Unique Paper Code: 32347507

Name of the paper: Data Analysis and Visualisation

Name of the Course: B.Sc. (Hons.) Computer Science

Semester: V

Duration: 3 Hours Maximum Marks: 75

Question No. 1 is compulsory.
Attempt any *four* questions out of Q. 2 to Q. 7.
Parts of a question must be answered together

Q1 a) Give output of the following code.

```
i. import pandas as pd
     obj3 = pd.Series(['wow', 'good', 'great'],
              index=[0, 2, 4])
     obj3.reindex(range(6), method='ffill')
     obj3
```

(2)

output

```
0       wow
1       wow
2      good
3      good
4     great
5     great
dtype: object
```

(2)

(2)

```
ii. matrix =[[j for j in range(3)]for i in range(3)]
    print(matrix)
```

output

```
[[0, 1, 2], [0, 1, 2], [0, 1, 2]]
```

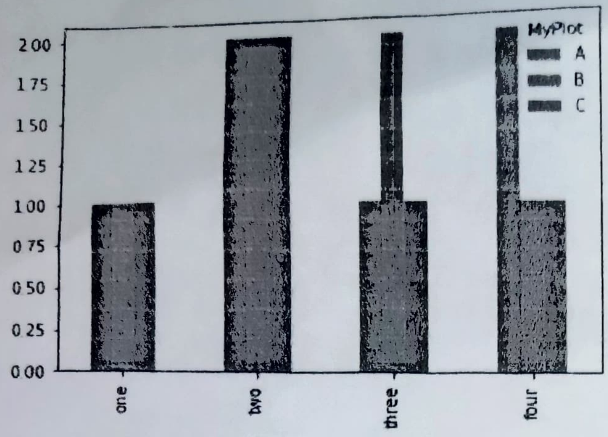
```
iii. import pandas as pd
     df=pd.DataFrame[[1,1,1],[2,2,2],[1,2,1],
     [2,1,1]],index=['one','two','three','four'],
     columns=pd.Index(['A','B','C'],name='MyPlot'))
```

Give the output for df.plot.bar().

Output

```
AxesSubplot(0.125,0.125;0.775x0.755)
```

[illegible]



b) What is a pivot table? Give one example. (2)

A pivot table is a data summarization tool frequently found in spreadsheet programs and other data analysis software. It aggregates a table of data by one or more keys, arranging the data in a rectangle with some of the group keys along the rows and some along the columns. Pivot tables in Python with pandas are made possible through the groupby facility combined with reshape opera-tions utilizing hierarchical indexing.
For example tips.pivot_table(index=['day','smoker'])

c) *Provide the output of following codes.* (3)
Given the value of string object s=3.1456 and
c="""This is a long string
that spans multiple lines"""

i.
```
fval= float(s)
type(fval)
```
Output
```
float
```
ii.
```
bool(s)
```
Output

true

iii.
```
c.count('\n')
```
Output
```
1
```

d) Consider a list seq= [1, 2, 0, 4, 6, 5, 2, 1]. Write a code to find the sum of elements of the value till element 5. (2)

Answer
```
sequence = [1, 2, 0, 4, 6, 5, 2, 1]
total_until_5 = 0
for value in sequence:
    if value == 5:
        break
    total_until_5 += value
```

print(total_until_5)

e) Consider the given arr = [1,2,8,9,3,4,7,5,10,6]. What will be the resulting array if these operations are performed arr[2:5], arr[-5: -1] and arr[::2]. (3)

Answer

```
i.    [8, 9, 3]
ii.   [4, 7, 5, 10]
iii.  [1, 8, 3, 7, 10]
```

f) Create a dataframe with four rows and three columns and populate it with random values. Index of the rows are 'Utah', 'Ohio', 'Texas', 'Oregon' and column indexes are 'b','d','e'. Write a lambda function to compute the difference between the maximum and minimum of each column. (3)

Answer

i.

```
import numpy as np
frame = pd.DataFrame(np.random.randn(4, 3),columns=list('bde'),index=['Utah',
'Ohio', 'Texas', 'Oregon'])
```

ii.

```
f = lambda x: x.max() - x.min()
frame.apply(f)
```

g) Create an array *num* of size 2 x 3 filled with all zeros then insert [[1,2,3], [4,5,6]] into array. Identify the shape of the array *num*. (3)

Answer

```
ar=np.zeros((2, 3)) ,   ar = [[1, 2, 3] , [4, 5, 6]]
ar.shape(3,2)
```

h) Write a code to read a CSV file with new delimiter as ';' and line terminator as '\n'. (3)

Answer

```
class my_dialect(csv.Dialect):
lineterminator = '\n'
delimiter = ';'
quotechar = '"'
quoting = csv.QUOTE_MINIMAL
reader = csv.reader(f, dialect=my_dialect)
```

i) Consider following piece of code and give the output. (3)

```
import pandas as pd
a = pd.DataFrame({'id': [1, 2, 9, 10],
    'val':['a', 'b', 'c', 'd']})
b = pd.DataFrame({'id': [1, 7, 10, 12,13, 7],
    'val': ['p', 'q', 'r', 's', 't', 'u']})
c = pd.merge(a, b, on='id', how='right')
```

i. How many 'NaN' values are in the dataframe 'c'?

```
   id    val_x val_y
0  1     a      p
1  7     NaN    q
2  10    d      r
3  12    NaN    s
4  13    NaN    t
5  7     NaN    u
```

4

ii. Drop duplicate values from dataframe '*b*' and keep the last duplicated value.

Answer
b.drop_duplicates(keep='last')

j) Generate DateTimeIndex of length 20 where each index will be Tuesday of the third week of a month starting from 10-Jan-2022. (3)

Answer
import pandas as pd
dates=pd.date_range('2020-01-10', periods=20, freq='WOM-3TUE')
dates

k) Consider dataframe df (4)

```
import pandas as pd
import numpy as np
df  =  pd.DataFrame({'key':  ['a',  'b',  'c']  *  4,
     'value': np.arange(12.0)})
```

What will be the output of the following statements?

i. Print the dataframe *df*.
```
df = pd.DataFrame({'key': ['a', 'b', 'c'] * 4,
'value': np.arange(12.0)})
df
```

ii. Write a code to group the dataframe using key.
```
g = df.groupby('key').value
print(g)
```

iii. Multiply each group value by 2.
```
g.transform(lambda x: x * 2)
```

Q2 a) Consider a dataframe df as (6)

```
import pandas as pd
import numpy as np
df = pd.DataFrame({'key1':['a','a','b','b','a'],
     'key2':['one','two','one','two','one'],
     'data1':np.random.randn(5),
     'data2' :np.random.randn(5)})
```

Provide the output for the following:

i. `print(df)`

	key1	key2	data1	data2
0	a	one	2.051693	-2.432268
1	a	two	0.196488	-0.134805
2	b	one	1.690703	-1.340778
3	b	two	-0.283880	-1.261686
4	a	one	-1.771815	-1.581653

ii.
```
m1 = df['data1'].groupby([df['key1'],
df['key2']]).mean()
print(m1)
```

```
key1  key2
a     one       0.139939
      two       0.196488
b     one       1.690703
      two      -0.283880
Name: data1, dtype: float64
```

iii.
```
m2 = df['data1'].groupby ([df['key1']).mean()
```

```
key1
a      0.158789
b      0.703411
Name: data1, dtype: float64
```

iv.
```
pieces = dict(list(df.groupby('key1')))
pieces['b']
```

	key1	key2	data1	data2
2	b	one	1.690703	-1.340778
3	b	two	-0.283880	-1.261686

v.
```
for(k1,k2),group in
df.groupby(['key1','key2']):
    print ((k1, k2))
    print(group)
```

```
('a', 'one')
  key1 key2     data1     data2
0    a  one  2.051693 -2.432268
4    a  one -1.771815 -1.581653
('a', 'two')
  key1 key2     data1     data2
1    a  two  0.196488 -0.134805
('b', 'one')
  key1 key2     data1     data2
2    b  one  1.690703 -1.340778
('b', 'two')
  key1 key2     data1     data2
3    b  two -0.28388 -1.261686
```

b) Give output of the following code. Justify.

```
    i.   val=['foo', 2, [4,2]]
         val[2]=(5,4)
         print(val)
```
(2)

```
output
['foo', 2, (5, 4)]
```
(2)

```
    ii.  var=(3, 5, (4,5))
         var[1]='two'
         print(var)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-15-36f8f7bc1575> in
      1 var=(3, 5, (4,5))
----> 2 var[1]='two'
      3 print(var)

TypeError: 'tuple' object does not support item assignment
```

Q3 a) Given the following list of strings
List1 = ['Amazon', 'Amazing Amazon', 'Apple', 'Microsoft ', 'Apple is good for health', 'I like Microsoft']. (5)
Using 'List1', generate the following dictionary 'Anydict' where key is the count of words in a string and value is the list of strings having that count.
Anydict={1:['Amazon', 'Apple', 'Microsoft'], 2: ['Amazing Amazon'], 3: ['I like Microsoft'], 4: ['Apple is good for health']}.

