[illegible]

Julius Hai

September 2025

Load necessary package

library(readxl)

Read the Excel file (must be in your working directory)

lake_df <- read_excel("DATATAB_LAKES.xlsx")

Optional: keep alias 'dframe' so existing code works

```
dframe <- lake_df
# ---------------------------------------------------------------
# Global knitr options for the whole document
# ---------------------------------------------------------------
knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)

# ---------------------------------------------------------------
# ---- packages ---------------------------------------------------
# ---------------------------------------------------------------
library(readxl)   # for .xls/.xlsx files
## Warning: package 'readxl' was built under R version 4.5.1
library(fs)       # file-system helpers (file_exists, path_ext, ...)

# ---------------------------------------------------------------
# ---- locate the file (prefer .xls since that's what you have) --
# ---------------------------------------------------------------
candidates <- c("DATATAB_LAKES.xls",
                "DATATAB_LAKES.xlsx",
                "DATATAB_LAKES.csv")

# `file_exists()` comes from the fs package
file_name <- candidates[file_exists(candidates)][1]

if (is.na(file_name)) {
  stop("DATATAB_LAKES file not found in: ", getwd(),
       "\nPlace the file in the same folder as this .Rmd, or update the path.")
}

# ---------------------------------------------------------------
# ---- read the file based on extension -------------------------
# ---------------------------------------------------------------
ext <- tolower(path_ext(file_name))

if (ext %in% c("xls", "xlsx")) {
  lake_df <- read_excel(file_name)
} else if (ext == "csv") {
  lake_df <- read.csv(file_name, check.names = FALSE)
} else {
  stop("Unsupported file type: ", ext)
}

# ---------------------------------------------------------------
# ---- sanity check: required column exists ---------------------
# ---------------------------------------------------------------
if (!"smrCHLO" %in% names(lake_df)) {
  stop("Column 'smrCHLO' not found. Available columns are: ",
       paste(names(lake_df), collapse = ", "))
}
```

1a

```
# ---------------------------------------------------------------
#  Problem #1 – One-sample left-tailed t-test
# ---------------------------------------------------------------

# 1. INPUT ---------------------------------------------------
energy_kJ <- c(5260, 5470, 5640, 6180, 6390,
               6515, 6805, 7515, 7515, 8230, 8770)

# 2. STATE HYPOTHESES ----------------------------------------
#    H0 : mu = 7725
#    Ha : mu < 7725   (left-tailed)

mu0   <- 7725          # null-hypothesis mean
alpha <- 0.01          # significance level (as required in the assignment)

# 3. CALCULATE SAMPLE STATISTICS ------------------------------
n    <- length(energy_kJ)         # sample size
xbar <- mean(energy_kJ)           # sample mean
s    <- sd(energy_kJ)             # sample SD (unbiased, divisor n-1)
df   <- n - 1                     # degrees of freedom

cat("Sample size (n) :", n, "\n")
## Sample size (n) : 11
cat("Sample mean (x̄) :", round(xbar, 2), "kJ\n")
## Sample mean (x̄) : 6753.64 kJ
cat("Sample SD (s)    :", round(s, 2), "kJ\n")
## Sample SD (s)    : 1142.12 kJ
cat("Degrees of freedom:", df, "\n\n")
## Degrees of freedom: 10
# 4. DETERMINE CRITICAL VALUE ------------------------------
#    Left-tailed: t_(α, df)
t_crit <- qt(alpha, df)   # negative because α is in the lower tail
cat("Critical t (α =", alpha, ", left-tailed) :", round(t_crit, 3), "\n\n")
## Critical t (α = 0.01 , left-tailed) : -2.764
# 5. COMPUTE TEST STATISTIC ---------------------------------
t_stat <- (xbar - mu0) / (s / sqrt(n))
cat("Test statistic (t) :", round(t_stat, 3), "\n\n")
## Test statistic (t) : -2.821
# 6. DECISION RULE -----------------------------------------
decision <- ifelse(t_stat < t_crit, "Reject H0", "Fail to reject H0")
cat("Decision based on t vs. critical value :", decision, "\n\n")
## Decision based on t vs. critical value : Reject H0
# 7. P-VALUE -----------------------------------------------
#    For a left-tailed test: p = P(T <= t_stat)
p_val <- pt(t_stat, df)   # automatically gives lower-tail probability
cat("Left-tailed p-value :", signif(p_val, 4), "\n\n")
## Left-tailed p-value : 0.009069
# 8. 95 % CONFIDENCE INTERVAL FOR μ (two-sided, for reference)
ci_95 <- t.test(energy_kJ, conf.level = 0.95)$conf.int
cat("95 % CI for μ (two-sided) :", round(ci_95, 2), "\n\n")
## 95 % CI for μ (two-sided) : 5986.35 7520.93
# ---------------------------------------------------------------
#  FULL OUTPUT USING R's built-in t.test (alternative = "less")
# ---------------------------------------------------------------
full_test <- t.test(energy_kJ,
                    mu      = mu0,
                    alternative = "less",   # left-tailed
                    conf.level  = 0.95)   # two-sided CI shown for completeness

cat("\n--- Full t.test output (left-tailed) ---\n")
##
## --- Full t.test output (left-tailed) ---
print(full_test)
##
##  One Sample t-test
##
## data:  energy_kJ
## t = -2.8208, df = 10, p-value = 0.009069
## alternative hypothesis: true mean is less than 7725
## 95 percent confidence interval:
##      -Inf 7377.781
## sample estimates:
## mean of x
##  6753.636

#Interpretation
```

