opps concept 1

March 12, 2024

[51]:

```
a=100
d=3221
c=322121
```

[2]:

```
s="sony"
```

[22]:

```
class bankaccount:

    def openaccount(self,name,email_id):
        print("open an account by taking name and email id")
        return name+email_id
    def deposite(self,amount):
        print("iam trying to deposite an ammount in my account")
    def withdraw(self,amt_with):
        print("withdraw the amount")
    def update_details(self,name_update,email_update):
        peint("this func will update my name and will if for account")
```

[23]:

```
sony=bankaccount()
```

[24]:

```
sony.openaccount("sony","jarupulasony32@gmail.com")
```

```
open an account by taking name and email id
```

[24]:

```
'sonyjarupulasony32@gmail.com'
```

[25]:

```
sony.deposite(8786)
```

```
iam trying to deposite an ammount in my account
```

[29]:

```
bankaccount.mro()
```

[29]:

```
[__main__.bankaccount, object]
```

[30]:

```
sony.withdraw(6736)
```

```
withdraw the amount
```

[10]:
```
    def openaccount(name,email_id):
        print("open an account by taking name and emailid")
        return name+email_id
```

[15]:
```
    def deposite(amount):
        print("iam trying to deposite an ammount in my account")
```

[16]:
```
openaccount("sony","jaryupulasjnk@123")
```

```
open an account by taking name and emailid
```

[16]:
```
'sonyjaryupulasjnk@123'
```

[18]:
```
deposite(2000)
```

```
iam trying to deposite an ammount in my account
```

[34]:
```
class list_ops:
    def exractfromindex(self,l,index):
        return l[index]
    def exractrangedata(self,l,start,end):
        return l[start:end]
    def exracteven(self,l):
        l1=[]
        for i in l:
            if i%2==0:
                l1.append(i)
        return l1
    def exractodd(self,l):
        l1=[]
        for i in l:
            if i%2!=0:
                l1.append(i)
        return l1
```

[36]:
```
first_obj=list_ops()
first_obj.exractrangedata([1,2,34,43,4,32,53,65,98],0,3)
```

[36]:
```
[1, 2, 34]
```

[40]:
```
first_obj.exractfromindex([213,23,34,43,4,34,23],3)
```

[40]:
```
43
```

[41]:
```
first_obj.exractodd([1,2,33,2,44,328970,766])
```

[41]:
```
[1, 33]
```

[33]:

```
l=[1,32,23,3222,3289]
l[3]
```

[33]:

```
3222
```

[91]:

```
class list_ops:
    #a=10
    #l=[1112,43,54,656,8765]
    def __init__(sony,l):
        sony.l1=l
        sony.l2=220
        sony.l3="sony"
        sony.l4=(1,2,3,2,332,98)
        sony.l5={"key":"dic"}
    def exractfromindex(sony,l,index):
        return l[index]
    def exractrangedata(sony,l,start,end):
        return l[start:end]
    def exracteven(sony,l):
        l1=[]
        for i in l:
            if i%2==0:
                l1.append(i)
        return l1
    def exractodd(self,l):
        l1=[]
        for i in l:
            if i%2!=0:
                l1.append(i)
        return l1
```

[92]:

```
second_obj=list_ops([1,2,3,4,4])
```

[93]:

```
second_obj.l1
```

[93]:

```
[1, 2, 3, 4, 4]
```

[95]:

```
second_obj.l5
```

[95]:

```
{'key': 'dic'}
```

[70]:

```
naresh=list_ops()
```

[71]:

```
naresh.exractodd(l)
```

[71]:

```
[3, 23, 23, 233]
```

[72]:
```
a
```

[72]:
```
100
```

[73]:
```
naresh.a
```

[73]:
```
10
```

[75]:
```
naresh.exractodd(naresh.p)
```

[75]:
```
[43, 8765]
```

[74]:
```
naresh.p
```

[74]:
```
[1112, 43, 54, 656, 8765]
```

[108]:
```
class book:
    def __init__(self,name,title,pageno):
        self.name_of_book=name
        self.title_of_book=title
        self.page_no=pageno
    def exract_details(self):
        print(self.name_of_book,self.title_of_book)

    def print_page_no(self):
        print(self.page_no)
```

[109]:
```
sony=book("dsa","practical dsa",765)
akhil=book("data science","emplementation of data science",9898)
```

[110]:
```
sony.exract_details()
```

```
dsa practical dsa
```

[111]:
```
sony.print_page_no()
```

```
765
```

[107]:
```
sony.title_of_book()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[107], line 1
----> 1 sony.title_of_book()

TypeError: 'str' object is not callable
```

[116]:

```
class book:
    def __init__(self):
        self.name_of_book="dsa"
        self.title_of_book="practical dsa"
        self.page_no=123
    def exract_details(self):
        print(self.name_of_book,self.title_of_book)

    def print_page_no(self):
        print(self.page_no)
```

[117]:

```
std1=book()
```

[119]:

```
std1.page_no()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[119], line 1
----> 1 std1.page_no()

TypeError: 'int' object is not callable
```

[1]:

```
l=[2,1,3,32,4,32,4343]
```

[2]:

```
ll
```

```
total 36
-rw-r--r-- 1 jovyan 15210 Mar 11 17:45 'opps concept 1.ipynb'
-rw-r--r-- 1 jovyan   198 Mar 11 15:36  README.md
-rw-r--r-- 1 jovyan  3151 Mar 11 15:36  sample-code.ipynb
-rw-r--r-- 1 jovyan    72 Mar 12 14:54  Untitled1.ipynb
-rw-r--r-- 1 jovyan    72 Mar 12 15:47  Untitled2.ipynb
-rw-r--r-- 1 jovyan   617 Mar 12 03:57  Untitled.ipynb
```

[6]:

```
class gmail_ops:
    def __init__(self,userid,password):
        self.userid=userid
        self.password=password
        self.url="https://mail.google.com/mail/u/1/#inbox"
    def login(self):
        print("take userid" + self.userid+"take password"+self.password+"hit␣
 ↪url"+self.url)
        print("login")
    def read_mail(self):
        print("reply mail for"+self.userid+" "+self.password)
    def reply_mail(self):
        print("reply mail for"+self.userid+" "+self.password)
```

[7]:

```
user1=gmail_ops("user1","user1pass")
user1.login()
```

```
take useriduser1take passworduser1passhit
urlhttps://mail.google.com/mail/u/1/#inbox
login
```

[19]:

```
from functools import reduce
class calculator:
    def __init__(self,username):
        self.username=username
    def add(self,*args):
         #*args are used to enter n number of input
        return self.username, sum(args)
    def subtraction(self,*args):
        return self.username, reduce(lambda a,b:a-b,args)
    def mult(self,*args):
        return self.username, reduce(lambda a,b:a*b,args)
    def div(self,*args):
        return self.username, reduce(lambda a,b:a/b,args)
```

[21]:

```
sony=calculator("sony")
```

[22]:

```
sony.mult(2,2,57,87,7654,43,43)
```

[22]:

```
('sony', 280723951656)
```

[23]:

```
sony.subtraction(21,3,34,54,34,6,4,76,865)
```

[23]:

```
('sony', -1055)
```

[29]:

```
naresh=calculator("naresh")
```

[30]:

```
naresh.div(21,43,32,4,43,32,43,32332)
```

[30]:

```
('naresh', 1.994441994071778e-12)
```

[ ]:

```

```

[ ]:

```

```