list functions

```
li = [10, 20, 30, 70, 40]
co = len(li)
print(co)
mx = max(li)
print(mx)
print(" Minimum no: ", min(li))
print(" Addition : ", sum(li))
print("Average : ", avg(li))
li.append(24)
print(li)
m = [1, 2, 3, 7]                [10,20,30,70,40,24,[1,2,3,7]]
li.append(m)                    [10,20,30,70,40,24,1,2,3,7].
print(li)
li.extend(m)
print(li)
li.sort()
print(li)
li.sort(reverse=True)
print(li)
li.reverse()
print(li)
n = li.pop()
print(n)  / print(li.pop())
co= li.count(24)
if(co >0)
    li.remove(24)
po= li.index(30)
print(po)
```

$$\begin{tikzcd} \text{dict} \\ \boxed{\text{key: value}} \arrow[u] \\ \{\ \} \end{tikzcd}$$

keys()

Dictionary

```
dic = {"FYABCA" : 180,
       "SYBCA" : 126,
       "TYBCA': 18,
       "FYMCA" : 120,
       "SYMCA":180
      }
print(dic)
ki= dic.keys()
print(ki)
val= dic.values()
print(val)
dic1= { "BCA": 3years",
        "MCA": 2years"
      }
dic1[" B.Tech"]="4 years"
print(dic1)
v= dic1.get("MCA")
print(v)
v=dic1.get("MBA")
print(v)
v=dic1.popitem()
print(v)
v=dic1.pop("BCA")
print (v)
```

Imp Program li = [1, 2, 2, 4, 5, 6, 7, 5, 8, 10]
nl = [1, 2,

```
li = []
print("Enter 10 elements for the list")
for i in range(10):
        n = int(input())
        li.append(n)
nl = []
for n in li :
      if (nl.count(n) == 0)
              nl.append(n)
print("Original list : ", li)
li = nl
print("After removing duplicate elements : ", li)
```

obj.x

```
class Test :
        x = 10
print ("Main Program")
obj = Test()
print ("Value is ", obj.x)
```

variable

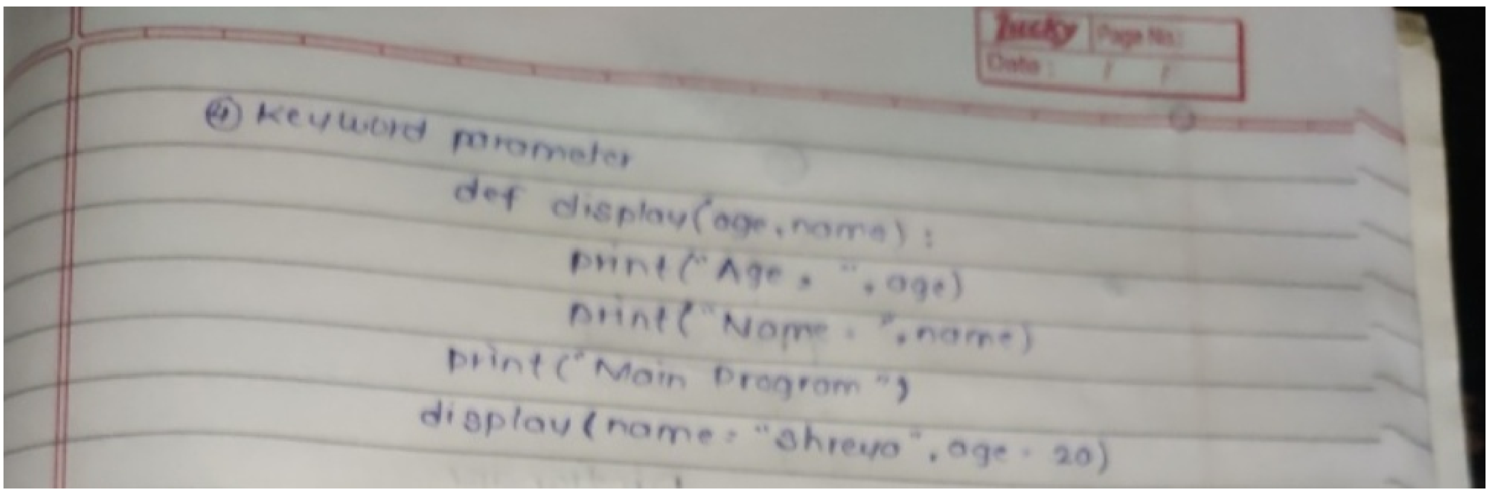
Types of parameters

• Positional Parameter :

```
def calculate(n1,n2,n3):
        sum= n1 +n2+n3
        avg= sum/3
        print("Average= ",avg)
print("Main Program")
calculate(5,7,10)
```

• variable length Parameter   key   Non key
        Non key

```
                                            tuple
def calculate(*arg):                          0 1 2 3 4
        sum=0
        count=0
        for i in arg:
                sum+=i
                count++
        avg = sum/count
        print("Average= ",avg)
def(display( **args):    Dictionary
        for key,value in args:
                print(key,value)
print( "Main Program")
calculate(10,22,23,24,26,29)
display(city="Kolhapur", city="Sangli")
```



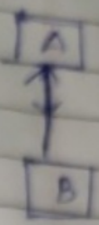
Lucky Page No.:
Date: / /

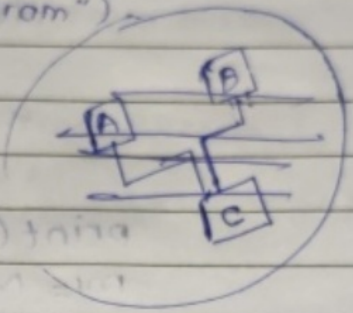
(4) Keyword parameter

```
def display(age,name):
        print("Age = ",age)
        print("Name = ",name)
print("Main Program")
display(name="Shreya",age=20)
```

Single

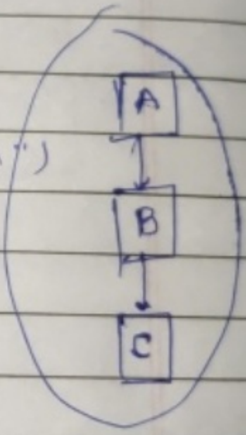
```
class A :
    def show(self):
        print("Show from A")

class B(A):
    def test(self):
        print("Test from B")

print("Main Program")
f1 = A()
F2 = B()
f1.show()
f2.show()
f2.test()
```





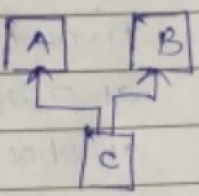
2. Multilevel

```
class A:
        def display(self):
            print("Display from A")
class B(A):
        def test(self):
            print("test from B")
class C(B):
        def get(self):
            print("Get from C")
print("Main Program")
a = A()
b = B()
c = C()
a.display()
b.test()
b.display()
c.get()      c.test()
```

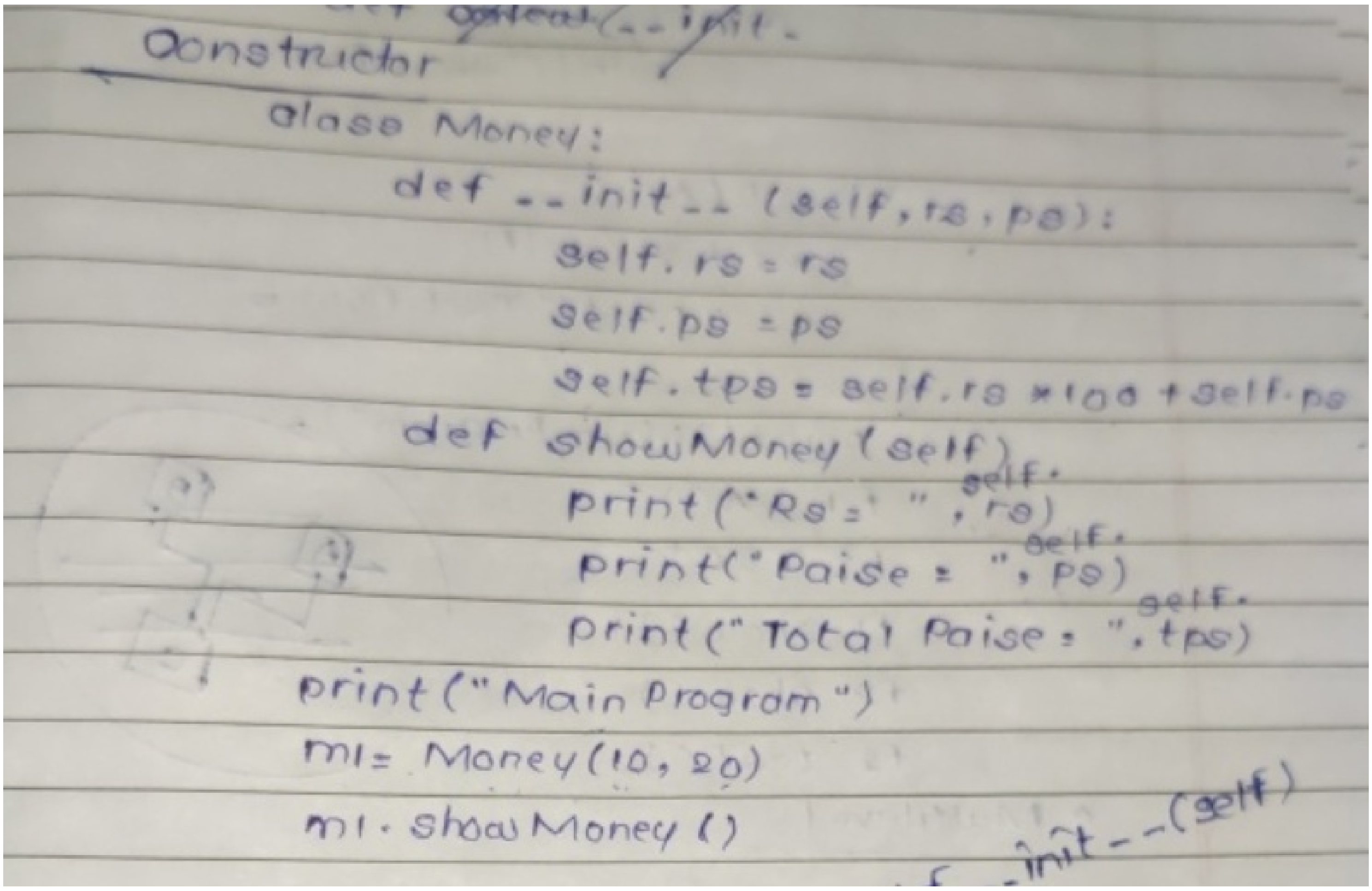


Multiple Inheritance

```
class A :
    def display(self):
        print("Display from A")

class B :
    def test(self):
        print("Test from B")
class C (A, B):
    def out(self):
        print("out from C")
print("Main Program")
a = A()
b = B()
c = C()
a.display()
b.test()
c.out()
c.display()
c.test()
```



Multiple



Constructor

```
class Money:
    def __init__(self,rs,ps):
        self.rs = rs
        self.ps = ps
        self.tps = self.rs *100 +self.ps
    def showMoney(self)
        print("Rs= ", self.rs)
        print("Paise = ", self.ps)
        print(" Total Paise= ", self.tps)
print("Main Program")
m1= Money(10, 20)
m1.showMoney()
```

__init__(self)

Page No.:
Date: / /

• function overloading

```
class Money:
        def assignValues(self, *args)
            if(len(args)==0):
                self.rs = int(input("Enter value for
                                        Rs"))
                 self.ps =int(input("Enter value
                                    for Paise"))
                self.tps = self.rs * 100 + self.ps
            elif(len(args)==1):
                self.tps= args[0]
                self.rs = self.tps/100
                Self.ps = selftps%100
            else:
                self.rs = args[0]
                self.ps=args[1]
                self.tps =self.rs *100 + self.ps
        def showMoney(self)
                print(" Rs =   ", self.rs)
                print(" Paise =  ", self.ps)
                print(" Total Paise = ", self.tps)
    • print ("Main")
    m1= Money()
    m2= Money()
    m3= Money()
    m1. assignValues( 1630)
    m1. showMoney()
    m2. assign Values(10,20)
    m2. showMoney()
    m3. assignValues( )
    m3. showMoney( )
```

• Default Argument

```
def calculate (n1, n2 = 15, n3 = 18) :
        sum = n1 + n2 + n3
        avg = sum / 3
        print (f" sum = {sum}")
        print (f" Average = {avg}")
print ("Main Program")
calculate (6, 18, 21)
calculate (18, 21)
calculate (21)
```

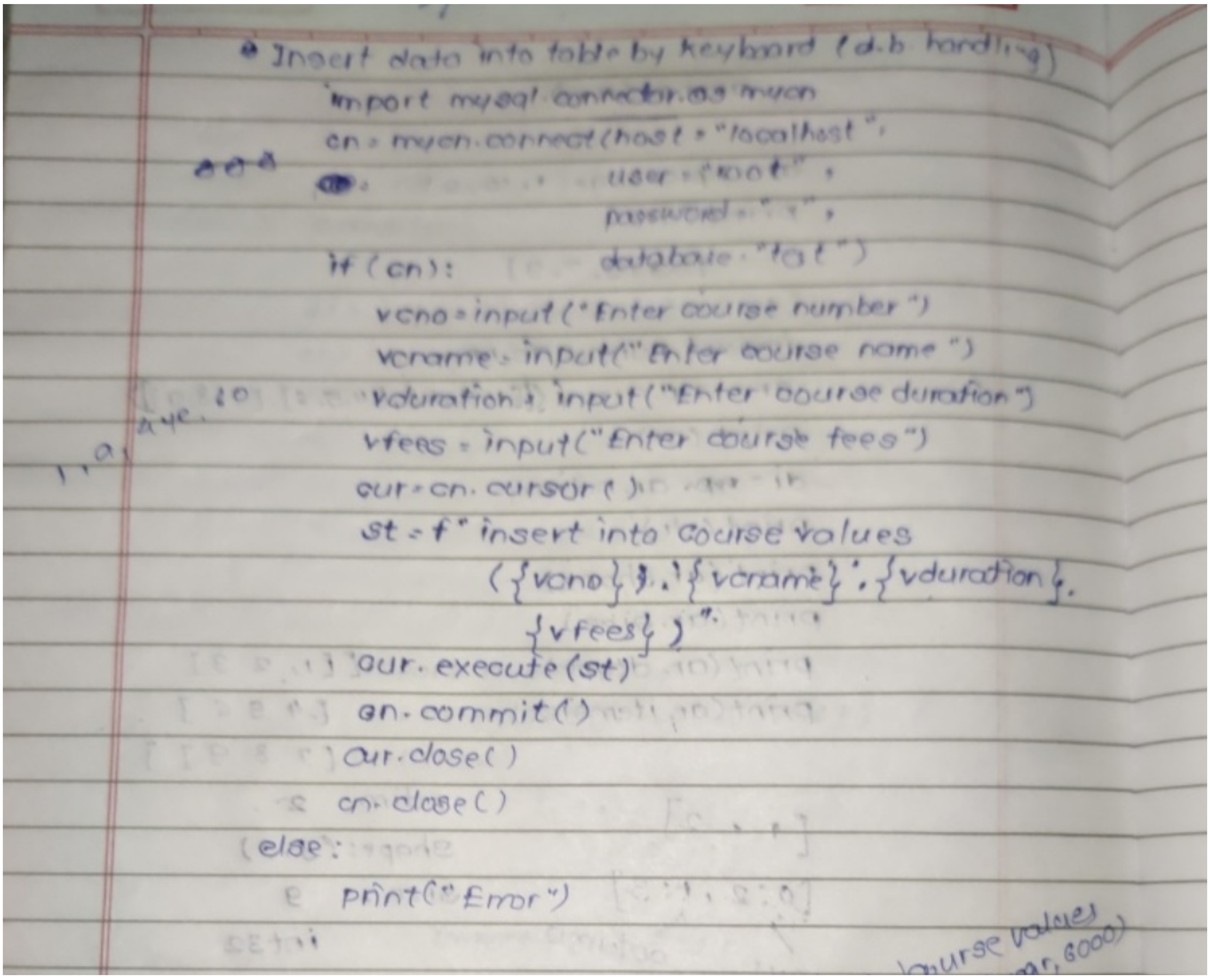
• Types of parameters

• Positional Parameter:

```
    def calculate (n1, n2, n3):
            sum = n1 + n2 + n3
            avg = sum / 3
            print (" Average = ", avg)
    print (" Main Program")
    calculate (5, 7, 10)
```

• variable length Parameter   Key   Non
          Non - key

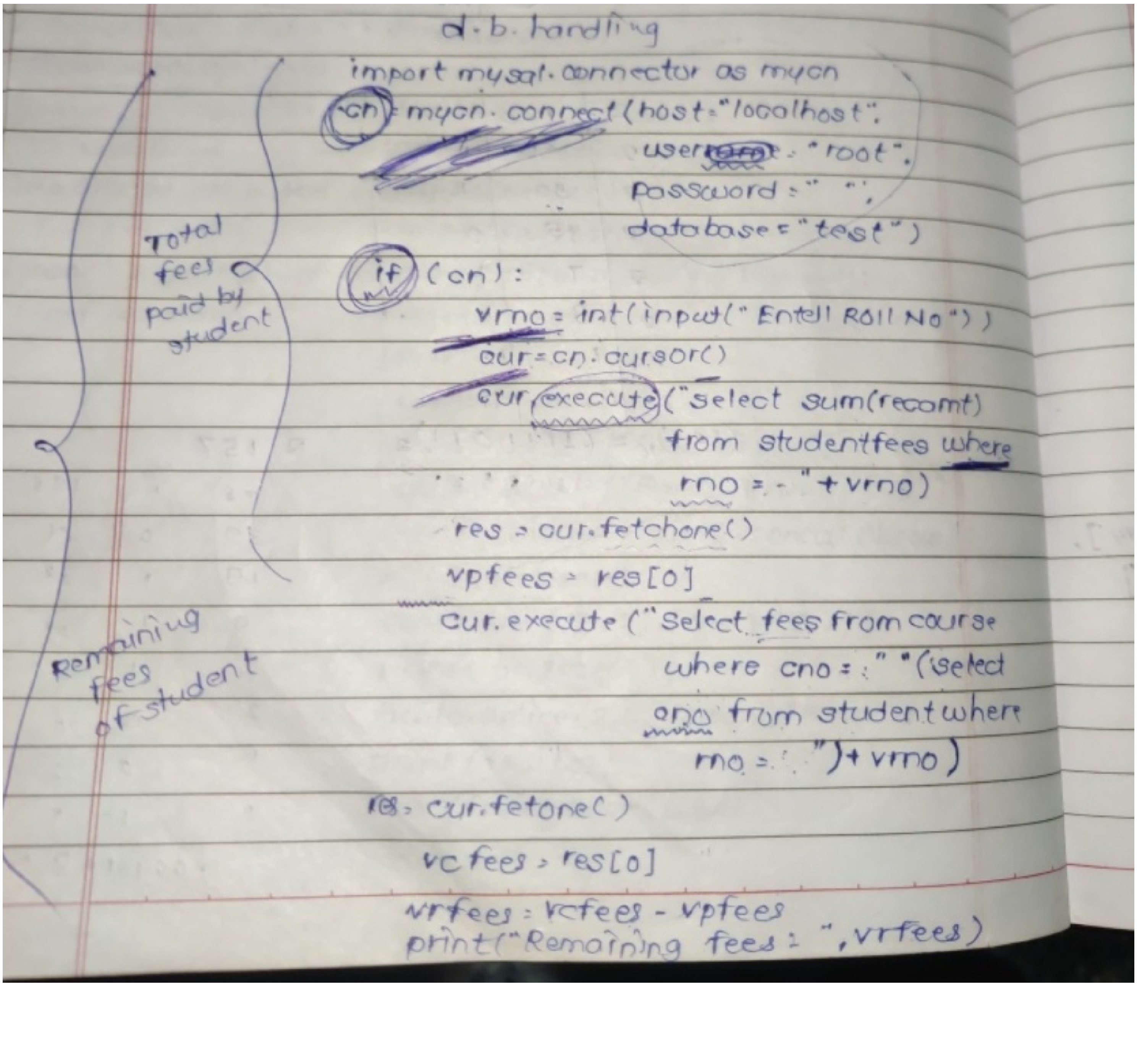
```
                                          tuple
        def calculate (*arg):
                sum = 0
                count = 0
                for i in arg:
                        sum += i
                        count++
                avg = sum / count
                print (" Average = ", avg)
        def (display ( **args):     _ Dictiona
                for key, value in args :
                        print (key, value)
    print ( " Main Program")
    calculate (10, 22, 23, 24, 26, 29)
    display (city = "Kolhapur", city = " Sangl
```



