

DE LA SALLE UNIVERSITY - MANILA

Hippopotamus (Hypothesis Testing Calculator)

A Term Project


Presented to Mr. Ramon Stephen L. Ruiz


In Partial Fulfillment of the

Requirements for the Course Programming Logic and Design (PROLOGI)

by

Alabado, Tayshaun Pierce 

Taraya, Marcus Alde 

Sy, Win Clarence 

EQ3

Monday and Thursday, 12:45 PM - 1:45 PM

April 2023

Table of Contents

Introduction	3
Background of the Study	4
Problem Statement	5
Objectives	6
General Objectives	6
Specific Objectives	6
Significance of the Project	6
Review of Related Literature	7
Hypothesis Testing	8
Hypothesis Testing and Reliability	9
Hypothesis Testing and Precision	10
Methodology	11
Conceptual Framework - IPO Chart (Input-Process-Output-Chart)	11
Hierarchy Chart	16
Flowchart	17
Pseudocode	19
Results	39
Discussion of the Results	41
Analysis, Conclusion, and Future Directives	41
References	43
Appendices	45
Appendix A - User's Manual	45
Appendix B - Source Code	45
Appendix C - Work Breakdown	66
Appendix D - Personal Data Sheet	67

I. Introduction

A crucial part of statistical analysis is hypothesis testing, which is used to determine the significance of study results and draw inferences about population characteristics. Hypothesis testing involves a number of steps, including formulating a hypothesis, choosing an appropriate statistical test, gathering and processing data, and interpreting the findings. However, carrying out these procedures by hand can be difficult and time-consuming, particularly for people without a strong statistical background.

In order to address this issue, hypothesis testing calculators have gained favor recently. These calculators give customers a quick and efficient way to test hypotheses, allowing for accurate statistical analysis. The usefulness of calculators for hypothesis testing has increased due to the expansion of big data and the rise in demand for data-driven decision-making.

In this research work, we describe the creation and assessment of a calculator for hypothesis testing with a user-friendly interface and precise statistical findings. We offer a variety of statistical tests in our calculator, from easy tests to choices about whether to accept or reject the null hypothesis. We also give users the option to tailor their analyses to meet their unique requirements, including defining the degree of significance, picking the specific test, and entering data.

To evaluate the effectiveness of our calculator, we conducted a series of tests comparing the results obtained from our calculator to those obtained from manual hypothesis testing. Our results indicate that our calculator provides highly accurate and reliable statistical analyses.

Overall, the development of a robust and reliable hypothesis testing calculator has the potential to greatly improve the accuracy and efficiency of statistical analysis, making it an essential tool for researchers and practitioners alike. Through our research, we hope to contribute to the advancement of statistical analysis and data-driven decision-making by providing a powerful and accessible tool for hypothesis testing.

A. Background of the Study

Hypothesis testing is a fundamental statistical tool that is widely used to determine whether a particular hypothesis or claim about a population is supported by the available evidence (Babbie, 2017; Field, 2013). It is a critical component of the scientific method and is used in a variety of fields, including social sciences, business, medicine, and engineering (Rosnow & Rosenthal, 1991; Welch, 1937).

By using hypothesis testing, researchers can draw conclusions about a population from a small sample of data. (Kline, 2015). It entails contrasting an alternative hypothesis, which is the null hypothesis's opposite, with a null hypothesis, which is the presumption that there is no difference or effect in the population. (Patten, 2018). The steps in the hypothesis testing procedure are data collection, computation of a test statistic, and evaluation of the statistical significance of the observed result. (Fisher, 1925).

With the availability of vast volumes of data in modern research, the use of hypothesis testing has grown in importance. (Hinton et al., 2019). However, the computations necessary for hypothesis testing can be challenging and demand a thorough knowledge of statistical theory and procedures. (Maxwell & Delaney, 2004). Furthermore, it can be difficult for some people to grasp how to make conclusions based on probability while conducting hypothesis testing. (Moore & McCabe, 2003).

To address these challenges, the development of hypothesis testing calculators has become an important area of research (Hogg & Tanis, 2018). These calculators allow researchers and practitioners to perform hypothesis tests quickly and accurately, without the need for extensive statistical knowledge (Zhang, 2015). They can be used to test a wide range of hypotheses, from simple comparisons of means to more complex analyses of variance and regression models (Freedman et al., 2007).

The development of hypothesis testing calculators has significant implications for both research and practice. They provide a powerful tool for researchers to analyze their data and draw conclusions about their populations (Cohen, 1994). In addition,

they can be used in applied settings, such as business and healthcare, to make decisions based on statistical evidence (Altman & Bland, 1994).

In this project, we aim to develop a hypothesis testing calculator that is accessible and user-friendly, while still providing accurate and reliable results. Our calculator will be designed to handle a wide range of hypothesis tests. By providing a powerful and accessible tool for hypothesis testing, we hope to contribute to the advancement of statistical analysis and promote evidence-based decision-making in a variety of fields.

B. Problem Statement

Hypothesis testing is a critical component of statistical analysis used to determine the significance of research findings and draw conclusions about population parameters. However, performing hypothesis testing manually can be time-consuming, complex, and prone to errors, particularly for individuals without a strong statistical background. While existing hypothesis testing calculators provide a solution, they are often limited in functionality, difficult to use, or provide inaccurate results.

The ideal scenario is that all researchers and practitioners have access to a reliable, user-friendly, and accurate hypothesis testing calculator that can perform a wide range of statistical tests, customize analysis based on specific needs, and provide clear interpretations of results. However, the current reality is that many available hypothesis testing calculators lack the necessary features and accuracy to provide accurate and reliable statistical analyses.

This gap in knowledge and available tools have significant consequences for the accuracy and efficiency of statistical analysis, potentially leading to incorrect conclusions and flawed decision-making. This problem highlights the need for a robust and reliable hypothesis testing calculator that can provide users with accurate and efficient statistical analysis, regardless of their statistical background.

Therefore, this research project aims to develop and evaluate a hypothesis testing calculator that addresses these issues, providing users with a powerful tool for conducting accurate and reliable statistical analysis.

C. Objectives

C.1. General Objective

The general objective of this project is to develop a hypothesis testing calculator that provides users with a user-friendly interface and accurate statistical results, allowing for quick and efficient statistical analysis.

C.2. Specific Objectives

1. To identify and evaluate the different types of hypothesis tests that will be included in the calculator.
2. To design and develop a user-friendly interface for the hypothesis testing calculator, including the input and output of data and the selection of statistical tests.
3. To implement and test the functionality of the hypothesis testing calculator, ensuring the accuracy and reliability of statistical results.
4. To evaluate the effectiveness of the hypothesis testing calculator through comparisons with manual hypothesis testing, ensuring the calculator's level of precision is comparable to that of experienced statisticians.

D. Significance of the Project

The development of a robust and reliable hypothesis testing calculator has significant implications for both research and practice in various fields, including science, engineering, business, and healthcare. The significance of this project is to improve accuracy and efficiency by providing users with a powerful and accurate tool for hypothesis testing, the calculator can significantly improve the accuracy and efficiency of statistical analysis. This can lead to more accurate conclusions, informed decision-making, and ultimately, better outcomes. Making a user-friendly interface that makes statistical analysis accessible to individuals with limited statistical knowledge, enabling them to perform hypothesis testing with ease. The calculator

significantly reduces the time required for statistical analysis, allowing researchers and practitioners to allocate more time to other critical tasks. By providing a reliable and accurate tool for hypothesis testing, the calculator can enhance the quality of research findings, increasing the credibility and validity of research outcomes. The calculator can increase productivity by reducing the time and effort required for statistical analysis, allowing individuals to focus on other critical tasks.

II. Review of Related Literature

Hypothesis testing is a statistical procedure that assesses the strength of evidence from a sample to make decisions about a population. The procedure involves testing the null hypothesis, which is assumed to be true, and rejecting it if the evidence suggests it is false, which supports the research hypothesis. The choice of the appropriate test statistic and significance level depends on the research topic and the underlying assumptions of the statistical model. Statistical power, which is the probability of correctly rejecting the null hypothesis, is influenced by the sample size, significance level, and effect size. Multiple testing can increase the risk of false positives, and confidence intervals can estimate the plausible values for a population parameter.

The paper by Haroutunian et al. (2008) provides a valuable resource for researchers and practitioners in the field of information theory and statistical hypothesis testing. The authors offer a comprehensive overview of the different reliability measures and their applications, highlighting the common principles and the trade-offs involved in maximizing reliability. This paper can be particularly relevant for individuals interested in developing a hypothesis calculator, as it provides important insights into the key considerations and factors that need to be taken into account when designing and evaluating such a tool.

The paper by Berger and Mortera provides a valuable perspective on the importance of precision in hypothesis testing. It highlights the limitations of traditional hypothesis testing methods and suggests that the use of precision measures, such as confidence intervals, can provide more informative results. For someone working on making a hypothesis calculator, this paper could serve as a useful reference for incorporating precision measures into their tool to provide more meaningful and accurate results.

A. Hypothesis Testing

According to Davis & Mukamal (2006), hypothesis testing is a procedure used to assess the strength of the evidence from the sample and offers a framework for making decisions pertaining to the population, i.e., it offers a method for comprehending how legitimately one can project observed findings in a sample under study to the larger population from which the sample was taken.

According to Allua and Thompson (2009), It is through hypothesis testing that the main research question is reduced to one of two possible hypothesis types: the null hypothesis (H_0), also known as the alternative hypothesis, or the research hypothesis (H_1), also known as the alternative hypothesis. We typically assume the opposite of our research question of interest because, in most cases, it is simpler to reject a statement that is false than it is to accept one that is true (the null hypothesis). Then, in order to support our research hypothesis, we test to see if we can disprove the null hypothesis. Hypothesis testing makes a suggestion that the research hypothesis is plausible rather than proving it to be true.

The procedure for hypothesis testing is described by Klein and Moeschberger in their paper from 2003. It entails selecting an appropriate test statistic and a significance level, computing the test statistic, and comparing it to the critical value or p-value to determine whether the null hypothesis should be rejected. The authors point out that the nature of the research topic and the underlying assumptions of the statistical model affect the choice of the test statistic.

The paper also discusses the concept of statistical power, which is the probability of correctly rejecting the null hypothesis when it is false. The authors explain that increasing the sample size and reducing the significance level can increase the power of a statistical test. They also note that statistical power is influenced by the effect size, which is the magnitude of the difference between the null and alternative hypotheses.

In addition to discussing the basic principles of hypothesis testing, Klein and Moeschberger (2003) also describe a number of advanced topics, such as multiple testing and the use of confidence intervals. The authors note that multiple testing can increase

the risk of false positive results and suggest adjusting the significance level or using a method such as the Bonferroni correction to control the overall error rate. They also explain how confidence intervals can be used to estimate the range of plausible values for a population parameter.

B. Hypothesis Testing and Reliability

The paper by Haroutunian et al. (2008) provides a comprehensive overview of the concept of reliability in both information theory and statistical hypothesis testing. The authors discuss the various measures of reliability and their applications in different fields, such as communications engineering, data analysis, and decision-making.

The paper begins by defining the concept of reliability in information theory, which is the probability of correctly decoding a transmitted message. The authors discuss different reliability criteria, such as the error probability, the outage probability, and the decoding complexity, and their relationship with various coding schemes. The paper then moves on to discuss the concept of reliability in statistical hypothesis testing, which is the probability of correctly accepting or rejecting a null hypothesis.

The authors explain the basic principles of statistical hypothesis testing and the role of various reliability measures, such as the significance level, the power of the test, and the confidence interval. They also discuss the trade-off between these measures and the impact of sample size on the reliability of the test.

The paper also delves into the relationship between information theory and statistical hypothesis testing, highlighting the common underlying principles and the shared objective of maximizing reliability. The authors argue that the principles of information theory can be applied to statistical hypothesis testing, particularly in the context of designing efficient tests that maximize the reliability of the decision-making process.

The paper concludes with a discussion of the practical applications of reliability criteria in different fields, such as telecommunications, data analysis, and decision-making. The authors emphasize the importance of selecting appropriate reliability measures based on the specific application and the underlying assumptions of

the problem. They also highlight the need for ongoing research to develop new reliability measures that are better suited to emerging applications and technologies.

C. Hypothesis Testing and Precision

The paper by Berger and Mortera discusses the concept of precision in hypothesis testing and how it can affect the interpretation of results. The authors argue that traditional hypothesis testing approaches, which rely heavily on p-values and significance levels, may not provide adequate information about the precision of the results obtained.

The authors start by highlighting the importance of precision in hypothesis testing, stating that it is a key factor in determining the scientific validity of a study. They then go on to argue that the use of p-values and significance levels, while useful, can be misleading in certain situations. Specifically, they point out that when sample sizes are large, even small differences between the null hypothesis and the alternative hypothesis can lead to significant results, despite the fact that the effect size may be negligible in practical terms.

The authors suggest using precision measurements, such as confidence intervals, in addition to conventional hypothesis testing techniques to address this problem. Given the data gathered, confidence intervals show a range of values where the true effect size is likely to fall. Narrower intervals signify better precision, and the width of the confidence interval indicates the accuracy of the estimate.

The authors then discuss the interpretation of confidence intervals and argue that they provide more useful information than p-values alone. Specifically, they suggest that researchers should focus on the width of the confidence interval, rather than just whether it includes zero. A narrow confidence interval indicates that the estimate is highly precise, while a wide interval suggests that the estimate is less precise.

III. Methodology

This project will utilize Python language with **Conda Kernel** since it has powerful libraries that we could use that would help us in this project. **Numpy** and **Math** would be used for calculations. **Scipy** would be used to get the critical value for the problem. **Colorma** would be used for coloring the program so that it would be easier for the user to see the important values. We will be utilizing several functions such as if-else statements. For the formula, we would be getting it from the course **FNDSTAT** we have taken from Term 1 AY2022-2023. We will be using modular functions to simplify the code and avoiding code duplication would make troubleshooting easier. We would be separating each case with its own functions so that it would be more user-friendly.

A. Conceptual Framework – IPO Chart (Input-Process-Output-Chart)

Input	Process	Output
Case A x, u_0, o, n, alpha Case B x, u_0, s, n, alpha Single Population Test Proportions x, n, p_0, alpha Case C x_1, s_1, n_1, x_2, s_2, n_2, d_0, alpha Case D x_1, s_1, n_1, x_2, s_2, n_2, d_0, alpha Case E	Import scipy.stats as st Case A $z = \frac{\bar{x} - \mu_0}{\left(\frac{\sigma}{\sqrt{n}}\right)}$ Case B $t = \frac{\bar{x} - \mu_0}{\left(\frac{s}{\sqrt{n}}\right)} \quad df = n - 1$ Single Population Test Proportion $\hat{p} = \frac{x}{n} \quad z = \frac{\hat{p} - p_0}{\sqrt{\frac{p_0 q_0}{n}}}$ Case C $z = \frac{(\bar{x}_1 - \bar{x}_2) - d_0}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$	Test Statistic P-Value (if any) Critical Value Degree of Freedom (if any) Sample Proportions (if any) Pooled Sample Proportion (if any) Pooled Standard Deviation (if any) Difference (if any) Difference Standard Deviation (if any) Whether to Reject or Not Reject the Null Hypothesis

<p>$x_1, s_1, n_1, x_2, s_2, n_2, d_0, \alpha$</p> <p>Case F</p> <p>d, d_0, s, n, α</p> <p>Two Populations Test Proportions</p> <p>$x_1, n_1, x_2, n_2, \alpha$</p> <p>One Population Variances</p> <p>n, s, o_0, α</p> <p>Two Population Variances</p> <p>$n_1, s_1, n_2, s_2, \alpha$</p>	<p>Case D</p> $s_p^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}$ $df = n_1 + n_2 - 2$ $t = \frac{(\bar{x}_1 - \bar{x}_2) - d_0}{s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$ <p>Case E</p> $t = \frac{(\bar{x}_1 - \bar{x}_2) - d_0}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$ $df = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{\left(\frac{s_1^2}{n_1}\right)^2}{n_1 - 1} + \frac{\left(\frac{s_2^2}{n_2}\right)^2}{n_2 - 1}}$ <p>Case F</p> $t = \frac{\bar{d} - d_0}{\left(s_d / \sqrt{n}\right)}$ $df = n - 1$ <p>Test Proportions for Two Population</p> $\hat{p}_1 = \frac{x_1}{n_1} \text{ and } \hat{p}_2 = \frac{x_2}{n_2} \quad \hat{p} = \frac{x_1 + x_2}{n_1 + n_2}$ $z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}\hat{q}\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}}$ <p>One Population Variances</p> $\chi^2 = \frac{(n - 1)s^2}{\sigma_0^2}$ $\nu = n - 1.$ <p>Two Population Variances</p>	
--	---	--

	$f = \frac{s_1^2}{s_2^2}$ <p>$v_1 = n_1 - 1$ and $v_2 = n_2 - 1$.</p> <p>Z-Statistic P-Value and Critical Point - Left p_value = st.norm.cdf(z_statistic)</p> <p>critical_value = st.norm.ppf(1 - alpha)</p> <p>Z-Statistic P-Value and Critical Point - Right critical_value = st.norm.ppf(1 - alpha)</p> <p>p_value = 1 - st.norm.cdf(z_statistic)</p> <p>Z-Statistic P-Value and Critical Point - Both alpha = alpha/2</p> <p>critical_value1 = st.norm.ppf(alpha)</p> <p>critical_value2 = st.norm.ppf(1 - alpha)</p> <p>p_value = 2 * (1 - st.norm.cdf(abs(z_statistic)))</p> <p>T-Statistic P-Value and Critical Point - Left critical_value = st.t.ppf(alpha, d_f)</p> <p>p_value = st.t.cdf(t_stat, d_f)</p> <p>T-Statistic P-Value and Critical Point - Right critical_value = st.t.ppf(1 -</p>	
--	---	--

	<p>alpha, d_f)</p> <p>p_value = st.t.cdf(1 - t_stat, d_f)</p> <p>T-Statistic P-Value and Critical Point - Both</p> <p>alpha = alpha/2</p> <p>critical_value1 = st.t.ppf(alpha, d_f)</p> <p>critical_value2 = st.t.ppf(1 - alpha, d_f)</p> <p>p_value = (1 - st.t.cdf(abs(t_stat), d_f)) * 2</p> <p>Single Population Variance P-Value</p> <p>p_value = st.chi2.sf(chi_squared, n - 1)</p> <p>Single Population Variance - Left</p> <p>critical_value = st.chi2.ppf(alpha, n - 1)</p> <p>Single Population Variance - Right</p> <p>critical_value = st.chi2.ppf(1 - alpha, n - 1)</p> <p>Single Population Variance - Both</p> <p>alpha = alpha/2</p> <p>critical_value1 = st.chi2.ppf(alpha, n - 1)</p> <p>critical_value2 = st.chi2.ppf(1 - alpha, n - 1)</p> <p>Two Population Variance P-Value</p> <p>p_value = st.f.sf(f_stats, v_1,</p>	
--	--	--

	$v_2)$ Two Population Variance - Left $\text{critical_value} = \text{st.f.ppf}(\alpha, v_1, v_2)$ Two Population Variance - Right $\text{critical_value} = \text{st.f.ppf}(1 - \alpha, v_1, v_2)$ Two Population Variance - Both $\alpha = \alpha/2$ $\text{critical_value1} = \text{st.f.ppf}(\alpha, v_1, v_2)$ $\text{critical_value2} = \text{st.f.ppf}(1 - \alpha, v_1, v_2)$	
--	---	--

Table 1.1

We can see that in the IPO figure found in Table 1.1, We can see that for the input, the user would input the data needed for the specific hypothesis test that he/she would like to do. In the processes, we can see that there is a formula for each test to get the test statistics (z or t or X or f), degree of freedom (df and v), proportions (p), pooled standard deviation (s_p^2), p-value (p_value), and critical values (critical_value, critical_value1, critical_value2). For the output, It would print out the following: test statistic, p-value (if any), critical value, degree of freedom (if any), sample proportions (if any), pooled sample proportion (if any), pooled standard deviation (if any), the difference (if any), difference standard deviation (if any), and whether to reject or not reject the null hypothesis

B. Hierarchy Chart

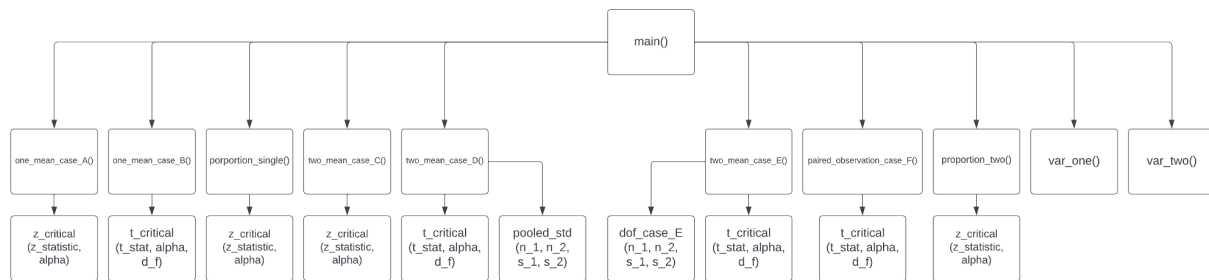


Figure 1.1

In the hierarchy chart found in Figure 1.1, we can see how each module are leaked to one another so that the program would be able to perform each selected tasks. It is important to have a hierarchy chart as this enables us to perform faster troubleshooting since we would know which are connected to which meaning that we will be able to find the broken function easily. We can also see that are several same functions being called and there is another example of the advantages of this which are that we would have fewer repeated codes meaning that modifying and adding new lines would be more efficient and troubleshooting would also be easier. An example would be the `main()` is connected to `one_mean_case_A()` and it is connected to `z_critical (Z_statistic, alpha)` to get the critical points and determine whether the program should reject or not reject the null hypothesis.

