Here are the solutions to the assignment problems using Python:

**Q1: 95% Confidence Interval for a Sample**

To calculate the 95% confidence interval:

import scipy.stats as stats

import math

# Given values

mean = 50

std\_dev = 5

n = 30 # Sample size

confidence\_level = 0.95

# Calculate the standard error

standard\_error = std\_dev / math.sqrt(n)

# Find the z-score for a 95% confidence level

z\_score = stats.norm.ppf(0.975)

# Calculate the confidence interval

confidence\_interval = (mean - z\_score \* standard\_error, mean + z\_score \* standard\_error)

print("95% Confidence Interval:", confidence\_interval)

**Interpretation:** The interval calculated represents the range where we expect the true population mean to lie with 95% confidence.

**Q2: Chi-Square Goodness of Fit Test (M&M Colors)**

import scipy.stats as stats

# Observed frequencies (in a sample bag of M&Ms)

observed = [20, 20, 20, 10, 10, 20]

# Expected frequencies (theoretical distribution)

expected = [0.20 \* sum(observed)] \* 6

# Chi-Square Test

chi\_square\_stat, p\_value = stats.chisquare(observed, expected)

print("Chi-Square Statistic:", chi\_square\_stat)

print("P-value:", p\_value)

**Interpretation:** If the p-value is less than the significance level (0.05), we reject the null hypothesis that the observed distribution matches the expected distribution.

**Q3: Chi-Square Test for Contingency Table**

import scipy.stats as stats

import numpy as np

# Contingency table

data = np.array([[20, 15], [10, 25], [15, 20]])

# Chi-Square Test for independence

chi\_square\_stat, p\_value, dof, expected = stats.chi2\_contingency(data)

print("Chi-Square Statistic:", chi\_square\_stat)

print("P-value:", p\_value)

**Interpretation:** If the p-value is less than 0.05, we conclude there is a significant association between the variables.

**Q4: Confidence Interval for a Proportion**

import statsmodels.api as sm

# Given data

n = 500 # Sample size

x = 60 # Number of smokers

# Proportion

p\_hat = x / n

# 95% Confidence Interval for proportion

conf\_int = sm.stats.proportion\_confint(x, n, alpha=0.05)

print("95% Confidence Interval for Proportion:", conf\_int)

**Interpretation:** This interval gives the range where we expect the true proportion of smokers in the population to lie.

**Q5: 90% Confidence Interval for a Sample**

# Given values

mean = 75

std\_dev = 12

n = 30

confidence\_level = 0.90

# Standard error

standard\_error = std\_dev / math.sqrt(n)

# Z-score for 90% confidence level

z\_score = stats.norm.ppf(0.95)

# Confidence interval

confidence\_interval = (mean - z\_score \* standard\_error, mean + z\_score \* standard\_error)

print("90% Confidence Interval:", confidence\_interval)

**Interpretation:** This interval shows where the population mean is likely to fall with 90% confidence.

**Q6: Plotting the Chi-Square Distribution**

import matplotlib.pyplot as plt

import numpy as np

# Chi-square distribution with 10 degrees of freedom

df = 10

x = np.linspace(0, 30, 100)

y = stats.chi2.pdf(x, df)

# Plotting the chi-square distribution

plt.plot(x, y, label="Chi-Square Distribution (df=10)")

plt.fill\_between(x, y, where=(x >= 15), color="red", alpha=0.5, label="Area for chi-square > 15")

plt.xlabel("Chi-Square Statistic")

plt.ylabel("Probability Density")

plt.title("Chi-Square Distribution with 10 Degrees of Freedom")

plt.legend()

plt.show()

**Q7: Confidence Interval for Proportion (Coke vs Pepsi)**

# Given data

n = 1000

x = 520

# Proportion

p\_hat = x / n

# 99% Confidence Interval for proportion

conf\_int = sm.stats.proportion\_confint(x, n, alpha=0.01)

print("99% Confidence Interval for Proportion:", conf\_int)

**Q8: Chi-Square Test for a Biased Coin**

# Observed frequencies (tails: 45, heads: 55)

observed = [45, 55]

expected = [50, 50] # Fair coin

# Chi-Square Test

chi\_square\_stat, p\_value = stats.chisquare(observed, expected)

print("Chi-Square Statistic:", chi\_square\_stat)

print("P-value:", p\_value)

**Q9: Chi-Square Test for Independence (Smoking and Lung Cancer)**

# Contingency table

data = np.array([[60, 140], [30, 170]])

# Chi-Square Test for independence

chi\_square\_stat, p\_value, dof, expected = stats.chi2\_contingency(data)

print("Chi-Square Statistic:", chi\_square\_stat)

print("P-value:", p\_value)

**Q10: Chi-Square Test for Independence (Chocolate Preference by Country)**

# Contingency table for chocolate preference

data = np.array([[100, 120, 150], [110, 130, 160]])

# Chi-Square Test for independence

chi\_square\_stat, p\_value, dof, expected = stats.chi2\_contingency(data)

print("Chi-Square Statistic:", chi\_square\_stat)

print("P-value:", p\_value)

**Q11: Hypothesis Test for Population Mean**

from scipy import stats

# Given data

n = 30

sample\_mean = 72

sample\_std\_dev = 10

population\_mean = 70

# Calculate t-statistic

t\_stat = (sample\_mean - population\_mean) / (sample\_std\_dev / math.sqrt(n))

# Calculate p-value for a two-tailed test

p\_value = 2 \* (1 - stats.t.cdf(abs(t\_stat), df=n-1))

print("T-statistic:", t\_stat)

print("P-value:", p\_value)

**Interpretation for all tests:** If the p-value is less than 0.05 (or the respective significance level), you reject the null hypothesis. Otherwise, you fail to reject it.

Let me know if you need further details or modifications!