

Name:-Vaishnavi Kedare

Roll no:- 173 PRN:-202201070094

Division:-A Batch:-A4

Assignment 1

Problem statement : Take/Prepare any text files for any real-life application. For Ex. "Stud.txt", "Placement.csv" and "Result. csv" files for result Analysis. Combine into "StudentDetails.csv". Perform all statistical analysis (Average, Max, Min, Count, Sum, Percentage) on it

Source code:

```
In [1]: #merge 2 data sets
f1 = open("emp.csv","r")
f2 = open("sal.csv","r")
f3 = open("emp_sal.csv","w")

contents1 = f1.read()
contents2 = f2.read()

nm = []
sal = []

lines1 = contents1.split("\n")
lines2 = contents2.split("\n")
for l1 in lines1:
    words1 = l1.split(",")

    for l2 in lines2:
        words2 = l2.split(",")
        if(words1[0]==words2[0]):
            l1 = l1 + "," + words2[1] + "," + words2[2] + "\n"
            f3.write(l1)

        nm.append(words1[1])
        sal.append(int(words2[2]))

f1.close()
f2.close()
f3.close()

#for finding index sal.index(max(sal))

hs = nm[sal.index(max(sal))]
print(hs ,"has the highest salary")

desgn = ["Manager","Sr. Manager","Ast. Manager","Supervisor","Employee"]
print("The employee with designation is Sr. Manager is :")
for i in range(len(desgn)):
    if desgn[i]=="Sr. Manager" or desgn[i]=="sr. Manager":
        print(nm[i],end=" ")
```

Output:

```
['Sanvi', 'Mrunmayee', 'Jayesh', 'Gouri', 'Mahesh']  
[100000, 150000, 90500, 100500, 85000]  
Largest salary is : 150000  
Least salary is : 85000  
Average salary is : 105200.0  
Mrunmayee has the highest salary  
The employee with designation is Sr. Manager is :  
Mrunmayee
```

assignment2

May 27, 2023

```
[ ]: product_details=[]
supplier_details=dict()
customer_details=[]
gender={}
fp1= open("sales.csv","r")
data=fp1.readline()
while(True):
    data=fp1.readline()
    if not data:
        break
    print(data)
    data= data.replace("\n","")
    temp= data.split(",")
    product_details.append(temp[1])
    customer_details.append(temp[3])
    supplier_details.update({temp[0]:temp[2]})
    gender.update({temp[3]:temp[4]})
```

