Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Science and Engineering (Data Science)
B.Tech. Sem: III Subject: Statistics for Data Science
Experiment 5

Date:	**Experiment Title: Confidence Interval**
Aim	To implement Confidence Interval using Python.
Software	Google Colab
Implementation	

```
import numpy as np import scipy
import statistics as st import
math import matplotlib.pyplot
as plt from scipy.stats import
norm
```

Using Python solve the following questions given below :

1. *If Z follows standard normal distribution, then find*

$(i)\ P(Z < 1.5)$ $(ii)\ P(Z > 0.5)$ $(iii)\ P(|Z| < 1.5)$ $(iv)\ P(|Z| > 0.5)$

$(v)\ P(-2.2 < Z < 1)$

Code:

```
P1 = norm.cdf(1.5)
print("P(z<1.5) = " + str(P1)) P2
= norm.sf(0.5)
print("P(Z> 0.5) = " + str(P2)) P3 =
norm.cdf(1.5) - norm.cdf(-1.5)
print("P(|Z| < 1.5) = " + str(P3)) P4
= norm.cdf(-0.5) + norm.sf(0.5)
print("P|Z > 0.5) = "+ str(P4)) P5 =
norm.cdf(1) - norm.cdf(-2.2)
print("P(-2.2 < Z < 1) = " + str(P5))
```

output:

1. `P(z<1.5) = 0.9331927987311419`
2. `P(Z> 0.5) = 0.3085375387259869`
3. `P(|Z| < 1.5) = 0.8663855974622838`
4. `P|Z > 0.5) = 0.6170750774519738`
5. `P(-2.2 < Z < 1) = 0.8274412985550443`

2. *If Z follows standard normal distribution, then find value of* Z_0 *satisfying the given equation*

$(i)\ P(Z < Z_0) = 0.90$ $(ii)\ P(Z < Z_0) = 0.95$ $(iii)\ P(Z < Z_0) = 0.99$

$(iv)\ P(|Z| < Z_0) = 0.90$ $(v)\ P(|Z| < Z_0) = 0.95$ $(vi)\ P(|Z| < Z_0) = 0.99$

Code:

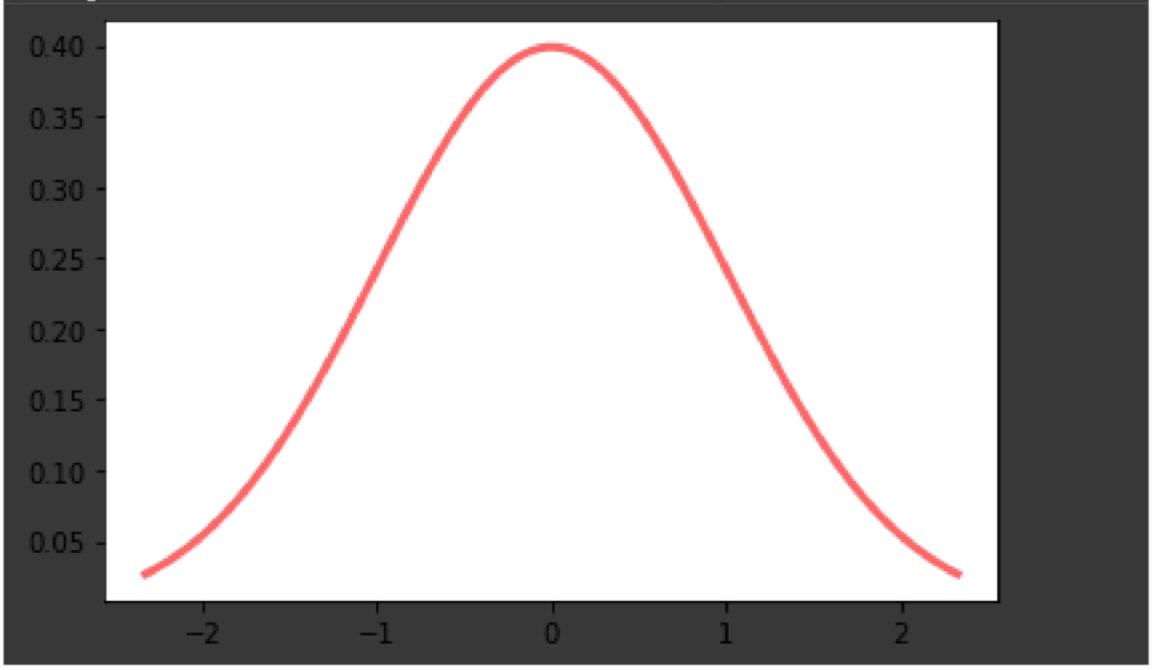
```
print("value of z_0 satsifying equation P(z< z_0) = 0.90 is " + str(norm.ppf(0.90)))
print("value of z_0 satsifying equation P(z< z_0) = 0.95 is " + str(norm.ppf(0.95)))
```

```
print("value of z_0 satsifying equation P(z< z_0) = 0.99 is " + str(norm.ppf(0.99)))
print("value of z_0 satsifying equation P(|z| < z_0) = 0.90 is " + str(norm.ppf(0.95)))
print("value of z_0 satsifying equation P(|z| < z_0) = 0.95 is " + str(norm.ppf(0.975)))
print("value of z_0 satsifying equation P(|z| < z_0) = 0.99 is " + str(norm.ppf(0.995)))

fig, ax = plt.subplots(1,1)
mean, var, skew, kurt = norm.stats(moments = 'mvsk')
print(mean, var, skew, kurt)
x = np.linspace(norm.ppf(0.01), norm.ppf(0.99), 100)
ax.plot(x, norm.pdf(x), 'r', lw=3, alpha = 0.6, label = 'norm pdf')
```

output:

```
value of z_0 satsifying equation P(z< z_0) = 0.90 is 1.2815515655446004
value of z_0 satsifying equation P(z< z_0) = 0.95 is 1.6448536269514722
value of z_0 satsifying equation P(z< z_0) = 0.99 is 2.3263478740408408
value of z_0 satsifying equation P(|z| < z_0) = 0.90 is 1.6448536269514722
value of z_0 satsifying equation P(|z| < z_0) = 0.95 is 1.959963984540054
value of z_0 satsifying equation P(|z| < z_0) = 0.99 is 2.5758293035489004
```

```
0.0 1.0 0.0 0.0
[<matplotlib.lines.Line2D at 0x7f5cacd7dc50>]
```



3. *If t follows students t distribution, then find*

(i) $P(t < 1.5)$ *with* $d.o.f. = 20$ (ii) $P(t > 0.5)$ *with* $d.o.f. = 15$

(iii) $P(|t| < 1.5)$ *with* $d.o.f. = 25$ (iv) $P(|t| > 0.5)$ *with* $d.o.f. = 35$

(v) $P(-2.2 < t < 1)$ *with* $d.o.f. = 42$

Code:

```
from scipy.stats import t
P1 = t.cdf(1.5, 20)
```

```
print("With df = 20, P(t < 1.5) = " + str(P1))
P2 = t.sf(0.5, 15)
print("With df = 15, P(t > 0.5) = " + str(P2))
P3 = t.cdf(1.5, 25) - t.cdf(-1.5, 25)
print("With df = 25, P(t < 1.5) = " + str(P3))
P4 = t.cdf(-0.5, 35) + t.sf(0.5, 35)
print("With df = 35, P(|t| > 0.5) = " + str(P4))
P5 = t.cdf(1, 42) - t.cdf(-2.2, 42)
print("With df = 42, P(-2.2 < t < 1) = " + str(P5))
```

output:

1. `With df = 20, P(t < 1.5) = 0.9253821144153737`
2. `With df = 15, P(t > 0.5) = 0.3121650567600378`
3. `With df = 25, P(t < 1.5) = 0.8538615348619807`
4. `With df = 35, P(|t| > 0.5) = 0.6202043032354958`
5. `With df = 42, P(-2.2 < t < 1) = 0.8218007171327162`

4. *If t follows students t distribution, then find value of* t_0 *satisfying the given equation*

(i) $P(t < t_0) = 0.90$ *with d.o.f.* $= 20$ (ii) $P(t < t_0) = 0.95$ *with d.o.f.* $= 15$

(iii) $P(t < t_0) = 0.99$ *with d.o.f.* $= 25$ (iv) $P(|t| < t_0) = 0.90$ *with d.o.f.* $= 30$

(v) $P(|t| < t_0) = 0.95$ *with d.o.f.* $= 42$ (vi) $P(|t| < t_0) = 0.99$ *with d.o.f.* $= 10$

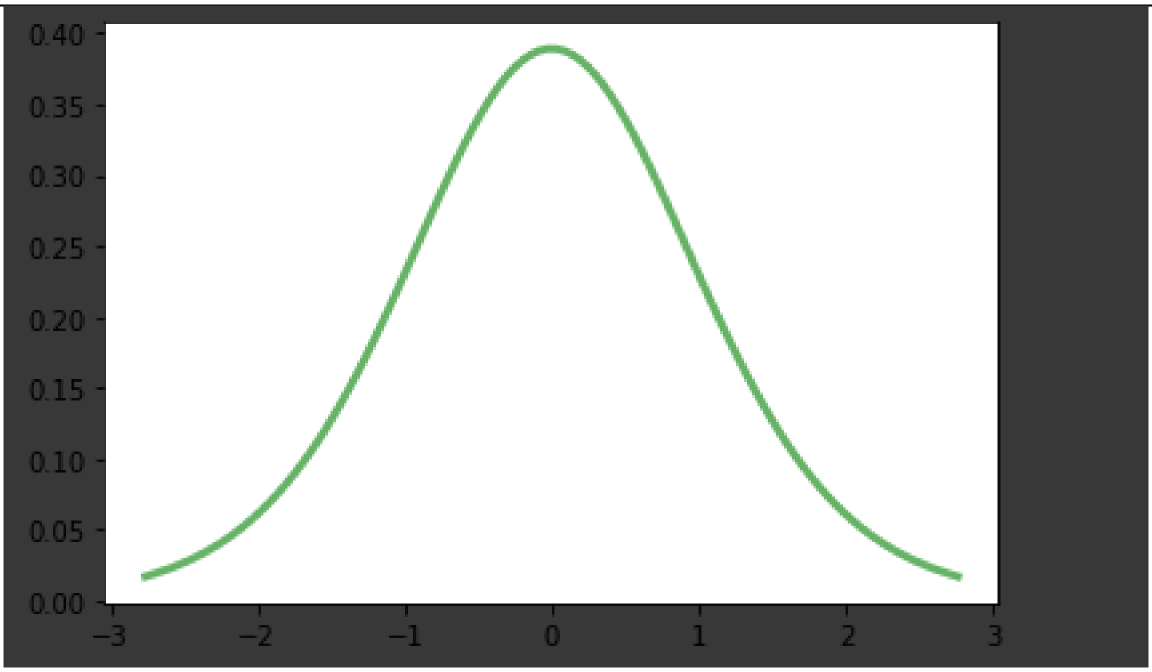
Code:

```
print("with df = 20 value of t_0 satisfisfying equation P(t < t_0) = 90 is " + str(t.ppf(0.90, 20)))

print("with df = 15 value of t_0 satisfisfying equation P(t < t_0) = 95 is " + str(t.ppf(0.95, 15)))

print("with df = 25 value of t_0 satisfisfying equation P(t < t_0) = 99 is " + str(t.ppf(0.99, 25)))

print("with df = 30 value of t_0 satisfisfying equation P(|t| < t_0) = 90 is " + str(t.ppf(0.95, 30)))

print("with df = 42 value of t_0 satisfisfying equation P(|t| < t_0) = 95 is " + str(t.ppf(0.975, 42)))


fig,ax = plt.subplots(1,1)
df = 10
mean, var, skew, kurt = t.stats(df, moments ='mvsk')
print(mean, var, skew, kurt)
x = np.linspace(t.ppf(0.01, df), t.ppf(0.99, df),100)
ax.plot(x, t.pdf(x, df), 'g', lw=3, alpha = 0.6, label = 't pdf')
```

output:

```
with df = 20 value of t_0 satisfisfying equation P(t < t_0) = 90 is 1.3253407069850462
with df = 15 value of t_0 satisfisfying equation P(t < t_0) = 95 is 1.7530503556925547
with df = 25 value of t_0 satisfisfying equation P(t < t_0) = 99 is 2.4851071754106413
with df = 30 value of t_0 satisfisfying equation P(|t| < t_0) = 90 is 1.6972608943617378
with df = 42 value of t_0 satisfisfying equation P(|t| < t_0) = 95 is 2.018081697095881
with df = 10 value of t_0 satisfisfying equation P(|t| < t_0) = 99 is 3.169272667175838
```

```
0.0 1.25 0.0 1.0
[<matplotlib.lines.Line2D at 0x7f5cacec1590>]
```



5. *If F follows Snedecor's F distribution, then find*

(i) $P(F < 1.5)$ *with* $df_1 = 5, df_2 = 14$ (ii) $P(F > 2.5)$ *with* $df_1 = 15, df_2 = 14$

(iii) $P(0.5 < F < 4.1)$ *with* $df_1 = 13, df_2 = 17$

Code:

```
from scipy.stats import f
P1 = f.cdf(1.5, 5, 14)
print("With df1 = 5, df2 = 14, P(F < 1.5) = " + str(P1))
P2 = f.sf(2.5, 15, 14)
print("With df1 = 15, df2 = 14, P(F > 2.5) = " + str(P2))
P3 = f.cdf(4.1, 13, 17) - f.cdf(0.5, 13, 17)
print("With df1 = 13, df2 = 17, P(0.5 < F < 4.1) = " + str(P3))
```

output:

1. `With df1 = 5, df2 = 14, P(F < 1.5) = 0.7480582329156877`
2. `With df1 = 15, df2 = 14, P(F > 2.5) = 0.04734368894060450 6`
3. `With df1 = 13, df2 = 17, P(0.5 < F < 4.1) = 0.8911203850553827`

6. *If F follows Snedecor's F distribution, then find value of* F_0 *satisfying the given equation*

