In [7]:
```
a="naidu"
a
```

Out[7]: 'naidu'

mylist()

In [2]:
```
5+3
```

Out[2]: 8

In [4]:
```
a=[]
```

In [5]:
```
type(a)
```

Out[5]: list

In [6]:
```
a
```

Out[6]: []

In [7]:
```
a=[1,100,"naidu","pavan",3]
```

In [8]:
```
a
```

Out[8]: [1, 100, 'naidu', 'pavan', 3]

In [10]:
```
a[0]
```

Out[10]: 1

In [9]:
```
a[-2]
```

Out[9]: 'pavan'

In [12]:
```
len(a)
```

Out[12]: 5

In [13]:
```
type(a)
```

Out[13]: list

In [14]:
```
a[-3]
```

Out[14]: 'naidu'

In [15]:
```
a[1:3]
```

Out[15]: [100, 'naidu']

In [16]:
```
a[0:3]
```

Out[16]:
```
[1, 100, 'naidu']
```

In [10]:
```
a[:]
```

Out[10]:
```
[1, 100, 'naidu', 'pavan', 3]
```

In [ ]:
```

```

In [18]:
```
a[:3]
```

Out[18]:
```
[1, 100, 'naidu']
```

In [19]:
```
b=[]
```

In [20]:
```
type(b)
```

Out[20]:
```
list
```

In [21]:
```
a[4]="pavan"
```

In [11]:
```
a
```

Out[11]:
```
[1, 100, 'naidu', 'pavan', 3]
```

In [15]:
```
c=[(1,2,5),20,30,(58,60,1000)]
```

In [16]:
```
c
```

Out[16]:
```
[(1, 2, 5), 20, 30, (58, 60, 1000)]
```

In [25]:
```
len(c)
```

Out[25]:
```
4
```

In [26]:
```
a[0]
```

Out[26]:
```
1
```

In [28]:
```
c[0][2]
```

Out[28]:
```
5
```

In [17]:
```
a+c
```

Out[17]:
```
[1, 100, 'naidu', 'pavan', 3, (1, 2, 5), 20, 30, (58, 60, 1000)]
```

In [18]:
```
a*3
```

Out[18]:
```
[1,
 100,
 'naidu',
 'pavan',
 3,
 1,
 100,
 'naidu',
 'pavan',
 3,
 1,
 100,
 'naidu',
 'pavan',
 3]
```

