

PANIMALAR INSTITUTE OF TECHNOLOGY

(A CHRISTIAN MINORITY INSTITUTION)

JAISAKTHI EDUCATIONAL TRUST

BANGALORE TRUNK ROAD, VARADHARAJAPURAM,
NASARATHPET, POONAMALLEE, CHENNAI 600 123



ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

AD8412 – DATA ANALYTICS LABORATORY

ACADEMIC YEAR: 2021 – 2022 (EVEN SEMESTER)

Name of the Student	:	
Register Number	:	
Roll Number	:	
Year & Semester	:	

PANIMALAR INSTITUTE OF TECHNOLOGY



REGISTER NUMBER:

Certified that this is a bonafide record of practical work done by _____ of II Year / III Semester of B.Tech Artificial Intelligence and Data Science in AD8412 – DATA ANALYTICS LABORATORY during the academic year 2021 - 22.

STAFF IN-CHARGE

HEAD OF THE DEPARTMENT

Submitted for the Anna University practical examination held on _____ at Panimalar Institute of Technology, Chennai – 600 123.

INTERNAL EXAMINER

EXTERNAL EXAMINER

INDEX

Expt. No.	Name of the Experiment	Page No	Date	Signature
1	Demonstration the random sampling			
2	Demonstrate the probability sampling from a known population			
3	Implementation of Z-Test – One Sample Z-Test and Two Sample Z-Test			
4	Implementation of Z-Test – using Titanic case study			
5	Implementation of T-Test – one sample t-test			
6	Implementation of T-Test – Two sample t-test and Paired T-Test			
7	Implementation of VARIANCE ANALYSIS (ANNOVA)			
8	Demonstration of Linear Regression			
9	Demonstration of Logistic Regression			
10	Demonstration of Multiple-Linear Regression			
11	Implementation of Time Series Analysis			

EX.NO: 01

PROGRAM

1 A) Picking Random Items in a List using 'random.choice()'

```
bmi_list = [29, 18, 20, 22, 19, 25, 30, 28, 22, 21, 18, 19, 20, 20, 22, 23]
```

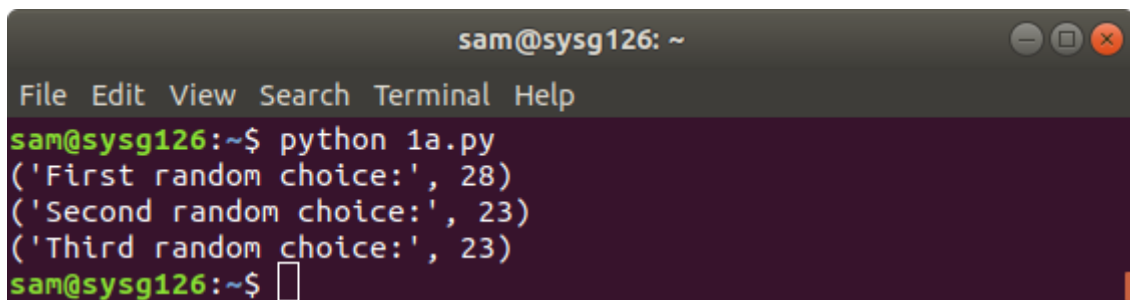
```
import random
```

```
print("First random choice:", random.choice(bmi_list))
```

```
print("Second random choice:", random.choice(bmi_list))
```

```
print("Third random choice:", random.choice(bmi_list))
```

OUTPUT:

A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'sam@sysg126:~\$'. The user enters 'python 1a.py'. The output shows three lines: ('First random choice:', 28), ('Second random choice:', 23), and ('Third random choice:', 23). The prompt returns to 'sam@sysg126:~\$' with a cursor.

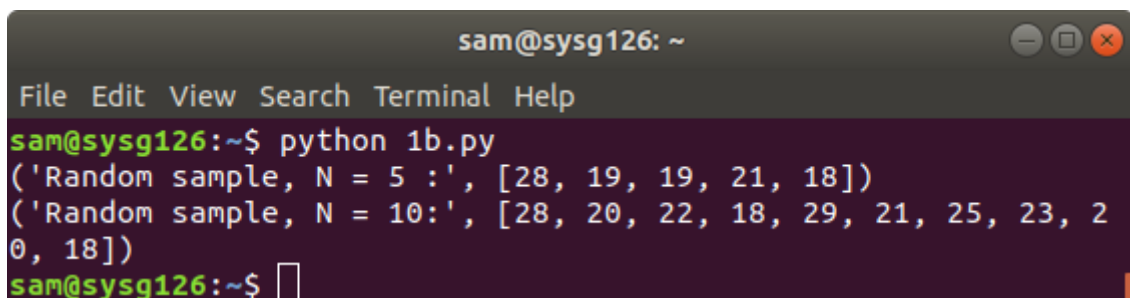
```
sam@sysg126:~$ python 1a.py
('First random choice:', 28)
('Second random choice:', 23)
('Third random choice:', 23)
sam@sysg126:~$
```

1 B) Picking Random Items in a List using 'random.sample()'

```
print("Random sample, N = 5 :", random.sample(bmi_list, 5))
```

```
print("Random sample, N = 10:", random.sample(bmi_list, 10))
```

OUTPUT:

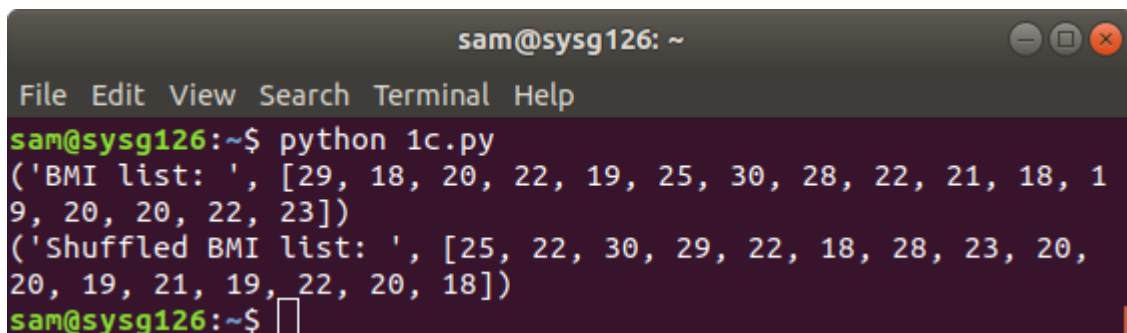
A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'sam@sysg126:~\$'. The user enters 'python 1b.py'. The output shows two lines: ('Random sample, N = 5 :', [28, 19, 19, 21, 18]) and ('Random sample, N = 10:', [28, 20, 22, 18, 29, 21, 25, 23, 20, 18]). The prompt returns to 'sam@sysg126:~\$' with a cursor.

```
sam@sysg126:~$ python 1b.py
('Random sample, N = 5 :', [28, 19, 19, 21, 18])
('Random sample, N = 10:', [28, 20, 22, 18, 29, 21, 25, 23, 20, 18])
sam@sysg126:~$
```

1 C) Randomly Shuffling Items in a List using 'random.shuffle()'

```
print("BMI list: ", bmi_list)
random.shuffle(bmi_list)
print("Shuffled BMI list: ", bmi_list)
```

OUTPUT:

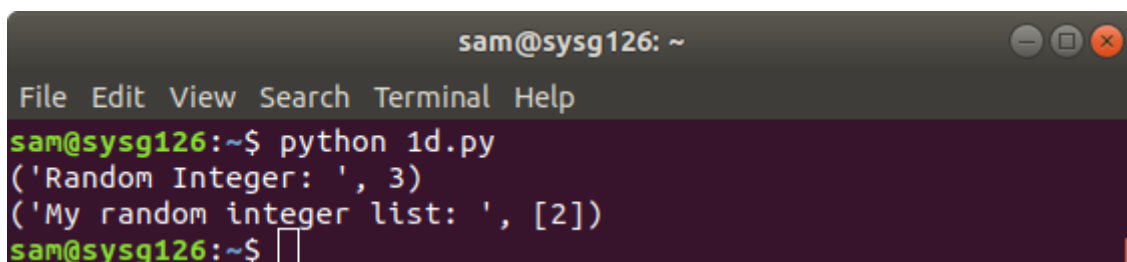
A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the execution of a Python script '1c.py'. The output is: ('BMI list: ', [29, 18, 20, 22, 19, 25, 30, 28, 22, 21, 18, 19, 20, 20, 22, 23]) followed by ('Shuffled BMI list: ', [25, 22, 30, 29, 22, 18, 28, 23, 20, 20, 19, 21, 19, 22, 20, 18]). The prompt 'sam@sysg126:~\$' is visible at the bottom.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 1c.py
('BMI list: ', [29, 18, 20, 22, 19, 25, 30, 28, 22, 21, 18, 19, 20, 20, 22, 23])
('Shuffled BMI list: ', [25, 22, 30, 29, 22, 18, 28, 23, 20, 20, 19, 21, 19, 22, 20, 18])
sam@sysg126:~$
```

1 D) Generating Random Integers using 'random.randint()'

```
print("Random Integer: ", random.randint(1,5))
random_ints_list = []
for i in range(1,50):
    n = random.randint(1,5)
    random_ints_list.append(n)
print("My random integer list: ", random_ints_list)
```

OUTPUT:

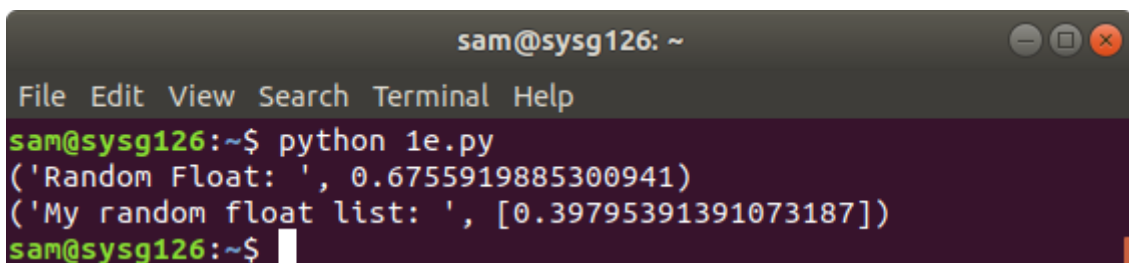
A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the execution of a Python script '1d.py'. The output is: ('Random Integer: ', 3) followed by ('My random integer list: ', [2]). The prompt 'sam@sysg126:~\$' is visible at the bottom.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 1d.py
('Random Integer: ', 3)
('My random integer list: ', [2])
sam@sysg126:~$
```

1 E) Generating Random Floating Point Values

```
print("Random Float: ", random.random())
random_float_list = []
for i in range(1,5):
    n = random.random()
    random_float_list.append(n)
print("My random float list: ", random_float_list)
```

OUTPUT:

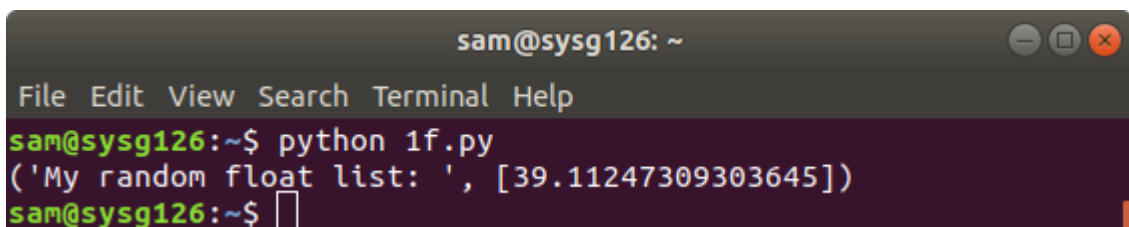
A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'sam@sysg126:~\$'. The user enters 'python 1e.py'. The output is: ('Random Float: ', 0.6755919885300941) and ('My random float list: ', [0.39795391391073187]). The prompt returns to 'sam@sysg126:~\$' with a cursor.

```
sam@sysg126:~$ python 1e.py
('Random Float: ', 0.6755919885300941)
('My random float list: ', [0.39795391391073187])
sam@sysg126:~$
```

Scale the random float numbers by multiplying our random number by 500

```
random_float_list = []
for i in range(1,5):
    n = random.random()*500
    random_float_list.append(n)
print("My random float list: ", random_float_list)
```

OUTPUT:

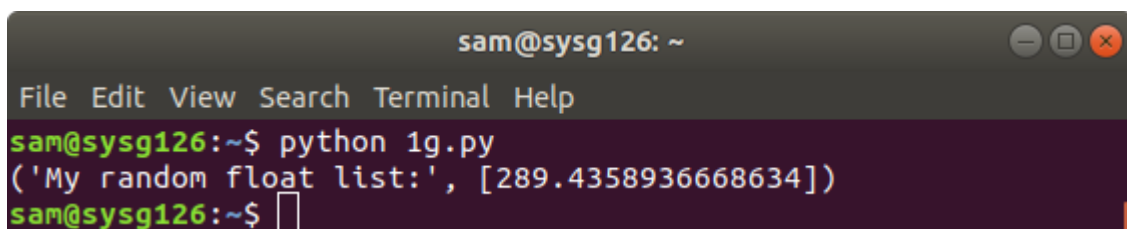
A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'sam@sysg126:~\$'. The user enters 'python 1f.py'. The output is: ('My random float list: ', [39.11247309303645]). The prompt returns to 'sam@sysg126:~\$' with a cursor.

```
sam@sysg126:~$ python 1f.py
('My random float list: ', [39.11247309303645])
sam@sysg126:~$
```