C. Flowchart

The Google Drive folder link below contains the flowchart for this project since when imported into the Word file, it would become very blurry and unreadable.

https://drive.google.com/drive/folders/10UjIdV-SWk1_epcxUHT7WwWsRHHYuHc1?usp=share_link

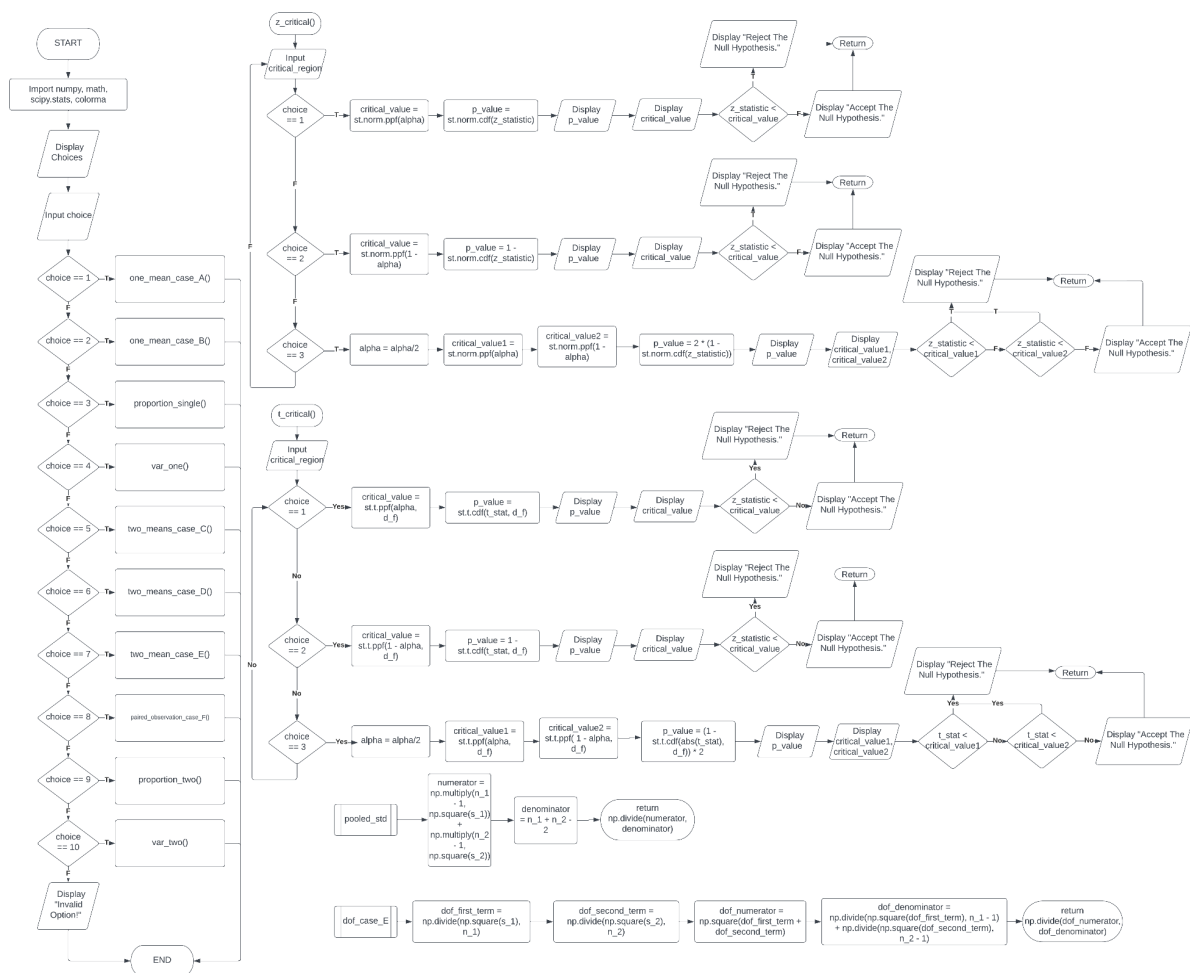


Figure 2.1

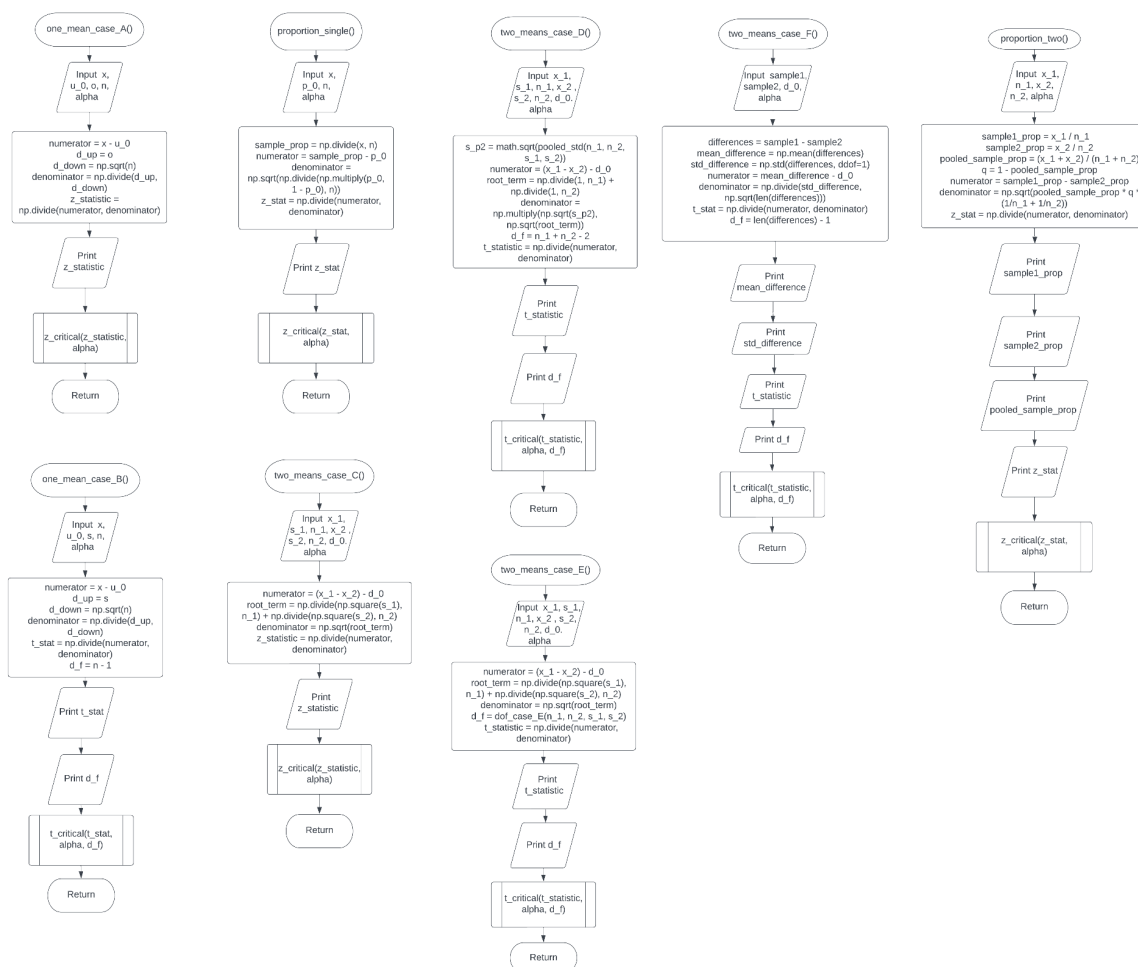


Figure 2.2

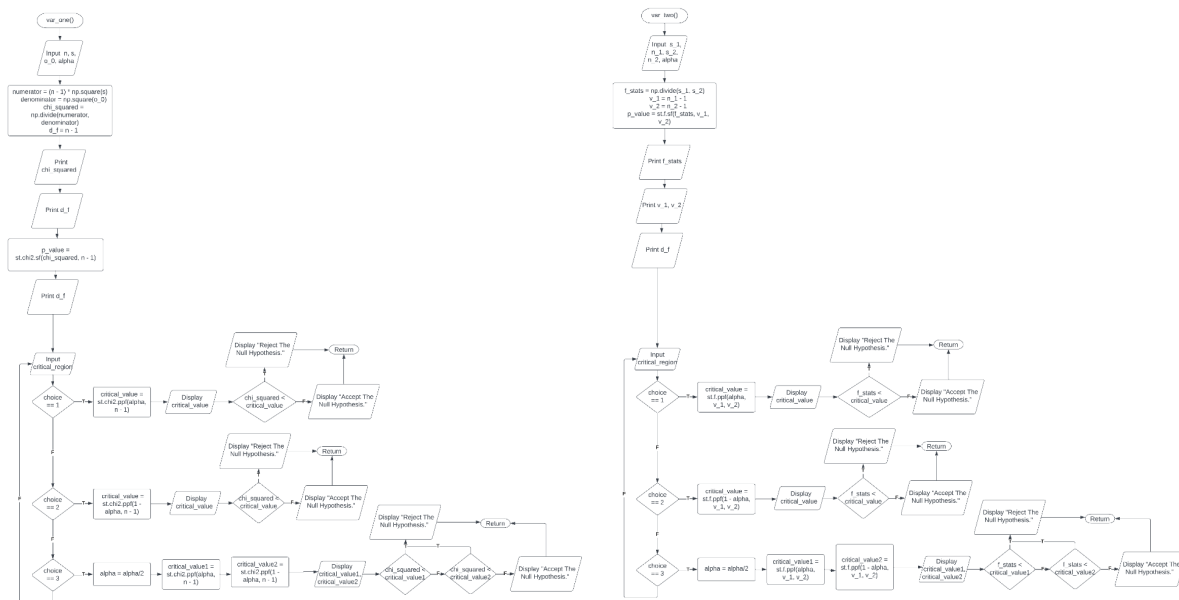


Figure 2.3

D. Pseudocode

importing of libraries

Import numpy as np

Import math

Import scipy.stats as st

from colorma import Fore, Style

print lines

def lines():

