Compute an approximate median value for the data

Input:

```
#age, frequency

age<-c(5,15,20,50,80,110)

frequency<-c(200,450,300,1500,700,44)

median(age)

median(frequency)
```

Input:

```
#mean,median,mode,quatile

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)

mean(age)

median(age)

mode_age<-names(table(age))[table(age)==max(table(age))]

mode_age

range(age)

quantile(age,.25)

quantile(age,.75)
```

a) Smoothing by bin meanb) Smoothing by bin median

c) Smoothing by bin boundaries

input:

```
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)bins <- 5

bin_indices <- cut(data, bins)

mean_smooth <- tapply(data, bin_indices, mean)

print(mean_smooth)

median_smooth <- tapply(data, bin_indices, median)

median_smooth

min_max_smooth <- tapply(data, bin_indices, function(x) c(min(x), max(x)))

print(min_max_smooth)
```

Input:

```
v<-c(23,23,27,27,39,41,47,49,50,52,54,54,56,57,58,58,60,61)

min<-0

max<-1

#min_max

min_max=((35-min(v))/(max(v)-min(v)))

print(min_max)

#z-score

m=mean(v)

s<-12.94

z_score=(35-m)/s

print(z_score)

#decimal scaling

m<-35

j=max(m)<1

decimal_scaling=m/10^j

print(decimal_scaling)
```

Input:

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

mean(pencils)

median(pencils)

mode=names(table(pencils))[table(pencils)==max(table(pencils))]

mode
```

input:

```
#scatterplot

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.smooth(x,y)
```

Input:

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

num_bins <- 3

bins_eq_frequency <- cut(marks, breaks = num_bins, labels = FALSE)

hist(marks, breaks = num_bins, col = "lightblue", xlab = "Marks", main = "Equal-Frequency
(Equi-Depth) Partitioning")

marks <- c(55, 60, 71, 63, 55, 65, 50, 55, 58, 59, 61, 63, 65, 67, 71, 72, 75)

bin_mean <- tapply(data, cut(data, num_bins), mean)

smoothed_data_by_mean <- unname(bin_mean[as.character(cut(data, num_bins))])

bin_median <- tapply(data, cut(data, num_bins), median)

smoothed_data_by_median <- unname(bin_median[as.character(cut(data, num_bins))])

bin_boundaries <- tapply(data, cut(data, num_bins), function(x) c(min(x), max(x)))

smoothed_data_by_boundaries <- unlist(bin_boundaries[as.character(cut(data, num_bins))])

print("Original data:")

print(data)

print("Smoothed data by bin mean:")

print(smoothed_data_by_mean)

print("Smoothed data by bin median:")

print(smoothed_data_by_median)

print("Smoothed data by bin boundaries:")

print(smoothed_data_by_boundaries)
```

Calculate the Inter quantile and standard deviation of the given data.

Input:

```
#IQR, SD

v<-c(78.3,81.8,82,74.2,83.4,84.5,82.9,77.5,80.9,70.6)

IQR(v)

sd(v)
```

Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?

Input:

```
#Q1, Q2

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)

quantile(age,.25)

quantile(age,.75)
```

Input:

```
data <- data.frame(

Age = rep(c("5-6 years", "7-8 years", "9-10 years"), each = 3),

A = c(18, 2, 20, 22, 28, 10, 20, 40, 40),

B = c(22, 28, 10, 20, 40, 40, 30, 45, 50),

C = c(20, 40, 40, 30, 45, 50, 15, 35, 25)

)

covariance_BC <- cov(data$B, data$C)

cat("Covariance between B and C:", covariance_BC, "\n")

covariance_matrix <- cov(data[, c("A", "B", "C")])

cat("Covariance matrix:\n", covariance_matrix, "\n")

correlation_BC <- cor(data$B, data$C)

cat("Correlation between B and C:", correlation_BC, "\n")

correlation_matrix <- cor(data[, c("A", "B", "C")])

cat("Correlation matrix:\n", correlation_matrix, "\n")
```

Input:

```
# Given 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, 30, 30)

# (i) Equal-frequency partitioning with bin equal to 3

equal_freq_bins <- cut(prices, breaks = 3, labels = FALSE)

cat("(i) Equal-frequency partitioning bins:\n", equal_freq_bins, "\n")
```

18, 18, 18, 20, 20, 20, 20, 20, 20, 20, 21, 21, 21, 21, 25, 25, 25, 25, 25, 28, 28, 30,

30, 30.