(i) $P(F < F_0) = 0.90$ *with* $df_1 = 5, df_2 = 14$

(ii) $P(F < F_0) = 0.95$ *with* $df_1 = 15, df_2 = 13$

(iii) $P(F < F_0) = 0.99$ *with* $df_1 = 25, df_2 = 28$

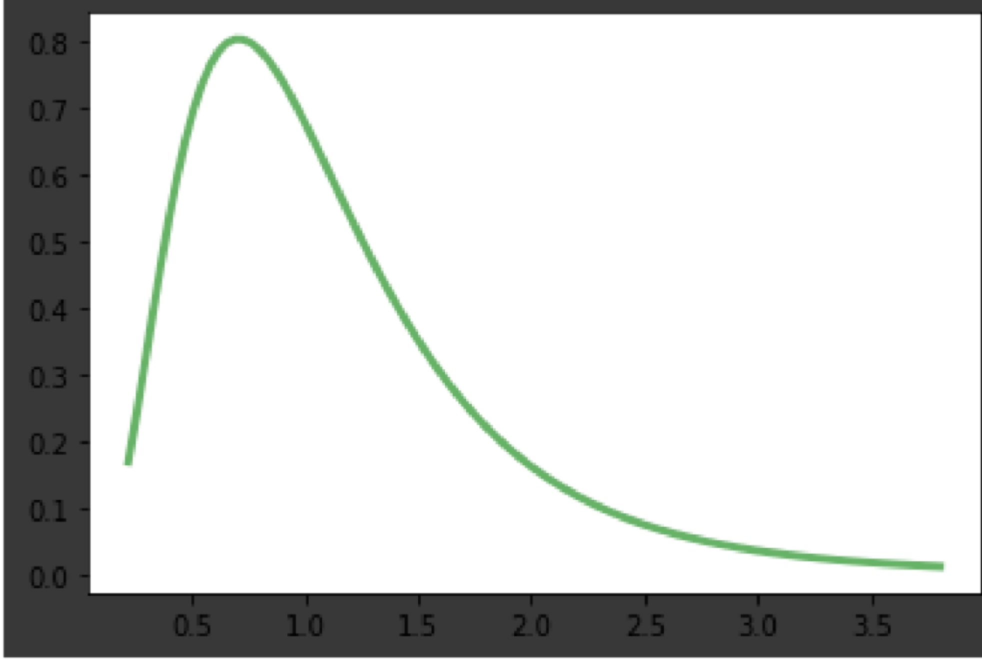
Code:

```
print("with df1 = 5, df2 = 14 value of f_0 satisfisfying equation P(F < F_0) = 0.90 is " + str(f.ppf(0.90, 5, 14)))
print("with df1 = 15, df2 = 13 value of f_0 satisfisfying equation P(F < F_0) = 0.95 is " + str(f.ppf(0.95, 15, 13)))
print("with df1 = 25, df2 = 28 value of f_0 satisfisfying equation P(F < F_0) = 0.99 is " + str(f.ppf(0.99, 25, 28)))
print("with df1 = 8, df2 = 29 value of F_0 satisfying equation P(F < F_0) = 0.99 is " + str(f.ppf(0.99,8,29)))

fig, ax = plt.subplots(1,1)
dfn, dfd = 10, 15
mean, var, skew, kurt = f.stats(dfn, dfd, moments ='mvsk')
print(mean, var, skew, kurt)
x = np.linspace(f.ppf(0.01, dfn, dfd), f.ppf(0.99, dfn, dfd),100)
ax.plot(x, f.pdf(x, dfn, dfd), 'g', lw=3, alpha = 0.6, label = 'F pdf')
```

output:

```
with df1 = 5, df2 = 14 value of f_0 satisfisfying equation P(F < F_0) = 0.90 is 2.3069430514007236
with df1 = 15, df2 = 13 value of f_0 satisfisfying equation P(F < F_0) = 0.95 is 2.533109983130745
with df1 = 25, df2 = 28 value of f_0 satisfisfying equation P(F < F_0) = 0.99 is 2.5060172667359417

with df1 = 8, df2 = 29 value of F_0 satisfying equation P(F < F_0) = 0.99 is 3.198218844688683
```

```
1.1538461538461537 0.556750941366326 2.268030280365984
11.634782608695652
[<matplotlib.lines.Line2D at 0x7f5caca2a1d0>]
```



7. *If X follows χ^2 − distribution, then find*

$(i)\ P(X < 1.5)\ with\ df = 10$ $\qquad (ii)\ P(X > 2.5)\ with\ df = 5$

$(iii)\ P(0.5 < X < 4.1)\ with\ df = 2$

Code:

```
from scipy.stats import chi2
P1 = chi2.cdf(1.5, 10)
print("With df = 10, P(X < 1.5) = " + str(P1))
```

```
P2 = chi2.sf(2.5, 5)
print("With df = 5, P(X > 2.5) = " + str(P2)) P3 =
chi2.cdf(4.1, 2) - chi2.cdf(0.5, 2) print("With df
= 2, P(0.5 < X < 4.1) = " + str(P3))
```

Code:

output:

1. `With df = 10, P(X < 1.5) = 0.0010646777727857928`
2. `With df = 5, P(X > 2.5) = 0.7764950711233227`
3. `With df = 2, P(0.5 < X < 4.1) = 0.6500658794836006`