Add a lower bound as well add a conditional statement before appending and generate random numbers between 100 and 500

```
random_float_list = []  
for i in range(1,10):  
    n = random.random()*500  
    if n>=100.0:  
        random_float_list.append(n)  
print("My random float list: ", random_float_list)
```

OUTPUT:

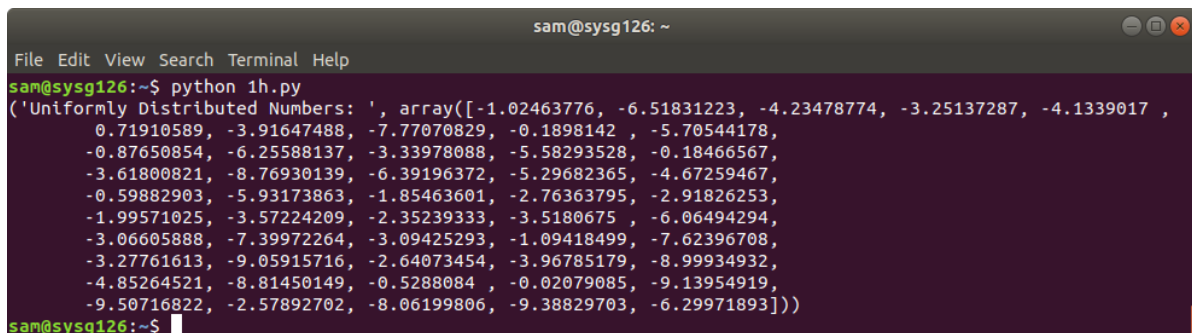


```
sam@sysg126: ~  
File Edit View Search Terminal Help  
sam@sysg126:~$ python 1g.py  
( 'My random float list:', [289.4358936668634])  
sam@sysg126:~$
```

1F) Computing Uniformly Distributed Numbers with ‘random.uniform()’

```
import numpy as np  
uniform_list = np.random.uniform(-10,1,50)  
print("Uniformly Distributed Numbers: ", uniform_list)
```

OUTPUT:

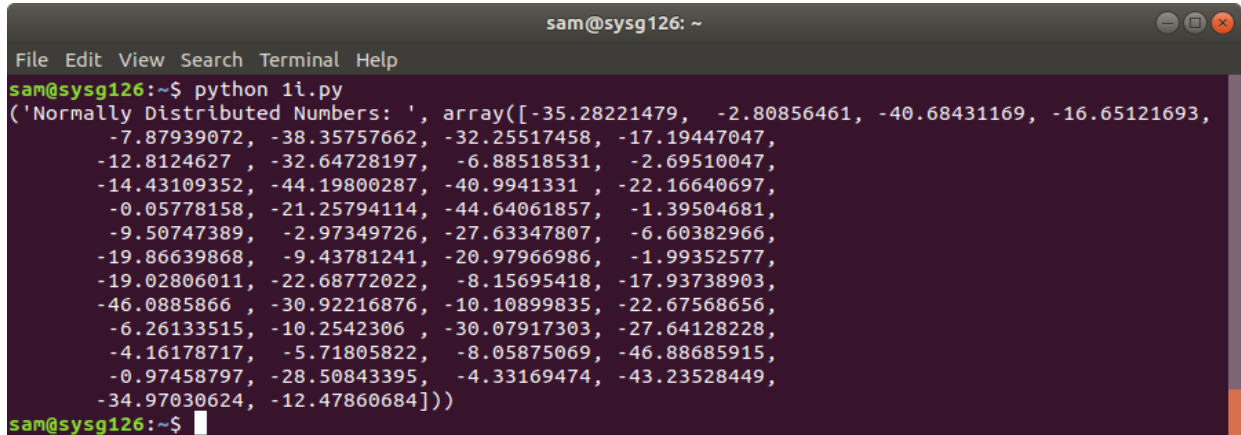


```
sam@sysg126: ~  
File Edit View Search Terminal Help  
sam@sysg126:~$ python 1h.py  
( 'Uniformly Distributed Numbers: ', array([-1.02463776, -6.51831223, -4.23478774, -3.25137287, -4.1339017 ,  
 0.71910589, -3.91647488, -7.77070829, -0.1898142 , -5.70544178,  
 -0.87650854, -6.25588137, -3.33978088, -5.58293528, -0.18466567,  
 -3.61800821, -8.76930139, -6.39196372, -5.29682365, -4.67259467,  
 -0.59882903, -5.93173863, -1.85463601, -2.76363795, -2.91826253,  
 -1.99571025, -3.57224209, -2.35239333, -3.5180675 , -6.06494294,  
 -3.06605888, -7.39972264, -3.09425293, -1.09418499, -7.62396708,  
 -3.27761613, -9.05915716, -2.64073454, -3.96785179, -8.99934932,  
 -4.85264521, -8.81450149, -0.5288084 , -0.02079085, -9.13954919,  
 -9.50716822, -2.57892702, -8.06199806, -9.38829703, -6.29971893]))  
sam@sysg126:~$
```

1 G) Computing Normally Distributed Numbers with 'random.gauss()'

```
normal_list = np.random.uniform(-50,0,50)
print("Normally Distributed Numbers: ", normal_list)
```

OUTPUT:

A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the execution of a Python script '1i.py'. The output is a list of 50 normally distributed numbers, displayed as a single line of text wrapped across multiple lines. The numbers are: [-35.28221479, -2.80856461, -40.68431169, -16.65121693, -7.87939072, -38.35757662, -32.25517458, -17.19447047, -12.8124627, -32.64728197, -6.88518531, -2.69510047, -14.43109352, -44.19800287, -40.9941331, -22.16640697, -0.05778158, -21.25794114, -44.64061857, -1.39504681, -9.50747389, -2.97349726, -27.63347807, -6.60382966, -19.86639868, -9.43781241, -20.97966986, -1.99352577, -19.02806011, -22.68772022, -8.15695418, -17.93738903, -46.0885866, -30.92216876, -10.10899835, -22.67568656, -6.26133515, -10.2542306, -30.07917303, -27.64128228, -4.16178717, -5.71805822, -8.05875069, -46.88685915, -0.97458797, -28.50843395, -4.33169474, -43.23528449, -34.97030624, -12.47860684].

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 1i.py
('Normally Distributed Numbers: ', array([-35.28221479, -2.80856461, -40.68431169, -16.65121693,
      -7.87939072, -38.35757662, -32.25517458, -17.19447047,
      -12.8124627, -32.64728197, -6.88518531, -2.69510047,
      -14.43109352, -44.19800287, -40.9941331, -22.16640697,
      -0.05778158, -21.25794114, -44.64061857, -1.39504681,
      -9.50747389, -2.97349726, -27.63347807, -6.60382966,
      -19.86639868, -9.43781241, -20.97966986, -1.99352577,
      -19.02806011, -22.68772022, -8.15695418, -17.93738903,
      -46.0885866, -30.92216876, -10.10899835, -22.67568656,
      -6.26133515, -10.2542306, -30.07917303, -27.64128228,
      -4.16178717, -5.71805822, -8.05875069, -46.88685915,
      -0.97458797, -28.50843395, -4.33169474, -43.23528449,
      -34.97030624, -12.47860684]))
sam@sysg126:~$
```

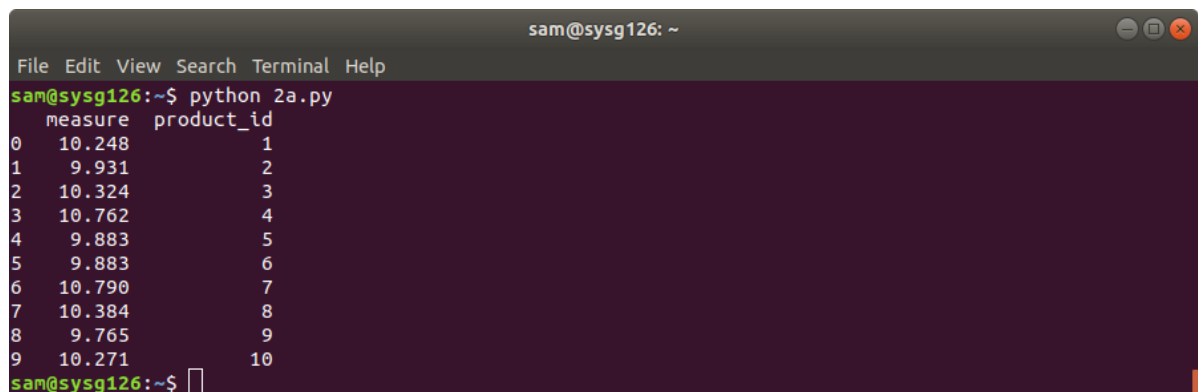

EX.NO 02

PROGRAM

2 A) Create Sample:

```
import numpy as np
import pandas as pd
np.random.seed(42)
number_of_products=10
data={'product_id':np.arange(1,number_of_products+1).tolist(),'measure':np.round(np
.random.normal(loc=10,scale=0.5,size=number_of_products),3)}
df=pd.DataFrame(data)
real_mean=round(df['measure'].mean(),3)
df
print(data)
```

OUTPUT:

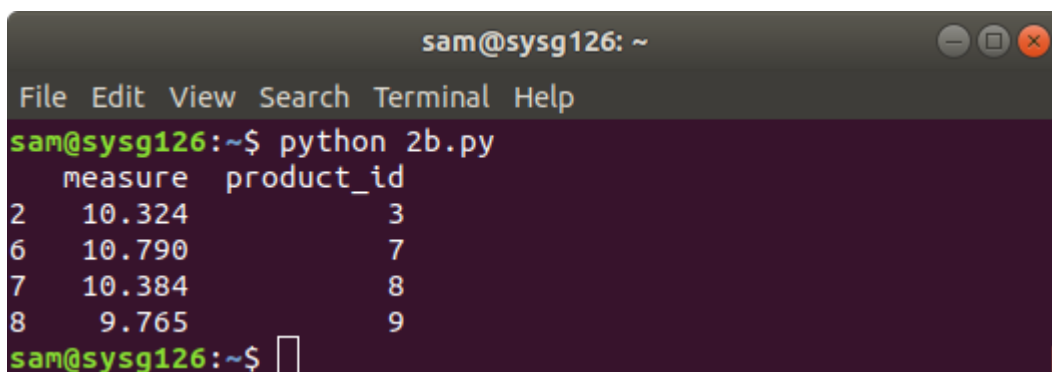


```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 2a.py
  measure  product_id
0  10.248           1
1   9.931           2
2  10.324           3
3  10.762           4
4   9.883           5
5   9.883           6
6  10.790           7
7  10.384           8
8   9.765           9
9  10.271          10
sam@sysg126:~$
```

2 B) Implement Simple Random Sampling:

```
import numpy as np
import pandas as pd
np.random.seed(42)
number_of_products=10
data={'product_id':np.arange(1,number_of_products+1).tolist(),'measure':np.round(np
.random.normal(loc=10,scale=0.5,size=number_of_products),3)}
df=pd.DataFrame(data)
simple_random_sample=df.sample(n=4).sort_values(by='product_id')
simple_random_mean=round(simple_random_sample['measure'].mean(),3)
simple_random_sample
print(simple_random_sample)
```

OUTPUT:

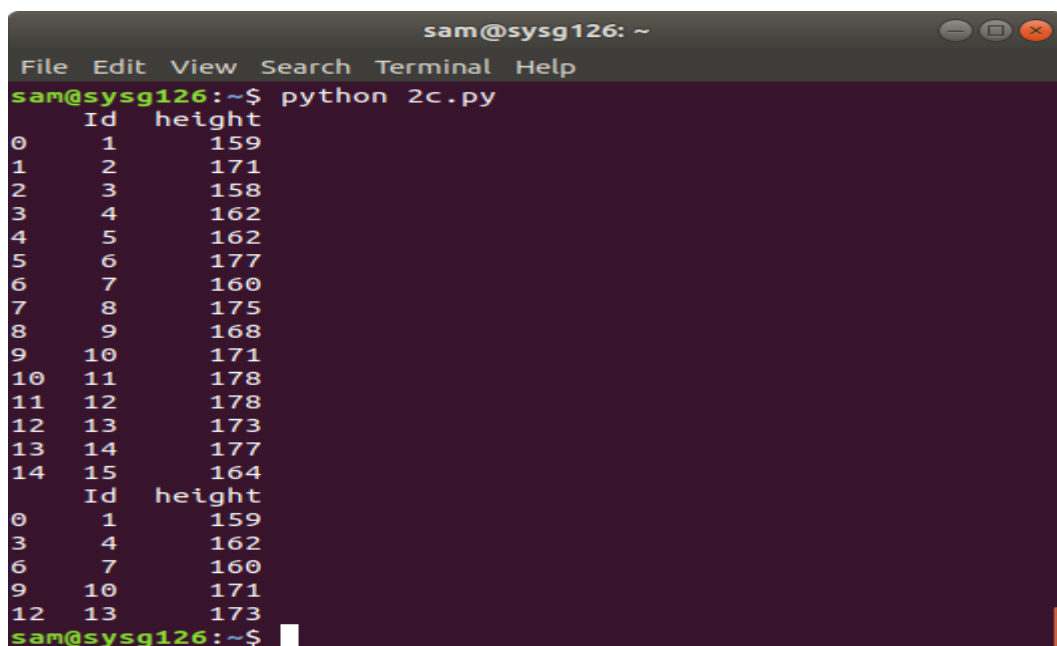