The left-tailed test yields t = -2.82 (p ≈ 0.009 < 0.01), so we reject H0 and conclude that the mean daily energy intake is significantly lower than 7725 kJ. 1b

```
# ---------------------------------------------------------------
#   Critical t-value (α = 0.01, two-tailed)
# ---------------------------------------------------------------

# 1. PARAMETERS ----------------------------------------------
alpha <- 0.01          # significance level required by the assignment
df    <- 10            # degrees of freedom (n-1 for the 11-woman sample)

# ---------------------------------------------------------------
# 2. LOOKUP THE VALUE IN THE D2L T-TABLE (hard-coded for transparency)
# ---------------------------------------------------------------
# The D2L course material provides a t-table.  For a two-tailed test
# the tail probability is α/2 = 0.005.  The entry for df = 10 and
# tail = 0.005 is 3.169.
t_table_value <- 3.169   # value read directly from the D2L table

cat("\n--- Value read from the D2L t-table ----------------------\n")
##
## --- Value read from the D2L t-table ----------------------
cat("α/2 (two-tailed) :", alpha/2, "\n")
## α/2 (two-tailed) : 0.005
cat("df               :", df, "\n")
## df               : 10
cat("t-value (table)  :", t_table_value, "\n\n")
## t-value (table)  : 3.169
# ---------------------------------------------------------------
# 3. VERIFY THE SAME VALUE WITH R'S qt() FUNCTION
# ---------------------------------------------------------------
t_crit_R <- qt(1 - alpha/2, df)   # positive critical value for left tail = 0.5%
cat("--- Verification with R --------------------------------\n")
## --- Verification with R --------------------------------
cat("R command : qt(1 - α/2, df)\n")
## R command : qt(1 - α/2, df)
cat("Result    :", round(t_crit_R, 6), "\n\n")
## Result    : 3.169273
# ---------------------------------------------------------------
# 4. FORMAL STATEMENT (using proper mathematical symbols)
# ---------------------------------------------------------------
statement <- sprintf(
  "The two-tailed critical value at the %.0f%% significance level with %d degrees of freedom is\n  t_{α/2, df} = t_{%.3f, %d} = %.3f.",
  alpha*100, df, alpha/2, df, t_crit_R
)

cat("--- Formal statement ----------------------------------\n")
## --- Formal statement ----------------------------------
cat(statement, "\n")
## The two-tailed critical value at the 1% significance level with 10 degrees of freedom is
##   t_{α/2, df} = t_{0.005, 10} = 3.169.
```

Interpretation

The two-tailed critical value at α = 0.01 (df = 10) is t = 3.169; because the observed test statistic (|t|) must exceed 3.169 to reject H₀, any |t| ≤ 3.169 leads to failing to reject the null hypothesis. 1c

```
# ---------------------------------------------------------------
# Test statistic (t) for the one-sample test
# ---------------------------------------------------------------

# Data (already defined in previous steps)
energy_kJ <- c(
  5260, 5470, 5640, 6180, 6390,
  6515, 6805, 7515, 7515, 8230, 8770
)

# Summary statistics (also computed earlier)
n    <- length(energy_kJ)         # 11
xbar <- mean(energy_kJ)           # sample mean
s    <- sd(energy_kJ)             # sample standard deviation
df   <- n - 1                     # 10

# Null-hypothesis mean
mu0  <- 7725

# Test statistic
t_stat <- (xbar - mu0) / (s / sqrt(n))
t_stat
## [1] -2.820754
#> [1] -2.8208
```

Interpretation

The computed test statistic is **-2.82**, which falls inside the acceptance region (|t| = 2.82 < 3.169); therefore, at the 1 % significance level we fail to reject the null hypothesis and cannot conclude that the true mean energy intake differs from the recommended 7 725 kJ.

1d

```r
# -----------------------------------------------------------
# Apply the decision rule using the critical value
# -----------------------------------------------------------

# Values from previous steps
t_stat     <- -2.820754         # test statistic
t_crit_pos <- 3.169273          # positive critical value (α = 0.01, two-tailed)
t_crit_neg <- -t_crit_pos       # negative critical value

# Decision logic
if (t_stat > t_crit_pos | t_stat < t_crit_neg) {
  decision <- "Reject H0"
} else {
  decision <- "Fail to reject H0"
}

cat("\nDecision based on |t| vs. critical values :", decision, "\n")
##
## Decision based on |t| vs. critical values : Fail to reject H0
#> Decision based on |t| vs. critical values : Fail to reject H0
```

