Constructor in Python

Python के अलावा C++ और Java में भी Constructor होता है लेकिन उन constructor में खुद class के नाम से ही constructor को बनाया जाता है |

लेकिन Python में Constructor को create करने के लिए \_\_init\_\_() function का इस्तेमाल किया जाता है |

जब class में \_\_init\_\_() function define किया जाता है और उस class का object बनाया जाता है तब ये function automatically call हो जाता है | उसे अलग से call करने की जरुरत नहीं पड़ती है |

Syntax for Constructor in Python

class className:

class\_body(Optional)

def \_\_init\_\_(parameter(s)): #Constructor

constructor\_body

class\_body(Optional)

Example for Constructor in Python

Example में MyClass में Constructor को define किया गया है और उसके बाद class के 'obj' call किया गया है | object create होते ही constructor call होता है |

Constructor का इस्तेमाल सामान्यतः variables को initialize करने के लिए किया जाता है |

Source Code :

class MyClass:

def \_\_init\_\_(self, a, b):

self.a = a

self.b = b

print(self.a,self.b)

print("Constructor invoked")

obj = MyClass(4, "Hello")

Output :

4 Hello

Constructor invoked

Destructor in Python

C++ और Java में destructor को ~(tilde) sign के साथ class के नाम की जरुरत पड़ती और वो object बनते ही अपने आप call होता है |  
लेकिन Python में Destructor के लिए '\_\_\_del\_\_()' function का इस्तेमाल किया जाता है और जब class का object बनाया जाता है तब वो automatically call नहीं होता है |  
Destructor को destroy करने के लिए 'del' operator से object को delete करना पड़ता है |

Example for Destructor in Python

Source Code :

class MyClass:

def \_\_init\_\_(self, a, b):

self.a = a

self.b = b

print(self.a,self.b)

print("Constructor invoked")

def \_\_del\_\_(self):

print("Destructor invoked")

if \_\_name\_\_ == "\_\_main\_\_":

obj = MyClass(4, "Hello")

del obj

Output :

4 Hello

Constructor invoked

Destructor invoked