In [19]:
```
a
```

Out[19]:
```
[1, 100, 'naidu', 'pavan', 3]
```

In [32]:
```
len(a)
```

Out[32]:
```
5
```

In [36]:
```
d=[1,20,30,80,100]
d
```

Out[36]:
```
[1, 20, 30, 80, 100]
```

In [37]:
```
min(d)
```

Out[37]:
```
1
```

In [38]:
```
max(d)
```

Out[38]:
```
100
```

In [ ]:
```

```

In [43]:
```
for j in b:
    print(j)
```

In [40]:
```
for i in d:
    print(i)
```

```
1
20
30
80
100
```

In [47]:
```
d.append(20)
```

In [48]:
```
d
```

Out[48]:
```
[1, 20, 30, 80, 100, 20]
```

In [49]:
```
d.append([20,90])
```

In [50]:
```
d
```

Out[50]:
```
[1, 20, 30, 80, 100, 20, [20, 90]]
```

In [51]:
```
d.extend([10,30,80,90])
```

In [52]:
```
d
```

Out[52]:
```
[1, 20, 30, 80, 100, 20, [20, 90], 10, 30, 80, 90]
```

In [53]:
```
a.insert(2,60)
```

In [54]:
```
a
```

Out[54]:
```
[1, 100, 60, 'naidu', 'pavani', 'pavan']
```

In [55]:
```
d.insert(2,60)
```

In [56]:
```
d
```

Out[56]:
```
[1, 20, 60, 30, 80, 100, 20, [20, 90], 10, 30, 80, 90]
```

In [57]:
```
d
```

Out[57]:
```
[1, 20, 60, 30, 80, 100, 20, [20, 90], 10, 30, 80, 90]
```

In [60]:
```
d.remove(60)
```

In [61]:
```
d
```

Out[61]:
```
[1, 20, 30, 80, 100, 20, [20, 90], 10, 30, 80, 90]
```

In [62]:
```
d.pop()
```

Out[62]:
```
90
```

In [63]:
```
d
```

Out[63]:
```
[1, 20, 30, 80, 100, 20, [20, 90], 10, 30, 80]
```

In [65]:
```
del d[2]
```

In [66]:
```
d
```

Out[66]:
```
[1, 20, 80, 100, 20, [20, 90], 10, 30, 80]
```

In [67]:
```
d
```

Out[67]:
```
[1, 20, 80, 100, 20, [20, 90], 10, 30, 80]
```

In [71]:
```
import random
```

In [72]:
```
random.shuffle(d)
```

In [73]:
```
d
```

Out[73]:
```
[10, 20, 20, 30, 100, [20, 90], 80, 1, 80]
```

In [74]:
```
d
```

Out[74]:
```
[10, 20, 20, 30, 100, [20, 90], 80, 1, 80]
```

In [78]:
```
d.sort()
```

In [76]:
```
del d[5]
```

In [77]:
```
d
```

Out[77]:
```
[10, 20, 20, 30, 100, 80, 1, 80]
```

In [79]:
```
d
```

Out[79]:
```
[1, 10, 20, 20, 30, 80, 80, 100]
```

In [80]:
```
d.sort(reverse=True)
```

In [81]:
```
d
```

Out[81]:
```
[100, 80, 80, 30, 20, 20, 10, 1]
```

In [ ]:
```

```

In [82]:
```
d.reverse()
```

In [83]:
```
d
```

Out[83]:
```
[1, 10, 20, 20, 30, 80, 80, 100]
```

In [169…
```
a=d*10
```

In [88]:
```
a
```

Out[88]:
```
[1,
 10,
 20,
 20,
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
```

```
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
```

```
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
```

```
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
```

```
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
```

```
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
```

```
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
```

```
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
```

```
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
```

```
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
```

```
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
```

```
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
```

```
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
30,
80,
80,
100,
1,
10,
20,
20,
```

```
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
 30,
 80,
 80,
 100,
 1,
 10,
 20,
 20,
 30,
 80,
 80,
 100]
```

In [90]:
```
a.count(20)
```

Out[90]: 200

In [91]:
```
a.index(20)
```

Out[91]: 2

In [92]:
```
a=[ele for ele in range(10) if ele%2==0]
```

In [93]:
```
a
```

Out[93]: [0, 2, 4, 6, 8]

In [94]:
```
t=()
```

In [95]:
```
type(t)
```

Out[95]: tuple

In [96]:
```
t
```

Out[96]: ()

In [97]:
```
t=(10,20,30,40,50,60,70)
```

In [98]:
```
t
```

Out[98]: (10, 20, 30, 40, 50, 60, 70)

In [99]:
```
min(t)
```

Out[99]: 10

In [100…
```
max(t)
```

Out[100]: 70

In [101…
```
len(t)
```

Out[101]: 7

In [102…
```
t[:]
```

Out[102]: (10, 20, 30, 40, 50, 60, 70)

In [103…
```
t[:3]
```

Out[103]: (10, 20, 30)

In [104…
```
t
```

Out[104]: (10, 20, 30, 40, 50, 60, 70)

In [ ]:
```
list
```

In [105…
```
l=list(t)
```

In [106…
```
l
```

Out[106]: [10, 20, 30, 40, 50, 60, 70]

In [107…
```
l.append(30)
```

In [108…
```
l
```

Out[108]: [10, 20, 30, 40, 50, 60, 70, 30]

In [109…
```
t=tuple(l)
```

In [110…
```
t
```

Out[110]: (10, 20, 30, 40, 50, 60, 70, 30)

