

**-->**

Restrict

access

to

method

**and**

variable

**-->**

Encapsulation

prevent

data

**from**

direct

modification

which

**is**

called

encapsulation

using

underscore

are

prifix

**-->**

i

.

e

.

single

"\_"

**or**

doubble

"\_\_"

*# Staticmethod :*

1

)

staticmethod

()

2

)

**@**

staticmethod

Static

:

Which

allocate

memory

once

**in**

a

programm

**class**

Sample

:

**def**

display

(

self

):

print

(

"Welocme Python"

)

obj

**=**

Sample

()

obj

.

display

()

1

2

3

4

5

6

7

8

1

2

3

4

5

6

1

2

3

4

5

6

In [1]:

In [3]:

Welcome Python

*# using staticmethod() :*

**class**

Sample

:

**def**

display

():

print

(

"Welcome Python"

)

Sample

.

display

**=**

staticmethod

(

Sample

.

display

)

Sample

.

display

()

1

2

3

4

5

6

7

8

In [5]:

Welcome Python

*# using @staticmethod :*

**class**

Sample

:

@staticmethod

**def**

display

():

print

(

"Welcome Python"

)

Sample

.

display

()

1

2

3

4

5

6

7

8

In [3]:

hello

*# \_\_name\_\_ == "\_\_main\_\_" = it is first call amin function*

**def**

greetings

():

print

(

"hello"

)

**if**

\_\_name\_\_

**==**

"\_\_main\_\_"

:

greetings

()

print

(

"welcome"

)

1

2

3

4

5

6

7

8

9

welcome In [ ]:

Welcome

*# Inheritance*

**-->**

**if**

we

add

a

method

**in**

child

**class**

**with**

the

same

name

**as**

**in**

parant

**class**

,

the

method

will

be

overridden

*#Abstraction :*

**-->**

it

**is**

a

concept

which

allopcate

only

few

information

**-**

**not**

allocated

background

**-->**

first

access

abc

(

ABC

)

:

Abstract

Base

Class

*# using inheritance :*

**class**

parant

:

**def**

display

(

self

):

print

(

"Welcome"

)

**class**

childA

(

parant

):

**pass**

**class**

childB

(

parant

):

**pass**

obj1

**=**

childA

()

obj1

.

display

()

obj2

**=**

childB

()

obj2

.

display

()

1

2

3

4

1

2

3

4

5

6

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

In [ ]:

In [5]: Welcome

In [6]:

I'm a child A

*# using Abstarction with Inheritance*

**from**

abc

**import**

ABC

**class**

parant

(

ABC

):

**def**

display

(

self

):

**pass**

**class**

childA

(

parant

):

**def**

display

(

self

):

print

(

"I'm a child A"

)

**class**

childB

(

parant

):

**def**

display

(

self

):

print

(

"I'm child B"

)

obj1

**=**

childA

()

obj1

.

display

()

obj2

**=**

childB

()

obj2

.

display

()

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

I'm child B In [ ]:

1