Answer

```
List1=['Amazon','Amazing Amazon','Apple','Microsoft','Apple is good for health','I like Microsoft']
Anydict={}
for i,v in enumerate(List1):
  l=len(v.split(' '))
  if l not in Anydict:
    Anydict[l]=[v]
```

```
else:
    Anydict[l].append(v)

Anydict
s = pd.Series(Anydict)
s
```

b) Write a code to read the data from a csv file. Find the number of rows and columns in the data, replace missing values with zero, and remove duplicate values. Write the modified data back to the original file. (5)

Answer

```
df = pd.read_csv('examples/ex1.csv')
df.fillna(0)
df.drop_duplicates()
df.to_csv('examples/ex1.csv')
```

Q4 a) What is the use of generator function? Write a generator function to print square of first n natural numbers where n is user input. (4)

Answer

```
def square_of_sequence(x):
    for i in range(x):
        yield i*i

squres = square_of_sequence(5)
for sqr in squres:
    print(sqr)
```

b) Write a code program to draw a scatter plot comparing marks of Mathematics= [88, 92, 80, 89, 100, 80, 60, 100, 80, 34] and Science = [35, 79, 79, 48, 100, 88, 32, 45, 20, 30] subjects. (6)
Import the necessary libraries.
Title the plot as 'Marks Comparison' and label y-axis as 'Marks Scored'.
Assign red color to mathematics marks points and blue color to science marks points.

Answer

```
import matplotlib.pyplot as plt
import pandas as pd
math_marks = [88, 92, 80, 89, 100, 80, 60, 100, 80, 34]
science_marks = [35, 79, 79, 48, 100, 88, 32, 45, 20, 30]
marks_range = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
plt.scatter(marks_range, math_marks, label='Math marks', color='r')
plt.scatter(marks_range, science_marks, label='Science marks', color='g')
plt.title('Marks Comparison')
plt.xlabel('Marks Scored')
plt.legend()
plt.show()
```

Q5 a) Consider the following data frame Family containing a family name, gender of the family member and her/his monthly income and expenditure in each record.

Name	Gender	Monthly Income	Expenditure
Shahin	Male	114000.00	58000.00
Vimal	Male	65000.00	32000.00
Vimala	Female	69500.00	38500.00
Vimala	Female	155000.00	70000.00
Karan	Male	103000.00	52000.00
Shahin	Male	55000.00	18000.00
Seema	Female	112400.00	60000.00
Seema	Female	81030.00	25000.00
Vimal	Male	71900.00	30000.00

i. Find correlation between *Monthly Income* and *Expenditure*. (1)

```
data['Monthly Income'].corr(data['Expenditure'])
```

ii. Use map function to convert each value of *Name* into uppercase. (2)

```
transform = lambda x: x.upper()
da=data.Name.map(transform)
da
```

iii. **Create a new data frame Info having a hierarchical index on columns *Name* and *Gender*.** (2)

```
Info=data.set_index(['Name','Gender'])
Info
```

b) **Consider the data array= [0.9296, 0.3164, 0.1839, 0.2046, 0.5677, 0.5955, 0.9645, 0.6532, 0.7489, 0.6536] of 10 floating-point values. Write code for following:**

i. Create 5 bins of the array using the cut method. (1)

```
import pandas as pd
arr= [0.9296, 0.3164, 0.1839, 0.2046, 0.5677, 0.5955, 0.9645,
0.6532, 0.7489, 0.6536]
pd.cut(arr,5)
```

ii. Create 5 bins of the array using the qcut method. (1)

```
import pandas as pd
arr= [0.9296, 0.3164, 0.1839, 0.2046, 0.5677, 0.5955, 0.9645,
0.6532, 0.7489, 0.6536]
pd.qcut(arr,5)
```

iii. Create 5 bins of the array with precision = 2 using cut method. Also explain the usage of parameter precision. (3)

```
import pandas as pd
arr= [0.9296, 0.3164, 0.1839, 0.2046, 0.5677, 0.5955, 0.9645,
0.6532, 0.7489, 0.6536]
```

pd.cut(arr,5, precision=2)

Q6 a) Consider the following code:

```
import pandas as pd
left =pd.DataFrame({'key1':['foo','foo','bar'],
      'key2':['one','two','one'],'lval':[1,2,3]})
right = pd.DataFrame({'key1':['foo','foo','bar',
        'bar'],'key2':['one','one','one','two'],
        'rval':[4,5,6,7]})
```

Provide output of the following:

i. `pd.merge(left, right, on=['key1'])` (2)

```
   key1  key2_x  lval  key2_y  rval
0  foo   one     1     one     4
1  foo   one     1     one     5
2  foo   two     2     one     4
3  foo   two     2     one     5
4  bar   one     3     one     6
5  bar   one     3     two     7
```

ii. (2)

```
prop_cumsum =left.sort_values(by='key2',
ascending=False).lval.cumsum()
print(prop_cumsum)
1    2
0    3
2    6
Name: lval, dtype: int64
```

iii. `left.append(right)` (2)

```
   key1  key2  lval  rval
0  foo   one   1.0   NaN
1  foo   two   2.0   NaN
2  bar   one   3.0   NaN
0  foo   one   NaN   4.0
1  foo   one   NaN   5.0
2  bar   one   NaN   6.0
3  bar   two   NaN   7.0
```

b) Consider a data given below:

EMP ID	EMP NAME	SALARY
1	Satish	5000
2	Vani	7500
3	Ramesh	10000
4	Rajesh	8000
5	Virat	9500

Write a code for the following:

i. ***Create a dataframe for the above data.*** (2)

```
import pandas as pd
Em = pd.DataFrame({'Em_ID': [1,2,3,4,5],
                  'Em_Name':
['Satesh','Vani','Ramesh','Rajesh','Virat'],
                   'Salary': [5000,7500,10000,8000,9500]})
Em
```

```
   Em_ID  Em_Name  Salary
0      1   Satesh    5000
1      2     Vani    7500
2      3   Ramesh   10000
3      4   Rajesh    8000
4      5    Virat    9500
```

ii. Print elements of 2nd to 4th column of 3rd to 5th row. (1)

```
Em.iloc[3:5,2:4]
```

iii. Print elements of all the columns for first two rows. (1)

```
Em.iloc[:2,]
```

Q7 a) Consider the code given below:

```
import pandas as pd
from datetime import datetime
dates =[datetime(2011,1,2),datetime(2011,1,5),
        datetime(2011,1,7),datetime(2011,1,8),
        datetime(2011,1,10),datetime(2011,1,12)]
ts = pd.Series(np.random.randn(6), index=dates)
```

Provide output for the following code:

i. `print(ts)` (1)

(1)

(1)

```
2011-01-02   -0.510303
2011-01-05    0.466675
2011-01-07   -2.073346
2011-01-08   -1.415322
2011-01-10    0.290394
2011-01-12   -1.828824
dtype: float64
```

ii. `print(ts + ts[::-1])`

```
2011-01-02   -1.020607
2011-01-05    0.933350
2011-01-07   -4.146693
2011-01-08   -2.830643
2011-01-10    0.580787
2011-01-12   -3.657648
dtype: float64
```

iii. `print(ts.index[0])`

```
2011-01-02 00:00:00
```

b) Write a code to convert string of date '2022-10-20' to string of date '20/10/2022'. (3)

```
import pandas as pd
st='2022-10-20'
from datetime import datetime
datetime.strptime(st,'%Y-%m-%d').strftime('%d/%m/%y')
```

c) Provide output of the following code: (4)

```
rng=pd.date_range('2010-01-01',periods=12,freq=
'T')
ts= pd.Series(np.arange(12), indexing=rng)
print(ts)
print(ts.resample('5min', closed= 'right').sum())
print(ts.resample('5min', closed= 'right', label=
'right', loffeset= '-1s').sum())
print(ts.resample('5min').ohlc())
```

Answer

```
2010-01-01 00:00:00     0
2010-01-01 00:01:00     1
2010-01-01 00:02:00     2
2010-01-01 00:03:00     3
2010-01-01 00:04:00     4
2010-01-01 00:05:00     5
2010-01-01 00:06:00     6
2010-01-01 00:07:00     7
2010-01-01 00:08:00     8
2010-01-01 00:09:00     9
2010-01-01 00:10:00    10
2010-01-01 00:11:00    11
Freq: T, dtype: int32
2009-12-31 23:55:00     0
2010-01-01 00:00:00    15
2010-01-01 00:05:00    40
2010-01-01 00:10:00    11
Freq: 5T, dtype: int32
2009-12-31 23:59:59     0
2010-01-01 00:04:59    15
2010-01-01 00:09:59    40
2010-01-01 00:14:59    11
Freq: 5T, dtype: int32
                     open  high  low  close
2010-01-01 00:00:00     0     4    0      4
2010-01-01 00:05:00     5     9    5      9
2010-01-01 00:10:00    10    11   10     11
```