```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 2b.py
   measure  product_id
2    10.324           3
6    10.790           7
7    10.384           8
8     9.765           9
sam@sysg126:~$
```

2 C) Implement Systematic Sampling:

```
import numpy as np
import pandas as pd
number_of_students = 15
data = {'Id': np.arange(1, number_of_students+1).tolist(),
        'height': [159, 171, 158, 162, 162, 177, 160, 175,
                    168, 171, 178, 178, 173, 177, 164]}
df = pd.DataFrame(data)
print(df)
def systematic_sampling(df, step):
    indexes = np.arange(0, len(df), step=step)
    systematic_sample = df.iloc[indexes]
    return systematic_sample
systematic_sample = systematic_sampling(df, 3)
print(systematic_sample)
```

OUTPUT:



```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 2c.py
  Id  height
0   1    159
1   2    171
2   3    158
3   4    162
4   5    162
5   6    177
6   7    160
7   8    175
8   9    168
9  10    171
10  11    178
11  12    178
12  13    173
13  14    177
14  15    164
  Id  height
0   1    159
3   4    162
6   7    160
9  10    171
12  13    173
sam@sysg126:~$
```

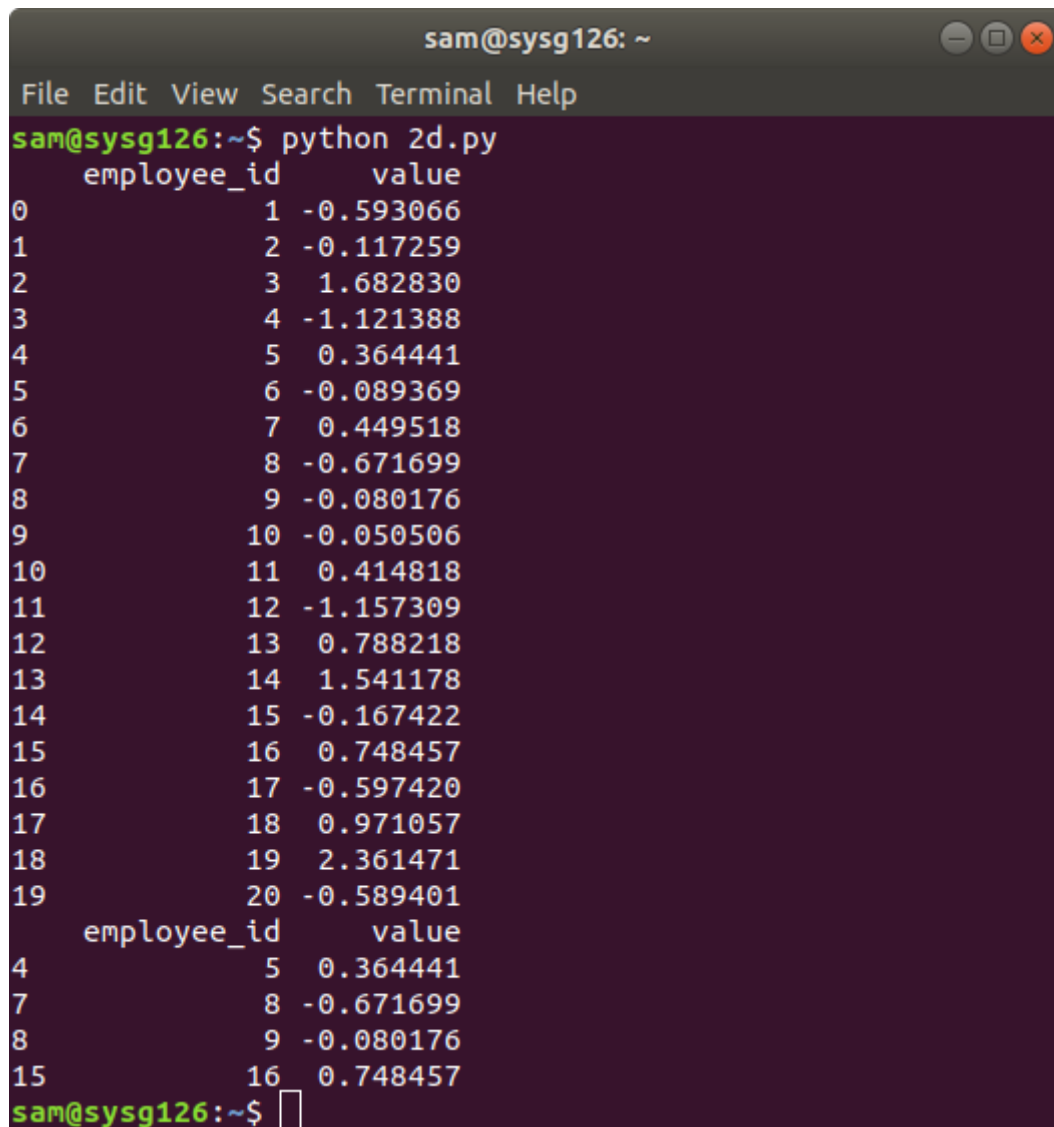
2 D) Implement Cluster Sampling:

```
import numpy as np
import pandas as pd

dic_data={"employee_id":np.arange(1,21),'value':np.random.randn(20)}
df=pd.DataFrame(dic_data)
print(df)

samples=df.sample(4).sort_values(by='employee_id')
print(samples)
```

OUTPUT:

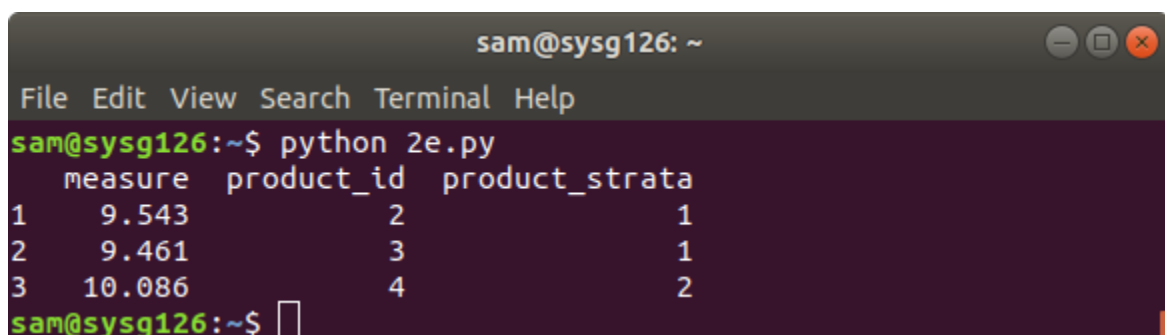


```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 2d.py
   employee_id  value
0            1 -0.593066
1            2 -0.117259
2            3  1.682830
3            4 -1.121388
4            5  0.364441
5            6 -0.089369
6            7  0.449518
7            8 -0.671699
8            9 -0.080176
9           10 -0.050506
10           11  0.414818
11           12 -1.157309
12           13  0.788218
13           14  1.541178
14           15 -0.167422
15           16  0.748457
16           17 -0.597420
17           18  0.971057
18           19  2.361471
19           20 -0.589401
   employee_id  value
4            5  0.364441
7            8 -0.671699
8            9 -0.080176
15           16  0.748457
sam@sysg126:~$
```

2 E) Implement Stratified Random Sampling:

```
import pandas as pd
import numpy as np
number_of_products=6
data = {'product_id':np.arange(1, number_of_products+1).tolist(),
        'product_strata':np.repeat([1,2], number_of_products/2).tolist(),
        'measure':np.round(np.random.normal(loc=10, scale=0.5, size=number_of_products),3)}
df = pd.DataFrame(data)
df
from sklearn.model_selection import StratifiedShuffleSplit
from sklearn.model_selection import train_test_split
split = StratifiedShuffleSplit(n_splits=1, test_size=0.4)
for x, y in split.split(df, df['product_strata']):
    stratified_random_sample = df.iloc[y].sort_values(by='product_id')
stratified_random_sample
stratified_random_sample.groupby('product_strata').mean().drop(['product_id'],axis=1)
print(stratified_random_sample)
```

OUTPUT:



The image shows a terminal window titled 'sam@sysg126: ~'. The terminal has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The command 'python 2e.py' has been executed, resulting in the following output:

	measure	product_id	product_strata
1	9.543	2	1
2	9.461	3	1
3	10.086	4	2

The terminal prompt is 'sam@sysg126:~\$' followed by a cursor.

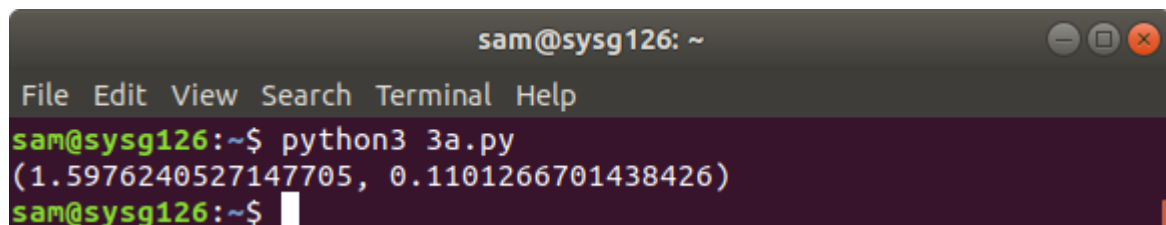
EX.NO 03

PROGRAM:

3 A) One Sample Z-Test in Python:

```
from statsmodels.stats.weightstats import ztest as ztest
data = [88, 92, 94, 94, 96, 97, 97, 97, 99, 99,
        105, 109, 109, 109, 110, 112, 112, 113, 114, 115]
ztest=ztest(data, value=100)
print(ztest)
```

OUTPUT:

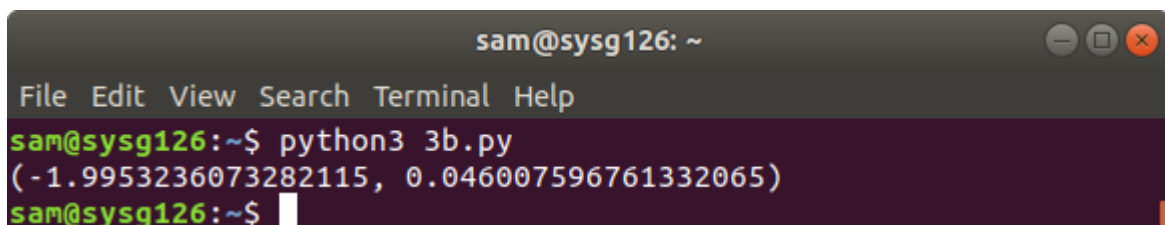
A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'sam@sysg126:~\$'. The user enters 'python3 3a.py'. The output is '(1.5976240527147705, 0.1101266701438426)'. The prompt returns to 'sam@sysg126:~\$' with a cursor.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python3 3a.py
(1.5976240527147705, 0.1101266701438426)
sam@sysg126:~$
```

3 B) Two Sample Z-Test in Python:

```
from statsmodels.stats.weightstats import ztest as ztest
cityA = [82, 84, 85, 89, 91, 91, 92, 94, 99, 99,
        105, 109, 109, 109, 110, 112, 112, 113, 114, 114]
cityB = [90, 91, 91, 91, 95, 95, 99, 99, 108, 109,
        109, 114, 115, 116, 117, 117, 128, 129, 130, 133]
ztest=ztest(cityA, cityB, value=0)
print(ztest)
```

OUTPUT:

A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'sam@sysg126:~\$'. The user enters 'python3 3b.py'. The output is '(-1.9953236073282115, 0.046007596761332065)'. The prompt returns to 'sam@sysg126:~\$' with a cursor.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python3 3b.py
(-1.9953236073282115, 0.046007596761332065)
sam@sysg126:~$
```

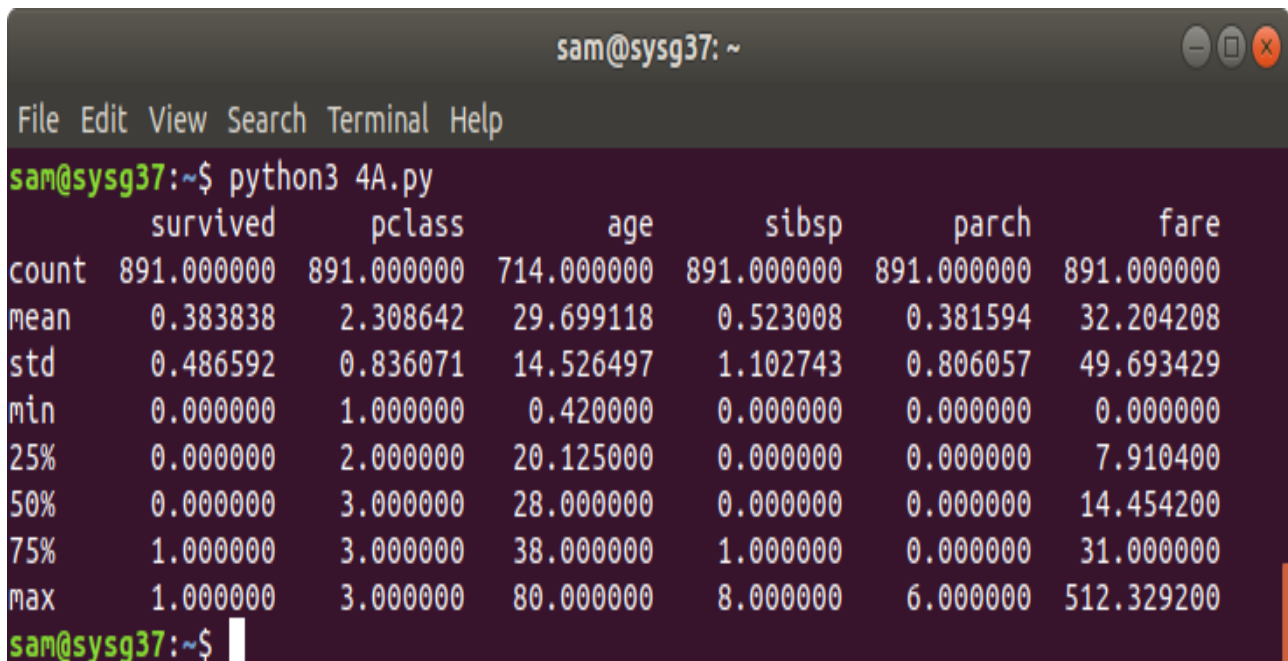
EX.NO 04

PROGRAM:

4 A) verview Of Data Set:

```
import pandas as pd
import numpy as np
import numpy.random
import seaborn as sb
titanic_trian_data=sb.load_dataset('titanic')
titanic_train_data.describe()
```

OUTPUT:



The screenshot shows a terminal window titled "sam@sysg37: ~" with a menu bar (File, Edit, View, Search, Terminal, Help). The command "python3 4A.py" has been executed, resulting in the following output:

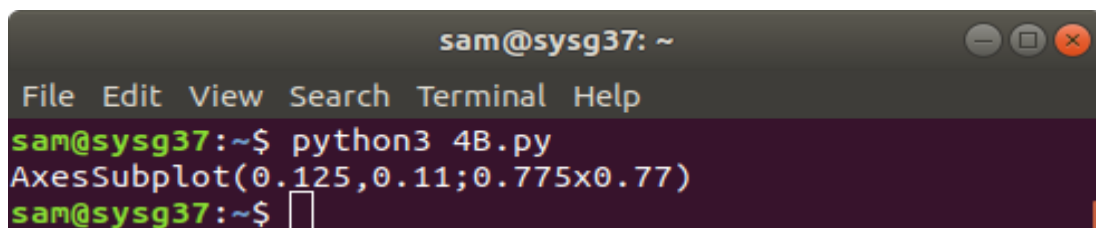
	survived	pclass	age	sibsp	parch	fare
count	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

The terminal prompt "sam@sysg37:~\$" is visible at the bottom of the window.

4 B) Titanic Train Data:

```
import pandas as pd
import numpy as np
import seaborn as sb
titanic_train_data=sb.load_dataset('titanic')
temp=titanic_train_data[titanic_train_data['age'].notna()].age
print(sb.histplot(temp))
```

OUTPUT:

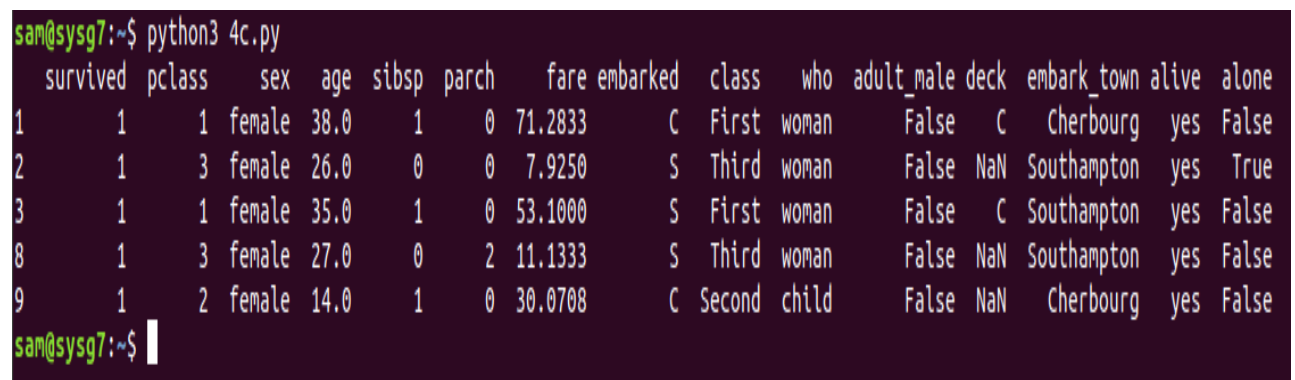


```
sam@sysg37: ~
File Edit View Search Terminal Help
sam@sysg37:~$ python3 4B.py
AxesSubplot(0.125,0.11;0.775x0.77)
sam@sysg37:~$
```

4 C) Survived Data:

```
survived_data = titanic_train_data(titanic_train_data['Survived']==1]
survived_data[survived_data['Age'].notna ()].head()
```

OUTPUT:



```
sam@sysg7:~$ python3 4c.py
survived  pclass  sex  age  sibsp  parch  fare embarked  class  who  adult_male  deck  embark_town  alive  alone
1         1      1  female  38.0    1    0  71.2833      C  First  woman      False   C   Cherbourg  yes  False
2         1      3  female  26.0    0    0   7.9250      S  Third  woman      False  NaN  Southampton  yes   True
3         1      1  female  35.0    1    0  53.1000      S  First  woman      False   C   Southampton  yes  False
8         1      3  female  27.0    0    2  11.1333      S  Third  woman      False  NaN  Southampton  yes  False
9         1      2  female  14.0    1    0  30.0708      C  Second  child      False  NaN   Cherbourg  yes  False
sam@sysg7:~$
```


4D) Sample List

```
sample_list = []
    for i in range (60):
        sample_list_temp = np.random.choice
(survived_data[survived_data['Age'].notna()].Age,60) .mean()
        sample_list.append(sample_list_temp)
sns.distplot (sample_list)
```

OUTPUT:

```
sam@sysg7:~$ python3 4d.py
AxesSubplot(0.125,0.11;0.775x0.77)
sam@sysg7:~$
```

4 E) Z test

```
from statsmodels.stats.weightstats import ztest
from statsmodels.stats.weightstats import zconfint

ztest Score, p_value= ztest (sample_list, value = mean_h0, alternative='larger')
print('p_value',p_value)
print('ztest Score',ztest Score)
if (p_value < 0.05): #The smaller the p-value, the stronger the evidence to reject the
H0
    print ()
    print("Reject H0 with", (1-p_value) *100, '% level of confidence');
    print ()
else:
    print("Fail to reject H0");
    print()
```

OUTPUT:

```
sam@sysg7:~$ python3 4e.py
p_value 0.026519184595012825
ztest Score 1.9346083940075893

Reject H0 with 97.34808154049873 % level of confidence

sam@sysg7:~$ python3 4e.py
p_value 0.4001628388995997
ztest Score 0.2529256372170385
Fail to reject H0
```

4 F) survived male data

```
sns.distplot(survived_male_data[survived_male_data.Age.notna()]).Age)
```

OUTPUT:

```
sam@sysg7:~$ python3 4f.py
survived  pclass  sex  age  sibsp  parch  fare embarked  class  who  adult_male  deck  embark_town  alive  alone
0         0       3  male  22.0    1     0   7.2500         S  Third  man        True   NaN  Southampton  no  False
4         0       3  male  35.0    0     0   8.0500         S  Third  man        True   NaN  Southampton  no  True
5         0       3  male  NaN     0     0   8.4583         Q  Third  man        True   NaN  Queenstown  no  True
6         0       1  male  54.0    0     0  51.8625         S  First  man        True    E  Southampton  no  True
7         0       3  male   2.0    3     1  21.0750         S  Third  child       False  NaN  Southampton  no  False
sam@sysg7:~$
```

4 G) Sample List male Using For Range (60)

```
sample_list_male = []
for i in range(60):
    sample_list_temp = np.random.choice(
        survived_male_data[survived_male_data.Age.notna()].Age, 60).Mean()
    sample_list_male.append(sample_list_temp)
sns.distplot(sample_list_male)
```

OUTPUT:

```
sam@sysg7:~$ python3 4g.py
AxesSubplot(0.125,0.11;0.775x0.77)
sam@sysg7:~$
```

4 H) survived_female_data

```
sns.distplot(survived_female_data[survived_female_data.Age.notna()]).Age)
```

OUTPUT:

```
sam@sysg7:~$ python3 4h.py
survived  pclass  sex  age  sibsp  parch  fare embarked  class  who  adult_male  deck  embark_town  alive  alone
1         1       1  female  38.0    1     0  71.2833         C  First  woman       False    C  Cherbourg  yes  False
2         1       3  female  26.0    0     0   7.9250         S  Third  woman       False  NaN  Southampton  yes  True
3         1       1  female  35.0    1     0  53.1000         S  First  woman       False    C  Southampton  yes  False
8         1       3  female  27.0    0     2  11.1333         S  Third  woman       False  NaN  Southampton  yes  False
9         1       2  female  14.0    1     0  30.0708         C  Second  child       False  NaN  Cherbourg  yes  False
sam@sysg7:~$
```

4 I) Sample List Female Using For Range (60)

```
sample_list_female = [ ]
    for i in range(60):
        sample_list_temp = np . random . choice
( survived_female_data[survived_female_date.Age . notna ()] . Age,60). Mean()
        sample_list_female.append(sample_list_temp)
sns.distplot (sample_list_female)
```

OUTPUT:

```
sam@sysg7:~$ python3 4i.py
AxesSubplot(0.125,0.11;0.775x0.77)
sam@sysg7:~$
```

4 J) Z test

```
from statsmodels.stats.weightstats import ztest
from statsmodels.stats.weightstats import zconfint

Ztest_Score, p_value = ztest(x1=sample_list_male, x2 sample_list_female, value = 0,
alternatives two-sided"]
    print('p_value,p_value)
    print('ztest Score',ztest Score)
    if (p_value < 0.05): #The smaller the p-value, the stronger the evidence to reject the H0
        print()
        print("Reject H0 with", (1-p_value)*100,' %level of confidence');
        print()
    else:
        print("Fail to reject H0 ");
        print()
        lower, upper = zconfint (x1=sample_list_male, value = 0,alpha=0.05, alternative = 'two-
sided")
        print("with 95% considence interval we can say thta, Passenger Mean age is between,
lower,"and", upper)
        lower, upper = zconfint(x1=sample_list_female, value = 0,alpha 0.05, alternative="two-
sided")
        print("With 95% considence interval we can say thta, Passenger Mean age is between",
lower, "and", upper)
```

OUTPUT:

```
sam@sysg7:~$ python3 4j.py
p_value nan
ztest Score nan
Fail to reject H0

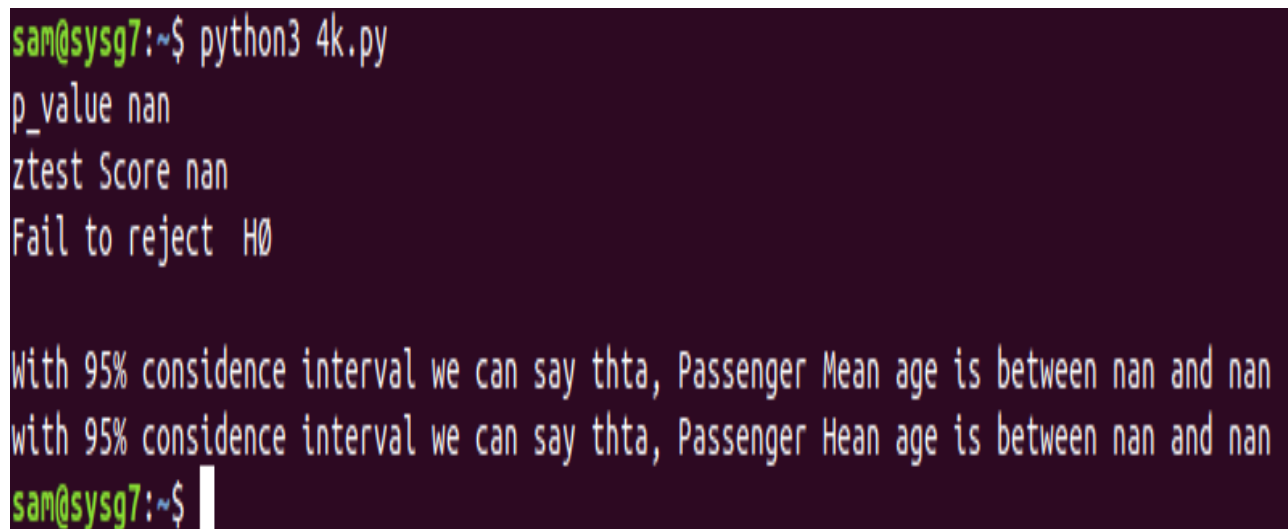
with 95% considence interval we can say thta, Passenger Mean age is between nan and nan
With 95% considence interval we can say thta, Passenger Mean age is between nan and nan
sam@sysg7:~$
```

4 K) z test