In [111…
```
sum(t)
```

Out[111]: 310

In [112…
```
t.count(30)
```

Out[112]: 2

In [113…
```
t.count(30)
```

Out[113]: 2

In [114…
```
t.index(30)
```

Out[114]: 2

In [115…
```
del t
```

In [117…
```
str="naidu"
```

In [118…
```
str
```

Out[118]:
```
'naidu'
```

In [119…
```
str[-1]
```

Out[119]:
```
'u'
```

In [120…
```
str[-2]
```

Out[120]:
```
'd'
```

In [121…
```
str[-3]
```

Out[121]:
```
'i'
```

In [122…
```
str[1:4]
```

Out[122]:
```
'aid'
```

In [123…
```
str
```

Out[123]:
```
'naidu'
```

In [124…
```
str="""naidu is one of the greatest person"""
```

In [125…
```
str
```

Out[125]:
```
'naidu is one of the greatest person'
```

In [126…
```
str[1:]
```

Out[126]:
```
'aidu is one of the greatest person'
```

In [128…
```
len(str)
```

Out[128]:
```
35
```

In [129…
```
s1="naidu"
```

In [130…
```
s2="python"
```

In [131…
```
s1+s2
```

Out[131]:
```
'naidupython'
```

In [132…
```
s="welcome to python"
```

In [133…
```
s.capitalize()
```

Out[133]: 'Welcome to python'

In [136…
```
s.center(21,"*")
```

Out[136]: '**welcome to python**'

In [137…
```
s.count("o")
```

Out[137]: 3

In [139…
```
s.endswith("on")
```

Out[139]: True

In [140…
```
s.startswith("we")
```

Out[140]: True

In [141…
```
s.find("o")
```

Out[141]: 4

In [142…
```
s.rfind("o")
```

Out[142]: 15

In [143…
```
s.index("o")
```

Out[143]: 4

In [144…
```
s.rindex("o")
```

Out[144]: 15

In [145…
```
s.isalnum()
```

Out[145]: False

In [146…
```
s.isalpha()
```

Out[146]: False

In [147…
```
s.isdigit()
```

Out[147]: False

In [148…
```
s.isspace()
```

Out[148]: False

In [149…
```
s.islower()
```

Out[149]: True

In [150…
```
s.isupper()
```

Out[150]: False

In [154…
```
s.istitle()
```

Out[154]: False

In [156…
```
s.ljust(30,"*")
```

Out[156]: 'welcome to python**********'

In [157…
```
s.rjust(30,"*")
```

Out[157]: '**********welcome to python'

In [158…
```
s.upper()
```

Out[158]: 'WELCOME TO PYTHON'

In [159…
```
s.lower()
```

Out[159]: 'welcome to python'

In [160…
```
s.strip()
```

Out[160]: 'welcome to python'

In [161…
```
s.lstrip()
```

Out[161]: 'welcome to python'

In [152…
```
s.title()
```

Out[152]: 'Welcome To Python'

In [162…
```
s.rstrip()
```

Out[162]: 'welcome to python'

In [163…
```
s.replace("python","java",1)
```

Out[163]: 'welcome to java'

In [164…
```
s.split(" ",2)
```

Out[164]: ['welcome', 'to', 'python']

In [165…
```
naidu="pavanADC"
```

In [4]:
```
naidu
```

Out[4]: <__main__.pavan at 0x1c1abc2c880>

In [3]:
```
naidu.swapcase()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[3], line 1
----> 1 naidu.swapcase()

AttributeError: 'pavan' object has no attribute 'swapcase'
```

In [168…
```
s.zfill(100)
```

Out[168]:
```
'0000000000000000000000000000000000000000000000000000000000000000000000000000000000welcome to python'
```