 Display"-----"

Z critical region test

```
def z_critical(z_statistic, alpha)
```

```
    # choosing of critical region
```

```
    Input critical region
```

```
    # left sided critical region
```

```
    If critical region = 1 Then:
```

```
        # calculation of the critical value and p-value
```

```
        Set critical value = st.norm.ppf alpha
```

```
        Set p value = st.norm.ppfz_statistic
```

```
        # displaying of the values
```

```
        Display p_value
```

```
        Display critical value
```

```
        # choosing to accept or reject the null hypothesis
```

```
        If z_statistic < critical_value then:
```

```
            Display "Reject the null hypothesis."
```

```
        else:
```

```
            Display "Accept the null hypothesis."
```

```
    # right sided critical region
```

```
    Else If critical region = 2 Then:
```

```
        # calculation of the critical value and p-value
```

```
        critical value = 1 - st.norm.ppfalpha
```

```
p value = 1- st.norm.ppf z_statistic
```

```
# displaying of the values
```

```
Display p_value
```

```
Display critical value
```

```
# choosing to accept or reject the null hypothesis
```

```
If z_statistic > critical_value then:
```

```
    Display "Reject the null hypothesis."
```

```
else:
```

```
    Display "Accept the null hypothesis."
```

```
Else If critical region = 2 Then:
```

```
# calculation of the two critical values and p-value
```

```
# the alpha is divided by 2 since there are 2 critical regions
```

```
alpha = alpha/2
```

```
critical value1 = st.norm.ppf alpha
```

```
critical value2 = st.norm.ppf 1-alpha
```

```
p value = 2*(1- st.norm.ppf z_statistic)
```

```
# displaying of the values
```

```
Display p_value
```

```
Display critical value1
```

```
Display critical value2
```

```
# choosing to accept or reject the null hypothesis
```

If $z_statistic < critical_value1$ or $z_statistic > critical_value2$ then:

Display "Reject the null hypothesis."

else:

Display "Accept the null hypothesis."

T critical region

def t_critical (T_stat, alpha, df)

choosing of critical region

Input critical_region

left sided critical region

If critical_region = 1 Then:

calculation of the critical value and p-value

critical_value = st.t.ppt (alpha, df)

p_value = st.t.cdf (t_stat df)

displaying of the values

Display p_value

Display critical_value

choosing to accept or reject the null hypothesis

If T_stat < critical_value then:

Display "Reject the null hypothesis"

Else:

Display "Accept the null hypothesis"

right sided critical region

Else If critical_region = 2 Then:

calculation of the critical value and p-value

```
critical_value = st.t.ppt (1 - alpha, df)
```

```
p_value = st.t.cdf (1 - t_stat df)
```

displaying of the values

```
Display p_value
```

```
Display critical_value
```

choosing to accept or reject the null hypothesis

```
If T_stat > critical_value then:
```

```
    Display "Reject the null hypothesis"
```

```
Else:
```

```
    Display "Accept the null hypothesis"
```

two sided critical region

```
Else If critical_region = 3 Then:
```

calculation of the two critical values and p-value**# the alpha is divided by 2 since there are 2 critical regions**

```
alpha = alpha/2
```

```
critical_value1 = st.t.ppt (alpha, df)
```

```
critical_value2 = st.t.ppt (1 - alpha, df)
```

```
p_value = st.t.cdf (1 - t_stat df) *2
```

displaying of the values

```
Display p_value
```

```
Display critical_value1
```

```
Display critical_value2
```

choosing to accept or reject the null hypothesis

If $T_{\text{stat}} < \text{critical_value1}$ or $T_{\text{stat}} > \text{critical_value2}$ then:

Display “Reject the null hypothesis”

Else:

Display “Accept the null hypothesis”

Case D Pooled Standard Deviation Claculation

def pooled_stn(n_1, n_2, s_1, s_2):

 numerator = $(n_1 - 1) * s_1^2 + (n_2 - 1) * s_2^2$

 denominator = $n_1 + n_2 - 2$

 return numerator / denominator

Case E Degree of Freedom

def dof_case_E(n_1, n_2, s_1, s_2):

 dof_first term = s_1^2 / n_1

 dof_second term = s_2^2 / n_2

 dof_numerator = $(\text{dof_first term} + \text{dof_second term})^2$

 dof_denominator = $(\text{dof_first term})^2 / (n_1 - 1) + (\text{dof_second term})^2 / (n_2)$

 return dof_numerator / dof_denominator

Case A: Single Population where sigma is known or sample is large

def one_mean_case_A():

asking for values

 Input x

 Input u_0

 Input o

 Input n

 Input alpha

calculation for the z statistics
$$\text{numerator} = x - u_0$$
$$d_up = o$$
$$d_down = \sqrt{n}$$
$$\text{denominator} = d_up / d_down$$
$$z_statistic = \text{numerator} / \text{denominator}$$
displaying the valueDisplay $z_statistic$ **# calling function for the remaining calculations (p-value, critical value, conclusion)**Call $z_critical(z_statistic, \alpha)$ **# Case B: Sigma is Unknown or Sample is Small**def $one_mean_case_B()$:**# asking for values**Input x Input u_0 Input o Input n Input α **# calculation for the t statistics and degree of freedom**
$$\text{numerator} = x - u_0$$
$$d_up = o$$
$$d_down = \sqrt{n}$$

```
denominator = d_up / d_down
```

```
t_stat = numerator / denominator
```

```
d_f = n - 1
```

```
# displaying of values
```

```
Display t_stat
```

```
Display d_f
```

```
# calling function for the remaining calculations (p-value, critical value, conclusion)
```

```
Call t_critical(t_stat, alpha, d_f)
```

```
# Test for Proportions: Single Population
```

```
# x is the number of “successes” in the sample, the sample size is  $n \geq 30$ 
```

```
def proportion_single():
```

```
# asking for values
```

```
Input x
```

```
Input n
```

```
Input p_0
```

```
Input alpha
```

```
# calculation for the sample proportion and z statistics
```

```
sample_prop = x / n
```

```
numerator = sample_prop - p_0
```

```
denominator = sqrt( (p_0* 1-p_0) / n)
```

```
z_stat = numerator / denominator
```

```
# displaying of value
```

Display z_stat

calling function for the remaining calculations (p-value, critical value, conclusion)

Call z_critical(z_statistic, alpha)

Case C: Two Mean Big Sample Size or Population Standard Deviation is Given

def two_means_case_C():

asking for values

Input x_1

Input s_1

Input n_1

Input x_2

Input s_2

Input n_2

Input d_0

Input alpha

calculation for the z statistics

numerator = (x_1 - x_2) - d_0

root_term = (1 / n_1) + (1 / n_2)

denominator = sqrt(root_term)

z_statistic = (numerator / denominator)

displaying of value

Display z_statistic

calling function for the remaining calculations (p-value, critical value, conclusion)

Call `z_critical(z_statistic, alpha)`

**# Case D: Two Mean Small Sample Size and Population Sample Standard Deviation
assumed Equal**

`def two_means_case_D():`

asking for values

Input `x_1`

Input `s_1`

Input `n_1`

Input `x_2`

Input `s_2`

Input `n_2`

Input `d_0`

Input `alpha`

calculation of the degree of freedom, pooled standard deviation, and t-statistic

`s_p2 = sqrt(pooled_std(n_1, n_2, s_1, s_2))`

`numerator = (x_1 - x_2) - d_0`

`root_term = (1 / n_1) + (1 / n_2)`

`denominator = sqrt(s_p2) * sqrt(root_term)`

`d_f = n_1 + n_2 - 2`

`t_statistic = (numerator / denominator)`

displaying of values

Display t_statistic

Display s_p2

Display d_f

calling function for the remaining calculations (p-value, critical value, conclusion)

Call t_critical(t_statistic, alpha, d_f)

Case E: Two Mean Small Sample Size and Population Sample Standard Deviation not Equal

def two_means_case_E():

asking for values

Input x_1

Input s_1

Input n_1

Input x_2

Input s_2

Input n_2

Input d_0

Input alpha

solving for the t-statistics

numerator = (x_1 - x_2) - d_0

root_term = ((s_1)^2 / n_1) + ((s_2)^2 / n_2)

denominator = sqrt(root_term)

t_statistic = (numerator / denominator)

calling the function to solve for the degree of freedom

d_f = dof_case_E(n_1, n_2, s_1, s_2)

displaying of values

Display t_statistic

Display d_f

calling function for the remaining calculations (p-value, critical value, conclusion)

Call t_critical(t_statistic, d_f)

Case F: Paired Observation

def one_mean_case_F():

asking for values

Input sample 1

Input sample 2

Input d_0

Input alpha

solving for the difference and difference standard deviation

differences = sample1 - sample2

mean_difference = np.mean(differences)

std_difference = np.std(differences, ddof=1)

calculation for the t-statistic and degree of freedom

```
numerator = mean_difference - d_0
denominator = std_difference / sqrt(differences)
t_stat = numerator / denominator
d_f = differences - 1
```

displaying of values

```
Display mean_differences
Display std_differences
Display t_stat
Display d_f
```

calling function for the remaining calculations (p-value, critical value, conclusion)

```
Call t_critical(t_stat, alpha, d_f)
```