(i) Partition the dataset using an equal-frequency partitioning method with bin equal to 3 (ii) apply data

smoothing using bin means and bin boundary.

(iii) Plot Histogram for the above frequency division

```
# (ii) Data smoothing using bin means and bin boundary

bin_means <- tapply(prices, equal_freq_bins, mean)

bin_boundaries <- unique(cut(prices, breaks = 3, labels = FALSE, include.lowest = TRUE))

cat("(ii) Bin Means:\n", bin_means, "\n")

cat("Bin Boundaries:\n", bin_boundaries, "\n")

# (iii) Plot Histogram

hist(prices, breaks = 3, main = "Histogram with Equal-frequency Partitioning", xlab =
"Prices", col = "lightblue", border = "black")
```

INPUT:

```
classA <- c(76, 35, 47, 64, 95, 66, 89, 36, 84)

classB <- c(51, 56, 84, 60, 59, 70, 63, 66, 50)

meanA <- mean(classA)

meanB <- mean(classB)

medianA <- median(classA)

medianB <- median(classB)

rangeA <- range(classA)

rangeB <- range(classB)

cat("(i) Class A vs Class B:\n")

cat("Mean: Class A -", meanA, " Class B -", meanB, "\n")

cat("Median: Class A -", medianA, " Class B -", medianB, "\n")

cat("Range: Class A -", diff(rangeA), " Class B -", diff(rangeB), "\n")

boxplot(classA, classB, names = c("Class A", "Class B"), col = c("lightblue", "lightgreen"),
main = "Boxplot - Class A vs Class B", ylab = "Scores")

cat("(ii) Inferences from Boxplot:\n")

cat(" - Class A has a wider range of scores compared to Class B.\n")

cat(" - The median score for Class A is higher than that for Class B.\n")

cat(" - Class A has an outlier which affects the mean.\n")
```

input:data("AirPassengers")

```
start_value <- 100

bin_width <- 150

bin_breaks <- seq(start_value, 700, by = bin_width)

hist(AirPassengers, breaks = bin_breaks, xlim = c(start_value, max(bin_breaks) +
bin_width),

main = "Histogram for AirPassengers Dataset",

xlab = "Passenger Count", ylab = "Frequency", col = "skyblue", border = "black")

legend("topright", legend = c("Passenger Count"), fill = c("skyblue"))
```

17.Obtain Multiple Lines in Line Chart using a single Plot Function in R.Use attributes“mpg”and“qsec”of the dataset “mtcars”

INPUT

```
data(mtcars)

plot(mtcars$mpg, type = "l", col = "blue", xlab = "Car Index", ylab = "Miles Per Gallon", main = "Multiple Lines Chart - mpg and qsec")

lines(mtcars$qsec, col = "red")

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

Input:

```
data <- c(200, 300, 400, 600, 1000)

min_max_custom <- function(x, min_val, max_val) {

return (x - min_val) / (max_val - min_val)

}

min_max_normalized_custom <- min_max_custom(data, 200, 1000)

min_max_normalized_default <- scale(data, center = min(data), scale = diff(range(data)))

z_score_normalized <- scale(data)

cat("Original Data: ", data, "\n\n")

cat("(a) Min-Max normalization with custom min and max values:\n")

cat("Normalized Data: ", min_max_normalized_custom, "\n\n")

cat("(b) Min-Max normalization with min = 0 and max = 1:\n")

cat("Normalized Data: ", min_max_normalized_default, "\n\n")

cat("(c) Z-score normalization:\n")

cat("Normalized Data: ", z_score_normalized, "\n")
```

Download the Dataset "water" From R dataset Link.Find out whether there is a linear relation between attributes"mortality" and"hardness" by plot function.Fit the Data into the Linear Regression model.Predict the mortality for the hardness==88.

INPUT