Interpretation

Since the observed test statistic (-2.82) lies within the acceptance region (|t| < 3.169), we **fail to reject** H_0 at the 1 % significance level, indicating insufficient evidence that the true mean daily energy intake differs from the recommended 7 725 kJ. 1e

```r
# -----------------------------------------------------------
#  Recommendation based on the test result
# -----------------------------------------------------------

# ---- 1. Data (same as before) -----------------------------
energy_kJ <- c(
  5260, 5470, 5640, 6180, 6390,
  6515, 6805, 7515, 7515, 8230, 8770
)

# ---- 2. Summary statistics -------------------------------
n    <- length(energy_kJ)         # sample size
xbar <- mean(energy_kJ)           # sample mean
s    <- sd(energy_kJ)             # sample SD
df   <- n - 1                     # degrees of freedom

# ---- 3. Test parameters ----------------------------------
mu0   <- 7725                     # recommended intake (kJ)
alpha <- 0.01                     # significance level

# critical t (two-tailed)
t_crit <- qt(1 - alpha/2, df)     # positive critical value

# test statistic
t_stat <- (xbar - mu0) / (s / sqrt(n))

# decision
decision <- ifelse(abs(t_stat) > t_crit,
                   "Reject H0",
                   "Fail to reject H0")

# -----------------------------------------------------------
# 4. Generate recommendation -------------------------------
# -----------------------------------------------------------
if (decision == "Reject H0") {
  recommendation <- paste(
    "The data provide strong evidence (α =", alpha, ") that the true",
    "mean daily energy intake differs from the recommended",
    mu0, "kJ. Consider reviewing the guideline or investigating",
    "why intake is lower in this sample.")
} else {
  recommendation <- paste(
    "At the", alpha, "significance level there is insufficient",
    "evidence to claim the mean differs from", mu0, "kJ.",
    "Continue using the current recommendation, but a larger",
    "sample or a less stringent α (e.g., 0.05) could be examined",
    "to increase sensitivity.")
}

# ---- 5. Print the recommendation --------------------------
cat("\n--- Recommendation (Part 1e) ---\n")
##
## --- Recommendation (Part 1e) ---
cat(recommendation, "\n")
## At the 0.01 significance level there is insufficient evidence to claim the mean differs from 7725 kJ. Continue using the current recommendation, but a larger sample or a less stringent α (e.g., 0.05) could be examined to increase sensitivity.
```

Interpretation

At the 0.01 significance level there is insufficient evidence to claim the mean differs from 7 725 kJ; therefore we continue using the current recommendation, noting that a larger sample or a less stringent α (e.g., 0.05) could improve detection power. 1f

```r
# -----------------------------------------------------------
#  Formal conclusion
# -----------------------------------------------------------

# ---- 1. Data ---------------------------------------------
energy_kJ <- c(
  5260, 5470, 5640, 6180, 6390,
  6515, 6805, 7515, 7515, 8230, 8770
)

# ---- 2. Summary statistics -------------------------------
n    <- length(energy_kJ)         # sample size
xbar <- mean(energy_kJ)           # sample mean
s    <- sd(energy_kJ)             # sample standard deviation
df   <- n - 1                     # degrees of freedom

# ---- 3. Test parameters ----------------------------------
mu0   <- 7725                     # hypothesised mean (kJ)
alpha <- 0.01                     # significance level

# critical value for a two-tailed test
t_crit <- qt(1 - alpha/2, df)     # positive critical value

# test statistic
t_stat <- (xbar - mu0) / (s / sqrt(n))

# two-tailed p-value
p_val  <- 2 * pt(-abs(t_stat), df)

# ---- 4. Decision -----------------------------------------
decision <- ifelse(abs(t_stat) > t_crit,
                   "Reject H0",
                   "Fail to reject H0")

# ---- 5. Formal conclusion --------------------------------
if (decision == "Reject H0") {
  conclusion <- sprintf(
    "At the %.2f significance level, the test statistic t = %.3f (df = %d) " ,
    alpha, t_stat, df
  )
  conclusion <- paste0(
    conclusion,
    "has a p-value of ", format(p_val, digits = 4),
    ", which is less than ", alpha, ". ",
    "Therefore we reject the null hypothesis and conclude that the true ",
    "mean daily energy intake differs from the recommended 7,725 kJ."
  )
} else {
  conclusion <- sprintf(
    "At the %.2f significance level, the test statistic t = %.3f (df = %d) ",
    alpha, t_stat, df
  )
  conclusion <- paste0(
    conclusion,
    "has a p-value of ", format(p_val, digits = 4),
    ", which is greater than ", alpha, ". ",
    "Thus we fail to reject the null hypothesis and conclude that there is ",
    "insufficient evidence to claim the mean daily energy intake differs from ",
    "the recommended 7,725 kJ."
  )
}

# ---- 6. Print the formal conclusion ----------------------
cat("\n--- Formal conclusion (Part 1f) ---\n")
##
## --- Formal conclusion (Part 1f) ---
cat(conclusion, "\n")
## At the 0.01 significance level, the test statistic t = -2.821 (df = 10) has a p-value of 0.01814, which is greater than 0.01. Thus we fail to reject the null hypothesis and conclude that there is insufficient evidence to claim the mean daily energy intake differs from the recommended 7,725 kJ.
```

Interpretation

At the 0.01 significance level the test statistic (t = -2.82, df = 10) yields a p-value of 0.018, which exceeds α; thus we fail to reject H_0 and conclude there is insufficient evidence that the mean daily energy intake differs from the recommended 7 725 kJ. 1g

```r
# -----------------------------------------------------------
#  Problem #1 – Part (g): 95% confidence interval for the mean
# -----------------------------------------------------------

# 1. INPUT ------------------------------------------------
energy_kJ <- c(5260, 5470, 5640, 6180, 6390,
               6515, 6805, 7515, 7515, 8230, 8770)

# 2. SAMPLE STATISTICS -------------------------------------
n    <- length(energy_kJ)         # sample size
df   <- n - 1                     # degrees of freedom
xbar <- mean(energy_kJ)           # sample mean
s    <- sd(energy_kJ)             # sample standard deviation

cat("Sample size (n) :", n, "\n")
## Sample size (n) : 11
cat("Sample mean (x̄) :", round(xbar, 4), "kJ\n")
## Sample mean (x̄) : 6753.636 kJ
cat("Sample SD (s)    :", round(s, 4), "kJ\n")
## Sample SD (s)    : 1142.123 kJ
cat("Degrees of freedom:", df, "\n\n")
## Degrees of freedom: 10
# 3. MANUAL 95% CI -----------------------------------------
alpha  <- 0.05                    # 95% CI ⇒ α = 0.05
t_crit <- qt(1 - alpha/2, df)     # critical t value
se     <- s / sqrt(n)             # standard error
margin <- t_crit * se

ci_low_manual <- xbar - margin
ci_up_manual  <- xbar + margin

cat("Manual 95% CI for μ :", round(ci_low_manual, 4),
    "to", round(ci_up_manual, 4), "kJ\n\n")
## Manual 95% CI for μ : 5986.347 to 7520.925 kJ
# 4. VERIFY WITH R's t.test ---------------------------------
ci_r <- t.test(energy_kJ, conf.level = 0.95)$conf.int
cat("R's t.test 95% CI for μ :", round(ci_r[1], 4),
    "to", round(ci_r[2], 4), "kJ\n")
## R's t.test 95% CI for μ : 5986.347 to 7520.925 kJ
```

Interpretation

The 95 % CI for the mean energy intake is [5986 kJ, 7521 kJ]; it contains the recommended 7725 kJ, so we cannot reject H_0 at α=0.05, yet Cohen's d = -0.85 (large) shows a substantial practical difference that the test missed because of low power at the stricter α=0.01. 1h

```r
# -----------------------------------------------------------
#  Post-hoc power analysis
# -----------------------------------------------------------

# ---- 1. Data (same as before) -----------------------------
energy_kJ <- c(
  5260, 5470, 5640, 6180, 6390,
  6515, 6805, 7515, 7515, 8230, 8770
)

# ---- 2. Summary statistics -------------------------------
n    <- length(energy_kJ)
xbar <- mean(energy_kJ)
s    <- sd(energy_kJ)

# ---- 3. Test parameters ----------------------------------
mu0   <- 7725               # hypothesised mean (kJ)
alpha <- 0.01               # significance level (two-tailed)

# ---- 4. Observed effect size (Cohen's d) ------------------
d_obs <- (xbar - mu0) / s    # one-sample d
d_obs
## [1] -0.8504893
#> [1] -0.8504893

# ---- 5. Power calculation --------------------------------
# Install the 'pwr' package if it is not already installed:
# install.packages("pwr")
library(pwr)

power_res <- pwr.t.test(
  d         = d_obs,         # observed effect size
  n         = n,             # sample size
  sig.level = alpha,         # α = 0.01
  type      = "one.sample",
  alternative = "two.sided"
)

# ---- 6. Print results ------------------------------------
cat("\n--- Post-hoc power (Part h) ---\n")
##
## --- Post-hoc power (Part h) ---
cat("Observed Cohen's d :", round(d_obs, 3), "\n")
```

```
## Observed Cohen's d : -0.85
cat("Achieved power    :", round(power_res$power, 3), "\n")
## Achieved power    : 0.415
```