```
from statsmodels.stats.weightstats import ztest
from statsmodels.stats.weightstats import zconfint

ztest_score,p_value=ztest(x1=sample_list_male,x2=sample_list_female,value=0,
alternative='larger')
    print('p_value,p_value)
    print('ztest Score',ztest Score)
    if (p_value < 0.05): #The smaller the p-value, the stronger the evidence to reject
the HO
        print()
        print("Reject HØ with", (1-p_value)*100,'%level of confidence');
        print()
else:
    print("Fail to reject HØ");
    print()
    lower, upper = zconfint(x1=sample_list_male, value = 0,alpha=0.05,
alternative='two-sided')
    print("With 95% considence interval we can say thta, Passenger Mean age is
between", lower,"and", upper)
    lower, upper = zconfint (x1=sample_list_female, value = 0,alpha=0.05,
alternatives=' two-sided')
    print("with 95% considence interval we can say thta, Passenger Hean age is
between", lower, "and", upper)
```

OUTPUT:



```
sam@sysg7:~$ python3 4k.py
p_value nan
ztest Score nan
Fail to reject HØ

With 95% considence interval we can say thta, Passenger Mean age is between nan and nan
with 95% considence interval we can say thta, Passenger Hean age is between nan and nan
sam@sysg7:~$
```

4 L) One Proportion Z-Test

```
from statsmodels.stats.weightstats import ztest
from statsmodels.stats.weightstats import zconfint
from statsmodels.stats.proportion import proportions_ztest
from statsmodels.stats.proportion import proportion_confint

p = 0.50
ztestScore,p_value=proportions_ztest(count=153,nobs=survived_data.Age_range.count(),value=p, alternative='large')

print("Below is the calculation using Library")
print('p_value,p_value')
print('ztest Score',ztest Score)
if (p_value < 0.05): #The smaller the p-value, the stronger the evidence to reject the H0
    print("Reject H0 with", (1-p_value)*100, "%level of confidence");
    print()
else:
    print("Fail to reject H0 ");
    print()
z_score= ((153/290)-0.5)/(math.sqrt((0.5+0.5)/714))
print("z_score,z_score")
lower,upper=proportion_confint(count=153,nobs=survived_data.Age_range.count(),method="normal")
print (lower *100, "% to", upper*100, "% of passengers who survived lie in the range of 20-40 age ")
```

OUTPUT:

```
sam@sysg7:~$ python3 4l.py
Below is the calculation using Library
p_value 0.17335611403977724
ztest Score 0.9409856206480031
Fail to reject H0

z_score 0.7371249222558574
47.01273352614269 % to", upper*100, "% of passengers who survived lie in the range of 20-40 age
sam@sysg7:~$
```

4 M) One Proportion Z-Test

```
from statsmodels.stats.weightstats import ztest
from statsmodels.stats.weightstats import zconfint
from statsmodels.stats.proportion import proportions_ztest
from statsmodels.stats.proportion import proportion_confint

P = 0.5

Ztest_Score, p_value = proportions_ztest(count=385, nobs=titanic_train_data.
Age_range.count(), value=p, alternative='large')

print("Below is the calculation using Library")
print('p_value', p_value)
print('ztest Score', ztest_Score)
if (p_value < 0.05): #The smaller the p-value, the stronger the evidence to reject the H0
print("Reject H0 with %. (1-p_value)*100, '% level of confidence");
print()
else:
    print("Fail to reject H0 ");
    print()
z_score = (0.539-0.5)/(math.sqrt((0.5*0.5)/714))
print("score", z_score)
Lower, upper = proportion_confint(count=385, nobs=titanic_train_data.Age_range.count(),
method="normal")
print(lower*100, "% to", upper*100, "% of passengers lie in range of 20-40 age ")
```

OUTPUT:

```
sam@sysg7:~$ python3 4m.py
Below is the calculation using Library
p_value 0.017766864459442252
ztest Score 2.1022231945651932
Reject H0 with 98.2233145577668008 % level of confidence

score 2.084220717678434
50.26537584422187 % to 57.57776141068008 % of passengers lie in range of 20-40 age
sam@sysg7:~$
```

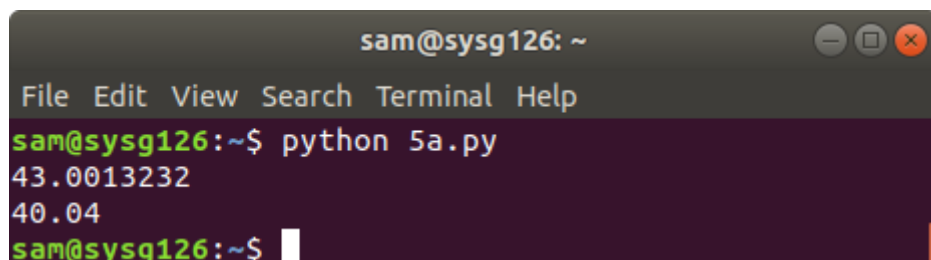
EX.NO 05

PROGRAM:

5 A) ONE-SAMPLE T-TEST:

```
import numpy as np
from scipy import stats
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=1500000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=1000000)
population_ages=np.concatenate((population_ages1,population_ages2))
minnesota_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
minnesota_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
minnesota_ages=np.concatenate((minnesota_ages1,minnesota_ages2))
print(population_ages.mean())
print(minnesota_ages.mean())
```

OUTPUT:

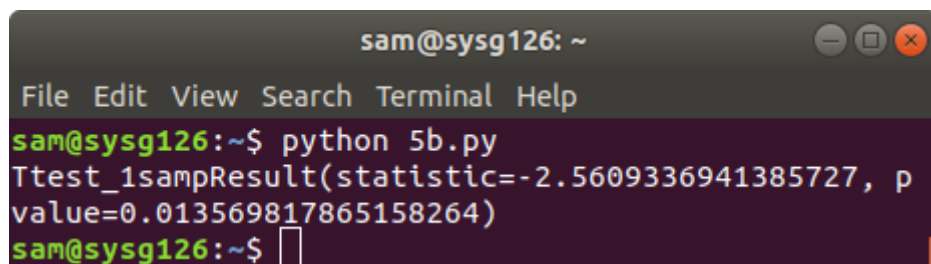
A screenshot of a terminal window titled 'sam@sysg126: ~'. The window has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the execution of a Python script '5a.py'. The output consists of two lines: '43.0013232' and '40.04'. The prompt 'sam@sysg126:~\$' is visible at the bottom, followed by a cursor.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 5a.py
43.0013232
40.04
sam@sysg126:~$
```

5 B)

```
import numpy as np
from scipy import stats
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=1500000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=1000000)
population_ages=np.concatenate((population_ages1,population_ages2))
minnesota_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
minnesota_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
minnesota_ages=np.concatenate((minnesota_ages1,minnesota_ages2))
sa=stats.ttest_1samp(a=minnesota_ages,popmean=population_ages.mean())
print(sa)
```

OUTPUT:

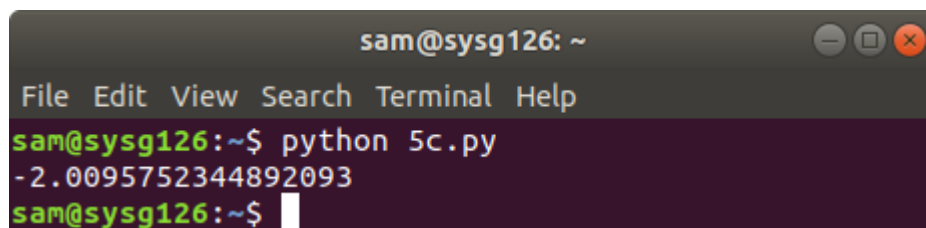
A terminal window titled 'sam@sysg126: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the command 'python 5b.py' being executed, followed by the output 'Ttest_1sampResult(statistic=-2.5609336941385727, p value=0.013569817865158264)'. The prompt 'sam@sysg126:~\$' is visible at the bottom with a cursor.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 5b.py
Ttest_1sampResult(statistic=-2.5609336941385727, p
value=0.013569817865158264)
sam@sysg126:~$
```


5 C)

```
import numpy as np
from scipy import stats
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=1500000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=1000000)
population_ages=np.concatenate((population_ages1,population_ages2))
minnesota_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
minnesota_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
minnesota_ages=np.concatenate((minnesota_ages1,minnesota_ages2))
sa=stats.t.ppf(q=0.025, df=49)
print(sa)
```

OUTPUT:

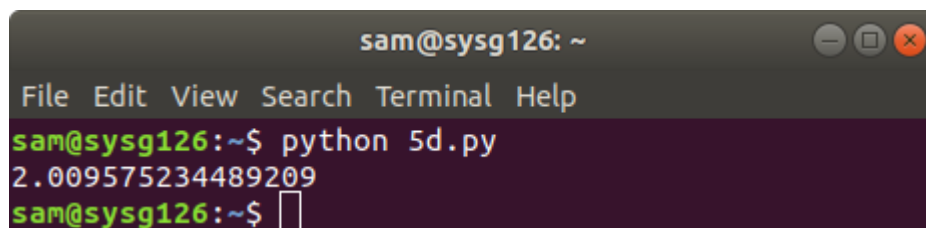
A terminal window titled 'sam@sysg126: ~' with standard window controls. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the command 'python 5c.py' being executed, which outputs the value '-2.0095752344892093'. The prompt 'sam@sysg126:~\$' is visible again on the next line.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 5c.py
-2.0095752344892093
sam@sysg126:~$
```

5 D)

```
import numpy as np
from scipy import stats
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=1500000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=1000000)
population_ages=np.concatenate((population_ages1,population_ages2))
minnesota_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
minnesota_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
minnesota_ages=np.concatenate((minnesota_ages1,minnesota_ages2))
sa=stats.t.ppf(q=0.975, df=49)
print(sa)
```

OUTPUT:

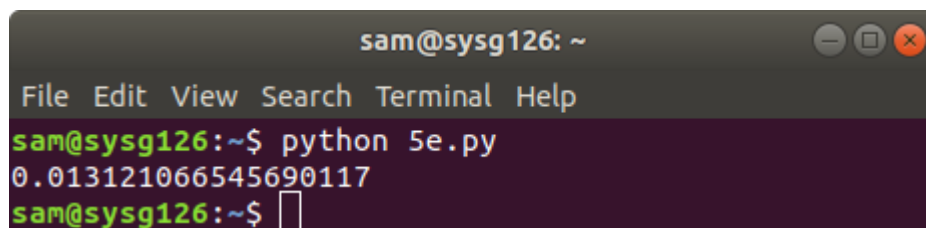
A terminal window titled 'sam@sysg126: ~' with standard window controls. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the command 'python 5d.py' being executed, which outputs the value '2.009575234489209'. The prompt 'sam@sysg126:~\$' is shown again with a cursor.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 5d.py
2.009575234489209
sam@sysg126:~$
```

5 E)

```
import numpy as np
from scipy import stats
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=1500000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=1000000)
population_ages=np.concatenate((population_ages1,population_ages2))
minnesota_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
minnesota_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
minnesota_ages=np.concatenate((minnesota_ages1,minnesota_ages2))
sa=stats.t.cdf(x=-2.5742, df=49)*2
print(sa)
```

OUTPUT:

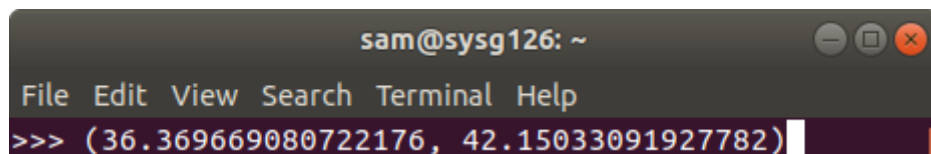
A screenshot of a terminal window titled 'sam@sysg126: ~'. The terminal has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The prompt is 'sam@sysg126:~\$'. The user has entered 'python 5e.py', and the output is '0.013121066545690117'. The prompt is now 'sam@sysg126:~\$' with a cursor.