P00001,Lenovo laptop,Raka Ele.,Kaustobh Mahajan,male

P00002,Samsung Laptop,Vijay Sales,Siddhi kivale,female

P00003,Realmi 10pro,Gada Ele.,Sanket Kandalkar,male

P00004,Oppo f21,Surya Ele.,Yash mali,male

P00005,Lenovo laptop,Raka Ele.,Yash Bagul,male

P00006,Samsung M31,Gada Ele.,Siddhi kivale,female

P00007,LG TV 32*,Vijay Sales,Sanket Kandalkar,male

P00008,Oppo f21,Surya Ele.,Kaustobh Mahajan,male

P00009,Lenovo laptop,Raka Ele.,Yash mali,male

P00010,Samsung M31,Gada Ele.,Siddhi kivale,female

P00011,LG TV 32*,Surya Ele.,Sanket Kandalkar,male

P00012,Lenovo laptop,Raka Ele.,Kaustobh Mahajan,male

P00013,Samsung M31,Surya Ele.,Yash mali,male

P00014,Realmi 10pro,Raka Ele.,Siddhi kivale,female

P00015,Lenovo laptop,Gada Ele.,Tanuja Mali,female

P00016,Oppo f21,Vijay Sales,Kaustobh Mahajan,male

P00017,LG TV 32*,Deshmukh Sales,Sanket Kandalkar,male

P00018,Lenovo laptop,Raka Ele.,Siddhi kivale,female

P00019,Samsung M21,Deshmukh Sales,Kaustobh Mahajan,male

P00020,LG TV 32*,Gada Ele.,Yash mali,male

```
[ ]: fp1.close()
```

```
[ ]: customer_details= tuple(customer_details)
print(type(customer_details))
print("\nproduct_details\n",product_details,end='')
print("\ncustomer_details\n",customer_details,end='')
print("\nsupplier_details\n",supplier_details,end='')
print("\ngender\n",gender,end='')
```

```
<class 'tuple'>
```

```
product_details
```

```
['Lenovo laptop', 'Samsung Laptop', 'Realmi 10pro', 'Oppo f21', 'Lenovo
laptop', 'Samsung M31', 'LG TV 32*', 'Oppo f21', 'Lenovo laptop', 'Samsung M31',
'LG TV 32*', 'Lenovo laptop', 'Samsung M31', 'Realmi 10pro', 'Lenovo laptop',
'Oppo f21', 'LG TV 32*', 'Lenovo laptop', 'Samsung M21', 'LG TV 32*']
```

```
customer_details
```

```
('Kaustobh Mahajan', 'Siddhi kivale', 'Sanket Kandalkar', 'Yash mali', 'Yash
Bagul', 'Siddhi kivale', 'Sanket Kandalkar', 'Kaustobh Mahajan', 'Yash mali',
'Siddhi kivale', 'Sanket Kandalkar', 'Kaustobh Mahajan', 'Yash mali', 'Siddhi
kivale', 'Tanuja Mali', 'Kaustobh Mahajan', 'Sanket Kandalkar', 'Siddhi
kivale', 'Kaustobh Mahajan', 'Yash mali')
```

```
supplier_details
```

```
{'P00001': 'Raka Ele.', 'P00002': 'Vijay Sales', 'P00003': 'Gada Ele.',
'P00004': 'Surya Ele.', 'P00005': 'Raka Ele.', 'P00006': 'Gada Ele.', 'P00007':
'Vijay Sales', 'P00008': 'Surya Ele.', 'P00009': 'Raka Ele.', 'P00010': 'Gada
Ele.', 'P00011': 'Surya Ele.', 'P00012': 'Raka Ele.', 'P00013': 'Surya Ele.',
```

```
'P00014': 'Raka Ele.', 'P00015': 'Gada Ele.', 'P00016': 'Vijay Sales', 'P00017':  
'Deshmukh Sales', 'P00018': 'Raka Ele.', 'P00019': 'Deshmukh Sales', 'P00020':  
'Gada Ele.'}
```

gender

```
{'Kaustobh Mahajan': 'male', 'Siddhi kivale': 'female', 'Sanket Kandalkar':  
'male', 'Yash mali': 'male', 'Yash Bagul': 'male', 'Tanuja Mali': 'female'}
```

```
[ ]: frequency= {}  
for item in product_details:  
    if item in frequency:  
        frequency[item] += 1  
    else:  
        frequency[item] = 1  
print(frequency)  
marklist= sorted(frequency.items(), key=lambda x: x[1],reverse=True)  
sortdict = dict(marklist)  
print(sortdict)  
print('The most popular product for sales',list(sortdict.  
↳keys())[0],'sold',list(sortdict.values())[0],'times')
```

```
{'Lenovo laptop': 6, 'Samsung Laptop': 1, 'Realmi 10pro': 2, 'Oppo f21': 3,  
'Samsung M31': 3, 'LG TV 32*': 4, 'Samsung M21': 1}
```

```
{'Lenovo laptop': 6, 'LG TV 32*': 4, 'Oppo f21': 3, 'Samsung M31': 3, 'Realmi  
10pro': 2, 'Samsung Laptop': 1, 'Samsung M21': 1}
```

The most popular product for sales Lenovo laptop sold 6 times

```
[ ]: from collections import Counter  
counter = dict(Counter(list(supplier_details.values())))  
sorted_counter = sorted(counter.items(), key= lambda x:x[1],reverse=True)  
sorted_counter = dict(sorted_counter)  
print('The most popular product for sales',list(sorted_counter.keys())[0],  
↳'sold',list(sorted_counter.values())[0],'Items')
```

The most popular product for sales Raka Ele. sold 6 Items

```
[ ]: frequency= {}  
for item in customer_details:  
    if item in frequency:  
        frequency[item] += 1  
    else:  
        frequency[item] = 1  
print('Frequency is as below:\n',frequency)  
marklist= sorted(frequency.items(), key=lambda x: x[1],reverse=True)  
sortdict = dict(marklist)  
print('\nSorted dict is as below:\n',sortdict)  
print('\n\nThe customer who buys most of the products',list(sortdict.  
↳keys())[0],'buy',list(sortdict.values())[0],'Items')
```

Frequency is as below:

```
{'Kaustoozh Mahajan': 5, 'Siddhi kivale': 5, 'Sanket Kandalkar': 4, 'Yash mali': 4, 'Yash Bagul': 1, 'Tanuja Mali': 1}
```

Sorted dict is as below:

```
{'Kaustoozh Mahajan': 5, 'Siddhi kivale': 5, 'Sanket Kandalkar': 4, 'Yash mali': 4, 'Yash Bagul': 1, 'Tanuja Mali': 1}
```