In [170…
```
set()
```

Out[170]:
```
set()
```

In [171…
```
s=set()
```

In [172…
```
s
```

Out[172]:
```
set()
```

In [173…
```
type(s)
```

Out[173]:
```
set
```

In [175…
```
s={10,20,30,40,50,10}
```

In [176…
```
len(s)
```

Out[176]:
```
5
```

In [177…
```
s.add(50)
```

In [178…
```
s
```

Out[178]:
```
{10, 20, 30, 40, 50}
```

In [179…
```
s1=set()
```

In [180…
```
s1={10,20}
```

In [181…
```
s1
```

Out[181]:
```
{10, 20}
```

In [182…
```
s.issubset(s1)
```

Out[182]:
```
False
```

In [183…
```
s1.issubset(s)
```

Out[183]:
```
True
```

In [184…
```
s1.issuperset(s)
```

Out[184]: False

In [185…
```
s.issuperset(s1)
```

Out[185]: True

In [186…
```
s.union(s1)
```

Out[186]: {10, 20, 30, 40, 50}

In [187…
```
s.intersection(s1)
```

Out[187]: {10, 20}

In [189…
```
s.difference(s1)
```

Out[189]: {30, 40, 50}

In [190…
```
s.add(100)
```

In [191…
```
s
```

Out[191]: {10, 20, 30, 40, 50, 100}

In [193…
```
s.symmetric_difference(s1)
```

Out[193]: {30, 40, 50, 100}

In [194…
```
s.update(s1)
```

In [195…
```
s
```

Out[195]: {10, 20, 30, 40, 50, 100}

In [197…
```
s.intersection_update(s1)
```

In [198…
```
s
```

Out[198]: {10, 20}

In [200…
```
s
```

Out[200]: {10, 20}

In [201…
```
d={}
```

In [202…
```
type(d)
```

Out[202]: dict

In [204…
```
d=dict()
```

In [205…
```
d
```

Out[205]: {}

In [207…
```
d={"name":"naidu","per":98}
```

In [208…
```
d
```

Out[208]: {'name': 'naidu', 'per': 98}

In [210…
```
d["college"]="lendi"
```

In [211…
```
d
```

Out[211]: {'name': 'naidu', 'per': 98, 'college': 'lendi'}

In [218…
```
def add(x, y):
    z = x + y
    return z

result = add(2, 3)
print(result)
```

```
5
```

In [219…
```
w=100
def add(x, y):
    w=123
    z = x + y+ w
    return z

result = add(2, 3)
print(result)
print(w)
```

```
128
100
```

In [221…
```
def factorial(n):
    if n==0:
        return 1
    else:
        return n*factorial(n-1)
result=factorial(5)
print(result)
```

```
120
```

In [223…
```
add=lambda a,b:a+b
result=add(2,3)
print(result)
```

```
5
```

In [224…
```
import math
```

In [225…
```
math.factorial(5)
```

Out[225]: 120

In [226…
```
pip install pytest
```

```
Requirement already satisfied: pytest in c:\pavan9\lib\site-packages (7.3.1)Note:
you may need to restart the kernel to use updated packages.

Requirement already satisfied: iniconfig in c:\pavan9\lib\site-packages (from pyt
est) (2.0.0)
Requirement already satisfied: packaging in c:\pavan9\lib\site-packages (from pyt
est) (23.1)
Requirement already satisfied: pluggy<2.0,>=0.12 in c:\pavan9\lib\site-packages
(from pytest) (1.0.0)
Requirement already satisfied: exceptiongroup>=1.0.0rc8 in c:\pavan9\lib\site-pac
kages (from pytest) (1.1.1)
Requirement already satisfied: tomli>=1.0.0 in c:\pavan9\lib\site-packages (from
pytest) (2.0.1)
Requirement already satisfied: colorama in c:\pavan9\lib\site-packages (from pyte
st) (0.4.6)
```

In [229…

```
a = int(input("Enter the number:"))

if a > 0:
    print("This is a positive number")
else:
    print("This is a negative number")
```

```
Enter the number:20
This is a positive number
```

In [230…

```
a = int(input("Enter the number:"))

if a > 0:
    print("This is a positive number")
else:
    print("This is a negative number")
```

```
Enter the number:-1
This is a negative number
```

In [231…

```
a=int(input("enter the a value"))
b=int(input("enter the b value"))
c=int(input("enter the c value"))
largest=max(a,b,c)
print("largest value is" ,largest)
```

```
enter the a value1
enter the b value2
enter the c value3
largest value is 3
```

In [232…

```
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
c = int(input("Enter the third number: "))

if a >= b and a >= c:
    largest = a
elif b >= a and b >= c:
    largest = b
else:
    largest = c

print("The largest number is:", largest)
```

```
Enter the first number: 12
Enter the second number: 13
Enter the third number: 56
The largest number is: 56
```

In [235…
```
n=int(input("enter the n value"))
i=1
while i<=n:
    print(i,end=" ")
    i=i+1
```

```
enter the n value12
1 2 3 4 5 6 7 8 9 10 11 12
```

In [238…
```
for i in range(1,101,20):
    print(i,end=" ")
```

```
1 21 41 61 81
```

