```
from scipy.stats import ttest_1samp
import numpy as np

# Creating a sample of ages
ages = [45, 89, 23, 46, 12, 69, 45, 24, 34, 67]
print(ages)

# Calculating the mean of the sample
mean = np.mean(ages)
print(mean)

# Performing the T-Test
t_test, p_val = ttest_1samp(ages, 30)
print("P-value is: ", p_val)

# taking the threshold value as 0.05 or 5%
if p_val < 0.05:
    print(" We can reject the null hypothesis")
else:
    print("We can accept the null hypothesis")
```

```
[45, 89, 23, 46, 12, 69, 45, 24, 34, 67]
45.4
P-value is:  0.07179988272763561
We can accept the null hypothesis
```

```
from scipy.stats import ttest_ind
import numpy as np

# Creating the data groups
data_group1 = np.array([12, 18, 12, 13, 15, 1, 7,
                        20, 21, 25, 19, 31, 21, 17,
                        17, 15, 19, 15, 12, 15])
data_group2 = np.array([23, 22, 24, 25, 21, 26, 21,
                        21, 25, 30, 24, 21, 23, 19,
                        14, 18, 14, 12, 19, 15])

# Calculating the mean of the two data groups
mean1 = np.mean(data_group1)
mean2 = np.mean(data_group2)

# Print mean values
print("Data group 1 mean value:", mean1)
print("Data group 2 mean value:", mean2)

# Calculating standard deviation
std1 = np.std(data_group1)
std2 = np.std(data_group2)

# Printing standard deviation values
print("Data group 1 std value:", std1)
print("Data group 2 std value:", std2)

# Implementing the t-test
t_test,p_val = ttest_ind(data_group1, data_group2)
print("The P-value is: ", p_val)

# taking the threshold value as 0.05 or 5%
if p_val < 0.05:
    print("We can reject the null hypothesis")
else:
    print("We can accept the null hypothesis")
```

```
Data group 1 mean value: 16.25
Data group 2 mean value: 20.85
Data group 1 std value: 6.171507109288622
Data group 2 std value: 4.452808102759426
The P-value is:  0.012117171124028792
We can reject the null hypothesis
```

```
import pandas as pd
from scipy import stats

# Creating two samples
sample1 = [29, 30, 33, 41, 38, 36,
       35, 31, 29, 30]
sample2 = [31, 32, 33, 39, 30, 33,
        30, 28, 29, 31]

# Performing paired sample t-test
t_test, p_val = stats.ttest_rel(sample1, sample2)
print("The P-value of the test is: ", p_val)

# taking the threshold value as 0.05 or 5%
if p_val < 0.05:
    print("We can reject the null hypothesis")
else:
    print("We can accept the null hypothesis")
```

```
     The P-value of the test is:  0.15266056244408904
     We can accept the null hypothesis
```

```
import pandas as pd
from scipy import stats
from statsmodels.stats import weightstats as stests

# Creating a dataset
data = [89, 93, 95, 93, 97, 98, 96, 99, 93, 97,
        110, 104, 119, 105, 104, 110, 110, 112, 115, 114]

# Performing the z-test
z_test ,p_val = stests.ztest(data, x2 = None, value = 160)
print(p_val)

# taking the threshold value as 0.05 or 5%
if p_val < 0.05:
    print("We can reject the null hypothesis")
else:
    print("We can accept the null hypothesis")
```

```
     2.417334226169332e-186
     We can reject the null hypothesis
```

```
import pandas as pd
from scipy import stats
from statsmodels.stats import weightstats as stests

# Creating a dataset
data1 = [83, 85, 86, 90, 90, 93, 93, 95, 97, 97,
         106, 108, 106, 108, 111, 113, 113, 112, 116, 111]

data2 = [92, 92, 90, 93, 93, 97, 94, 98, 109, 108,
         110, 117, 110, 115, 114, 114, 130, 130, 149, 131]

# Implementing the two-sample z-test
z_test ,p_val = stests.ztest(data1, x2 = data2, value = 0, alternative = 'two-sided')
print(p_val)

# taking the threshold value as 0.05 or 5%
if p_val < 0.05:
    print("We can reject the null hypothesis")
else:
    print("We can accept the null hypothesis")
```

```
     0.04813782199434202
     We can reject the null hypothesis
```

```
import scipy.stats

# Creating sample data
data1 = [0.0842, 0.0368, 0.0847, 0.0935, 0.0376, 0.0963, 0.0684,
             0.0758, 0.0854, 0.0855]
data2 = [0.0785, 0.0845, 0.0758, 0.0853, 0.0946, 0.0785, 0.0853,
           0.0685]
data3 = [0.0864, 0.2522, 0.0894, 0.2724, 0.0853, 0.1367, 0.853]

# Performing the F-Test
f_test, p_val = scipy.stats.f_oneway(data1, data2, data3)
print("p-value is: ", p_val)

# taking the threshold value as 0.05 or 5%
if p_val < 0.05:
    print(" We can reject the null hypothesis")
else:
    print("We can accept the null hypothesis")
```

```
p-value is:  0.04043792126789144
 We can reject the null hypothesis
```

```
# Python program to perform 2-way F-test

# Importing the required modules
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.formula.api import ols

# Create a dataframe
df = pd.DataFrame({'Frequency_fertilizers': np.repeat(['daily', 'weekly'], 16),
                          'Frequency_Watering': np.repeat(['daily', 'weekly'], 16),
                          'Crop_height': [12, 14, 15, 12, 17, 21, 19, 8,
                                          5, 12, 19, 23, 23, 14, 16, 21,
                                          25, 16, 17, 13, 24, 9, 19, 4,
                                          12, 14, 15, 12, 17, 15, 18, 14]})

# Performing the two-way ANOVA test
f_test = ols('Crop_height ~ C(Frequency_fertilizers) * C(Frequency_Watering) + \
C(Frequency_fertilizers):C(Frequency_Watering)',
            data = df).fit()
result = sm.stats.anova_lm(f_test, type = 2)

# Printing the result
print(result)
```

```
                                                  df      sum_sq     mean_sq  \
C(Frequency_fertilizers)                         1.0    1.531250    1.531250
C(Frequency_Watering)                            1.0  117.945205  117.945205
C(Frequency_fertilizers):C(Frequency_Watering)   1.0    1.643988    1.643988
Residual                                        30.0  802.437500   26.747917

                                                       F    PR(>F)
C(Frequency_fertilizers)                        0.057247  0.812528
C(Frequency_Watering)                           4.409510  0.044253
```