8. *If X follows* $\chi^2 -$ *distribution, then find value of* X_0 *satisfying the given equation*

$(i)\ P(X < X_0) = 0.90\ with\ df = 1$ $\quad (ii)\ P(X < X_0) = 0.95\ with\ df = 3$

$(iii)\ P(X < X_0) = 0.99\ with\ df = 2$ $\quad (iv)\ P(X < X_0) = 0.05\ with\ df = 1$

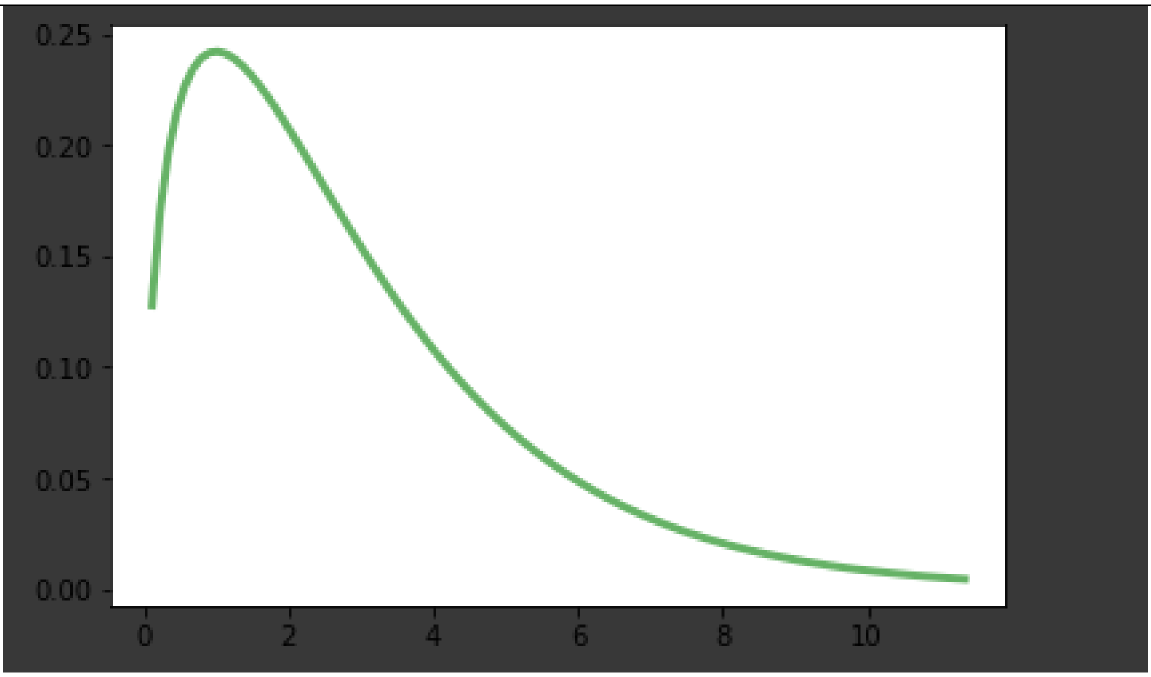
$(v)\ P(X < X_0) = 0.025\ with\ df = 3$ $\quad (vi)\ P(X < X_0) = 0.005\ with\ df = 2$

```
print("with df = 1 value of x_0 satisfisfying equation P(x < x_0) = 90 is " + str(chi2.ppf(0.90, 1))) print("with
df = 3 value of x_0 satisfisfying equation P(x < x_0) = 95 is " + str(chi2.ppf(0.95, 3))) print("with df = 2
value of x_0 satisfisfying equation P(x < x_0) = 99 is " + str(chi2.ppf(0.99, 2))) print("with df = 1 value of
x_0 satisfisfying equation P(x < x_0) = 0.05 is " + str(chi2.ppf(0.05, 1))) print("with df = 3 value of x_0
satisfisfying equation P(x < x_0) = 0.025 is " + str(chi2.ppf(0.025, 3))) print("with df = 2 value of x_0
satisfisfying equation P(x < x_0) = 0.005 is " + str(chi2.ppf(0.005, 2)))


fig,ax = plt.subplots(1,1) df = 3 mean, var, skew, kurt =
chi2.stats(df, moments ='mvsk') print(mean, var, skew,
kurt) x = np.linspace(chi2.ppf(0.01, df), chi2.ppf(0.99,
df),100)
ax.plot(x, chi2.pdf(x, df), 'g', lw=3, alpha = 0.6, label = 'chi2 pdf')
```