The customer who buys most of the products Kaustoozh Mahajan buy 5 Items

```
[ ]: from collections import Counter
counter = dict(Counter(customer_details))
sorted_counter = sorted(counter.items(), key= lambda x:x[1],reverse=True)
sorted_counter = dict(sorted_counter)
print('The customer who buys most of the products',list(sorted_counter.
↳keys())[0], 'buy',list(sorted_counter.values())[0], 'Items')
```

The customer who buys most of the products Kaustoozh Mahajan buy 5 Items

```
[ ]: from collections import Counter
counter = dict(Counter(customer_details))
names = list(counter.keys())
print(names)
male = 0
female = 0
for name in names:
    if gender[name]=='male':
        male = male+1
    if gender[name]=='female':
        female=female+1
print('Total no of Male=',male)
print('Total no of Female=',female)
```

```
['Kaustoozh Mahajan', 'Siddhi kivale', 'Sanket Kandalkar', 'Yash mali', 'Yash
Bagul', 'Tanuja Mali']
Total no of Male= 4
Total no of Female= 2
```

Vaishnavi Prakash Kedare

Div:- A(A4)

Roll no:- 173

```
import numpy as np
array1=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(array1)
```

Output

```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
array2=np.array([[11,12,13],[14,15,16],[17,18,19]])
print(array2)
```

Output

```
array([[11, 12, 13],
       [14, 15, 16],
       [17, 18, 19]])
```

1. Matrix Operation

1.1 Addition

```
resultarray=array1+array2
print("\nUsing Operator:\n",resultarray)
resultarray=np.add(array1,array2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output

```
Using Operator:
[[12 14 16]
 [18 20 22]
 [24 26 28]]
```

Using Numpy Function:

```
[[12 14 16]
```

```
[18 20 22]
```

```
[24 26 28]]
```

1.2. Subtraction

```
resultarray=array1-array2
```

```
print("\nUsing Operator:\n",resultarray)
```

```
resultarray=np.subtract(array1,array2)
```

```
print("\nUsing Numpy Function:\n",resultarray)
```

Output

Using Operator:

```
[[ -10 -10 -10]
```

```
[ -10 -10 -10]
```

```
[ -10 -10 -10]]
```

Using Numpy Function:

```
[[ -10 -10 -10]
```

```
[ -10 -10 -10]
```

```
[ -10 -10 -10]]
```

1.3. Multiplication

```
resultarray=array1*array2
```

```
print("\nUsing Operator:\n",resultarray)
```

```
resultarray=np.multiply(array1,array2)
```

```
print("\nUsing Numpy Function:\n",resultarray)
```

Output

Using Operator:

```
[[ 11 24 39]
```

```
[ 56 75 96]
```

```
[119 144 171]]
```

Using Numpy Function:

```
[[ 11 24 39]
```



```
[ 56 75 96]
[119 144 171]]
```

1.4. Division

```
resultarray=array1/array2
print("\nUsing Operator:\n",resultarray)
resultarray=np.divide(array1,array2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output

```
Using Operator:
[[0.09090909 0.16666667 0.23076923]
 [0.28571429 0.33333333 0.375 ]
 [0.41176471 0.44444444 0.47368421]]
Using Numpy Function:
[[0.09090909 0.16666667 0.23076923]
 [0.28571429 0.33333333 0.375 ]
 [0.41176471 0.44444444 0.47368421]]
```

1.5. Mod

```
resultarray=array1%array2
print("\nUsing Operator:\n",resultarray)
resultarray=np.mod(array1,array2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output

```
Using Operator:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Using Numpy Function:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

1.6. dot Product

```
resultarray=np.dot(array1,array2)  
print("",resultarray)
```

Output

```
[[ 90 96 102]  
 [216 231 246]  
 [342 366 390]]
```

.7. Transpose

```
resultarray=np.transpose(array1)  
print(resultarray)  
#Or  
resultarray=array1.transpose()  
print(resultarray)
```

Output

```
[[1 4 7]  
 [2 5 8]  
 [3 6 9]]  
[[1 4 7]  
 [2 5 8]  
 [3 6 9]]
```

2. Horizontal and vertical stacking of Numpy Arrays

2.1. Horizontal Stacking

```
resultarray=np.hstack((array1,array2))  
resultarray
```

Output

```
array([[ 1, 2, 3, 11, 12, 13],  
 [ 4, 5, 6, 14, 15, 16],
```

```
[ 7, 8, 9, 17, 18, 19]])
```

2.2. Vertical Stacking

```
resultarray=np.vstack((array1,array2))
```

```
resultarray
```

Output

```
array([[ 1, 2, 3],  
[ 4, 5, 6],  
[ 7, 8, 9],  
[11, 12, 13],  
[14, 15, 16],  
[17, 18, 19]])
```