```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 5e.py
0.013121066545690117
sam@sysg126:~$
```

5 F)

```
import numpy as np
from scipy import stats
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=1500000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=1000000)
population_ages=np.concatenate((population_ages1,population_ages2))
minnesota_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
minnesota_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
minnesota_ages=np.concatenate((minnesota_ages1,minnesota_ages2))
sa=stats.t.interval(0.95, df=49, loc=minnesota_ages.mean(), scale=sigma)
print(sa)
```

OUTPUT:

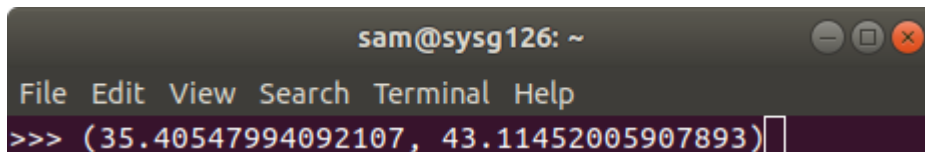
A screenshot of a terminal window with a dark background. The title bar at the top reads "sam@sysg126: ~" and includes standard window control buttons (minimize, maximize, close). Below the title bar is a menu bar with the options "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows a command prompt ">>>" followed by the output "(36.369669080722176, 42.15033091927782)" in a light-colored font. A white cursor is positioned at the end of the output line.

```
sam@sysg126: ~
File Edit View Search Terminal Help
>>> (36.369669080722176, 42.15033091927782)
```

5 G)

```
import numpy as np
from scipy import stats
population_ages1=stats.poisson.rvs(loc=18,mu=35,size=1500000)
population_ages2=stats.poisson.rvs(loc=18,mu=10,size=1000000)
population_ages=np.concatenate((population_ages1,population_ages2))
minnesota_ages1=stats.poisson.rvs(loc=18,mu=30,size=30)
minnesota_ages2=stats.poisson.rvs(loc=18,mu=10,size=20)
minnesota_ages=np.concatenate((minnesota_ages1,minnesota_ages2))
sa=stats.t.interval(alpha=0.99, df=49, loc=minnesota_ages.mean(), scale=sigma)
print(sa)
```

OUTPUT:

A screenshot of a terminal window with a dark background. The title bar at the top reads "sam@sysg126: ~" and includes standard window control buttons (minimize, maximize, close). Below the title bar is a menu bar with the options "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows the command prompt ">>>" followed by the output "(35.40547994092107, 43.11452005907893)" in a light-colored font. A cursor is visible at the end of the output line.

```
sam@sysg126: ~
File Edit View Search Terminal Help
>>> (35.40547994092107, 43.11452005907893)
```

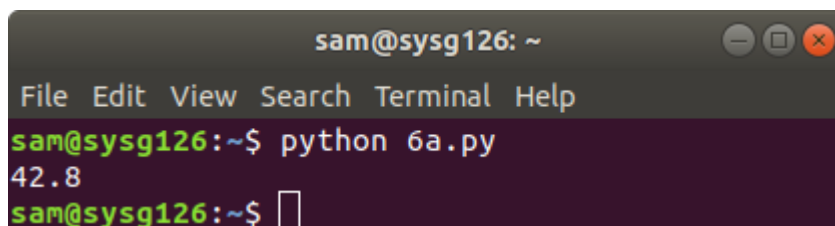
EX.NO 06

PROGRAM

6 A) Two-Sample T-Test:

```
import pandas as pd
import numpy as np
from scipy import stats
from scipy.stats import norm
import matplotlib.pyplot as plt
np.random.seed(12)
wisconsin_ages1 = stats.poisson.rvs(loc=18, mu=33, size=30)
wisconsin_ages2 = stats.poisson.rvs(loc=18, mu=13, size=20)
wisconsin_ages = np.concatenate((wisconsin_ages1, wisconsin_ages2))
print(wisconsin_ages.mean() )
stats.ttest_ind(a= minnesota_ages,
                b= wisconsin_ages,
                equal_var=False) # Assume samples have equal variance?
Ttest_indResult(statistic=-1.7083870793286842, pvalue=0.09073104343957748)
```

OUTPUT:



```
sam@sysg126: ~
File Edit View Search Terminal Help
sam@sysg126:~$ python 6a.py
42.8
sam@sysg126:~$
```

6 B) Paired T-Test:

```
np.random.seed(11)

before= stats.norm.rvs(scale=30, loc=250, size=100)
```

```
after = before + stats.norm.rvs(scale=5, loc=-1.25, size=100)

weight_df = pd.DataFrame({"weight_before":before,
                           "weight_after":after,
                           "weight_change":after-before})

weight_df.describe()

stats.ttest_rel(a = before, b = after)

Ttest_relResult(statistic=2.5720175998568284, pvalue=0.011596444318439857)
```

OUTPUT:

sam@sysg126: ~

File Edit View Search Terminal Help

sam@sysg126:~\$ python 6b.py

	weight_after	weight_before	weight_change
0	305.605006	302.483642	3.121364
1	240.526071	241.417810	-0.891739
2	226.017788	235.463046	-9.445258
3	165.913930	170.400443	-4.486513
4	252.590309	249.751461	2.838848
5	239.345033	240.411059	-1.066026
6	232.407606	233.901119	-1.493513
7	267.169736	259.462080	7.707655
8	272.390803	262.631521	9.759282
9	216.596576	218.031911	-1.435335
10	231.827337	223.412810	8.414527
11	224.510138	235.727995	-11.217858
12	259.195183	270.690469	-11.495286
13	269.921041	266.835765	3.085276
14	208.273739	210.833545	-2.559805
15	218.060598	216.415742	1.644856
16	273.437884	272.105122	1.332762
17	295.542004	297.239022	-1.697018
18	251.228396	249.067747	2.160649
19	229.000211	229.496601	-0.496390
20	273.958178	282.868891	-8.910713
21	240.557044	240.712701	-0.155657
22	271.450883	271.772567	-0.321684
23	304.386032	296.472149	7.913883
24	262.007923	268.902395	-6.894472
25	251.039782	252.204797	-1.165016
26	268.595996	271.968141	-3.372144
27	228.808189	230.722738	-1.914549
28	251.703228	244.657205	7.046023
29	233.424567	232.781363	0.643203
..
70	262.348372	265.482521	-3.134149
71	267.011699	263.661786	3.349913


```

sam@sysg126: ~
File Edit View Search Terminal Help
70      262.348372      265.482521      -3.134149
71      267.011699      263.661786      3.349913
72      264.668473      267.810586      -3.142113
73      257.420313      261.115190      -3.694877
74      285.889373      290.361342      -4.471969
75      282.723969      280.478264      2.245704
76      267.985387      267.858354      0.127033
77      229.197578      229.515952      -0.318373
78      221.644174      228.593202      -6.949028
79      192.210338      192.751314      -0.540976
80      262.165774      270.619370      -8.453597
81      201.442277      195.309712      6.132564
82      274.465647      276.373414      -1.907767
83      305.113092      305.390946      -0.277854
84      216.374622      218.150368      -1.775746
85      220.295647      229.465460      -9.169813
86      227.604697      235.713566      -8.108868
87      273.517721      274.909313      -1.391592
88      230.311782      224.099425      6.212357
89      239.257570      246.083756      -6.826186
90      229.530301      234.307371      -4.777070
91      240.975529      242.461692      -1.486163
92      279.051839      288.738242      -9.686403
93      211.337144      221.073854      -9.736711
94      248.845524      252.152793      -3.307270
95      256.671813      258.148189      -1.476376
96      272.554150      275.760015      -3.205866
97      218.498901      212.077790      6.421112
98      281.375144      283.446109      -2.070965
99      263.595143      263.043310      0.551833

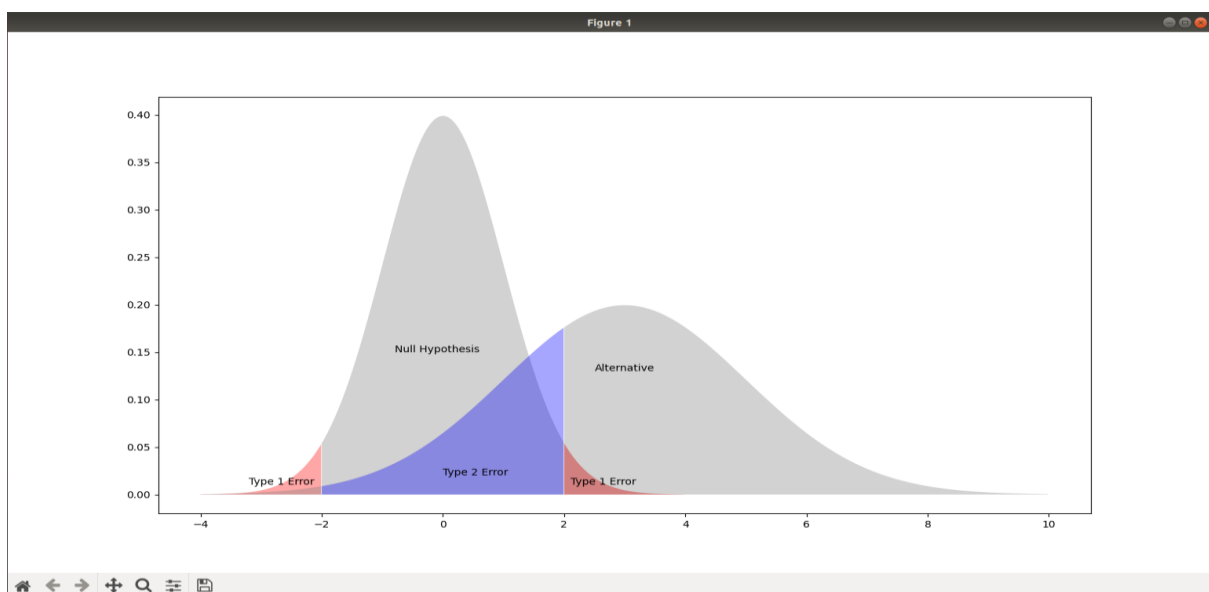
[100 rows x 3 columns]
sam@sysg126:~$
```

6 C) Type I and Type II Error:

```
import pandas as pd
import numpy as np
from scipy import stats
from scipy.stats import norm
import matplotlib.pyplot as plt

plt.figure(figsize=(12,10))
plt.fill_between(x=np.arange(-4,-2,0.01),y1= stats.norm.pdf(np.arange(-4,-2,0.01))
,facecolor='red',alpha=0.35)
plt.fill_between(x=np.arange(-2,2,0.01),y1= stats.norm.pdf(np.arange(-2,2,0.01))
,facecolor='grey',alpha=0.35)
plt.fill_between(x=np.arange(2,4,0.01),y1= stats.norm.pdf(np.arange(2,4,0.01))
,facecolor='red',alpha=0.5)
plt.fill_between(x=np.arange(-4,-2,0.01),y1= stats.norm.pdf(np.arange(-4,-2,0.01),loc=3,
scale=2) ,facecolor='grey',alpha=0.35)
plt.fill_between(x=np.arange(-2,2,0.01),y1= stats.norm.pdf(np.arange(-2,2,0.01),loc=3,
scale=2) ,facecolor='blue',alpha=0.35)
plt.fill_between(x=np.arange(2,10,0.01),y1= stats.norm.pdf(np.arange(2,10,0.01),loc=3,
scale=2),facecolor='grey',alpha=0.35)
plt.text(x=-0.8, y=0.15, s= "Null Hypothesis")
plt.text(x=2.5, y=0.13, s= "Alternative")
plt.text(x=2.1, y=0.01, s= "Type 1 Error")
plt.text(x=-3.2, y=0.01, s= "Type 1 Error")
plt.text(x=0, y=0.02, s= "Type 2 Error")
plt.show()
```

OUTPUT:



EX.NO 07

PROGRAM:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
a=[25,25,27,30,23,20]
b=[30,30,21,24,26,28]
c=[18,30,29,29,24,26]
list_of_tuples = list(zip(a, b,c))
df = pd.DataFrame(list_of_tuples, columns = ['A', 'B', 'C'])
df
m1=np.mean(a)
m2=np.mean(b)
m3=np.mean(c)
print('Average mark for college A: {}'.format(m1))
print('Average mark for college B: {}'.format(m2))
print('Average mark for college C: {}'.format(m3))
m=(m1+m2+m3)/3
print('Overall mean: {}'.format(m))
SSb=6*((m1-m)**2+(m2-m)**2+(m3-m)**2)
print('Between-groups Sum of Squared Differences: {}'.format(SSb))
MSb=SSb/2
print('Between-groups Mean Square value: {}'.format(MSb))
err_a=list(a-m1)
err_b=list(b-m2)
err_c=list(c-m3)
err=err_a+err_b+err_c
ssw=[]
for i in err:
    ssw.append(i**2)
SSw=np.sum(ssw)
print('Within-group Sum of Squared Differences: {}'.format(SSw))
MSw=SSw/15
print('Within-group Mean Square value: {}'.format(MSw))
F=MSb/MSw
print('F-score: {}'.format(F))
print(stats.f_oneway(a,b,c))
```