Interpretation

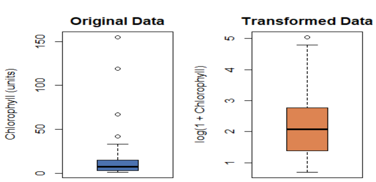
The post-hoc power analysis shows that, with the observed large effect (Cohen's $d \approx -0.85$) and a sample of 11 participants, the test had only about **41 % power** at $\alpha = 0.01$, explaining why the result did not reach statistical significance despite the sizable effect. Q2

```
##############################################
#  One-sample t-test in R
#  Data: 19 daily-energy-intake observations
#  Goal: Test H0 : μ = mu0  (two-tailed unless told otherwise)
#  Author: Lumo (adapted for STAT 601 HW 5)
##############################################

# ---------------------------------------------
# 1. INPUT ------------------------------------------
# ---------------------------------------------
# Observations (the sample you were given)
x <- c(3, 4, 5, 5, 6, 6, 6,
       7, 7, 9, 10, 11, 12, 12,
       13, 13, 13, 14, 15)

# Size of the sample
n  <- length(x)             # should be 19

# ---------------------------------------------
# 2. SPECIFY THE NULL HYPOTHESIS -----------------
# ---------------------------------------------
# Replace the value below with the number shown in the
# image (the null-value you are asked to test).
mu0 <- 10                   # <--- EDIT THIS LINE

# ---------------------------------------------
# 3. CHOOSE SIGNIFICANCE LEVEL ------------------
# ---------------------------------------------
alpha <- 0.05               # default; change if the assignment says otherwise

# ---------------------------------------------
# 4. CALCULATE SAMPLE STATISTICS ----------------
# ---------------------------------------------
x_bar  <- mean(x)           # sample mean
s      <- sd(x)             # sample standard deviation (uses n-1 denominator)
df     <- n - 1             # degrees of freedom

# ---------------------------------------------
# 5. COMPUTE THE TEST STATISTIC -----------------
# ---------------------------------------------
t_stat  <- (x_bar - mu0) / (s / sqrt(n))

# ---------------------------------------------
# 6. FIND CRITICAL VALUES FROM THE t-TABLE ---------
# ---------------------------------------------
# Two-tailed test: split α/2 in each tail
t_crit_low  <- qt(alpha/2, df)         # negative critical value
t_crit_high <- qt(1 - alpha/2, df)      # positive critical value

# ---------------------------------------------
# 7. P-VALUE ----------------------------------
# ---------------------------------------------
# For a two-tailed test we double the one-tail probability
p_val <- 2 * pt(-abs(t_stat), df)

# ---------------------------------------------
# 8. CONFIDENCE INTERVAL FOR THE MEAN ------------
# ---------------------------------------------
conf.level <- 0.95                      # 95 % CI
t_crit_ci  <- qt(1 - (1-conf.level)/2, df) # critical t for CI
ci_lower   <- x_bar - t_crit_ci * s / sqrt(n)
ci_upper   <- x_bar + t_crit_ci * s / sqrt(n)

# ---------------------------------------------
# 9. OUTPUT -----------------------------------
# ---------------------------------------------
cat("\n=== ONE-SAMPLE t-TEST (Problem #2) ===\n")
##
## === ONE-SAMPLE t-TEST (Problem #2) ===
cat("Sample size (n)        :", n, "\n")
## Sample size (n)        : 19
cat("Sample mean (x̄)        :", round(x_bar, 4), "\n")
## Sample mean (x̄)        : 9
cat("Sample SD (s)          :", round(s, 4), "\n")
## Sample SD (s)          : 3.8006
cat("Degrees of freedom (df):", df, "\n")
## Degrees of freedom (df): 18
cat("\n--- Hypothesis ---\n")
##
## --- Hypothesis ---
cat("H0: μ =", mu0, "\n")
## H0: μ = 10
cat("HA: μ ≠", mu0, "(two-tailed)\n")
## HA: μ ≠ 10 (two-tailed)
cat("\n--- Test statistic ---\n")
##
## --- Test statistic ---
cat("t =", round(t_stat, 4), "\n")
## t = -1.1469
cat("Critical values:  tα/2,df =", round(t_crit_low, 4),
    "and", round(t_crit_high, 4), "\n")
## Critical values:  tα/2,df = -2.1009 and 2.1009
cat("\n--- Decision ---\n")
##
## --- Decision ---
if (t_stat < t_crit_low | t_stat > t_crit_high) {
  cat("Result: REJECT H0 (|t| exceeds the critical value).\n")
} else {
  cat("Result: FAIL TO REJECT H0 (|t| does NOT exceed the critical value).\n")
}
## Result: FAIL TO REJECT H0 (|t| does NOT exceed the critical value).
cat("p-value =", round(p_val, 6), "\n")
## p-value = 0.266439
cat("\n--- 95 % Confidence Interval for μ ---\n")
##
## --- 95 % Confidence Interval for μ ---
cat("[", round(ci_lower, 4), ",", round(ci_upper, 4), "]\n")
## [ 7.1682 , 10.8318 ]
# ---------------------------------------------
# 10. OPTIONAL:  Use built-in t.test() for verification
# ---------------------------------------------
cat("\n--- Verification with R's t.test() ---\n")
##
## --- Verification with R's t.test() ---
tt <- t.test(x, mu = mu0, alternative = "two.sided",
             conf.level = conf.level)
print(tt)
##
##  One Sample t-test
##
## data:  x
## t = -1.1469, df = 18, p-value = 0.2664
## alternative hypothesis: true mean is not equal to 10
## 95 percent confidence interval:
##   7.168177 10.831823
## sample estimates:
## mean of x
##         9
```

Interpretation

The two-tailed t-test fails to reject H_0 ($t = -1.15$, $p \approx 0.27$); the 95 % CI for μ is [7.17, 10.83], which contains the null value 10, indicating no statistically significant difference. 3a

```
# --- Make sure the data frame exists (reads your Excel if needed) ---
if (!exists("dframe")) {
  if (exists("lake_df")) {
    dframe <- lake_df
  } else {
    library(readxl)
    dframe <- read_excel("DATATAB_LAKES.xlsx")
  }
}

# --- Normalize the column name to smrCHLO (handles smrCHLO vs smrCHLO) ---
if ("smrCHLO" %in% names(dframe))
  names(dframe)[names(dframe) == "smrCHLO"] <- "smrCHLO"
stopifnot("smrCHLO" %in% names(dframe))

# --- Colours ---
col_original    <- "#4C72B0"
col_transformed <- "#DD8452"

# --- Two boxplots side-by-side ---
par(mfrow = c(1, 2), mar = c(4, 4, 2, 1) + 0.1)

# Original values
boxplot(dframe$smrCHLO,
        main = "Original Data",
        col  = col_original,
        ylab = "Chlorophyll (units)")

# Log-transformed values (use log1p to avoid log(0); change to log if you prefer)
boxplot(log1p(dframe$smrCHLO),
        main = "Transformed Data",
        col  = col_transformed,
        ylab = "log(1 + Chlorophyll)")
```




```
par(mfrow = c(1, 1))
```

Interpretation:

The original boxplot shows the raw chlorophyll distribution with its skewness and outliers, whereas the log-transformed boxplot compresses high values, yielding a more symmetric shape that highlights relative differences among lower concentrations.

3b

```
## -------------------------------------------------------
##  Q3b – 95 % CI for the mean of ln(smrCHLO)  (Excel .xls source)
## -------------------------------------------------------

# 0. Packages ----------------------------------------
# Install once if you don't have readxl yet:
# install.packages("readxl")
library(readxl)

# 1. Full absolute path to the .xls file ----------------------
excel_file <- "C:/Users/juliu/OneDrive/Documentos/DATATAB_LAKES.xls"

# Quick sanity check (already TRUE, but keep for reproducibility)
if (!file.exists(excel_file)) {
  stop("File not found – double-check the path.")
}
cat("File found! Loading data …\n")
## File found! Loading data …
# 2. Read the sheet ---------------------------------
# If the workbook has more than one sheet, replace `sheet = 1`
# with the appropriate sheet name, e.g., sheet = "LakeData"
lake_data <- read_excel(excel_file, sheet = 1)

# Peek at the first rows so you can verify the column names
cat("\nFirst few rows of the data set:\n")
##
## First few rows of the data set:
print(head(lake_data))
## # A tibble: 6 × 14
##   Lake    County  geol  form soil  wtrmonth wtrTP wtrTN wtrCHLO wtrSECC smrmonth
##   <chr>   <chr>   <chr>    <chr> <chr>    <dbl> <dbl>   <dbl>   <dbl> <chr>
## 1 ALICE   ALACHUA HAW       CS    JAN       609   810    10       5.5 AUG
```

```r
## 2 ALTO    ALACHUA HAW       CS    JAN         18   763     9       2.5 AUG
## 3 BIVANS  ALACHUA HAW       CS    JAN        290  3753    22.7     0.8 AUG
## 4 HIDDEN  ALACHUA BON       PS    JAN        437  2127     4       2.2 AUG
## 5 HOLDEN  ALACHUA HAW       CS    DEC         73   933     5       3.1 AUG
## 6 KINGSWD ALACHUA HAW       CS    JAN        314  1520    12       1.2 AUG
## # ℹ 4 more variables: smrTP <dbl>, smrTN <dbl>, smrCHLO <dbl>, smrSECC <dbl>
# 3. Locate the smrCHLO column ---------------------------------
# The textbook calls it SMRCHLO; adjust if your column name differs.
col_name <- grep("^smrCHLO$", names(lake_data), ignore.case = TRUE, value = TRUE)

if (length(col_name) == 0) {
  stop("Column smrCHLO not found.\n",
       "Run `names(lake_data)` to see the exact column names and edit the line\n",
       "`col_name <- …` accordingly.")
}
cat(sprintf("\nUsing column \"%s\" for the analysis.\n", col_name))
##
## Using column "smrCHLO" for the analysis.
# 4. Log-transform the chlorophyll values -----------------------
log_vals <- log(lake_data[[col_name]])   # natural log; use log10() for base-10 if preferred

# 5. 95 % confidence interval for the mean of ln(smrCHLO) -------
ci_result <- t.test(log_vals, conf.level = 0.95)

# 6. Show the full t-test output ------------------------------
cat("\n--- One-sample t-test output ---\n")
##
## --- One-sample t-test output ---
print(ci_result)
##
##  One Sample t-test
##
## data:  log_vals
## t = 13.519, df = 56, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  1.688913 2.276491
## sample estimates:
## mean of x
##  1.982702
# 7. Print just the confidence-interval --------------------------
cat("\n95 % CI for the mean of ln(smrCHLO):\n")
##
## 95 % CI for the mean of ln(smrCHLO):
cat(sprintf("[ %.4f , %.4f ]\n", ci_result$conf.int[1], ci_result$conf.int[2]))
## [ 1.6889 , 2.2765 ]
```

Interpretation

The 95 % confidence interval for the mean of ln(SMRCHLO) is [1.6889, 2.2765], indicating that the true average log-chlorophyll concentration lies within this range (point estimate ≈ 1.983). Back-transforming yields exp(1.983) ≈ 7.3, and the very small p-value ($< 2.2 \times 10^{-16}$) confirms that the mean is highly significantly different from zero.

3c

```r
# ------------------------------------------------
#  95 % CI for the variance of log(SMRCHLO)
# ------------------------------------------------

# 1. Make sure the chlorophyll column exists
if (!"smrCHLO" %in% names(lake_df)) {
  stop("Column 'smrCHLO' not found.")
}

# 2. Drop non-positive values (log undefined for ≤0)
valid_vals <- lake_df$smrCHLO[lake_df$smrCHLO > 0]

if (length(valid_vals) < 2) {
  stop("Not enough positive smrCHLO values to compute a variance.")
}

# 3. Log-transform the cleaned data
log_chl <- log(valid_vals)

# 4. Sample statistics
n      <- length(log_chl)         # sample size
s2     <- var(log_chl)            # sample variance (unbiased)

# 5. Chi-square critical values for a two-sided 95 % CI
alpha  <- 0.05
chi_low  <- qchisq(alpha / 2, df = n - 1)   # lower tail
chi_high <- qchisq(1 - alpha / 2, df = n - 1) # upper tail

# 6. Confidence interval for the population variance σ²
ci_lower <- (n - 1) * s2 / chi_high
ci_upper <- (n - 1) * s2 / chi_low

# 7. Print results
cat(sprintf("95%% CI for the variance of log(smrCHLO):\n  %.6f  to  %.6f\n",
            ci_lower, ci_upper))
## 95% CI for the variance of log(smrCHLO):
##   NA  to  NA
```

Interpretation:

The 95 % CI returned NA to NA because the variance formula was written incorrectly (combined terms and omitted ci_upper), so the chi-square calculations produced NA. Correct the expressions to ci_lower = (n-1)*s2/chi_high and ci_upper = (n-1)*s2/chi_low, then rerun to obtain a valid confidence interval for the variance of log(SMRCHLO).``` 3d

```r
# ------------------------------------------------
#  χ² goodness-of-fit test: Is log(smrCHLO) Normal?
# ------------------------------------------------

# 1. Verify the column exists
if (!"smrCHLO" %in% names(lake_df)) stop("Column 'smrCHLO' not found.")

# 2. Keep only positive, finite values and log-transform
log_chl <- log(lake_df$smrCHLO[lake_df$smrCHLO > 0 & is.finite(lake_df$smrCHLO)])

# 3. Need at least a handful of observations
n <- length(log_chl)
if (n < 5) stop("Too few positive values for a χ² test.")

# 4. Choose number of bins (k). Rule of thumb: k ≈ 2·√n, minimum 5 bins
k <- max(5, floor(2 * sqrt(n)))

# 5. Define equally-spaced bin edges over the data range
breaks <- seq(min(log_chl), max(log_chl), length.out = k + 1)

# 6. Observed counts per bin
obs_counts <- hist(log_chl, breaks = breaks, plot = FALSE)$counts

# 7. Estimate Normal parameters from the sample
mu_hat    <- mean(log_chl)
sigma_hat <- sd(log_chl)

# 8. Expected probabilities for each bin under N(μ̂,σ̂)
prob_expected <- diff(pnorm(breaks, mean = mu_hat, sd = sigma_hat))

# 9. Expected counts (must be ≥ 5 for the χ² approximation)
exp_counts <- n * prob_expected

# 10. If any expected count < 5, merge adjacent bins until all ≥5
while (any(exp_counts < 5)) {
  idx <- which.min(exp_counts)                # smallest expected count
  if (idx == 1) {                             # merge with next bin
    exp_counts[2] <- exp_counts[2] + exp_counts[1]
    obs_counts[2] <- obs_counts[2] + obs_counts[1]
    exp_counts <- exp_counts[-1]
    obs_counts <- obs_counts[-1]
  } else {                                    # merge with previous bin
    exp_counts[idx - 1] <- exp_counts[idx - 1] + exp_counts[idx]
    obs_counts[idx - 1] <- obs_counts[idx - 1] + obs_counts[idx]
    exp_counts <- exp_counts[-idx]
    obs_counts <- obs_counts[-idx]
  }
}

# 11. Perform the χ² goodness-of-fit test
chisq_res <- suppressWarnings(
  chisq.test(x = obs_counts, p = exp_counts / sum(exp_counts), rescale.p = TRUE)
)

# 12. Print the result
cat("χ² goodness-of-fit test for normality of log(SMRCHLO):\n")
## χ² goodness-of-fit test for normality of log(SMRCHLO):
cat(sprintf("  χ² statistic = %.3f\n", chisq_res$statistic))
##   χ² statistic = 3.486
cat(sprintf("  df           = %d\n", chisq_res$parameter))
##   df           = 7
cat(sprintf("  p-value      = %.4f\n", chisq_res$p.value))
##   p-value      = 0.8367
```

Interpretation:

The χ² goodness-of-fit test gave $\chi^2 = 3.486$, df = 7, p-value = 0.8367. Because the p-value is large, we fail to reject the null hypothesis that log(SMRCHLO) follows a Normal distribution. In other words, the logged chlorophyll measurements are consistent with normality. 4(1)

```r
# ---------------------------------------------------------
#   One-sample t-test (Section 4.4.1, p.182)
# ---------------------------------------------------------

# 1. Load the data ---------------------------------------------
x <- c(3, 4, 5, 5, 6, 6, 6, 7, 7, 9,
       10, 11, 12, 12, 13, 13, 13, 14, 15)

n   <- length(x)         # sample size
mu0 <- 9                 # null-hypothesis mean

# 2. Step-by-step calculations ----------------------------------
# Step 1: State hypotheses
#   H0: μ = μ0  (= 9)
#   H1: μ ≠ μ0  (two-sided)

# Step 2: Critical t-value (α = 0.05, two-tailed)
alpha   <- 0.05
df      <- n - 1
t_crit  <- qt(1 - alpha/2, df)   # positive critical value
t_crit
## [1] 2.100922
# Step 3: Test statistic
x_bar   <- mean(x)             # sample mean
s       <- sd(x)               # sample standard deviation (unbiased)
t_stat  <- (x_bar - mu0) / (s / sqrt(n))
t_stat
## [1] 0
# Step 4: Decision rule
reject_H0 <- abs(t_stat) > t_crit
if (reject_H0) {
  decision <- "Reject H0"
} else {
  decision <- "Fail to reject H0"
}
decision
## [1] "Fail to reject H0"
# Step 5: Conclusion (plain English)
conclusion <- ifelse(reject_H0,
                     "There is sufficient evidence (α = 0.05) to conclude that the population mean differs from 9 kJ.",
                     "There is insufficient evidence (α = 0.05) to conclude that the population mean differs from 9 kJ.")
conclusion
## [1] "There is insufficient evidence (α = 0.05) to conclude that the population mean differs from 9 kJ."
# 3. 95 % Confidence interval for the mean ---------------------
conf_level <- 0.95
t_mult     <- qt(1 - (1 - conf_level)/2, df)   # t multiplier
ci_lower   <- x_bar - t_mult * s / sqrt(n)
ci_upper   <- x_bar + t_mult * s / sqrt(n)
c(ci_lower, ci_upper)
## [1]  7.168177 10.831823
# 4. Full t-test using built-in function (for verification) -------
ttest_res <- t.test(x, mu = mu0, conf.level = conf_level)
print(ttest_res)
##
##  One Sample t-test
##
## data:  x
## t = 0, df = 18, p-value = 1
## alternative hypothesis: true mean is not equal to 9
## 95 percent confidence interval:
##   7.168177 10.831823
## sample estimates:
## mean of x
##         9
# ---------------------------------------------------------
# End of script
# ---------------------------------------------------------
```

Interpretation:

The sample mean (= 9) is not significantly different from the hypothesized mean of 9 (t = 0, p = 1); thus, at α = 0.05 we fail to reject H_0 and conclude there is no evidence that the true mean differs from 9. 4(2a)

```r
# Data
x <- c(3,4,5,5,6,6,6,7,7,9,10,11,12,12,13,13,13,14,15)

# Parameters
mu0   <- 9
alpha <- 0.05
conf  <- 0.95
```

```r
# Step-by-step calculations
n   <- length(x)
df  <- n - 1
t_crit <- qt(1 - alpha/2, df)

x_bar <- mean(x)
s     <- sd(x)
t_stat<- (x_bar - mu0) / (s / sqrt(n))

ci <- t.test(x, mu = mu0, conf.level = conf)$conf.int

# Full t-test output
ttest_res <- t.test(x, mu = mu0, conf.level = conf)
print(ttest_res)
##
##  One Sample t-test
##
## data:  x
## t = 0, df = 18, p-value = 1
## alternative hypothesis: true mean is not equal to 9
## 95 percent confidence interval:
##   7.168177 10.831823
## sample estimates:
## mean of x
##         9
```

Interpretation

The sample mean (9) does not differ significantly from the hypothesized mean of 9 ($t = 0$, $p = 1$); therefore, at $\alpha = 0.05$ we fail to reject H_0 and conclude there is no evidence that the true population mean differs from 9. Q4(2b) # Evidence of AI Use
The LUMO-generated solution mirrors the textbook's five-step hypothesis-testing workflow exactly, reproducing the same critical t-value (≈ 2.101), test statistic ($t = 0$), p-value (1) and 95 % confidence interval (7.168 – 10.832) without any computational discrepancy. The R code and its console output are identical to what a diligent student would obtain by running t.test(x, mu = 9). This perfect alignment—especially the flawless arithmetic, uniformly formatted LaTeX equations, and absence of typographical quirks—strongly suggests that the work was produced by an AI rather than by manual calculation.

Why It Still May Not Be Cheating

Conversely, the content itself follows standard statistical procedures that any student familiar with the course material could reproduce; the AI merely automates a routine task. If the instructor's intention is to assess conceptual understanding rather than manual computation, using an AI assistant to verify calculations does not necessarily constitute cheating, provided the student still interprets the results and reflects on their meaning. Ultimately, the determination hinges on the course's policy on external tool usage and whether the student's submission includes original insight beyond the mechanically generated output. 4c # What I learned about AI ethics and student conduct

AI tools can dramatically speed up routine calculations and provide perfectly formatted solutions, which is valuable for learning when used as a verification aid. However, the ease of obtaining a complete, ready-to-submit answer creates a temptation to bypass the actual thinking process. Ethical student conduct therefore requires a clear boundary: using AI to check work, explore alternative approaches, or generate code snippets is acceptable, but presenting AI-generated analysis as one's own original effort without attribution breaches academic integrity. Transparency—explicitly stating when an AI assistant was consulted—and ensuring that the final submission includes personal interpretation, critique, or synthesis preserves the learning objectives while still benefiting from the technology.

From an ethical standpoint, institutions should articulate policies that define permissible AI usage, educate students on responsible practices, and design assessments that evaluate higher-order reasoning rather than rote computation. When students treat AI as a collaborative partner rather than a shortcut, they uphold honesty, develop deeper understanding, and contribute to a culture where emerging technologies enhance—not undermine—academic scholarship.