• Insert data into table by keyboard (d.b handling)

```
import mysql.connector as mycon
cn = mycon.connect(host = "localhost",
                   user = "root",
                   password = "1",
                   database = "tat")
if (cn):
    vcno = input("Enter course number")
    vcname = input("Enter course name")
    vduration = input("Enter course duration")
    vfees = input("Enter course fees")
    cur = cn.cursor()
    st = f"insert into course values
            ({vcno},'{vcname}',{vduration},
             {vfees})"
    cur.execute(st)
    cn.commit()
    cur.close()
    cn.close()
else:
    print("Error")
```

course values
ar, 6000)



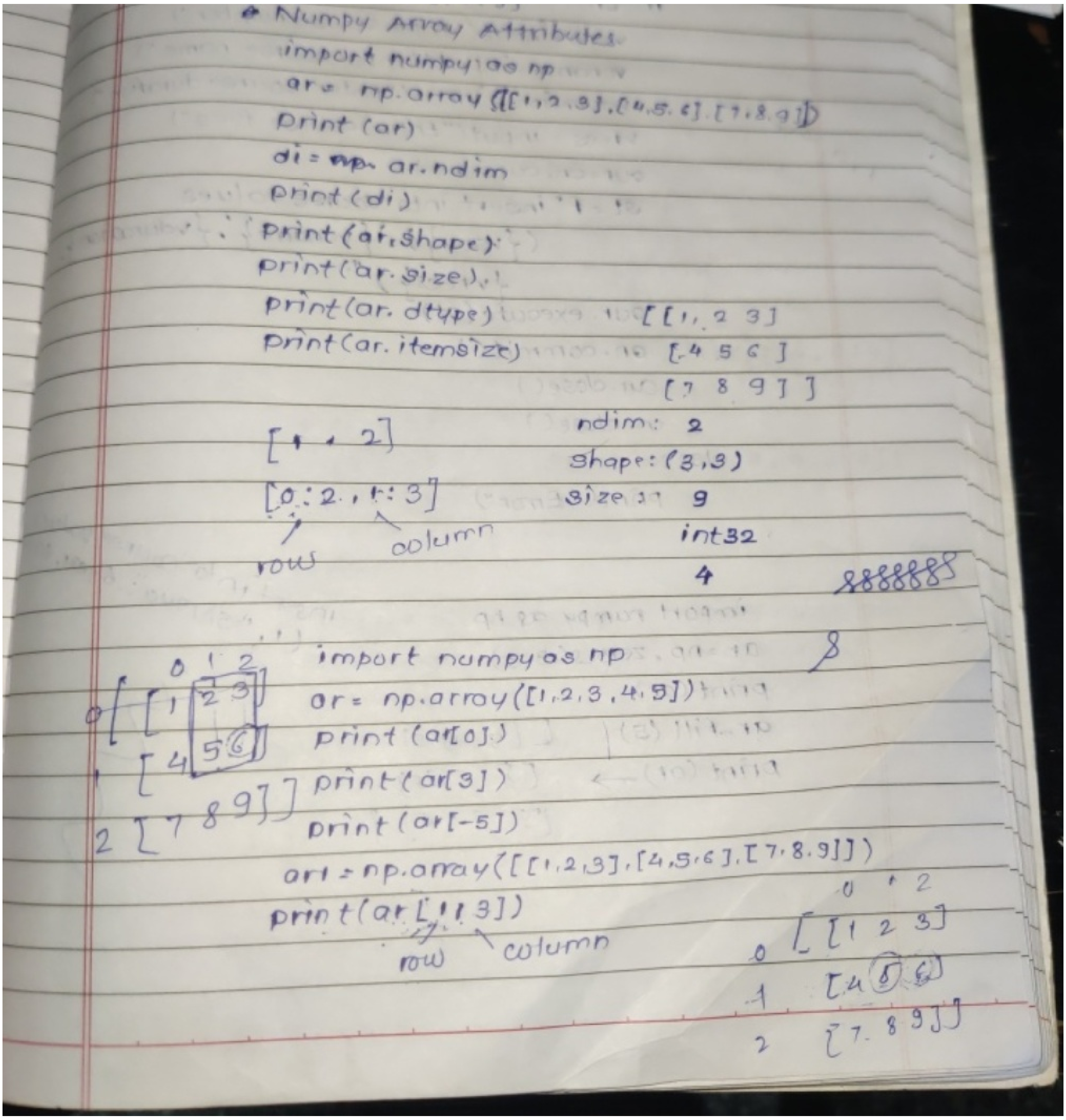
d.b. handling

Total fees paid by student

```
import mysql.connector as mycn
cn = mycn.connect(host="localhost",
                  user = "root",
                  password = " ",
                  database = "test")
if (cn):
    vrno = int(input("Entell Roll No"))
    cur=cn.cursor()
    cur.execcute("select sum(recamt)
           from studentfees where
           rno = " + vrno)
    res = cur.fetchone()
    vpfees = res[0]
```

Remaining fees of student

```
    cur.execute ("Select fees from course
           where cno = " "(select
           cno from student where
           rno = ")+ vrno)
res = cur.fetone()
    vcfees = res[0]
vrfees = vcfees - vpfees
print("Remaining fees = ",vrfees)
```



Numpy Array Attributes

```
import numpy as np
ar = np.array([[1,2,3],[4,5,6],[7,8,9]])
print (ar)
di = ~~np.~~ ar.ndim
print(di)
print(ar.shape)
print(ar.size)
print(ar.dtype)
print(ar.itemsize)
```

```
[[1, 2 3]
 [4 5 6]
 [7 8 9]]
ndim: 2
shape: (3,3)
size: 9
int32
4
```

~~888888~~

8

[1 , 2]

[0:2 , 1:3]

row column

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 1 & 2 & 3 \\ 1 & 4 & 5 & 6 \\ 2 & 7 & 8 & 9 \end{array}$$

```
import numpy as np
ar = np.array([1,2,3,4,5])
print (ar[0])
print(ar[3])
print(ar[-5])
ar1 = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(ar[1,1:3])
```

row column

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 1 & 2 & 3 \\ 1 & 4 & 5 & 6 \\ 2 & 7 & 8 & 9 \end{array}$$