output:

```
with df = 1 value of x_0 satisfisfying equation P(x < x_0) = 90 is 2.705543454095404
with df = 3 value of x_0 satisfisfying equation P(x < x_0) = 95 is 7.814727903251179
with df = 2 value of x_0 satisfisfying equation P(x < x_0) = 99 is 9.21034037197618 with
df = 1 value of x_0 satisfisfying equation P(x < x_0) = 0.05 is
0.003932140000019522
with df = 3 value of x_0 satisfisfying equation P(x < x_0) = 0.025 is
0.2157952826238978
```

```
3.0 6.0 1.632993161855452 4.0
[<matplotlib.lines.Line2D at 0x7f5cac5b3490>]
```



9. *Construct a* 95 % *confidence interval for population mean in an experiment that found the sample mean temperature for a certain city in August was* 101.82, *with a population standard deviation of* 1.2.*There were* 6 *samples in this experiment*.

Code:

```
s_m = 101.82 #sample mean
pop_sd = 1.2 n = 6
#sample Size  alpha =
0.05
z_0 = norm.ppf(1-alpha/2)
L = s_m - z_0*pop_sd/math.sqrt(n) U
= s_m + z_0*pop_sd/math.sqrt(n)
conf_interval = (L, U)
print("95% confidence interval for guiven population mean is "+ str(conf_interval))
```

output:

```
95%
```

10. *Construct a* 98% *Confidence Interval for population mean based on the following data*:

45, 55, 67, 45, 68, 79, 98, 87, 84, 82.

```
confidence interval for given population mean is (100.85981766472894, 102.78018233527105)
```

Code:

```
from scipy.stats import t
data = [45, 55, 67, 45, 68, 79, 98, 87, 84, 82] s_m
= st.mean(data)  #sample mean
s_sd = st.stdev(data)  #sample standard deviation
n = len(data)   #sampe size df = n - 1  alpha =
0.02
t_0 = t.ppf(1-alpha/2, df)
L = s_m - t_0*s_sd/math.sqrt(n)
U = s_m + t_0*s_sd/math.sqrt(n)
```

<table>
<tr><td></td><td>

```
conf_interval = (L, U)
print("98% confidence interval for given population mean =" + str(conf_interval))
```

output:

```
98% confidence interval for given population mean =(54.78661680024991, 87.2133831997501)
```

11. 510 *people applied to the Bachelor's in Elementary Education program at Florida State College. Of those applicants,* 57 *were men. Find the* 90% *CI of the true proportion of men who applied to the program.*

Code:

```
X = 57 n = 510 p_hat =
X/n q_hat = 1 - p_hat
alpha = 0.1 z_0 =
norm.ppf(1-alpha/2)
L = p_hat - z_0*math.sqrt(p_hat*q_hat/n) U = p_hat + z_0*math.sqrt(p_hat*q_hat/n)
conf_interval = (L, U) print("98% confidence interval for guiven population proportion is
"+ str(conf_interval))
```

output:

```
98% confidence interval for guiven population proportion is
(0.08881598300884237, 0.13471342875586353)
```

</td></tr>
<tr><td>Conclusion</td><td>Different python and scipy libraries made it easy to calculate the probabilities of different distributiions and vice versa.</td></tr>
</table>

Signature of Faculty