3. Custom sequence generation

3.1. Range

```
import numpy as np
```

```
nparray=np.arange(0,12,1).reshape(3,4)
```

```
nparray
```

Output

```
array([[ 0, 1, 2, 3],  
[ 4, 5, 6, 7],  
[ 8, 9, 10, 11]])
```

3.2. Linearly Separable

```
nparray=np.linspace(start=0,stop=24,num=12).reshape(3,4)
```

```
nparray
```

Output

```
array([[ 0. , 2.18181818, 4.36363636, 6.54545455],
 [ 8.72727273, 10.90909091, 13.09090909, 15.27272727],
 [17.45454545, 19.63636364, 21.81818182, 24. ]])
```

3.3. Empty Array

```
nparray=np.empty((3,3),int)
```

```
nparray
```

Output

```
array([[ 11, 24, 39],
 [ 56, 75, 96],
 [119, 144, 171]])
```

3.4. Empty Like Some other array

```
nparray=np.empty_like(array1)
```

```
nparray
```

Output

```
array([[ 90, 96, 102],
 [216, 231, 246],
 [342, 366, 390]])
```

3.5. Identity Matrix

```
nparray=np.identity(3)
```

```
nparray
```

Output

```
array([[1., 0., 0.],
 [0., 1., 0.],
 [0., 0., 1.]])
```

4. Arithmetic and Statistical Operations, Mathematical Operations, Bitwise Operators

4.1. Arithmetic Operation

```
array1=np.array([1,2,3,4,5])  
array2=np.array([11,12,13,14,15])  
print(array1)  
print(array2)
```

Output

```
[1 2 3 4 5]  
[11 12 13 14 15]
```

Addition

```
print(np.add(array1,array2))
```

Subtraction

```
print(np.subtract(array1,array2))
```

Multiplication

```
print(np.multiply(array1,array2))
```

Division

```
print(np.divide(array1,array2))
```

Output

```
[12 14 16 18 20]  
[-10 -10 -10 -10 -10]  
[11 24 39 56 75]  
[0.09090909 0.16666667 0.23076923 0.28571429  
0.33333333]
```

4.2. Statistical and Mathematical Operations

```
array1=np.array([1,2,3,4,5,9,6,7,8,9,9])
```

Standard Deviation

```
print(np.std(array1))
```

#Minimum

```
print(np.min(array1))
```

#Summation

```

print(np.sum(array1))
#Median
print(np.median(array1))
#Mean
print(np.mean(array1))
#Mode
from scipy import stats
print("Most Frequent element=",stats.mode(array1)[0])
print("Number of Occarances=",stats.mode(array1)[1])
# Variance
print(np.var(array1))

```

Output

2.7990553306073913

1

63

6.0

5.7272727272727275

Most Frequent element= [9]

Number of Occarances= [3]

7.834710743801653

4.3. Bitwise Operations

```
array1=np.array([1,2,3],dtype=np.uint8)
```

```
array2=np.array([4,5,6])
```

AND

```
resultarray=np.bitwise_and(array1,array2)
```

```
print(resultarray)
```

OR

```
resultarray=np.bitwise_or(array1,array2)
```

```
print(resultarray)
```

#LeftShift

```
resultarray=np.left_shift(array1,2)
```

```
print(resultarray)
```

#RightShift

```
resultarray=np.right_shift(array1,2)
```

```
print(resultarray)
```

Output

[0 0 2]

[5 7 7]

[4 8 12]

[0 0 0]

You can get Binary Representation of Number

```
print(np.binary_repr(10,8))
```

```
resultarray=np.left_shift(10,2)
```

```
print(resultarray)
```

```
print(np.binary_repr(np.left_shift(10,2),8))
```

Output

00001010

40

00101000

5.Copying and viewing arrays

5.1 Copy

```
array1=np.arange(1,10)
```

```
print(array1)
```

```
newarray=array1.copy()
```

```
print(newarray)
```

##modification in Original Array

```
array1[0]=100
```

```
print(array1)
```

```
print(newarray)
```

Output

[1 2 3 4 5 6 7 8 9]

[1 2 3 4 5 6 7 8 9]

[100 2 3 4 5 6 7 8 9]

[1 2 3 4 5 6 7 8 9]

5.2 View

```
array1=np.arange(1,10)
print(array1)
newarray=array1.view()
print(newarray)
##modification in Original Array
array1[0]=100
print(array1)
print(newarray)
```