In [239…
```
str="python"
for i in str:
    print(i)
```

```
p
y
t
h
o
n
```

In [240…
```
list=[10,12,16,56,98]
for i in list:
    print(i)
```

```
10
12
16
56
98
```

In [242…
```
for i in range(1,11):
    if i<=5:
        print("True")
        break
    else:
        print(i)
```

```
True
```

In [245…
```
for i in range(1,11):
    if i==5:
        continue
    else:
        print(i)
```

```
1
2
3
4
6
7
8
9
10
```

In [246…
```
i=1
while i<=10:
    if i<=4:
        pass
    print(i)
    i=i+1
```

```
1
2
3
4
5
6
7
8
9
10
```

In [247…
```
import math
```

In [248…
```
math.sqrt(36)
```

Out[248]: 6.0

In [251…
```
math.factorial(6)
```

Out[251]: 720

math.factorial(6)

In [252…
```
math.gcd(12,16)
```

Out[252]: 4

In [253…
```
math.pow(12,3)
```

Out[253]: 1728.0

In [255…
```
import random
```

In [256…
```
l=[10,20,36,100]
```

In [257…
```
l
```

Out[257]: [10, 20, 36, 100]

In [259…
```
random.choice(l)
```

Out[259]: 10

In [260…
```
random.choices(l,k=2)
```

Out[260]: [20, 20]

In [261…
```
random.choices(l,k=2)
```

Out[261]: [36, 20]

In [ ]:
```
#python Oops#
```

In [263…
```
class naidu:
    color="black&white"#attributes
    height=5.5
    def run(self):    #method#
        print("running")
    def walk(self):
        print("walking")

pavani=naidu()#object#
print(pavani.color,pavani.height)

pavani.run()
```

```
black&white 5.5
running
```

In [265…
```
class naidu:
    def __init__(self):
        print("constructor")
    def run(self):    #method#
        print("running")
    def walk(self):
        print("walking")
    pavani=naidu()#Object create ,it goes to constructor#
```

```
constructor
```

In [269…
```
class naidu:
    def __init__(self,c,h):
        self.color=c
        self.height=h
    def run(self):    #method#
        print("running")
    def walk(self):
        print("walking")


pavani=naidu("white",7)
print(pavani.color,pavani.height)
saradhi=naidu("white",10)
print(saradhi.color,saradhi.height)
```

```
white 7
white 10
```

In [ ]:
```
class Calculator:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def addition(self):
        return self.x + self.y

    def multiplication(self):
        return self.x * self.y

# Create an instance of the Calculator class
calc = Calculator(5, 3)

# Perform addition and multiplication operations
result_addition = calc.addition()
result_multiplication = calc.multiplication()

# Print the results
print("Addition result:", result_addition)
print("Multiplication result:", result_multiplication)
```

In [273…
```
class calcu:
    def __init__(self,x,y):
        self.x = x
        self.y = y
    def add(self):
        return self.x+self.y
    def mal(self):
        return self.x*self.y

cal=calcu(5,3)

res1=cal.add()
print(res1)
res2=cal.mal()
print(res2)

naidu=calcu(10,3)
res3=naidu.add()
print(res3)
```

```
8
15
13
```

In [ ]:
```
#inheritance#
```

In [276…
```
class Baseclass:
    a=10
    b=100
    def display(self):
        print("Baseclass")#  accesing derived class#
class Derivedclass:
    c=20
    d=200
    def show(self):
        print("Derivedclass")
dobject=Derivedclass()
```

```
print(dobject.c,dobject.d)
dobject.show()
```

```
20 200
Derivedclass
```

In [277…

```
class Baseclass:
    a=10
    b=100
    def display(self):
        print("Baseclass")# single Inhertance   #
class Derivedclass(Baseclass):#important#    #Child#
    c=20
    d=200
    def show(self):
        print("Derivedclass")
dobject=Derivedclass()
print(dobject.c,dobject.d,dobject.a,dobject.b)
dobject.show()
dobject.display()
```

```
20 200 10 100
Derivedclass
Baseclass
```

