```
#1
import numpy as np
from scipy import stats

def variance_test(data1,data2):
   var1=np.var(data1,ddof=11)
   var2=np.var(data2,ddof=1)
   F=var1/var2
   df1=len(data)-1
   df2=len(data)-1
   p_value=1-stats.f.cdf(f,df1,df2)
   return F,p_value;

data1=[22,23,44,55,66]
data2=[34,45,66,89,90]
F, p_value = variance_ratio_test(data1, data2)

print(f"F-value: {F}")
print(f"P-value: {p_value}")



import numpy as np
from scipy import stats

def variance_ratio_test(data1, data2):
   # Calculate variances of both datasets
```

```
    var1 = np.var(data1, ddof=1)  # Sample variance (ddof=1 for unbiased estimator)
    var2 = np.var(data2, ddof=1)

    # Calculate the F-statistic
    F = var1 / var2

    # Degrees of freedom
    df1 = len(data1) - 1  # degrees of freedom for the first sample
    df2 = len(data2) - 1  # degrees of freedom for the second sample

    # Calculate the p-value using the F-distribution CDF
    p_value = 1 - stats.f.cdf(F, df1, df2)

    return F, p_value

# Example data
data1 = [10, 12, 13, 15, 16, 18]
data2 = [20, 22, 24, 25, 30]

# Get F-value and p-value
F, p_value = variance_ratio_test(data1, data2)

print(f"F-value: {F}")
print(f"P-value: {p_value}")
```

**Q2.Given a significance level of 0.05 and the degrees of freedom for the numerator and denominator of an**

**F-distribution, write a Python function that returns the critical F-value for a two-tailed test.**

```
from scipy import stats


def criticle_f_value(alpha,data1,data2):
    upper_critical_value=stats.f.ppf(1-alpha,data1,data2)
    lower_critical_value=1/upper_criticle_value
    return upper_criticle_value,lower_criticle_value
    alpha=0.05
    data1=5
    data2=10
    lower_critical, upper_critical = critical_f_value(alpha, data1, data2)
print(f"Lower critical F-value: {upper_critical}")
print(f"Upper critical F-value: {lower_critical}")
```

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

```
def f_test_for_variances(sample1, sample2):
    # Calculate variances of both datasets
    var1 = np.var(sample1, ddof=1)  # Sample variance (ddof=1 for unbiased estimator)
    var2 = np.var(sample2, ddof=1)

    # Calculate the F-statistic (ratio of variances)
    F = var1 / var2 if var1 > var2 else var2 / var1  # Ensure F > 1

    # Degrees of freedom for each sample
    df1 = len(sample1) - 1  # degrees of freedom for the first sample
    df2 = len(sample2) - 1  # degrees of freedom for the second sample

    # Calculate the p-value using the F-distribution CDF
    p_value = 1 - stats.f.cdf(F, df1, df2)

    return F, df1, df2, p_value

# Generate random samples from two normal distributions with known variances
np.random.seed(42)  # For reproducibility

# Parameters for the normal distributions: mean, variance, and sample size
mean1, var1, size1 = 10, 4, 30  # mean=10, variance=4, sample size=30
mean2, var2, size2 = 20, 9, 30  # mean=20, variance=9, sample size=30

# Generate the samples using numpy's normal distribution function
```

```
sample1 = np.random.normal(mean1, np.sqrt(var1), size1)

sample2 = np.random.normal(mean2, np.sqrt(var2), size2)


# Perform the F-test for variances

F, df1, df2, p_value = f_test_for_variances(sample1, sample2)


# Output the results

print(f"F-value: {F}")

print(f"Degrees of freedom for sample 1: {df1}")

print(f"Degrees of freedom for sample 2: {df2}")

print(f"P-value: {p_value}")


#4

import scipy.stats as stats


# Given values

var1 = 10  # Variance of first population

var2 = 15  # Variance of second population

n1 = 12  # Sample size for population 1

n2 = 12  # Sample size for population 2

alpha = 0.05  # Significance level


# F-statistic calculation

F = var1 / var2 if var1 > var2 else var2 / var1  # Ensure F > 1
```

```
# Degrees of freedom
df1 = n1 - 1  # Degrees of freedom for sample 1
df2 = n2 - 1  # Degrees of freedom for sample 2

# p-value calculation
p_value = 1 - stats.f.cdf(F, df1, df2)

print(f"F-value: {F}")
print(f"Degrees of freedom for sample 1: {df1}")
print(f"Degrees of freedom for sample 2: {df2}")
print(f"P-value: {p_value}")

#5
# Given values
claimed_variance = 0.005  # Claim variance
sample_variance = 0.006  # Sample variance
n = 25  # Sample size
alpha = 0.01  # Significance level

# F-statistic calculation
F = sample_variance / claimed_variance

# Degrees of freedom
df1 = n - 1  # Degrees of freedom for sample
df2 = n - 1  # Degrees of freedom for the population (same as sample)
```

```
# p-value calculation
p_value = 1 - stats.f.cdf(F, df1, df2)

print(f"F-value: {F}")
print(f"Degrees of freedom: {df1}")
print(f"P-value: {p_value}")

#6
def f_distribution_mean_variance(df1, df2):
  if df1 <= 2:
    mean = None
  else:
    mean = df1 / (df1 - 2)

  if df2 <= 4:
    variance = None
  else:
    variance = (2 * df2 * (df1 + df2 - 2)) / (df1 * (df2 - 2)**2 * (df2 - 4))

  return mean, variance

# Example degrees of freedom
df1 = 5
df2 = 10
```

```
mean, variance = f_distribution_mean_variance(df1, df2)

print(f"Mean of F-distribution: {mean}")

print(f"Variance of F-distribution: {variance}")


#7

# Given values

var1 = 25  # Sample variance for population 1

var2 = 20  # Sample variance for population 2

n1 = 10  # Sample size for population 1

n2 = 15  # Sample size for population 2

alpha = 0.10  # Significance level


# F-statistic calculation

F = var1 / var2 if var1 > var2 else var2 / var1  # Ensure F > 1


# Degrees of freedom

df1 = n1 - 1  # Degrees of freedom for sample 1

df2 = n2 - 1  # Degrees of freedom for sample 2


# p-value calculation

p_value = 1 - stats.f.cdf(F, df1, df2)


print(f"F-value: {F}")

print(f"Degrees of freedom for sample 1: {df1}")
```

```
print(f"Degrees of freedom for sample 2: {df2}")

print(f"P-value: {p_value}")


#8

# Data for restaurants

restaurant_a = [24, 25, 28, 23, 22, 20, 27]

restaurant_b = [31, 33, 35, 30, 32, 36]


# Calculate variances

var_a = np.var(restaurant_a, ddof=1)

var_b = np.var(restaurant_b, ddof=1)


# Sample sizes

n1 = len(restaurant_a)

n2 = len(restaurant_b)

alpha = 0.05  # Significance level


# F-statistic calculation

F = var_a / var_b if var_a > var_b else var_b / var_a


# Degrees of freedom

df1 = n1 - 1

df2 = n2 - 1


# p-value calculation
```

```
p_value = 1 - stats.f.cdf(F, df1, df2)


print(f"F-value: {F}")

print(f"Degrees of freedom for sample 1: {df1}")

print(f"Degrees of freedom for sample 2: {df2}")

print(f"P-value: {p_value}")


#9

# Data for groups

group_a = [80, 85, 90, 92, 87, 83]

group_b = [75, 78, 82, 79, 81, 84]


# Calculate variances

var_a = np.var(group_a, ddof=1)

var_b = np.var(group_b, ddof=1)


# Sample sizes

n1 = len(group_a)

n2 = len(group_b)

alpha = 0.01  # Significance level


# F-statistic calculation

F = var_a / var_b if var_a > var_b else var_b / var_a


# Degrees of freedom
```

```
df1 = n1 - 1

df2 = n2 - 1


# p-value calculation

p_value = 1 - stats.f.cdf(F, df1, df2)


print(f"F-value: {F}")

print(f"Degrees of freedom for sample 1: {df1}")

print(f"Degrees of freedom for sample 2: {df2}")

print(f"P-value: {p_value}")


#10
```