Output

```
[1 2 3 4 5 6 7 8 9]
[1 2 3 4 5 6 7 8 9]
[100 2 3 4 5 6 7 8 9]
[100 2 3 4 5 6 7 8 9]
```

6. Searching

```
array1=np.array([[1,2,3,12,5,7],[94,5,6,7,89,44],[7,8,9,11,13,14]])
print(array1)
```

Output

```
[[ 1 2 3 12 5 7]
 [94 5 6 7 89 44]
 [ 7 8 9 11 13 14]]
np.sort(array1,axis=0)
```

Output

```
array([[ 1, 2, 3, 7, 5, 7],
 [ 7, 5, 6, 11, 13, 14],
 [94, 8, 9, 12, 89, 44]])
np.sort(array1,axis=1)
```


Output

```
array([[ 1, 2, 3, 5, 7, 12],  
       [ 5, 6, 7, 44, 89, 94],  
       [ 7, 8, 9, 11, 13, 14]])
```

7. Searching

```
array1=np.array([1,2,3,12,5,7])  
np.searchsorted(array1,7,side="left")#Perform Search After sorting
```

Output

3

8. Counting

```
array1=np.array([1,2,3,12,5,7,0])  
print(np.count_nonzero(array1))#Return total Non Zero element  
print(np.nonzero(array1))#Return Index  
print(array1.size)#Total Element
```

Output

6

```
(array([0, 1, 2, 3, 4, 5], dtype=int64),)
```

7

9. Data Stacking

```
array1=np.array(np.arange(1,5).reshape(2,2))  
print(array1)  
array2=np.array(np.arange(11,15).reshape(2,2))  
print(array2)
```

Output

```
[[1 2]
 [3 4]]
[[11 12]
 [13 14]]
newarray=np.stack([array1,array2],axis=0)
print(newarray)
```

Output

```
[[1 2]
 [3 4]]
[[11 12]
 [13 14]]
newarray=np.stack([array1,array2],axis=1)
print(newarray)
```

Output

```
[[1 2]
 [11 12]]
[[3 4]
 [13 14]]
```

10. Append

```
array1=np.arange(1,10).reshape(3,3)
print(array1)
array2=np.arange(21,30).reshape(3,3)
print(array2)
```

Output

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

```
[[21 22 23]
 [24 25 26]
 [27 28 29]]
np.append(array1,array2,axis=0)
```

Output

```
array([[ 1, 2, 3],
 [ 4, 5, 6],
 [ 7, 8, 9],
 [21, 22, 23],
 [24, 25, 26],
 [27, 28, 29]])
np.append(array1,array2,axis=1)
```

Output

```
array([[ 1, 2, 3, 21, 22, 23],
 [ 4, 5, 6, 24, 25, 26],
 [ 7, 8, 9, 27, 28, 29]])
```

11. Concat

```
array1=np.arange(1,10).reshape(3,3)
print(array1)
array2=np.arange(21,30).reshape(3,3)
print(array2)
```

Output

```
[[1 2 3]
 [4 5 6]
```

```
[7 8 9]]
[[21 22 23]
 [24 25 26]
 [27 28 29]]
np.concatenate((array1,array2),axis=0)
```

Output

```
array([[ 1, 2, 3],
 [ 4, 5, 6],
 [ 7, 8, 9],
 [21, 22, 23],
 [24, 25, 26],
 [27, 28, 29]])
np.concatenate((array1,array2),axis=1)
```

Output

```
array([[ 1, 2, 3, 21, 22, 23],
 [ 4, 5, 6, 24, 25, 26],
 [ 7, 8, 9, 27, 28, 29]])
import numpy as np
# using loadtxt()
arr =
np.loadtxt("F:\\ISO\\EDS\\NOTES\\dataset\\testmarks1.csv",delimiter=",",skipr
ows=1)
print(type(arr))
arr.shape
```

Output

```
<class 'numpy.ndarray'>
```

```
(10, 5)
```

```
EDS=arr[:,1]
```

```
print(EDS)
```

Output

```
[43.05 43.47 42.24 39.24 40.9 39.47 41.68 42.19 44.75  
46.95]
```

```
SON=arr[:,2]
```

```
print(SON)
```

Output

```
[27.79 28.52 28.16 26.16 26.03 26.31 25.63 27.61  
28.35 28.88]
```

NAME :Vaishnavi
Kedare
CLASS : A
BATCH : A4
ROLL NO : 173

```
import pandas as pd
import numpy as np
f1 = open("F:\grainsales.csv","r")
data = pd.read_csv(f1)
df = pd.DataFrame(data)
maindata = df
df['Sales'].describe()
df=df.groupby('Months').sum()
df=df.sort_values(by= ['Sales'], ascending=False) df.head(1)
print("Best Month for Sales: July")
print("Revenue Earned was: 16000000")
df
maindata
df = df.groupby("GrainName").sum()
df = df.sort_values(by=["Sales"], ascending = False)
df.head (1)
print("Most Sold Grain is: Wheat")
print("The Best Month for sales is July and this product has occurred in July
so this is most sold product with highest sales")
df
maindata
df= df.groupby("City").sum()
df = df.sort_values (by = ['Sales'], ascending= False)
df.head (1)
print("'Asansole' Has sold highest no. of products")
df
maindata
df = df.groupby('State').sum()
df = df.sort_values (by = ['Sales'], ascending = False) print("West
Bengal has highest sales")
```

Best Month for Sales: July

Revenue Earned was: 16000000

Most Sold Grain is: Wheat

The Best Month for sales is July and this product has occurred in July so this is most sold product with highest sales

'Asansole' Has sold highest no. of products

West Bengal has highest sales.

Name -Vaishnavi P Kedare

Roll no-173

Batch-A4

Prn no-202201070094

ASSIGNMENT -5

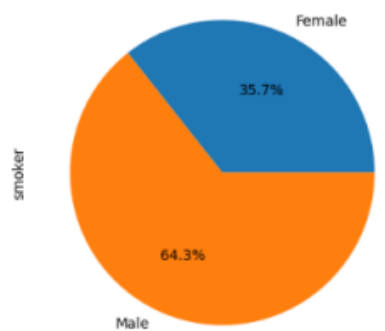
```
[ ] import pandas as pd
import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
print(d)
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
..
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

[244 rows x 7 columns]

```
import pandas as pd
import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
#print(d)
t1 = d.groupby("sex").count()
t1["smoker"].plot(kind = "pie", autopct = '%1.1f%%')
plt.show()
print(t1)
```

☐



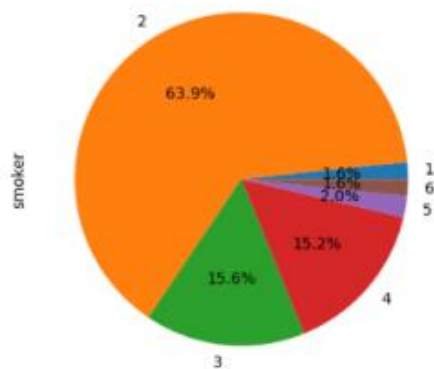
	total_bill	tip	smoker	day	time	size
sex						
Female	87	87	87	87	87	87
Male	157	157	157	157	157	157


```

import pandas as pd
import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
#print(d)
t1 = d.groupby("size").count()
t1["smoker"].plot(kind = "pie",autopct = '%1.1f%%')
plt.show()
print(t1)

```

3



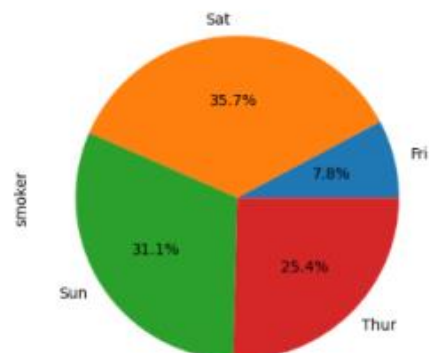
size	total_bill	tip	sex	smoker	day	time
1	4	4	4	4	4	4
2	156	156	156	156	156	156
3	38	38	38	38	38	38
4	37	37	37	37	37	37
5	5	5	5	5	5	5
6	4	4	4	4	4	4

```

import pandas as pd
import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
#print(d)
t1 = d.groupby("day").count()
t1["smoker"].plot(kind = "pie",autopct = '%1.1f%%')
plt.show()
print(t1)

```

4



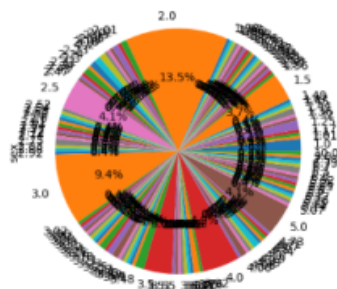
day	total_bill	tip	sex	smoker	time	size
Fri	19	19	19	19	19	19
Sat	87	87	87	87	87	87
Sun	76	76	76	76	76	76
Thur	62	62	62	62	62	62