In [279…

```
class grandparent:
    a=10
    b=100
    def display(self):
        print("grandparent")# multilevel Inhertance   #
class parent(grandparent):#important#
    c=20
    d=200
    def show(self):
        print("parent")
class child(parent):#important#
    e=200
    f=2000
    def cinema(self):
        print("child")
dobject=child()
print(dobject.c,dobject.d,dobject.a,dobject.b,dobject.e,dobject.f)
dobject.show()
dobject.display()
dobject.cinema()
```

```
20 200 10 100 200 2000
parent
grandparent
child
```

In [283…

```
class Parent:  #Hierarchical inheritance#
    a = 20
    b = 30

    def display(self):
        print("parent")

class Son(Parent):
    c = 30
    d = 60
```

```
    def show(self):
        print("son")

class Daughter(Parent):
    e = 69
    f = 23

    def cinema(self):
        print("daughter")

s_object = Son()
print(s_object.a, s_object.b, s_object.c, s_object.d)
s_object.display()
s_object.show()
```

```
20 30 30 60
parent
son
```

In [284…

```
class father:  #multilevel inheritance#
    a = 20
    b = 30
    def fun_1(self):
        print("father class")
class mother:
    c=20
    d=102
    def fun_2(self):
        print("mother class")
class child(father,mother):
    e=30
    f=45
    def fun_3(self):
        print("child class")

co=child()
print(co.a,co.b,co.c,co.d,co.e,co.f)
co.fun_1()
co.fun_2()
co.fun_3()
```

```
20 30 20 102 30 45
father class
mother class
child class
```

In [ ]:

```
Polymorphism

1.compile time Polymorphism---method overloading----direct not support----using
2.Run Time Polymorphism----method overiding
```

In [286…

```
class demo:   #Polymorphism method overloading#
    def add(self,a,b,c=100):
```

```
        print(a+b+c)
obj=demo()
obj.add(100,200)
obj.add(100,200,300)
obj.add(100,400,9000)
```

```
400
600
9500
```

In [ ]:
```
Run Time Polymorphism
```

In [287…
```
class parent:
    def add(self,a,b):
        print(a+b)
class child(parent):#note-same method names and same no.of arguments,Only the fu
    def add(self,a,b):
        print(a*b)
ob=child()
ob.add(20,30)
```

```
600
```

In [ ]:
```
Abstraction

Hiding the Implementation and show esstential information Only
```

In [291…
```
from abc import ABC, abstractmethod

class AbstractDemo(ABC):
    @abstractmethod
    def HousingInterest(self):
        pass

    @abstractmethod
    def VehicleInterest(self):  # Abstract Method
        pass

class SBI(AbstractDemo):
    def HousingInterest(self):
        print("HousingInterest: 8.5%")

    def VehicleInterest(self):
        print("VehicleInterest: 9.2%")

sbi_obj = SBI()
sbi_obj.HousingInterest()
sbi_obj.VehicleInterest()
#we are method overriding concept Here#
```

```
HousingInterest: 8.5%
VehicleInterest: 9.2%
```

In [ ]:
```
Encapsulation:
    Combining the data&methods
Data Security:By using Access Specifiers
```

In [292…
```
class Demo:
    a=10
    b=100
```

```
    def display(self):
        print("Display the method in Demo Class")

obj=Demo()
print(obj.a)
obj.display()
```

```
10
Display the method in Demo Class
```

In [293…

```
class Demo:
    __a=10
    b=100
    def display(self):
        print("Display the method in Demo Class") #From this output we can obser

obj=Demo()
print(obj.a)
obj.display()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[293], line 8
      5         print("Display the method in Demo Class")
      7 obj=Demo()
----> 8 print(obj.a)
      9 obj.display()

AttributeError: 'Demo' object has no attribute 'a'
```

In [295…

```
28%3
```

Out[295]:

```
1
```

In [297…

```
class Demo:
    __a = 30
    b = 25

    def display(self):
        print(self.__a)
        print("demo of Private Variable")


obj = Demo()
print(obj.b)
obj.display()
```

```
25
30
demo of Private Variable
```

In [298…

```
num=int(input("enter the number"))
if num>1:
    for i range(2,num):
        if num%i==0:
            print("These are the not prime numbers",i)
        break
else:
    print("These are the prime numbers",i)
```

```
  Cell In[298], line 3
    for i range(2,num):
          ^
SyntaxError: invalid syntax
```