OUTPUT:

```
sam@sysg126: ~  
File Edit View Search Terminal Help  
sam@sysg126:~$ python 7.py  
Average mark for college A: 25.0  
Average mark for college B: 26.5  
Average mark for college C: 26.0  
Overall mean: 25.8333333333  
Between-groups Sum of Squared Differences: 7.0  
Between-groups Mean Square value: 3.5  
Within-group Sum of Squared Differences: 223.5  
Within-group Mean Square value: 14.9  
F-score: 0.234899328859  
F_onewayResult(statistic=0.2348993288590604, pvalue=0.793504662732833)  
sam@sysg126:~$
```

EX.NO 08

PROGRAM:

```
import numpy as np
import matplotlib.pyplot as plt

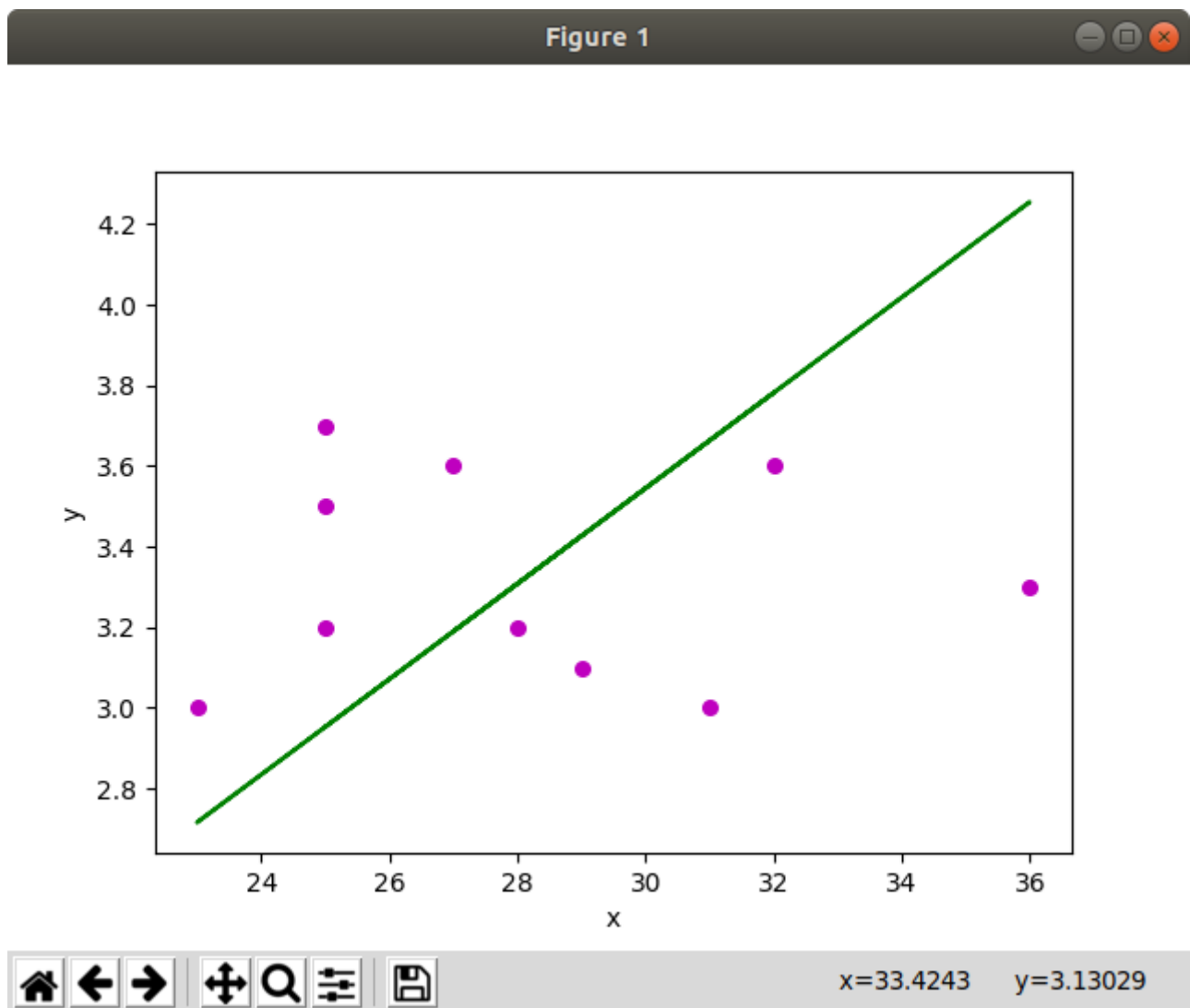
def estimate_coef(x, y):
    n = np.size(x)
    m_x, m_y = np.mean(x), np.mean(y)
    SS_xy = np.sum(y*x - n*m_y*m_x)
    SS_xx = np.sum(x*x - n*m_x*m_x)
    b_1 = SS_xy / SS_xx
    b_0 = m_y - b_1*m_x
    return(b_0, b_1)

def plot_regression_line(x, y, b):
    plt.scatter(x, y, color = "m", marker = "o", s = 30)
    y_pred = b[0] + b[1]*x
    plt.plot(x, y_pred, color = "g")
    plt.xlabel('x')
    plt.ylabel('y')
    plt.show()

x = np.array([25, 23, 25, 31, 32, 25, 36, 27, 28, 29])
y = np.array([3.2, 3, 3.5, 3, 3.6, 3.7, 3.3, 3.6, 3.2, 3.1])
b = estimate_coef(x, y)
print("Estimated coefficients:\nb_0 = {} nb_1 = {}".format(b[0], b[1]))
plot_regression_line(x, y, b)
```

OUTPUT:

```
sam@sysg126: ~  
File Edit View Search Terminal Help  
sam@sysg126:~$ python 8.py  
Estimated coefficients:  
b_0 = -0.00677659964468 nb_1 = 0.118390626322
```



EX.NO 09

PROGRAM

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics
import seaborn as sn
import matplotlib.pyplot as plt

candidates = {
    'gmat': [780, 750, 690, 710, 680, 730, 690, 720, 740, 690, 610, 690, 710, 680, 770, 610, 580, 650, 540, 590, 620, 600, 550, 550, 570, 670, 660, 580, 650, 660, 640, 620, 660, 660, 680, 650, 670, 580, 590, 690],
    'gpa': [4, 3.9, 3.3, 3.7, 3.9, 3.7, 2.3, 3.3, 3.3, 1.7, 2.7, 3.7, 3.7, 3.3, 3.3, 3.3, 2.7, 3.7, 2.7, 2.3, 3.3, 2.2, 3.2, 2.7, 3.3, 3.7, 2.3, 3.7, 3.3, 3.3, 2.7, 4, 3.3, 3.3, 2.3, 2.7, 3.3, 1.7, 3.7],
    'work_experience': [3, 4, 3, 5, 4, 6, 1, 4, 5, 1, 3, 5, 6, 4, 3, 1, 4, 6, 2, 3, 2, 1, 4, 1, 2, 6, 4, 2, 6, 5, 1, 2, 4, 6, 5, 1, 2, 1, 4, 5],
    'admitted': [1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1]
}

df = pd.DataFrame(candidates, columns= ['gmat', 'gpa', 'work_experience', 'admitted'])
print(df)

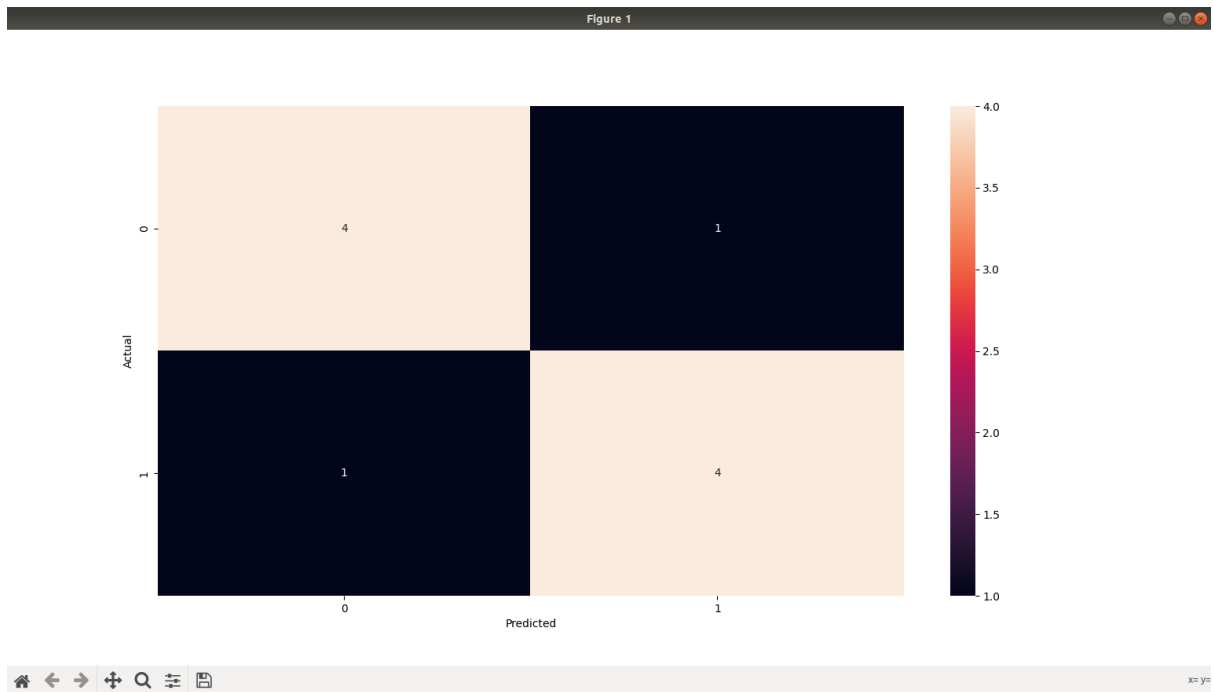
X = df[['gmat', 'gpa', 'work_experience']]
y = df['admitted']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
print(X_train)
print(y_train)

logistic_regression = LogisticRegression()
logistic_regression.fit(X_train, y_train)
y_pred = logistic_regression.predict(X_test)

confusion_matrix = pd.crosstab(y_test, y_pred, rownames=['Actual'], colnames=['Predicted'])
sn.heatmap(confusion_matrix, annot=True)
print('Accuracy: ', metrics.accuracy_score(y_test, y_pred))
print(X_test)
print(y_pred)
print('confusion_matrix:', confusion_matrix, sep='\n', end='\n\n')
plt.show()
```

OUTPUT:



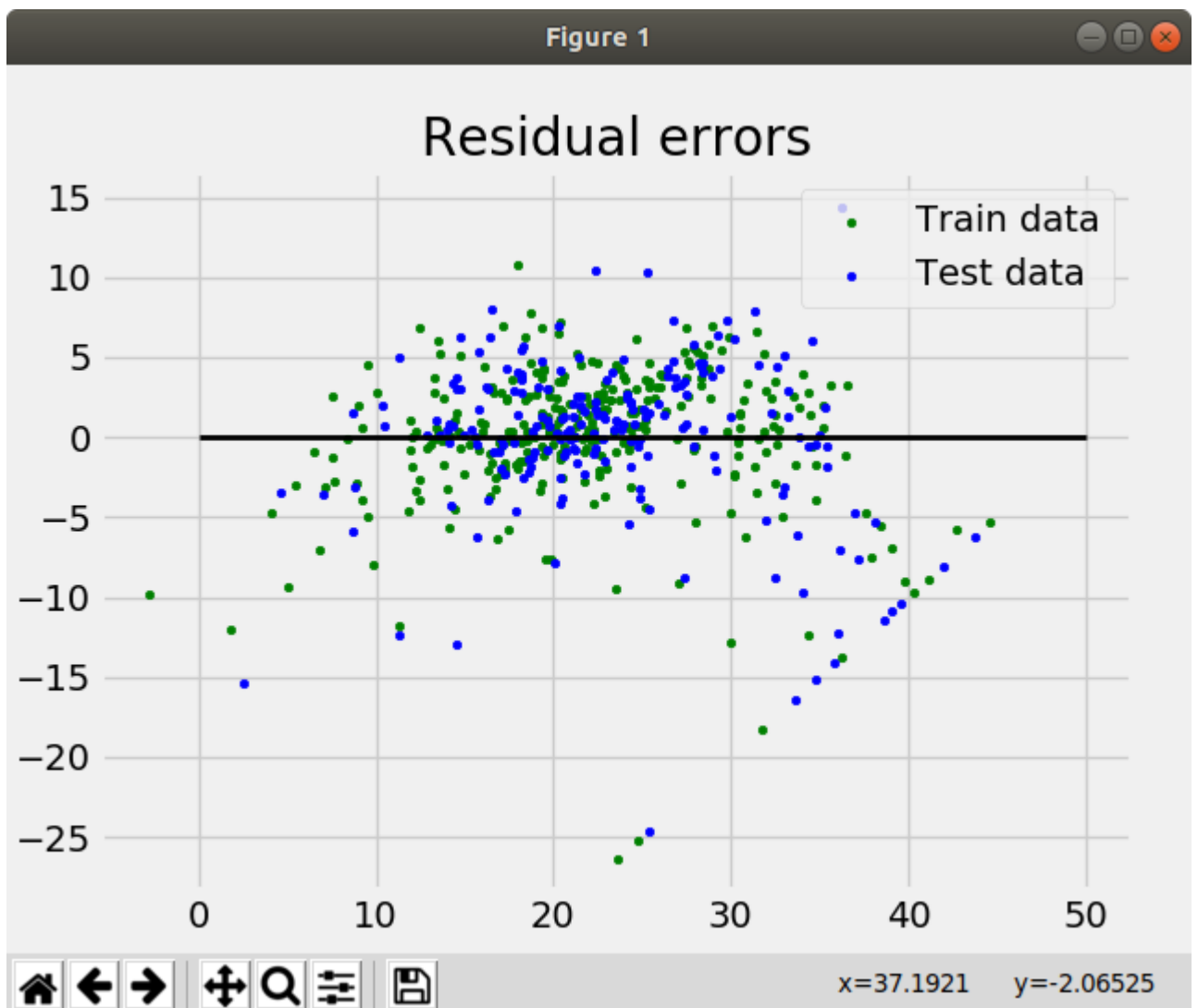
EX.NO 10

PROGRAM:

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model, metrics
boston = datasets.load_boston(return_X_y=False)
X = boston.data
y = boston.target
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=1)
reg = linear_model.LinearRegression()
reg.fit(X_train, y_train)
print('Coefficients: ', reg.coef_)
print('Variance score: {}'.format(reg.score(X_test, y_test)))
plt.style.use('fivethirtyeight')
plt.scatter(reg.predict(X_train), reg.predict(X_train) - y_train,
            color = "green", s = 10, label = 'Train data')
plt.scatter(reg.predict(X_test), reg.predict(X_test) - y_test,
            color = "blue", s = 10, label = 'Test data')
plt.hlines(y = 0, xmin = 0, xmax = 50, linewidth = 2)
plt.legend(loc = 'upper right')
plt.title("Residual errors")
plt.show()
```