```

import pandas as pd
import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
#print(d)
t1 = d.groupby("tip").count()
t1["sex"].plot(kind = "pie",autopct = '%1.1f%%')
plt.show()
print(t1)

```

☐



tip	total_bill	sex	smoker	day	time	size
1.00	4	4	4	4	4	4
1.01	1	1	1	1	1	1
1.10	1	1	1	1	1	1
1.17	1	1	1	1	1	1
1.25	3	3	3	3	3	3
...
6.70	1	1	1	1	1	1
6.73	1	1	1	1	1	1
7.58	1	1	1	1	1	1
9.00	1	1	1	1	1	1
10.00	1	1	1	1	1	1

[123 rows x 6 columns]

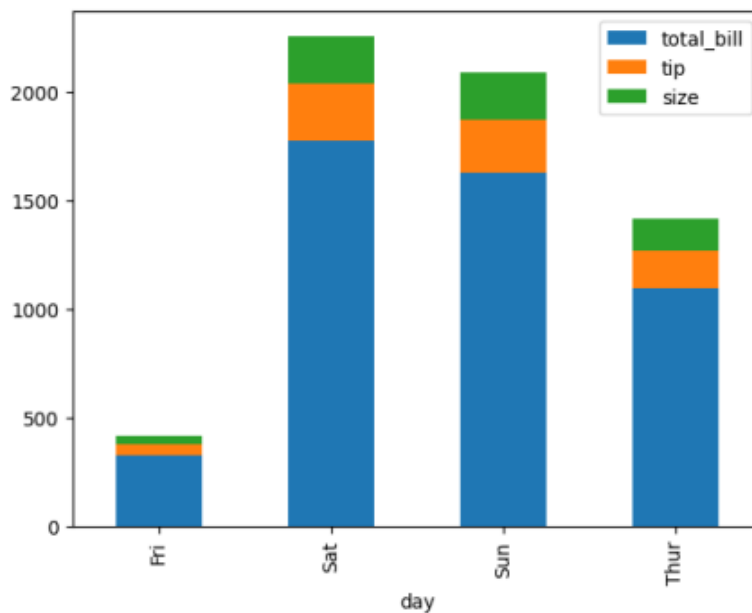
```

▶ t2 = d.groupby("day").sum("total_bill")
t2.plot(kind = "bar",stacked = True)

```

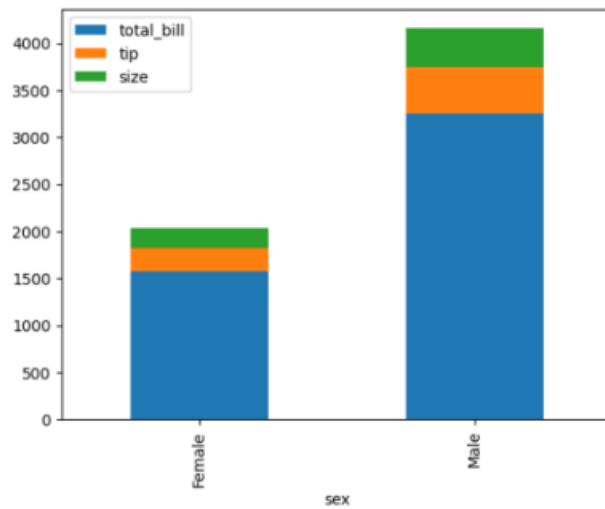
☐

<Axes: xlabel='day'>



```
t2 = d.groupby("sex").sum("smoker")
t2.plot(kind = "bar", stacked = True)
```

<Axes: xlabel='sex'>



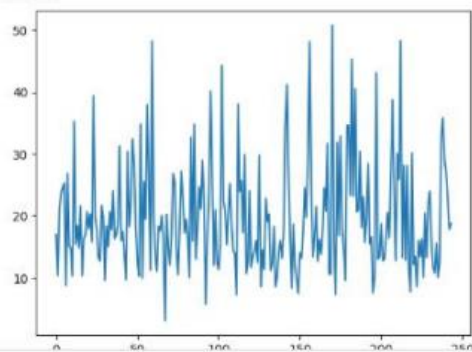
```
[ ] import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
#print(d)
d1 = d.groupby("total_bill").count()
print(d1)
```

```
total_bill  tip  sex  smoker  day  time  size
3.07        1    1      1      1      1      1
5.75        1    1      1      1      1      1
7.25        2    2      2      2      2      2
7.51        1    1      1      1      1      1
7.56        1    1      1      1      1      1
...
45.35       1    1      1      1      1      1
48.17       1    1      1      1      1      1
48.27       1    1      1      1      1      1
48.33       1    1      1      1      1      1
50.81       1    1      1      1      1      1
```

[229 rows x 6 columns]

```
d["total_bill"].plot()
```

<Axes: >



```

import pandas as pd
import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
print(d)
d1 = d.groupby("size").count()
print(d1)

```

```

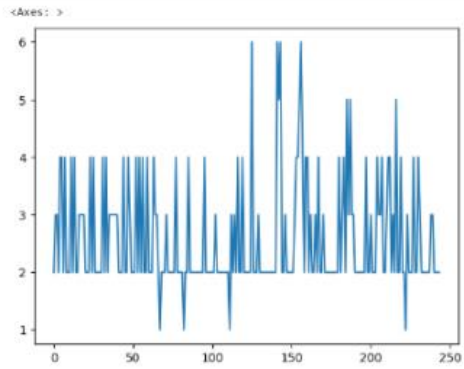
total_bill  tip  sex  smoker  day  time
size
1          4    4    4        4    4    4
2       156  156  156       156  156  156
3        38   38   38        38   38   38
4        37   37   37        37   37   37
5         5    5    5         5    5    5
6         4    4    4         4    4    4

```

```

d["size"].plot()

```

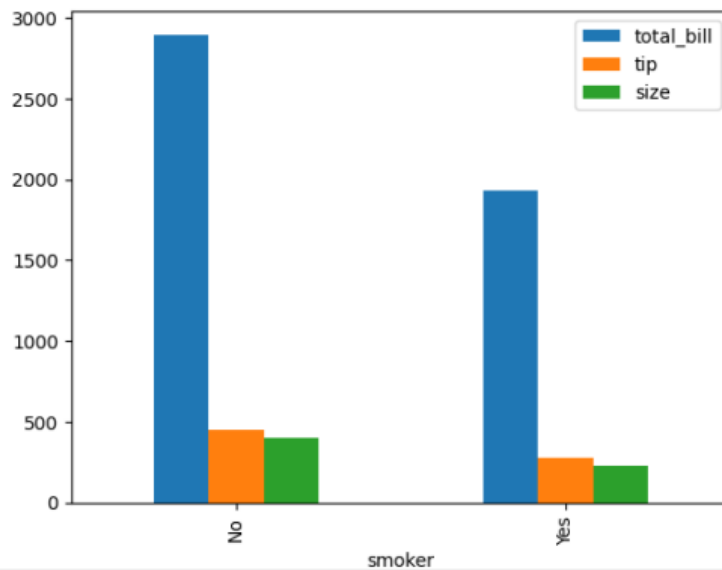


```

t2 = d.groupby("smoker").sum("tip")
t2.plot(kind = "bar",stacked = False)

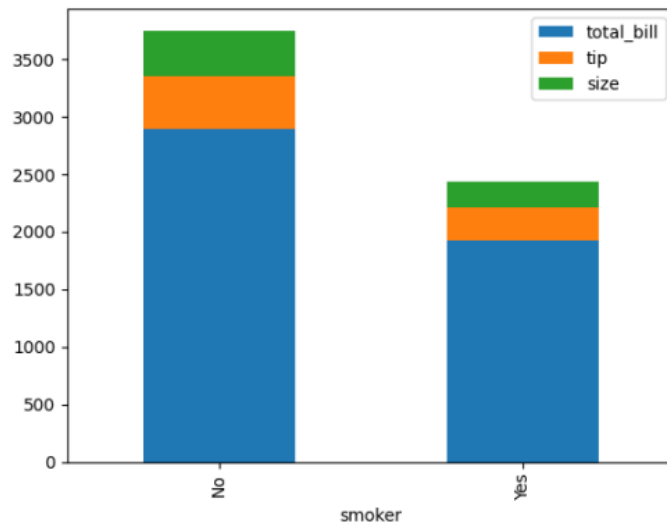
```

<Axes: xlabel='smoker'>

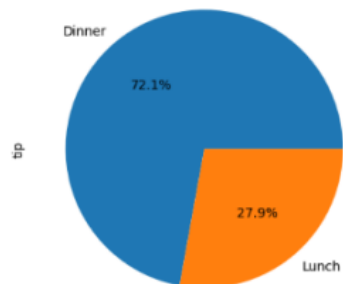


```
t2 = d.groupby("smoker").sum("time")
t2.plot(kind = "bar",stacked = True)
```

<Axes: xlabel='smoker'>



```
[ ] import pandas as pd
import matplotlib.pyplot as plt
d = pd.read_csv('/content/tips.csv')
#print(d)
t1 = d.groupby("time").count()
t1["tip"].plot(kind = "pie",autopct = '%1.1f%%')
plt.show()
print(t1)
```



time	total_bill	tip	sex	smoker	day	size
Dinner	176	176	176	176	176	176
Lunch	68	68	68	68	68	68