In [299…
```
num = int(input("Enter the number: "))
if num > 1:
    for i in range(2, num):
        if num % i == 0:
            print("These are not prime numbers:", i)
            break
    else:
        print("This is a prime number:", num)
else:
    print("This is a prime number:", num)
```

```
Enter the number: 29
This is a prime number: 29
```

In [1]:
```
def add (x,y):
    z=x+y
    return z
a=add(1,3)
print(a)
```

```
4
```

In [2]:
```
def display(*marks):
    print(marks)

a=display(90,99,98,87)
print(a)
```

```
(90, 99, 98, 87)
None
```

In [5]:
```
add=lambda a,b:a+b
a=add(1,3)
print(a)
```

```
4
```

In [6]:
```

```

```
120
```

In [20]:
```
a=list()
```

In [21]:
```
type(a)
```

Out[21]:
```
list
```

In [22]:
```
a=[10,20,30,40,40,50,60]
```

In [23]:
```
print(a)
```

```
[10, 20, 30, 40, 40, 50, 60]
```

In [27]:
```
a=[10,20,30,40,50]
sum=0
for i in a:
```

```
    sum=sum+i
    print(sum)
```

```
10
30
60
100
150
```

In [29]:
```
a = [10, 20, 30, 40, 50]
a.sort()
for i in a:
    print(i)
```

```
10
20
30
40
50
```

In [30]:
```
c=0
for i in a:
    c+=1
    print(c)
```

```
1
2
3
4
5
```

In [31]:
```
t=tuple()
```

In [33]:
```
type(t)
```

Out[33]:
```
tuple
```

In [35]:
```
s=set()
type(s)
```

Out[35]:
```
set
```

In [36]:
```
s={10,20,10,30,40,50,60,70,8}
s
```

Out[36]:
```
{8, 10, 20, 30, 40, 50, 60, 70}
```

In [37]:
```
d={}
```

In [38]:
```
type(d)
```

Out[38]:
```
dict
```

In [39]:
```
for i in range(1,10):
    if i==7:
        print(i)
```

```
7
```

In [41]:
```
for i in range(10,16):
    if i==12:
        break
    else:
        print(i)
```

```
10
11
```

In [44]:
```
for i in range(10,15):
    if i==13:
        continue
    else:
        print(i)
```

```
10
11
12
14
```

In [48]:
```
i=0
while i==10:
     pass
    print(i)
    i+=1
```

```
  File <tokenize>:4
    print(i)
    ^
IndentationError: unindent does not match any outer indentation level
```

In [49]:
```
i = 0
while i <= 10:
    print(i)
    i += 1
```

```
0
1
2
3
4
5
6
7
8
9
10
```

In [50]:
```
i=1
while i<=10:
    if i<=4:
        pass
    print(i)
    i=i+1
```

```
1
2
3
4
5
6
7
8
9
10
```

In [2]:
```
class Naidu:
    def __init__(self, a, b):
        self.a = a
        self.b = b

    def add(self):
        c = self.a + self.b
        return c

pavan = Naidu(100, 200)
result = pavan.add()
print(result)
```

```
300
```

In [3]:
```
class naidu:
    def __init__(self,a,b):
        self.a=a
        self.b=b
    def mul(self):
        c=self.a*self.b
        return c
pavan=naidu(100,300)
result=pavan.mul()
print(result)
```

```
30000
```

In [4]:
```
class tdf:
    def __init__(self,a,b):
        self.a=a
        self.b=b
    def add(self):
        c=self.a+self.b
        return c

pavan=tdf(500,700)
result=pavan.add()
print(result)
```

```
1200
```

In [6]:
```
class Manuh:
    def __init__(self, a, b):
        self.a = a
        self.b = b

    def add(self):
        c = self.a + self.b
        return c
```

```
tdf = Manuh(100, 300)
d = tdf.add()
print(d)
```

```
400
```

In [1]:
```
class pavan:
    def __init__(self,a,b):
        self.a=a
        self.b=b
    def add(self):
        c=self.a+self.b
        return c
naidu=pavan(142,123)
mar=naidu.add()
print(mar)
```

```
265
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: