DAY -1

Ex.1

```
# Given data
age_intervals <- c("1-5", "5-15", "15-20", "20-50", "50-80", "80-110")
frequencies <- c(200, 450, 300, 1500, 700, 44)

# Calculate cumulative frequencies
cumulative_freq <- cumsum(frequencies)

# Find the class interval containing the median
N <- sum(frequencies)
median_class_index <- which(cumulative_freq >= N / 2)[1]
median_class <- age_intervals[median_class_index]

# Extract lower and upper bounds of the median class
median_class_bounds <- as.numeric(strsplit(median_class, "-")[[1]])
L <- median_class_bounds[1]
U <- median_class_bounds[2]

# Calculate the approximate median
F <- cumulative_freq[median_class_index - 1]
f <- frequencies[median_class_index]
w <- U - L
median <- L + ((N / 2 - F) / f) * w
median
```

Ex. 2.

```
# Given data
age <- c(13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35,
35, 35, 36, 40, 45, 46, 52, 70)

# (a) Mean and Median
mean_age <- mean(age)
median_age <- median(age)

# (b) Mode
mode_age <- names(table(age))[which.max(table(age))]

# (c) Midrange
midrange_age <- (max(age) + min(age)) / 2

# (d) Quartiles
Q1 <- quantile(age, 0.25)
Q3 <- quantile(age, 0.75)

# Print the results
print(paste("Mean:", mean_age))
print(paste("Median:", median_age))
print(paste("Mode:", mode_age))
print(paste("Midrange:", midrange_age))
print(paste("Q1:", Q1))
```

	print(paste("Q3:", Q3))

Ex. 3
# Given data 	
data <- c(200, 300, 400, 600, 1000)
 	
# Min-max normalization 	
min_max <- function(x) {
  (x - min(x)) / (max(x) - min(x))
}
 	
min_max_normalized <- min_max(data)
min_max_normalized
 	
# Z-score normalization 	
z_score <- function(x) {
  (x - mean(x)) / sd(x)
}
 	
z_score_normalized <- z_score(data)
z_score_normalized
	Ex. 4.
	# Given data
	data <- c(11, 13, 13, 15, 15, 16, 19, 20, 20, 20, 21, 21, 22, 23, 24, 30, 40, 45, 45, 45, 71, 72, 73, 75)
	
	# Number of bins
	num_bins <- 5
	
	# Bin width
	bin_width <- ceiling((max(data) - min(data)) / num_bins)
	
	# Bin boundaries
	bin_boundaries <- seq(min(data), max(data), by = bin_width)
	
	# Bin indices
	bin_indices <- cut(data, breaks = bin_boundaries, labels = FALSE)
	
	# Smoothing by bin mean
	bin_means <- tapply(data, bin_indices, mean)
	smoothed_mean <- sapply(bin_indices, function(i) bin_means[i])
	
	# Smoothing by bin median
	bin_medians <- tapply(data, bin_indices, median)
	smoothed_median <- sapply(bin_indices, function(i) bin_medians[i])

```
# Smoothing by bin boundaries
smoothed_boundaries <- sapply(bin_indices, function(i) bin_boundaries[i])

# Print the results
print("Smoothing by bin mean:")
print(smoothed_mean)

print("Smoothing by bin median:")
print(smoothed_median)

print("Smoothing by bin boundaries:")
print(smoothed_boundaries)
```

Ex. 5.

```
# Age and body fat data
age <- c(23, 23, 27, 27, 39, 41, 47, 49, 50, 52, 54, 54, 56, 57, 58, 58, 60, 61)
body_fat <- c(9.5, 26.5, 7.8, 17.8, 31.4, 25.9, 27.4, 27.2, 31.2, 34.6, 42.5, 28.8,
33.4, 30.2, 34.1, 32.9, 41.2, 35.7)

# (a) Calculate mean, median, and standard deviation
mean_age <- mean(age)
median_age <- median(age)
sd_age <- sd(age)

mean_fat <- mean(body_fat)
median_fat <- median(body_fat)
```

Ex. 6.

```
# Age and body fat data
age <- c(23, 23, 27, 27, 39, 41, 47, 49, 50, 52, 54, 54, 56, 57, 58, 58, 60, 61)
body_fat <- c(9.5, 26.5, 7.8, 17.8, 31.4, 25.9, 27.4, 27.2, 31.2, 34.6, 42.5, 28.8,
33.4, 30.2, 34.1, 32.9, 41.2, 35.7)

# (a) Calculate mean, median, and standard deviation
mean_age <- mean(age)
median_age <- median(age)
sd_age <- sd(age)

mean_fat <- mean(body_fat)
median_fat <- median(body_fat)
```

```
sd_fat <- sd(body_fat)


# (b) Boxplots
par(mfrow = c(1, 2))
boxplot(age, main = "Age", ylab = "Age", col = "lightblue")
boxplot(body_fat, main = "Body Fat %", ylab = "Body Fat %", col = "lightgreen")


# (c) Scatter plot and q-q plot
par(mfrow = c(1, 2))
plot(age, body_fat, xlab = "Age", ylab = "Body Fat %", main = "Scatter Plot")


qqplot_age <- qqplot(age, main = "Q-Q Plot: Age")
qqline(age)


qqplot_fat <- qqplot(body_fat, main = "Q-Q Plot: Body Fat %")
qqline(body_fat)


# Print the results
cat("Age: Mean =", mean_age, ", Median =", median_age, ", SD =", sd_age, "\n")
cat("Body Fat %: Mean =", mean_fat, ", Median =", median_fat, ", SD =", sd_fat,
"\n")
```

```
sd_fat <- sd(body_fat)


# (b) Boxplots
par(mfrow = c(1, 2))
boxplot(age, main = "Age", ylab = "Age", col = "lightblue")
boxplot(body_fat, main = "Body Fat %", ylab = "Body Fat %", col = "lightgreen")


# (c) Scatter plot and q-q plot
par(mfrow = c(1, 2))
plot(age, body_fat, xlab = "Age", ylab = "Body Fat %", main = "Scatter Plot")


qqplot_age <- qqplot(age, main = "Q-Q Plot: Age")
qqline(age)


qqplot_fat <- qqplot(body_fat, main = "Q-Q Plot: Body Fat %")
qqline(body_fat)


# Print the results
cat("Age: Mean =", mean_age, ", Median =", median_age, ", SD =", sd_age, "\n")
cat("Body Fat %: Mean =", mean_fat, ", Median =", median_fat, ", SD =", sd_fat,
"\n")
```

Ex. 7.

```
# Given value for age
age_value <- 35
```

Ex. 8.

```
# Given data
pencils <- c(9, 25, 23, 12, 11, 6, 7, 8, 9, 10)
```

```r
# (i) Min-max normalization
min_age <- min(age)
max_age <- max(age)
min_max_age <- (age_value - min_age) / (max_age - min_age)

# (ii) Z-score normalization
mean_age <- mean(age)
sd_age <- 12.94
z_score_age <- (age_value - mean_age) / sd_age

# (iii) Normalization by decimal scaling
scale_factor <- 10 ^ ceiling(log10(max(age)))
decimal_scaled_age <- age_value / scale_factor

# Print the results
cat("Min-max normalization for age:", min_max_age, "\n")
cat("Z-score normalization for age:", z_score_age, "\n")
cat("Normalization by decimal scaling for age:", decimal_scaled_age, "\n")
```

```r

# Mean
mean_pencils <- mean(pencils)

# Median
median_pencils <- median(pencils)

# Mode (using the 'Mode' function)
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}
mode_pencils <- Mode(pencils)

# Print the results
cat("Mean:", mean_pencils, "\n")
cat("Median:", median_pencils, "\n")
cat("Mode:", mode_pencils, "\n")
```

Ex. 9.

```r
# Given data
x <- c(4, 1, 5, 7, 10, 2, 50, 25, 90, 36)
y <- c(12, 5, 13, 19, 31, 7, 153, 72, 275, 110)

# Scatter plot
```

Ex. 10.

```r
# Given data
marks <- c(55, 60, 71, 63, 55, 65, 50, 55, 58, 59, 61, 63, 65, 67, 71, 72, 75)

# (a) Equal-frequency (equi-depth) partitioning
equal_freq_bins <- cut(marks, breaks = 3, labels = FALSE)
```

```
plot(x, y, xlab = "Number of Mobile Phones Sold", ylab = "Money", main =
"Scatter Plot of Mobile Phones Sold vs Money")
```

```
# (b) Equal-width partitioning
min_mark <- min(marks)
max_mark <- max(marks)
width <- (max_mark - min_mark) / 3
equal_width_bins <- cut(marks, breaks = seq(min_mark, max_mark + width, by =
width), labels = FALSE)

# Plotting histogram
par(mfrow = c(1, 2))
hist(marks, breaks = 3, main = "Equal-frequency Partitioning", xlab = "Marks",
ylab = "Frequency", col = "lightblue", border = "black")
abline(v = breaks, col = "red", lty = 2)

hist(marks, breaks = seq(min_mark, max_mark + width, by = width), main =
"Equal-width Partitioning", xlab = "Marks", ylab = "Frequency", col =
"lightgreen", border = "black")
abline(v = breaks, col = "red", lty = 2)

# Reset plot layout
par(mfrow = c(1, 1))
```

Ex.11.

```
# Given data
speed <- c(78.3, 81.8, 82, 74.2, 83.4, 84.5, 82.9, 77.5, 80.9, 70.6)
```

Ex.12.

```
# Given data
age <- c(13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35,
35, 35, 36, 40, 45, 46, 52, 70)
```

```
# Interquartile range (IQR)
q1 <- quantile(speed, 0.25)
q3 <- quantile(speed, 0.75)
iqr <- q3 - q1

# Standard deviation
sd_speed <- sd(speed)

# Print the results
cat("Interquartile range (IQR):", iqr, "\n")
cat("Standard deviation:", sd_speed, "\n")
```

```
# First quartile (Q1)
q1 <- quantile(age, 0.25)

# Third quartile (Q3)
q3 <- quantile(age, 0.75)

# Print the results
cat("First quartile (Q1):", q1, "\n")
cat("Third quartile (Q3):", q3, "\n")
```

DAY -2

Ex.1

```
# Create a matrix for the data
data <- matrix(c(18, 2, 20, 22, 28, 10, 20, 40, 40), nrow = 3, byrow = TRUE)
rownames(data) <- c("5-6 years", "7-8 years", "9-10 years")
colnames(data) <- c("A", "B", "C")

# Calculate covariance between B and C
cov_bc <- cov(data[, "B"], data[, "C"])
print(paste("Covariance between B and C:", cov_bc))

# Calculate sample covariance matrix for the preferences
cov_matrix <- cov(data)
```

Ex.2

```
# Prices data
prices <- c(1, 1, 5, 5, 5, 5, 5, 8, 8, 10, 10, 10, 10, 12, 14, 14, 14, 15, 15, 15, 15, 15,
15, 18, 18, 18, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25,
25, 25, 25, 28, 28, 30)

# Equal-frequency partitioning with bin equal to 3
num_bins <- 3
bin_labels <- cut(prices, breaks = num_bins, labels = FALSE)

# Data smoothing using bin boundaries
bin_boundaries <- cut(prices, breaks = num_bins)
```

```
print("Sample Covariance Matrix:")
print(cov_matrix)

# Calculate correlation between B and C
cor_bc <- cor(data[, "B"], data[, "C"])
print(paste("Correlation between B and C:", cor_bc))

# Calculate sample correlation matrix for the preferences
cor_matrix <- cor(data)
print("Sample Correlation Matrix:")
print(cor_matrix)
```

```
bin_boundaries_clean <- as.numeric(as.character(bin_boundaries))

# Plot histogram for frequency division
hist(prices, breaks = num_bins, main = "Histogram of Prices", xlab = "Price", col
= "lightblue")
```

Ex.3

```
# Data for Class A and Class B
class_a <- c(76, 35, 47, 64, 95, 66, 89, 36, 84)
class_b <- c(51, 56, 84, 60, 59, 70, 63, 66, 50)

# Calculate mean, median, and range for each class
mean_a <- mean(class_a)
mean_b <- mean(class_b)

median_a <- median(class_a)
median_b <- median(class_b)

range_a <- max(class_a) - min(class_a)
range_b <- max(class_b) - min(class_b)
```

Ex.4

```
# Define the given data
data <- c(200, 300, 400, 600, 1000)

# Min-max normalization by setting min = 0 and max = 1
min_max_normalized <- (data - min(data)) / (max(data) - min(data))

# Z-score normalization
z_score_normalized <- (data - mean(data)) / sd(data)

# Print the normalized values
cat("Min-max normalized data:", min_max_normalized, "\n")
cat("Z-score normalized data:", z_score_normalized, "\n")
```

# Determine which class scored higher mean, median, and range 	
mean_comparison <- ifelse(mean_a > mean_b, "Class A", "Class B")
median_comparison <- ifelse(median_a > median_b, "Class A", "Class B")
range_comparison <- ifelse(range_a > range_b, "Class A", "Class B")
 	
# Print the results 	
cat("Mean:", mean_comparison, "had a higher mean.\n")
cat("Median:", median_comparison, "had a higher median.\n")
cat("Range:", range_comparison, "had a higher range.\n")
 	
# Plot boxplot 	
boxplot(class_a, class_b, names = c("Class A", "Class B"), col = c("skyblue", "lightgreen"),
main = "Comparison of Exam Scores for Class A and Class B",
ylab = "Scores", xlab = "Class")	

Ex.5
# Load the AirPassengers dataset 	
data("AirPassengers")
 	
# Create a histogram with specified bins 	
hist(AirPassengers, breaks = seq(100, 700, by = 150), xlim = c(100, 700),
main = "Histogram of AirPassengers dataset",
xlab = "Passenger Count", ylab = "Frequency")
	Ex.6
	# Load the mtcars dataset
	data(mtcars)
	
	# Plot the first line
	plot(mtcars$mpg, type = "l", col = "blue", xlab = "Index", ylab = "mpg and qsec")
	# Add the second line to the plot
	lines(mtcars$qsec, type = "l", col = "red")
	# Add a legend

	legend("topright", legend = c("mpg", "qsec"), col = c("blue", "red"), lty = 1)

Ex.7
# Read the CSV file 	
data <- read.csv("D:/phd/Dataset/water_potability.csv")
 	
# Check the data types of columns 	
str(data)
 	
# Convert Potability column to factor 	
data$Potability <- as.factor(data$Potability)
 	
# Convert Potability factor levels to numeric 	
data$Potability <- as.numeric(data$Potability)
 	
# Remove rows with missing or infinite values in Hardness 	
data <- data[is.finite(data$Hardness),]
 	
# Replace NA values with mean of Hardness 	
data$Hardness[is.na(data$Hardness)] <- mean(data$Hardness, na.rm = TRUE)
 	
# Replace NA values with mean of Potability 	
data$Potability[is.na(data$Potability)] <- mean(data$Potability, na.rm = TRUE)
 	
# Plotting Potability vs. Hardness 	
	Ex.8
	# Load the mtcars dataset
	data("mtcars")
	
	# Create a boxplot for mpg vs. cyl
	boxplot(mpg ~ cyl, data = mtcars,
	xlab = "Number of Cylinders",
	ylab = "Miles per Gallon",
	main = "Boxplot of Miles per Gallon vs. Number of Cylinders")

plot(data$Hardness, data$Potability, xlab = "Hardness", ylab = "Potability", main = "Scatter plot of Potability vs. Hardness")
 	
# Fit linear regression model 	
model <- lm(Potability ~ Hardness, data = data)
 	
# Add the regression line to the plot 	
abline(model, col = "red")
 	
# Predict Potability for Hardness=88 	
predict_value <- predict(model, newdata = data.frame(Hardness = 88))
cat("Predicted Potability for Hardness=88:", predict_value, "\n")
 	
# Summary of the linear regression model 	
summary(model)	

Ex.9
# Sample data 	
set.seed(42)
points <- c(50, 60, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175, 180, 185, 190, 195, 200, 205, 210, 215, 220, 225, 230, 235, 240, 245, 250, 255, 260, 265, 270, 275, 280, 285, 290, 295, 300, 350, 400)
# Add a few outliers 	
outliers <- c(20, 25, 30, 35, 400, 450, 500)
points <- c(points, outliers)
	Ex.10
	# Load the dataset
	diabetes <- read.csv("D:/phd/Dataset/diabetes_dataset.csv")
	
	# Scatterplot
	plot(diabetes$Age, diabetes$BloodPressure,
	    xlab = "Age", ylab = "Blood Pressure",
	    main = "Scatterplot of Blood Pressure vs. Age")

# Create a boxplot 	
boxplot(points,
    main = "Boxplot of Points Scored by Tennis Team Players",
    ylab = "Points",
    ylim = c(0, 500),
    notch = TRUE, # Add a notch to the box
    outline = TRUE, # Show outliers as individual points
    col = "lightblue" # Box color
)
	# Create age groups
	age_groups <- cut(diabetes$Age, breaks = 4)
	
	# Calculate average BloodPressure for each age group
	avg_bp <- tapply(diabetes$BloodPressure, age_groups, mean)
	
	# Bar chart
	barplot(avg_bp,
	    xlab = "Age Group", ylab = "Average Blood Pressure",
	    main = "Average Blood Pressure by Age Group",
	    col = "skyblue")