OUTPUT:

```
sam@sysg126: ~  
File Edit View Search Terminal Help  
sam@sysg126:~$ python 10.py  
( 'Coefficients: ', array([-8.95714048e-02,  6.73132853e-02,  5.04649248e-02,  2.  
18579583e+00,  
    -1.72053975e+01,  3.63606995e+00,  2.05579939e-03, -1.36602886e+00,  
    2.89576718e-01, -1.22700072e-02, -8.34881849e-01,  9.40360790e-03,  
    -5.04008320e-01]))  
Variance score: 0.720905667266
```



EX. NO. 11

PROGRAM

11 A)

```
from dateutil.parser import parse
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd
plt.rcParams.update({'figure.figsize': (10, 7), 'figure.dpi': 120})
df = pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/a10.csv',
parse_dates=['date'])
df.head()
df =
pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/MarketArrivals.csv')
df = df.loc[df.market=='MUMBAI', :]
df.head().
import matplotlib.pyplot as plt
df = pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/a10.csv',
parse_dates=['date'], index_col='date')
def plot_df(df, x, y, title="", xlabel='Date', ylabel='Value', dpi=100):
    plt.figure(figsize=(16,5), dpi=dpi)
    plt.plot(x, y, color='tab:red')
    plt.gca().set(title=title, xlabel=xlabel, ylabel=ylabel)
    plt.show()
plot_df(df, x=df.index, y=df.value, title='Monthly anti-diabetic drug sales in Australia from
1992 to 2008.')
```

OUTPUT

	date	value
0	1991-07-01	3.526591
1	1991-08-01	3.180891
2	1991-09-01	3.252221
3	1991-10-01	3.611003
4	1991-11-01	3.565869

11 B)

```
df =  
pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/MarketArrivals.csv')  
df = df.loc[df.market=='MUMBAI', :]  
df.head()
```

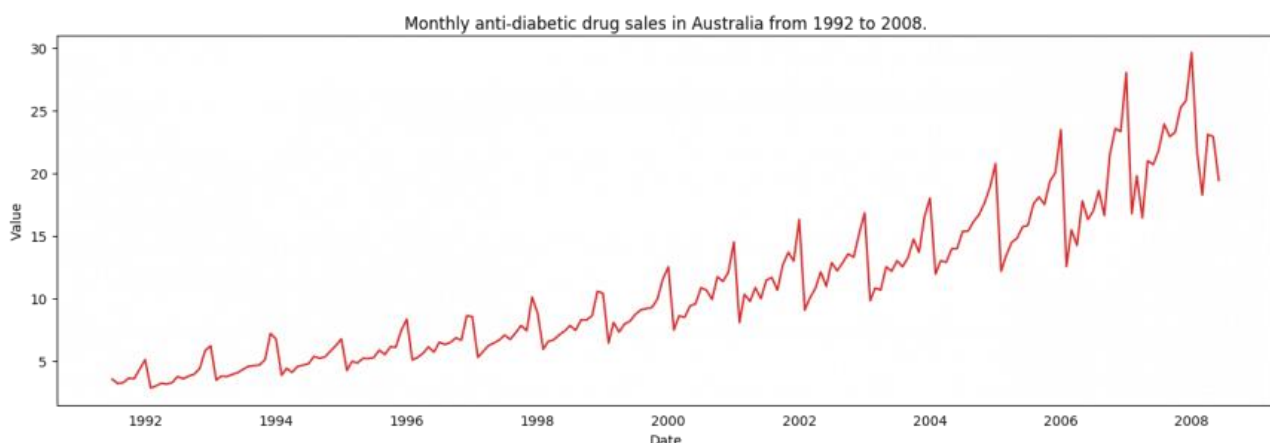
OUTPUT:

	market	month	year	quantity	priceMin	priceMax	priceMod	state	city	date
6654	MUMBAI	January	2004	267100	719	971	849	MS	MUMBAI	January-2004
6655	MUMBAI	January	2005	275845	261	513	387	MS	MUMBAI	January-2005
6656	MUMBAI	January	2006	228000	315	488	402	MS	MUMBAI	January-2006
6657	MUMBAI	January	2007	205200	866	1136	997	MS	MUMBAI	January-2007
6658	MUMBAI	January	2008	267550	348	550	448	MS	MUMBAI	January-2008

11 C)

```
import matplotlib.pyplot as plt  
df = pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/a10.csv',  
parse_dates=['date'], index_col='date')  
  
def plot_df(df, x, y, title="", xlabel='Date', ylabel='Value', dpi=100):  
    plt.figure(figsize=(16,5), dpi=dpi)  
    plt.plot(x, y, color='tab:red')  
    plt.gca().set(title=title, xlabel=xlabel, ylabel=ylabel)  
    plt.show()  
  
plot_df(df, x=df.index, y=df.value, title='Monthly anti-diabetic drug sales in Australia from  
1992 to 2008.')  
df = pd.read_csv('datasets/AirPassengers.csv', parse_dates=['date'])  
x = df['date'].values  
y1 = df['value'].values
```

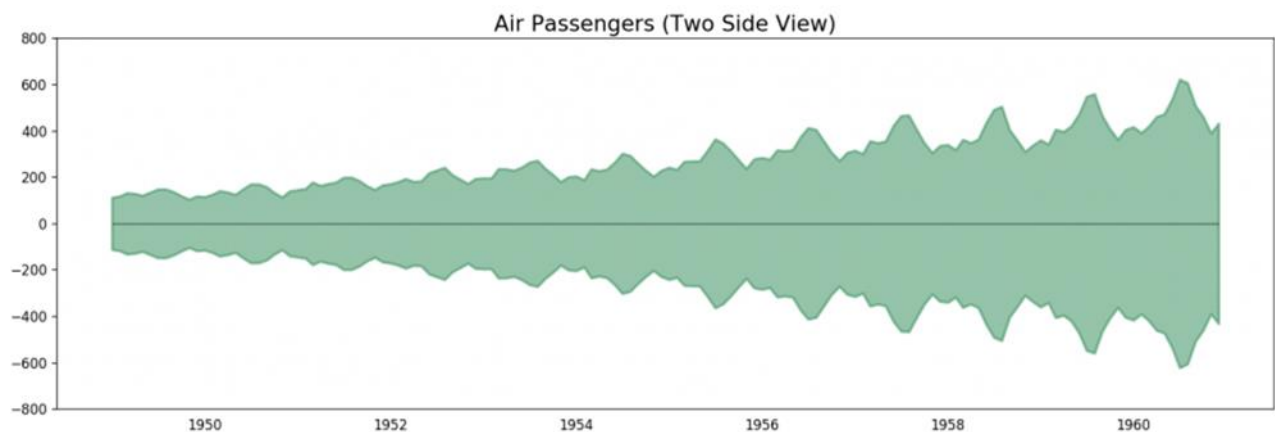
OUTPUT:



11 D) PLOT

```
fig, ax = plt.subplots(1, 1, figsize=(16,5), dpi= 120)
plt.fill_between(x, y1=y1, y2=-y1, alpha=0.5, linewidth=2, color='seagreen')
plt.ylim(-800, 800)
plt.title('Air Passengers (Two Side View)', fontsize=16)
plt.hlines(y=0, xmin=np.min(df.date), xmax=np.max(df.date), linewidth=.5)
plt.show()
```

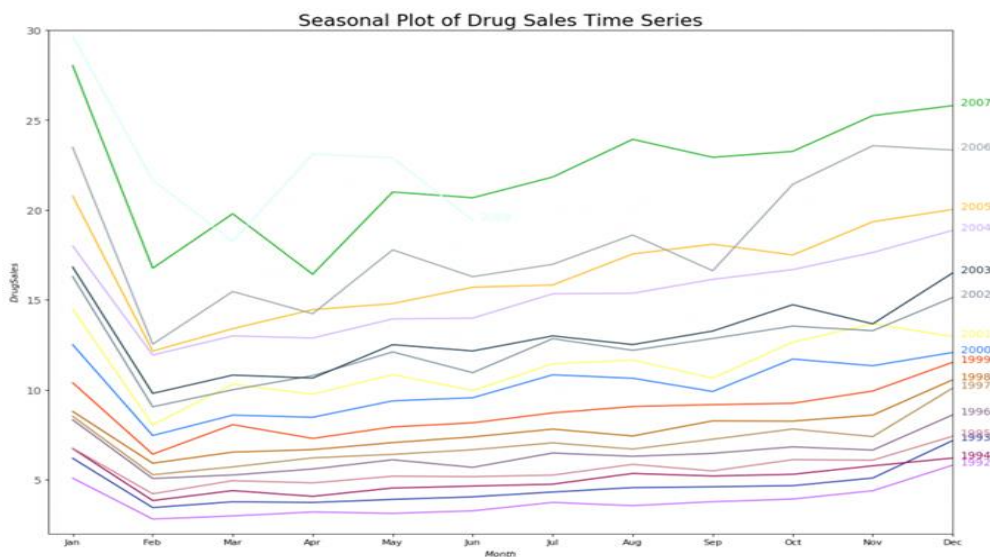
OUTPUT:



11 E)

```
df = pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/a10.csv',
parse_dates=['date'], index_col='date')
df.reset_index(inplace=True)
df['year'] = [d.year for d in df.date]
df['month'] = [d.strftime('%b') for d in df.date]
years = df['year'].unique()
np.random.seed(100)
mycolors = np.random.choice(list(mpl.colors.XKCD_COLORS.keys()), len(years),
replace=False)
plt.figure(figsize=(16,12), dpi= 80)
for i, y in enumerate(years):
    if i > 0:
        plt.plot('month', 'value', data=df.loc[df.year==y, :], color=mycolors[i], label=y)
        plt.text(df.loc[df.year==y, :].shape[0]-.9, df.loc[df.year==y, 'value'][-1:].values[0], y,
        fontsize=12, color=mycolors[i])
plt.gca().set(xlim=(-0.3, 11), ylim=(2, 30), ylabel='$Drug Sales$', xlabel='$Month$')
plt.yticks(fontsize=12, alpha=.7)
plt.title("Seasonal Plot of Drug Sales Time Series", fontsize=20)
plt.show()
```

OUTPUT:



11 F)

```
df = pd.read_csv('https://raw.githubusercontent.com/selva86/datasets/master/a10.csv',
parse_dates=['date'], index_col='date')
df.reset_index(inplace=True)
df['year'] = [d.year for d in df.date]
df['month'] = [d.strftime('%b') for d in df.date]
years = df['year'].unique()
fig, axes = plt.subplots(1, 2, figsize=(20,7), dpi= 80)
sns.boxplot(x='year', y='value', data=df, ax=axes[0])
sns.boxplot(x='month', y='value', data=df.loc[~df.year.isin([1991, 2008]), :])
axes[0].set_title('Year-wise Box Plot\n(The Trend)', fontsize=18);
axes[1].set_title('Month-wise Box Plot\n(The Seasonality)', fontsize=18)
plt.show()
```

OUTPUT:

