Garbage Collector

In [ ]:
```
Garbage Collector -->  In old programming langauge like c/c++ we need to deleted the unused variables
                       and data. programmer is reponsible to deleted or remove the irrelavent data.
                  -->  In Python we have on assitant whose name is garbage collection and he will take care of useless thing
                       Garbage collection automatically delete the usless datat from the memeory
                  -->  If the object doesnot have any reference variable then garbage collector will automcatically destroyed
                       that object.
                  -->  If we are writing any program internally garbage collector is running and he will destroy all the
                       useless data  and reference variable
```

How to check weather garbage collector is enabled for every program or not?

In [6]:
```
import gc
print(gc.isenabled())
gc.disable()
print(gc.isenabled())
```

```
True
False
```

Destructor

In [ ]:
```
Destructor --> it is also a special method  and its name should be __del__.
           --> Just before destrying the object garbage collector always calls destructors for performing the
               cleanup activities(closing database, relese memory)
           --> Once Destructors completed clearnup activites then garbage collector destroy the object.
```

Example

In [10]:
```
class Test:
    def __init__(self):
        print("Constructor called")
    def __del__(self):
        print("Destructor called")

x=Test()
x=None
```

```
Constructor called
Destructor called
```

Counting of Objects and References

In [ ]:
```
class Test:
    pass

t1=Test()
t2=t1
t3=t1
t4=t1


Object -->  1
Reference -->  4
```

Banking Application With Menu Driven Program

In [1]:
```
import sys
class Customer:
    bank_name="Indian Bank"
    def __init__(self,name,account_number,balance=0):
        self.name=name
        self.balance=balance  #100
        self.account_number=account_number

    def deposit(self,amount):
        self.balance=self.balance+amount

        print("Balance After Deposit ",self.balance)

    def withdraw(self,amount):  #balance=100 , amount=1000
        if amount>self.balance:
            print("Insufficient balance we cannot process your request")
            sys.exit()
        self.balance=self.balance-amount    #100   #1000 --> 1000-100 -->900
        print("Balance After Withdrawal :",self.balance)

    def check_balance(self):
        print("Balance Available :",self.balance)

print("Welcome to",Customer.bank_name)
name=input("Please enter your name")
account_number=int(input("Enter Your Account Number"))
print("--------------------------------------------------------------------------------")
print("Customer Name is :", name)
print("Customer Account Number is ",account_number)
print("--------------------------------------------------------------------------------")
c=Customer(name,account_number)


while True:
    print("*********************************************************************************************")
    print("Press D -- FOR DEPOSIT MONEY ")
    print("Press W -- FOR WITHDRAWAL MONEY")
    print("Press B -- FOR BALANCE CHECKING")
    print("Press E -- FOR EXIT")
    print("**********************************************************************************************")
    option = input("Choose your Option for Transaction")
    if option == "D" or option =="d":
        amount=int(input("Enter the Amount You want to Deposit"))
        c.deposit(amount)
    elif option == "W" or option =="w":
        amount=int(input("Enter the Amount You want to Withdrawal"))
        c.withdraw(amount)
    elif option == "B" or option == "b":
        c.check_balance()
    elif option == "E" or option == "e":
        print("Thanks for Banking!!!")
        sys.exit()
    else:
        print("Please enter valid input")
```

```
Welcome to Indian Bank
Please enter your namePratyush
Enter Your Account Number120
--------------------------------------------------------------------------------
Customer Name is : Pratyush
Customer Account Number is  120
--------------------------------------------------------------------------------
*********************************************************************************************
Press D -- FOR DEPOSIT MONEY
Press W -- FOR WITHDRAWAL MONEY
Press B -- FOR BALANCE CHECKING
Press E -- FOR EXIT
**********************************************************************************************
Choose your Option for TransactionD
Enter the Amount You want to Deposit1500
Balance After Deposit  1500
*********************************************************************************************
Press D -- FOR DEPOSIT MONEY
Press W -- FOR WITHDRAWAL MONEY
Press B -- FOR BALANCE CHECKING
Press E -- FOR EXIT
**********************************************************************************************
Choose your Option for TransactionE
Thanks for Banking!!!
```

```
An exception has occurred, use %tb to see the full traceback.

SystemExit

C:\Users\praty\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3377: UserWarning: To exit: use 'exit', 'quit', or Ctrl-D.
  warn("To exit: use 'exit', 'quit', or Ctrl-D.", stacklevel=1)
```

Banking Application Using Inheritance and Super() Method

In [26]:
```
class Account:
    def __init__(self,name,balance,min_balance):
        self.name=name
        self.balance=balance
        self.min_balance=min_balance

    def deposit(self,amount):
        self.balance=self.balance+amount

        print("Balance After Deposit ",self.balance)

    def withdraw(self,amount):  #balance=100 , amount=1000
        if amount>self.balance:
            print("Insufficient balance we cannot process your request")
            sys.exit()
        self.balance=self.balance-amount    #100   #1000 --> 1000-100 -->900
        print("Balance After Withdrawal :",self.balance)

    def printstatement(self):
        print("Updated Balance is ",self.balance)


class Saving(Account):
    def __init__(self,name,balance):
        super().__init__(name,balance,min_balance=0)
    def user_details(self):
        print("User Name is ,",self.name)
        print("User Balance is ,",self.balance)




class Current(Account):
    def __init__(self,name,balance):
        super().__init__(name,balance,min_balance=1000 )
    def user_details(self):
        print("User Name is ,",self.name)
        print("User Balance is ,",self.balance)
c=Saving("Pratyush",1000)
c.deposit(100)
c.withdraw(500)
```

```
Balance After Deposit  1100
Balance After Withdrawal : 600
```