In [1]:
```
import numpy as np
import scipy.stats as st
```

Common Variable for all Tests

In [2]:
```
# H0: mu = 46
# H1: mu > 46  --> For Right Tailed Test
# H1: mu < 46  --> For Left  Tailed Test
# H1: mu != 46 --> For Two   Tailed test
n=30 #No of sample data
xbar = 48.6 #Mean of Sample Data
mu = 46 #Mean of Population Data
sigma = 4.2 #Standard Deviation of Population Data
alpha = 0.05
```

Z - Test

In [3]:
```
z_critical = abs(st.norm.ppf(alpha))
z_critical
```

Out[3]:
```
1.6448536269514729
```

In [4]:
```
z = (xbar-mu)/(sigma/np.sqrt(n))
z
```

Out[4]:
```
3.3906634512224585
```

Right-tailed Test

In [5]:
```
if (z < z_critical):
    print("Cannot be Rejected")
else:
    print("Rejected")
```

```
Rejected
```

Left-tailed Test

In [6]:
```
if (z >= -z_critical):
    print("Cannot be Rejected")
else:
    print("Rejected")
```

```
Cannot be Rejected
```

Two-tailed Test

In [7]:
```
z_critical = abs(st.norm.ppf(alpha/2))
z_critical
```

Out[7]:
```
1.9599639845400545
```

In [8]:
```
z = (xbar-mu)/(sigma/np.sqrt(n))
z
```

```
3.3906634512224585
```

Out[8]:

In [9]:
```
if (-z_critical < z < z_critical):
    print("Cannot be Rejected")
else:
    print("Rejected")
```

```
Rejected
```

In [ ]:

In [ ]:

Z - Test using P-Value

In [10]:
```
z= (xbar-mu)/(sigma/np.sqrt(n))
z
```

Out[10]:
```
3.3906634512224585
```

Two - Tail test

In [11]:
```
p_val = (1-st.norm.cdf(abs(z))) * 2
p_val
```

Out[11]:
```
0.0006972366404673913
```

In [12]:
```
if(p_val > alpha):
    print("Cannot be Rejected")
else:
    print("Rejected")
```

```
Rejected
```

Left - Tail Test

In [13]:
```
p_val = (1-st.norm.cdf(abs(z)))
p_val
```

Out[13]:
```
0.0003486183202336956 6
```

In [14]:
```
if(p_val < alpha):
    print("Cannot be Rejected")
else:
    print("Rejected")
```

```
Cannot be Rejected
```

Right - Tail Test

In [15]:
```
p_val = (1-st.norm.cdf(abs(z)))
p_val
```

Out[15]:
```
0.0003486183202336956 6
```

In [16]:
```
if(p_val > alpha):
    print("Cannot be Rejected")
```

```
else:
    print("Rejected")
```

```
Rejected
```

In [ ]:

In [ ]:

t - Test

If the Variable is Normally Distributed

In [17]:
```
#H0 : mu = 16.3
#H1 : mu != 16.3 --> Two-Tailed Test
#H2 : mu > 16.3 --> Right-Tailed Test
#H3 : mu < 16.3 --> Left-Tailed test
n=10
degrees_of_freedom = n-1
xbar = 17.7
mu = 16.3
s = 1.8 #Standard deviation of Sample
alpha = 0.05
```

In [18]:
```
t = (xbar - mu)/(s / np.sqrt(n))
t
```

Out[18]:
```
2.4595492912420704
```

Right-Tail test

In [19]:
```
t_critical = st.t.ppf(alpha,degrees_of_freedom)
t_critical
```

Out[19]:
```
-1.8331129326536337
```

In [20]:
```
if (t <= t_critical):
    print("Cannot be rejected")
else:
    print("Rejected")
```

```
Rejected
```

Left-Tail Test

In [21]:
```
t_critical = st.t.ppf(alpha,degrees_of_freedom)
t_critical
```

Out[21]:
```
-1.8331129326536337
```

In [22]:
```
if (t >= -t_critical):
    print("Cannot be rejected")
else:
    print("Rejected")
```

```
Cannot be rejected
```

Two - tail test

In [23]:
```
t_critical = st.t.ppf(alpha/2,degrees_of_freedom)
t_critical=abs(t_critical)
```

In [24]:
```
if (-t_critical<t<t_critical):
    print("Cannot be rejected")
else:
    print("Rejected")
```

```
Rejected
```

In [ ]:

In [ ]:

t-test using P Value

In [25]:
```
t = (xbar - mu)/(s / np.sqrt(n))
t
```

Out[25]:
```
2.4595492912420704
```

Right - Tailed Test

In [26]:
```
p_val = (1-st.t.cdf(abs(t),degrees_of_freedom))
p_val
```

Out[26]:
```
0.018092696370958006
```

In [27]:
```
if (p_val > alpha):
    print("Cannot be rejected")
else:
    print("Rejected")
```

```
Rejected
```

Left - tailed test

In [28]:
```
p_val = (1-st.t.cdf(abs(t),degrees_of_freedom))
p_val
```

Out[28]:
```
0.018092696370958006
```

In [29]:
```
if (p_val < alpha):
    print("Cannot be rejected")
else:
    print("Rejected")
```

```
Cannot be rejected
```

Two-Tailed Test

In [30]:
```
p_val = (1-st.t.cdf(abs(t),degrees_of_freedom))*2
p_val
```

Out[30]:
```
0.03618539274191601
```

In [31]:
```
if (p_val > alpha):
    print("Cannot be rejected")
```

```
else:
    print("Rejected")
```

```
Rejected
```

In [ ]: