```python
import scipy.stats as stats
import pandas as pd
from statsmodels.stats.multicomp import pairwise_tukeyhsd

# -=-=-=-=-=-=-=-=-=-=-=-=-=-
# EXCERCISE PROBLEM 1
# -=-=-=-=-=-=-=-=-=-=-=-=-=-

data = pd.read_csv('flavors.csv')
d = pd.DataFrame(data)

#print(d)
USA = d[d['Location'] == 'U.S.A.']
France = d[d['Location'] == 'France']
Canada = d[d['Location'] == 'Canada']

#print(Canada.Rating)

fvalue, pvalue = stats.f_oneway(USA.Rating, France.Rating, Canada.Rating)

print(fvalue, pvalue)

data = pd.concat([USA['Rating'], France['Rating'], Canada['Rating']])

groups = ['USA'] * len(USA) + ['France'] * len(France) + ['Canada'] * len(Canada)

fvalue, pvalue = stats.f_oneway(USA['Rating'], France['Rating'], Canada['Rating'])
print("ANOVA p-value:", pvalue)

tukey_results = pairwise_tukeyhsd(data, groups)

print(tukey_results.summary())


# Perform ANOVA
anova_result = stats.f_oneway(*[group['Rating'] for name, group in d.groupby('ReviewDate')])

# Print ANOVA results
print("ANOVA p-value:", anova_result.pvalue)

# Perform Tukey-Kramer post hoc test
tukey_results = pairwise_tukeyhsd(d['Rating'], d['ReviewDate'])

# Print the summary of the post hoc test
print(tukey_results.summary())


# -=-=-=-=-=-=-=-=-=-=-=-=-=-
# EXCERCISE PROBLEM 2
# -=-=-=-=-=-=-=-=-=-=-=-=-=-


import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

data = pd.read_csv('longley.csv')
d = pd.DataFrame(data)


slope, intercept = np.polyfit(d.GNP, d.Employed, 1)

predicted_values = slope * d.GNP + intercept

plt.scatter(d.GNP, d.Employed, label='Data Points')
```

```python
plt.plot(d.GNP, predicted_values, label='Regression Line', color='red')

plt.xlabel('GNP')
plt.ylabel('Employed')

plt.legend()

plt.show()

print('Slope: ' + str(slope))
print('Intercept: ' + str(intercept))


# -=-=-=-=-=-=-=-=-=-=-=-=-=-
# EXCERCISE PROBLEM 3
# -=-=-=-=-=-=-=-=-=-=-=-=-=-


import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import mean_squared_error

data = pd.read_csv('abalone2.csv')
d = pd.DataFrame(data)

slope, intercept = np.polyfit(d.ShellWeight, d.Rings, 1)

predicted_values = slope * d.ShellWeight + intercept

plt.scatter(d.ShellWeight, d.Rings, label='Data Points')

plt.plot(d.ShellWeight, predicted_values, label='Regression Line', color='red')

plt.xlabel('Shell Weight')
plt.ylabel('Age (Rings)')

plt.legend()

plt.show()

mse = mean_squared_error(d.Rings, predicted_values)

rmse = np.sqrt(mse)

print("Mean Squared Error (MSE): " + str(mse))
print("Root Mean Squared Error (RMSE): " + str(rmse))
```