Test for Proportions: Two Population

x is the number of “successes” in the sample, the sample size is $n \geq 30$

```
def proportion_two():
```

asking for values

```
Input x_1
```

```
Input n_1
```

```
Input x_2
```

```
Input n_2
```

```
Input alpha
```

solving for the 2 sample proportions and pooled sampled proportion

```
sample1_prop = (x_1 / n_1)
sample2_prop = (x_2 / n_2)
pooled_sample_prop = (x_1 + x_2) / (n_1 + n_2)
q = 1 - pooled_sample_prop
```

calculation for z-statistic value

```
numerator = sample1_prop - sample2_prop
denominator = sqrt(pooled_sample_prop * q * (1/n_1 + 1/n_2))
z_stat = numerator / denominator
```

displaying of values

```
Display sample1_prop
Display sample2_prop
Display pooled_sample_prop
Display z_stat
```

calling function for the remaining calculations (p-value, critical value, conclusion)

```
Call z_critical(z_stat, alpha)
```

Variance For Single Population

```
def var_one():
```

asking for values

```
Input n
Input s
```


Input σ_0

Input α

calculation for chi-square statistic and degree of freedom

$\text{numerator} = (n - 1) * s^2$

$\text{denominator} = \sigma_0^2$

$\text{chi_squared} = \text{numerator} / \text{denominator}$

$d_f = n - 1$

displaying of values

Display chi_squared

Display d_f

$p_value = \text{st.chi2.sf}(\text{chi_squared}, n - 1)$

Display p_value

choosing of critical region

Input critical_region

left sided critical region

If $\text{critical_region} == 1$ then:

calculating and displaying the value

$\text{critical_value} = \text{st.chi2.ppf}(\alpha, n - 1)$

Display critical_value

choosing whether to accept or reject the null hypothesis

if $\text{chi_squared} < \text{critical_value}$ then:

```
        Display "Reject the Null Hypothesis"
    else:
        Display "Accept the Null Hypothesis"
```

right sided critical region

```
Elif critical_region == 2 then:
```

calculating and displaying the critical value

```
critical_value = st.chi2.ppf(1 - alpha, n - 1)
Display critical_value
```

choosing whether to accept or reject the null hypothesis

```
if chi_squared > critical_value then:
    Display "Reject the Null Hypothesis"
else:
    Display "Accept the Null Hypothesis"
```

two sided critical region

```
Elif critical_region == 3 then:
```

calculating and displaying the two critical values

alpha is divided by 2 since there are 2 critical regions

```
critical_value1 = st.chi2.ppf(alpha, n - 1)
critical_value2 = st.chi2.ppf(1 - alpha, n - 1)
Display critical_value1
Display critical_value2
```

choosing whether to accept or reject the null hypothesis

```
if chi_squared < critical_value or chi_squared > critical_value2 then:  
    Display "Reject the Null Hypothesis"  
else:  
    Display "Accept the Null Hypothesis"
```

Variance For Two Populations

```
def var_two():
```

```
    # asking for values
```

```
    Input n_1
```

```
    Input s_1
```

```
    Input n_2
```

```
    Input s_2
```

```
    Input alpha
```

```
    # calculation for the f-statistic, 2 degree of freedom, and p-value
```

```
    f_stats = s_1 / s_2
```

```
    v_1 = n_1 - 1
```

```
    v_2 = n_2 - 1
```

```
    p_value = st.f.sf(f_stats, v_1, v_2)
```

```
    # displaying of values
```

```
    Display f_stats
```

```
    Display v_1
```

```
    Display v_2
```

```
    Display p_value
```

```
    # choosing of critical region
```

Input critical_region

left sided critical region

If critical_region == 1 then:

calculating and displaying the critical value

critical_value = st.chi2.ppf(alpha, v_1, v_2)

Display critical_value

choosing whether to accept or reject the null hypothesis

if chi_squared < critical_value then:

Display "Reject the Null Hypothesis"

else:

Display "Accept the Null Hypothesis"

right sided critical region

Elif critical_region == 2 then:

calculating and displaying the critical value

critical_value = st.chi2.ppf(1 - alpha, v_1, v_2)

Display critical_value

choosing whether to accept or reject the null hypothesis

if chi_squared > critical_value then:

Display "Reject the Null Hypothesis"

else:

Display "Accept the Null Hypothesis"

two sided critical region

Elif critical_region == 3 then:

calculating and displaying the critical value

alpha is divided by 2 since there are 2 critical regions

alpha = alpha/2

critical_value1 = st.chi2.ppf(alpha, v_1, v_2)

critical_value2 = st.chi2.ppf(1 - alpha, v_1, v_2)

Display critical_value1

Display critical_value2

choosing whether to accept or reject the null hypothesis

if chi_squared < critical_value or chi_squared > critical_value2 then:

Display "Reject the Null Hypothesis"

else:

Display "Accept the Null Hypothesis"

Main Program that will print the welcome message

It also shows choice for different choices for the user to choose from

Display "Welcome to Hippopotamus - Hypothesis Testing Calculator"

Display "There following are the choices for Single Population:"

Display "1. Case A: Single Population Where Sigma is Known or Sample is Large"

Display "2. Case B: Sigma is Unknown or Sample is Small"

Display "3. Test for Proportions: Single Population"

Display "4. Variance For Single Population"

lines()

Display “The following are the choices for Two Population:”

Display “5. Case C: Two Mean Big Sample Size or Population Standard Deviation is Given”

Display “6. Case D: Two Mean Small Sample Size and Population Sample Standard Deviation assumed Equal”

Display “7. Case E: Two Mean Small Sample Size and Population Sample Standard Deviation not Equal”

Display “8. Case F: Paired Observation”

Display “9. Test for Proportions: Two Population”

Display “10. Variance For Two Populations”

lines()

Input choice

if-else statement to execute specified task based on choice option

if the choice is not in the selected range then the program will display "invalid option"

If choice == 1 then:

 one_mean_case_A()

Elif choice == 2 then:

 one_mean_case_B()

Elif choice == 3 then:

 proportion_single()

Elif choice == 4 then:

 var_one()

Elif choice == 5 then:

 two_means_case_C()

Elif choice == 6 then:

 two_means_case_D()

Elif choice == 7 then:

 two_means_case_E()

Elif choice == 8 then:

 paired_observation_case_F()

Elif choice == 9 then:

 proportion_two()

Elif choice == 10 then:

 var_two()

Else:

 Display “Invalid Option!”

IV. Results

Test #1 - Example 8.3.1 from Hypothesis Testing of Single Mean LibreText Statistics

```
Welcome to Hippotamus - Hypothesis Testing Calculator
-----
There following are the choices for Single Population:
-----
1. Case A: Single Population Where Sigma is Known or Sample is Large
2. Case B: Sigma is Unknown or Sample is Small
3. Test for Proportions: Single Population
4. Variance For Single Population
-----
There following are the choices for Two Population:
-----
5. Case C: Two Mean Big Sample Size or Population Stnadard Deviation is Given
6. Case D: Two Mean Small Sample Size and Population Sample Standard Deviation assumed Equal
7. Case E: Two Mean Small Sample Size and Population Sample Standard Deviation not Equal
8. Case F: Paired Observation
9. Test for Proportions: Two Population
10. Variance For Two Populations
-----
Enter your choice: 1
-----
You have Chosen Case A: Single Population where sigma is known or sample is large.
Enter the Sample Mean: 169
Enter the Population Mean: 179
Enter the Population Standard Deviation: 10.39
Enter the Sample Size: 5
Enter the Significance Level: 0.05
-----
The Z Score is -2.1521.
-----
Enter the Critical Region (1) Left, (2) Right, (3) Both: 1
-----
The p-value is 0.0156933725.
The Critical Value is -1.6449.
Reject The Null Hypothesis.
```

Test #2 - Example from FNDSTAT Homework 2 by Ms. Cherry Frondoza

```

Welcome to Hippotamus - Hypothesis Testing Calculator
-----
There following are the choices for Single Population:
-----
1. Case A: Single Population Where Sigma is Known or Sample is Large
2. Case B: Sigma is Unknown or Sample is Small
3. Test for Proportions: Single Population
4. Variance For Single Population
-----
There following are the choices for Two Population:
-----
5. Case C: Two Mean Big Sample Size or Population Stnadard Deviation is Given
6. Case D: Two Mean Small Sample Size and Population Sample Standard Deviation assumed Equal
7. Case E: Two Mean Small Sample Size and Population Sample Standard Deviation not Equal
8. Case F: Paired Observation
9. Test for Proportions: Two Population
10. Variance For Two Populations
-----
Enter your choice: 9
-----
You have Chosen Test for Proportions: Two Population.
Enter the 1st Number of Successes: 280
Enter the 1st Sample Size: 500
Enter the 2nd Number of Successes: 230
Enter the 2nd Sample Size: 500
Enter the Significance Level: 0.01
The 1st Sample Proportion is 0.56.
The 2nd Sample Proportion is 0.46.
The Pooled Sample Proportion is 0.51.
The Z-Statistic is 3.1629.
-----
Enter the Critical Region (1) Left, (2) Right, (3) Both: 2
-----
The p-value is 0.0007810022.
The Critical Value is 2.3263.
Reject The Null Hypothesis.

```

Test 3 - Example 9.1.2 from Comparison of Two Population Means- Large, Independent Samples
LibreText Statistics

```

Welcome to Hippotamus - Hypothesis Testing Calculator
-----
There following are the choices for Single Population:
-----
1. Case A: Single Population Where Sigma is Known or Sample is Large
2. Case B: Sigma is Unknown or Sample is Small
3. Test for Proportions: Single Population
4. Variance For Single Population
-----
There following are the choices for Two Population:
-----
5. Case C: Two Mean Big Sample Size or Population Stnadard Deviation is Given
6. Case D: Two Mean Small Sample Size and Population Sample Standard Deviation assumed Equal
7. Case E: Two Mean Small Sample Size and Population Sample Standard Deviation not Equal
8. Case F: Paired Observation
9. Test for Proportions: Two Population
10. Variance For Two Populations
-----
You have Chosen Case C: Two Mean Big Sample Size or Population Stnadard Deviation is Given.
Enter the 1st Sample Mean: 3.51
Enter the 1st Sample Standard Deviation: 0.51
Enter the 1st Sample Size: 174
Enter the 2nd Sample Mean: 3.24
Enter the 2nd Sample Standard Deviation: 0.52
Enter the 2nd Sample Size: 355
Enter the Difference Between the Population Means: 0
Enter the Significance Level: 0.01
The Z-Statistic is 5.6839.
-----
Enter the Critical Region (1) Left, (2) Right, (3) Both: 2
-----
The p-value is 6.6e-09.
The Critical Value is 2.3263.
Reject The Null Hypothesis.

```


V. Discussion of Results

The project on making the hypothesis testing calculator has been successful. Hypothesis testing is a critical component of statistical analysis that enables researchers to make informed decisions about the significance of their results. The successful implementation of a hypothesis testing calculator can help simplify the process for researchers and analysts, making their work more efficient and effective.

One of the key benefits of the hypothesis testing calculator is its ability to help users understand the statistical significance of their data. With this tool, users can easily input their data and run statistical tests to determine whether the results are significant or not. This can be particularly useful for researchers who may not have the statistical background necessary to perform these tests manually.

The success of the project also demonstrates the potential of machine learning and artificial intelligence to improve research and analysis in various fields. The use of machine learning algorithms can help automate many of the complex calculations involved in hypothesis testing, reducing the risk of errors and improving the accuracy of the results.

Another benefit of the hypothesis testing calculator is that it can help researchers identify patterns and trends in their data that may not have been immediately apparent. By using statistical tests to analyze their data, researchers can identify correlations and other relationships that may not have been immediately apparent. This can help inform future research and analysis and improve the accuracy of the results.

VI. Analysis, Conclusion, and Future Directives

For the analysis, the project involved creating a hypothesis testing calculator that enables users to perform statistical hypothesis testing. The project was successful in achieving its objectives, as the calculator provided accurate and reliable results for hypothesis testing. The calculator was designed to be user-friendly, with easy-to-use features and a straightforward interface that allows users to input their data and get the results quickly.

Coming to the conclusion, the successful completion of the hypothesis testing calculator project is a significant achievement. This tool has the potential to simplify the process of

hypothesis testing, making it more accessible to researchers and analysts. Additionally, the use of machine learning algorithms can help automate many of the complex calculations involved in statistical analysis, improving the accuracy and efficiency of the process. Overall, this project demonstrates the potential of machine learning and artificial intelligence to improve research and analysis in various fields.

As the project has been successful, there are several future directions that can be explored. For instance, the calculator can be further enhanced by integrating more advanced statistical methods, such as multiple regression analysis and factor analysis. Additionally, the calculator can be made more interactive by providing users with the ability to visualize the data and results, making it easier for them to interpret and communicate their findings. Moreover, the calculator can be integrated into larger software applications to enhance their analytical capabilities. In summary, the successful development of the hypothesis testing calculator has opened up a new frontier in statistical analysis, and further research and development can lead to even more exciting possibilities in the future.

References

- Allua, S., & Thompson, C. A. (2009). Hypothesis Testing. *Air Medical Journal*, 28(3), 108–153.
<https://doi.org/10.1016/j.amj.2009.03.002>
- Altman, D. G., & Bland, J. M. (1994). Diagnostic tests 2: Predictive values. *BMJ (Clinical research ed.)*, 309(6947), 102. <https://doi.org/10.1136/bmj.309.6947.102>
- Babbie, E. R. (2017). *The Basics of Social Research, Fourth Edition. Thomson Higher Education*. ISBN-13: 978-0-495-09468-5
- Berger, J. M., & Mortera, J. (1991). Interpreting the Stars in Precise Hypothesis Testing. *International Statistical Review*. <https://doi.org/10.2307/1403691>
- Cohen, J. (1994). The earth is round ($p < .05$). *American Psychologist*, 49(12), 997–1003.
<https://doi.org/10.1037/0003-066X.49.12.997>
- Davis, R. J., & Mukamal, K. J. (2006). Hypothesis Testing. *Circulation*, 114(10), 1078–1082.
<https://doi.org/10.1161/circulationaha.105.586461>
- Freedman, D., Pisani, R., & Purves, R. (2007). *Statistics. WW Norton & Company*.
- Field, A. P. (2017). *Discovering statistics using IBM SPSS statistics, 5th edition. SAGE Publications eBooks*. <http://sro.sussex.ac.uk/id/eprint/77862/>
- Fisher, R. (1926). Statistical Methods for Research Workers. *Journal of the Royal Statistical Society*, 92(1), 101. <https://doi.org/10.2307/2341440>
- Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2019). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6), 82-97.
<https://doi.org/10.1109/MSP.2012.2205597>
- Hogg, R. S., & Tanis, E. A. (2008). Probability and statistical inference. *Choice Reviews Online*, 45(11), 45–6219. <https://doi.org/10.5860/choice.45-6219>
- Klein, J.P., Moeschberger, M.L. (2003). Hypothesis Testing. In: *Survival Analysis. Statistics for Biology and Health*. Springer, New York, NY. https://doi.org/10.1007/0-387-21645-6_7

Kline, R. B. (2015). *Principles and Practice of Structural Equation Modeling*, Fourth Edition. Guilford Publications.

Mathematics and Statistics Department. (2022). FNDSTAT Hypothesis Testing Formula Sheet For Variances. De La Salle University. <https://drive.google.com/drive/folders/1Y1mhqy8yeHQB9YC7WBPkQCNMH8Icvorh?usp=sharing>

Mathematics and Statistics Department. (2022). FNDSTAT Hypothesis Testing Formula Sheet for Two Populations. De La Salle University. <https://drive.google.com/drive/folders/1Y1mhqy8yeHQB9YC7WBPkQCNMH8Icvorh?usp=sharing>

Mathematics and Statistics Department. (2022). FNDSTAT Hypothesis Testing Formula Sheet for Single Populations. De La Salle University. <https://drive.google.com/drive/folders/1Y1mhqy8yeHQB9YC7WBPkQCNMH8Icvorh?usp=sharing>

Maxwell, S. E., & Delaney, H. D. (2004). Designing experiments and analyzing data: A model comparison perspective. *Psychology Press*. <https://doi.org/10.4324/9781315642956>

Moore, D., & McCabe, G. P. (2004). Introduction to the Practice of Statistics. *Technometrics*, 46(1), 118–119. <https://doi.org/10.1198/tech.2004.s753>

Patten, M. L. (2018). *Understanding research methods: An overview of the essentials*. Routledge.

Rosnow, R. L., & Rosenthal, R. (1991). Effect sizes for experimenting psychologists. *Canadian Journal of Experimental Psychology*, 45(3), 308. <https://awspntest.apa.org/doi/10.1037/h0087427>

Welch, B. L. (1938). The Significance of the Difference Between Two Means when the Population Variances are Unequal. *Biometrika*, 29(3/4), 350–362. <https://doi.org/10.2307/2332010>

Zhang, Z. (2015). A flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behavior research methods*, 47(2), 800-816. <https://doi.org/10.3758/bf03193146>

Appendices

A. User's Manual

1. The program would display an introduction and choice of hypothesis testing scenario.
2. The user would input the selected choice with the number beside the scenario of choice although the choices are only from 1-10 and if the user inputs a value greater than or lesser than the range then the program would show “**Invalid Option!**”
3. After choosing the scenario, the program would ask for the values needed and would also display the important values such as test statistic, p-value, degree of freedom, etc.
4. The user would then be asked to choose the side of critical region (left, right, both sides) and the user would input the number value of the choice being 1 for left, 2 for right, and 3 for both sides.
5. The program would display the critical value/s and determine whether to accept or reject the null hypothesis.

B. Source Code

```
# importing libraries

import numpy as np
import math
import scipy.stats as st
from colorama import Fore, Style

# Print Lines
def lines():
    print("-----")

# Z critical region test
def z_critical(z_statistic, alpha):
```

```
# choosing of critical region

lines()

critical_region = int(input(Fore.YELLOW + "Enter the Critical Region (1) Left, (2)
Right, (3) Both: " + Style.RESET_ALL))

lines()

# left sided critical region

if critical_region == 1:

    #calculation of the critical value and p-value

    critical_value = st.norm.ppf(alpha)

    p_value = st.norm.cdf(z_statistic)

# displaying of the values

print("The p-value is " + str(round(p_value,10)) + ".")

print("The Critical Value is " + str(round(critical_value,4)) + ".")

# choosing to accept or reject the null hypothesis

if z_statistic < critical_value:

    print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)

else:

    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

# right sided critical region

elif critical_region == 2:

    #computation for the critical value and the p-value

    critical_value = st.norm.ppf(1 - alpha)

    p_value = 1 - st.norm.cdf(z_statistic)

#printing of the values
```

```
print("The p-value is " + str(round(p_value,10)) + ".")
print("The Critical Value is " + str(round(critical_value,4)) + ".")

# choosing the accept or reject the null hypothesis
if z_statistic > critical_value:
    print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)
else:
    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

# two-sided critical region
elif critical_region == 3:
    # calculation of the two critical values and p-value
    # the alpha is divided by 2 since there are 2 critical regions
    alpha = alpha/2
    critical_value1 = st.norm.ppf(alpha)
    critical_value2 = st.norm.ppf(1 - alpha)
    p_value = 2 * (1 - st.norm.cdf(abs(z_statistic)))

#printing of the values
print("The p-value is " + str(round(p_value,10)) + ".")
print("The Left Critical Value is " + str(round(critical_value1,4)) + ".")
print("The Right Critical Value is " + str(round(critical_value2,4)) + ".")

# choosing the accept or reject the null hypothesis
if z_statistic < critical_value1 or z_statistic > critical_value2:
    print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)
else:
    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)
```

```
# T critical Region
def t_critical(t_stat, alpha, d_f):
    # choosing of critical region
    lines()

    critical_region = int(input(Fore.YELLOW + "Enter the Critical Region (1) Left, (2)
    Right, (3) Both: " + Style.RESET_ALL))
    lines()

    # left sided critical region
    if critical_region == 1:
        #calculation of the critical value and p-value
        critical_value = st.t.ppf(alpha, d_f)
        p_value = st.t.cdf(t_stat, d_f)

        #displaying of the values
        print("The P-Value is " + str(round(p_value, 5)) + ".")
        print("The Critical Value is " + str(round(critical_value, 4)) + ".")

        # choosing the accept or reject the null hypothesis
        if t_stat < critical_value:
            print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)
        else:
            print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

    # right sided critical region
    elif critical_region == 2:
        # calculation of the critical value and p-value
        critical_value = st.t.ppf(1 - alpha, d_f)
```



```

p_value = 1 - st.t.cdf(t_stat, d_f)

#printing of the values
print("The P-Value is " + str(round(p_value, 5)) + ".")
print("The Critical Value is " + str(round(critical_value, 4)) + ".")

# choosing the accept or reject the null hypothesis
if t_stat > critical_value:
    print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)
else:
    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

#two sided critical value
elif critical_region == 3:
    # calculation of the two critical values and p-value
    # the alpha is divided by 2 since there are 2 critical regions
    alpha = alpha/2
    critical_value1 = st.t.ppf(alpha, d_f)
    critical_value2 = st.t.ppf(1 - alpha, d_f)
    p_value = (1 - st.t.cdf(abs(t_stat), d_f)) * 2

    #printing of the values
    print("The P-Value is " + str(round(p_value, 5)) + ".")
    print("The Left Critical Value is " + str(round(critical_value1, 4)) + ".")
    print("The Right Critical Value is " + str(round(critical_value2, 4)) + ".")

    # choosing the accept or reject the null hypothesis
    if t_stat < critical_value1 or t_stat > critical_value2:

```

```

        print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)
    else:
        print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

# Case D Pooled Standard Deviation Claculation
def pooled_std(n_1, n_2, s_1, s_2):
    numerator = np.multiply(n_1 - 1, np.square(s_1)) + np.multiply(n_2 - 1,
np.square(s_2))
    denominator = n_1 + n_2 - 2
    return np.divide(numerator, denominator)

# Case E Degree of Freedom
def dof_case_E(n_1, n_2, s_1, s_2):
    dof_first_term = np.divide(np.square(s_1), n_1)
    dof_second_term = np.divide(np.square(s_2), n_2)
    dof_numerator = np.square(dof_first_term + dof_second_term)
    dof_denominator = np.divide(np.square(dof_first_term), n_1 - 1) +
np.divide(np.square(dof_second_term), n_2 - 1)

    return np.divide(dof_numerator, dof_denominator)

# Case A: Single Population where sigma is known or sample is large
def one_mean_case_A():
    # asking for values

    print(Fore.GREEN + "You have Chosen Case A: Single Population where sigma is
known or sample is large." + Style.RESET_ALL)

    x = float(input("Enter the Sample Mean: "))
    u_0 = float(input("Enter the Population Mean: "))
    o = float(input("Enter the Population Standard Deviation: "))

```

```
n = int(input("Enter the Sample Size: "))
alpha = float(input("Enter the Significance Level: "))

# calculation for the z statistics
numerator = x - u_0
d_up = o
d_down = np.sqrt(n)
denominator = np.divide(d_up, d_down)
z_statistic = np.divide(numerator, denominator)

# displaying the value
lines()
print("The Z-Statistic Value is " + str(round(z_statistic,4)) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)
z_critical(z_statistic, alpha)

# Case B: Sigma is Unknown or Sample is Small
def one_mean_case_B():
    # asking for values
    print(Fore.GREEN + "You have chosen Case B: Sigma is Unknown or Sample is
Small." + Style.RESET_ALL)
    x = float(input("Enter the Sample Mean: "))
    u_0 = float(input("Enter the Population Mean: "))
    s = float(input("Enter the Sample Standard Deviation: "))
    n = int(input("Enter the Sample Size: "))
    alpha = float(input("Enter the Significance Level: "))
```

```

# calculation for the t statistics and degree of freedom
numerator = x - u_0
d_up = s
d_down = np.sqrt(n)
denominator = np.divide(d_up, d_down)
t_stat = np.divide(numerator, denominator)
d_f = n - 1

# displaying of values
lines()
print("The T-Statistic Value is " + str(round(t_stat, 4)) + ".")
print("The Degree of Freedom is " + str(d_f) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)
t_critical(t_stat, alpha, d_f)

# Test for Proportions: Single Population
# x is the number of “successes” in the sample, the sample size is  $n \geq 30$ 
def proportion_single():
    # asking for values
    print(Fore.GREEN + "You have Chosen Test for Proportions: Single Population." +
Style.RESET_ALL)
    x = int(input("Enter the Number of Successes: "))
    n = int(input("Enter the Sample Size: "))
    p_0 = float(input("Enter the Population Proportion: "))
    alpha = float(input("Enter the Significance Level: "))

    # calculation for the sample proportion and the z statistics

```

```
sample_prop = np.divide(x, n)
numerator = sample_prop - p_0
denominator = np.sqrt(np.divide(np.multiply(p_0, 1 - p_0), n))
z_stat = np.divide(numerator, denominator)

#displaying of value
lines()
print("The Z-Statistic Value is " + str(round(z_stat, 4)) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)
z_critical(z_stat, alpha)

# Case C: Two Mean Big Sample Size or Population Standard Deviation is Given
def two_means_case_C():
    # asking for values
    print(Fore.GREEN + "You have Chosen Case C: Two Mean Big Sample Size or
Population Standard Deviation is Given." + Style.RESET_ALL)
    x_1 = float(input("Enter the 1st Sample Mean: "))
    s_1 = float(input("Enter the 1st Sample Standard Deviation: "))
    n_1 = int(input("Enter the 1st Sample Size: "))
    x_2 = float(input("Enter the 2nd Sample Mean: "))
    s_2 = float(input("Enter the 2nd Sample Standard Deviation: "))
    n_2 = int(input("Enter the 2nd Sample Size: "))
    d_0 = float(input("Enter the Difference Between the Population Means: "))
    alpha = float(input("Enter the Significance Level: "))

    # calculation for the z statistics
    numerator = (x_1 - x_2) - d_0
```

```
root_term = np.divide(np.square(s_1), n_1) + np.divide(np.square(s_2), n_2)
denominator = np.sqrt(root_term)
z_statistic = np.divide(numerator, denominator)

# displaying of value
print("The Z-Statistic Value is " + str(round(z_statistic, 4)) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)
z_critical(z_statistic, alpha)

# Case D: Two Mean Small Sample Size and Population Sample Standard Deviation
assumed Equal
def two_means_case_D():
    # asking for values
    print(Fore.GREEN + "You have Chosen Case D: Two Mean Small Sample Size and
Population Sample Standard Deviation assumed Equal." + Style.RESET_ALL)
    x_1 = float(input("Enter the 1st Sample Mean: "))
    s_1 = float(input("Enter the 1st Sample Standard Deviation: "))
    n_1 = int(input("Enter the 1st Sample Size: "))
    x_2 = float(input("Enter the 2nd Sample Mean: "))
    s_2 = float(input("Enter the 2nd Sample Standard Deviation: "))
    n_2 = int(input("Enter the 2nd Sample Size: "))
    d_0 = float(input("Enter the Difference Between the Population Means: "))
    alpha = float(input("Enter the Significance Level: "))
    lines()

# calculation of the degree of freedom, pooled standard deviation, and t-statistic
s_p2 = math.sqrt(pooled_std(n_1, n_2, s_1, s_2))
```

```

numerator = (x_1 - x_2) - d_0
root_term = np.divide(1, n_1) + np.divide(1, n_2)
denominator = np.multiply(np.sqrt(s_p2), np.sqrt(root_term))
d_f = n_1 + n_2 - 2
t_statistic = np.divide(numerator, denominator)

# displaying of values
print("The T-Statistic is " + str(round(t_statistic, 4)) + ".")
print("The Pooled Standard Deviation is " + str(round(s_p2,4)) + ".")
print("The Degree of Freedom is " + str(d_f) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)

t_critical(t_statistic, alpha, d_f)

# Case E: Two Mean Small Sample Size and Population Sample Standard Deviation not
Equal
def two_means_case_E():
    # asking for values
    print(Fore.GREEN + "You have Chosen Case E: Two Mean Small Sample Size and
Population Sample Standard Deviation not Equal." + Style.RESET_ALL)
    x_1 = float(input("Enter the 1st Sample Mean: "))
    s_1 = float(input("Enter the 1st Sample Standard Deviation: "))
    n_1 = int(input("Enter the 1st Sample Size: "))
    x_2 = float(input("Enter the 2nd Sample Mean: "))
    s_2 = float(input("Enter the 2nd Sample Standard Deviation: "))
    n_2 = int(input("Enter the 2nd Sample Size: "))
    d_0 = float(input("Enter the Difference Between the Population Means: "))
    alpha = float(input("Enter the Significance Level: "))

```

```

# solving for the t-statistics
numerator = (x_1 - x_2) - d_0
root_term = np.divide(np.square(s_1), n_1) + np.divide(np.square(s_2), n_2)
denominator = np.sqrt(root_term)
t_statistic = np.divide(numerator, denominator)

# calling the function to solve for the degree of freedom
d_f = dof_case_E(n_1, n_2, s_1, s_2)

# displaying of values
lines()
print("The T-Statistic is " + str(round(t_statistic, 4)) + ".")
print("The Degree of Freedom is " + str(round(d_f, 4)) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)
t_critical(t_statistic, alpha, d_f)

# Case F: Paired Observation
def paired_observation_case_F():
    # asking for values
    print(Fore.GREEN + "You have Chosen Case F: Paired Observation." +
Style.RESET_ALL)

    sample1 = np.array([float(x) for x in input("Enter the data for the 1st Sample (Separate
by Spaces): ").split()])

    sample2 = np.array([float(x) for x in input("Enter the data for the 2nd Sample
(Separated by Spaces): ").split()])

    d_0 = int(input("Enter the Difference Between the Samples: "))

    alpha = float(input("Enter the Significance Level: "))

```



```
# solving for the difference and difference standard deviation
differences = sample1 - sample2
mean_difference = np.mean(differences)
std_difference = np.std(differences, ddof=1)

# calculation for the t-statistic and degree of freedom
numerator = mean_difference - d_0
denominator = np.divide(std_difference, np.sqrt(len(differences)))
t_stat = np.divide(numerator, denominator)
d_f = len(differences) - 1

# displaying of values
lines()
print("The Difference is " + str(round(mean_difference, 4)) + ".")
print("The Difference Standard Deviation is " + str(round(std_difference, 4)) + ".")
print("The T-Statistic is " + str(round(t_stat, 4)) + ".")
print("The Degree of Freedom is " + str(round(d_f, 4)) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)
t_critical(t_stat, alpha, d_f)

# Test for Proportions: Two Population
# x is the number of “successes” in the sample, the sample size is  $n \geq 30$ 
def proportion_two():
    # asking for values
    print(Fore.GREEN + "You have Chosen Test for Proportions: Two Population." +
Style.RESET_ALL)
```

```
x_1 = int(input("Enter the 1st Number of Successes: "))
n_1 = int(input("Enter the 1st Sample Size: "))
x_2 = int(input("Enter the 2nd Number of Successes: "))
n_2 = int(input("Enter the 2nd Sample Size: "))
alpha = float(input("Enter the Significance Level: "))

# solving for the 2 sample proportions and pooled sampled proportion
sample1_prop = x_1 / n_1
sample2_prop = x_2 / n_2
pooled_sample_prop = (x_1 + x_2) / (n_1 + n_2)
q = 1 - pooled_sample_prop

# calculation for z-statistic Value
numerator = sample1_prop - sample2_prop
denominator = np.sqrt(pooled_sample_prop * q * (1/n_1 + 1/n_2))
z_stat = np.divide(numerator, denominator)

# displaying of values
print("The 1st Sample Proportion is " + str(round(sample1_prop,4)) + ".")
print("The 2nd Sample Proportion is " + str(round(sample2_prop,4)) + ".")
print("The Pooled Sample Proportion is " + str(round(pooled_sample_prop,4)) + ".")
print("The Z-Statistic Value is " + str(round(z_stat, 4)) + ".")

# calling the function for the remaining calculations (p-value, critical value,
conclusion)

z_critical(z_stat, alpha)

# Variance For Single Population
def var_one():
```

```
# asking for values

print(Fore.GREEN + "You have Chosen Variance For Single Population." +
Style.RESET_ALL)

n = int(input("Enter the Sample Size: "))
s = float(input("Enter the Sample Variance: "))
o_0 = float(input("Enter the Population Variance: "))
alpha = float(input("Enter the Significance Level: "))

# calculation for chi-square statistic and degree of freedom
numerator = (n - 1) * np.square(s)
denominator = np.square(o_0)
chi_squared = np.divide(numerator, denominator)
d_f = n - 1

# displaying of values
lines()
print("The Chi-Square Value is " + str(round(chi_squared,4)) + ".")
print("The Degree of Freedom is " + str(d_f) + ".")

# calculating and displaying the p-value
p_value = st.chi2.sf(chi_squared, n - 1)
print("The P-Value is " + str(round(p_value,6)) + ".")
lines()

# choosing of critical region
critical_region = int(input(Fore.YELLOW + "Enter the Critical Region (1) Left, (2)
Right, (3) Both: " + Style.RESET_ALL))
lines()
```

```
# left sided critical region
if critical_region == 1:
    # calculating and displaying the critical value
    critical_value = st.chi2.ppf(alpha, n - 1)
    print("The Critical Value is " + str(round(critical_value,4)) + ".")

    # choosing whether to accept or reject the null hypothesis
    if chi_squared < critical_value:
        print(Fore.RED + "Reject The Null Hypothesis."+ Style.RESET_ALL)
    else:
        print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

# right sided critical region
elif critical_region == 2:
    # calculating and displaying the critical value
    critical_value = st.chi2.ppf(1 - alpha, n - 1)
    print("The Critical Value is " + str(round(critical_value,4)) + ".")

    # choosing whether to accept or reject the null hypothesis
    if chi_squared > critical_value:
        print(Fore.RED + "Reject The Null Hypothesis."+ Style.RESET_ALL)
    else:
        print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

# two sided critical region
elif critical_region == 3:
    # calculating and displaying the two critical values
    # alpha is divided by 2 since there are 2 critical regions
```

```

alpha = alpha/2
critical_value1 = st.chi2.ppf(alpha, n - 1)
critical_value2 = st.chi2.ppf(1 - alpha, n - 1)
print("The Left Critical Value is " + str(round(critical_value1,4)) + ".")
print("The Right Critical Value is " + str(round(critical_value2,4)) + ".")

# choosing whether to accept or reject the null hypothesis
if chi_squared < critical_value1 or chi_squared > critical_value2:
    print(Fore.RED + "Reject The Null Hypothesis."+ Style.RESET_ALL)
else:
    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

# Variance For Two Populations
def var_two():
    # asking for values
    print(Fore.GREEN + "You have Chosen Variance For Two Population." +
    Style.RESET_ALL)
    s_1 = float(input("Enter the 1st Sample Variance: "))
    n_1 = int(input("Enter the 1st Sample Size: "))
    s_2 = float(input("Enter the 2nd Sample Variance: "))
    n_2 = int(input("Enter the 2nd Sample Size: "))
    alpha = float(input("Enter the Significance Level: "))

    # calculation for the f-statistic, 2 degree of freedom, and p-value
    f_stats = np.divide(s_1, s_2)
    v_1 = n_1 - 1
    v_2 = n_2 - 1
    p_value = st.f.sf(f_stats, v_1, v_2)

```

```
# displaying of values

lines()

print("The F-Ratio Value is " + str(round(f_stats,4)) + ".")
print("The 1st Degree of Freedom is " + str(v_1) + ".")
print("The 2nd Degree of Freedom is " + str(v_2) + ".")
print("The P-Value is " + str(round(p_value,6)) + ".")

lines()

# choosing of critical region

critical_region = int(input(Fore.YELLOW + "Enter the Critical Region (1) Left, (2)
Right, (3) Both: " + Style.RESET_ALL))

lines()

# left sided critical region

if critical_region == 1:

    # calculating and displaying the critical value

    critical_value = st.f.ppf(alpha, v_1, v_2)

    print("The Critical Value is " + str(round(critical_value,4)) + ".")

# choosing whether to accept or reject the null hypothesis

if f_stats < critical_value:

    print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)

else:

    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

elif critical_region == 2:

    # calculating and displaying the critical value

    critical_value = st.f.ppf(1 - alpha, v_1, v_2)

    print("The Critical Value is " + str(round(critical_value,4)) + ".")
```

```

# choosing whether to accept or reject the null hypothesis
if f_stats > critical_value:
    print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)
else:
    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

elif critical_region == 3:
    # calculating and displaying the critical value
    # alpha is divided by 2 since there are 2 critical regions
    alpha = alpha/2
    critical_value1 = st.f.ppf(alpha, v_1, v_2)
    critical_value2 = st.f.ppf(1 - alpha, v_1, v_2)
    print("The Left Critical Value is " + str(round(critical_value1,4)) + ".")
    print("The Right Critical Value is " + str(round(critical_value2,4)) + ".")

# choosing whether to accept or reject the null hypothesis
if f_stats < critical_value1 or f_stats > critical_value2:
    print(Fore.RED + "Reject The Null Hypothesis." + Style.RESET_ALL)
else:
    print(Fore.GREEN + "Accept The Null Hypothesis." + Style.RESET_ALL)

# Main Program that will print the welcome message
# It also shows choice for different choices for the user to choose from

print(Fore.BLUE + '\nWelcome to Hippopotamus - Hypothesis Testing Calculator' +
Style.RESET_ALL)

lines()

print(Fore.GREEN + "The following are the choices for Single Population:" +
Style.RESET_ALL)

```

```
lines()

print("1. Case A: Single Population Where Sigma is Known or Sample is Large")
print("2. Case B: Sigma is Unknown or Sample is Small")
print("3. Test for Proportions: Single Population")
print("4. Variance For Single Population")

lines()

print(Fore.GREEN + "The following are the choices for Two Population:" +
Style.RESET_ALL)

lines()

print("5. Case C: Two Mean Big Sample Size or Population Stnadard Deviation is
Given")

print("6. Case D: Two Mean Small Sample Size and Population Sample Standard
Deviation assumed Equal")

print("7. Case E: Two Mean Small Sample Size and Population Sample Standard
Deviation not Equal")

print("8. Case F: Paired Observation")

print("9. Test for Proportions: Two Population")

print("10. Variance For Two Populations")

lines()

choice = int(input(Fore.BLUE + "Enter your choice: " + Style.RESET_ALL))

lines()

# if-else statement to execute specified task based on choice option
# if the choice is not in the selected range then the program will display "invalid option"
if choice == 1:
    one_mean_case_A()
elif choice == 2:
    one_mean_case_B()
elif choice == 3:
    proportion_single()
```



```
elif choice == 4:
    var_one()
elif choice == 5:
    two_means_case_C()
elif choice == 6:
    two_means_case_D()
elif choice == 7:
    two_means_case_E()
elif choice == 8:
    paired_observation_case_F()
elif choice == 9:
    proportion_two()
elif choice == 10:
    var_two()
else:
    print(Fore.RED + "Invalid Option!" + Style.RESET_ALL)
```

C. Work breakdown

Student Name	Tasks Assigned	Percentage of the Work Contribution
Win Clarence Sy	Source Code User's Manual Review of Related Literature Result Discussion of the Results Analysis, Conclusion, and Future Directives	40%
Tayshaun Pierce Alabado	Introduction Flowchart Review of Related Literature Background of the Study Problem Statement	30%
Marcus Alden Taraya	Pseudocode Hierarchy Chart Objectives Significance of the Project IPO Chart	30%

D. Personal Data Sheet

Tayshaun Pierce Alabado

**Student's Information Sheet**Subject: PROLOGI Section: EQ3 Schedule: _____Tri-Academic Year: 1st Professor: Ramon Ruiz**Personal:**Name: Tayshaun Pierce G. AlabadoDegree Program: Computer EngineeringScholarship: N/AAddress: 1466 Sta. Maria St. Tondo, Manila, Metro ManilaTelephone No: N/A Cell phone No: 9664407812E-mail Address: tayshaun_alabado@dlsu.edu.phBirthday: 07/10/2004 Age: 18**Family:**Father: N/A Occupation: N/AMother: Mary Grace Alabado Occupation: Tech Support Agent**Schools:**Elementary - High School: Manila Cathedral School

Win Clarence Sy

Student's Information Sheet

Subject: PROLOGI Section: EQ3 Schedule: _____
Tri-Academic Year: 1st Professor: Ramon Stephen Ruiz

Personal:

Name: Win Clarence Sy
Degree Program: Computer Engineering
Scholarship: N/A
Address: City Place Residences, Felipe II St., Binondo, Manila
Telephone No: N/A Cell phone No: 09950460933
E-mail Address: win_sy@dlsu.edu.ph
Birthday: July 29, 2004 Age: 18

Family:

Father: _____ Occupation: _____
Mother: _____ Occupation: _____

Schools:

Elementary-High School: Uno High School

Marcus Alden Taraya

Student's Information SheetSubject: P R O L O G I Section: E Q 3 Schedule: _____Tri-Academic Year: 1 S T Professor: R a m o n R u i z**Personal:**Name: M a r c u s A l d e n D . T a r a y aDegree Program: C O M P U T E R E N G I N E E R I N GScholarship: N / AAddress: 26 Editha Vital St., BFRV Las Pinas CityTelephone No: _____ Cell phone No: 0963034712E-mail Address: marcus_taraya@dlsu.edu.phBirthday: 1 8 / 0 1 / 2 0 0 4 Age: 1 9**Family:**Father: Alvin B. Taraya Occupation: Taekwondo CoachMother: Maricar D. Taraya Occupation: Industrial Manager**Schools:**

Elizabeth Seton School

Merry Treasure School