```
data(mtcars)

mortality <- mtcars$mpg

hardness <- mtcars$hp

plot(hardness, mortality, main = "Linear Regression: Mortality vs. Hardness",

xlab = "Hardness", ylab = "Mortality", pch = 16, col = "blue")

linear_model <- lm(mortality ~ hardness)

abline(linear_model, col = "red")

new_data <- data.frame(hardness = 88)

predicted_mortality <- predict(linear_model, newdata = new_data)

cat("Predicted Mortality for Hardness=88:", predicted_mortality, "\n")
```

19.Create a Boxplot graph for the relation between "mpg"(miles per galloon) and "cyl"(number of Cylinders) for the dataset "mtcars" available in R Environment.

Input:

```
data(mtcars)

boxplot(mpg ~ cyl, data = mtcars, main = "Boxplot: mpg vs. cyl",

xlab = "Number of Cylinders", ylab = "Miles Per Gallon", col = "skyblue")
```

Give suitable example using Boxplot visualization

technique.

Input:

```
points_scored <- c(35, 42, 48, 52, 56, 60, 62, 65, 68, 72, 76, 80, 85, 88, 92, 100, 110)

boxplot(points_scored, main = "Boxplot: Points Scored by Tennis Players",

xlab = "Players", ylab = "Points Scored", col = "lightblue", border = "black")
```

10. Implement using R language in which age group of people are affected by blood pressure based on the diabetes dataset show it using scatter plot and bar chart (that is BloodPressure vs Age using dataset “diabetes.csv”)

Input:

```
# Sample data (assuming the structure of your dataset)

data <- data.frame(

  Age = c(25, 30, 35, 40, 45, 50, 55, 60, 65, 70),

  BloodPressure = c(120, 130, 140, 150, 135, 145, 155, 160, 150, 140)

)

# Scatterplot

plot(data$Age, data$BloodPressure, main = "Blood Pressure vs Age", xlab = "Age", ylab = "Blood Pressure", pch = 16, col = "blue")

# Bar chart

barplot(data$BloodPressure, names.arg = data$Age, main = "Blood Pressure vs Age", xlab = "Age", ylab = "Blood Pressure", col = "green", border = "black")
```

model exam problem

Input:

```
marks <- c(55, 60, 71, 63, 55, 65, 50, 55, 58, 59, 61, 63, 65,

bins_a <- cut(marks, breaks = 3, labels = c("Low", "Medium", "High"))

bins_b <- cut(marks, breaks = seq(min(marks), max(marks), length.out = 4), labels = c("Low", "Medium", "High"))

k <- 3

clusters <- kmeans(matrix(marks), centers = k)

bins_c <- cut(clusters$centers[clusters$cluster], breaks = 3, labels = c("Low", "Medium", "High"))

par(mfrow = c(1, 3))

hist(marks, main = "Equal-frequency (equi-depth) partitioning", col = "skyblue", breaks = 3)

hist(marks, main = "Equal-width partitioning", col = "lightgreen", breaks = seq(min(marks), max(marks), length.out = 4))

hist(marks, main = "Clustering", col = "lightpink", breaks = 3)
```

strike rates tournament problem:
Input:

```
strike_rates <- c(100, 70, 60, 90, 90)

min_max_normalization <- function(x) {

(x - min(x)) / (max(x) - min(x))

}

normalized_min_max <- min_max_normalization(strike_rates)

z_score_normalization <- function(x) {

(x - mean(x)) / sd(x)

}

normalized_z_score <- z_score_normalization(strike_rates)

mad_normalization <- function(x) {

(x - mean(x)) / mad(x)

}

normalized_mad <- mad_normalization(strike_rates)

decimal_scaling_normalization <- function(x) {

x / 10^(ceiling(log10(max(x))))

}

normalized_decimal_scaling <- decimal_scaling_normalization(strike_rates)

cat("Original Data:", strike_rates, "\n\n")

cat("(a) Min-Max Normalization:", normalized_min_max, "\n")

cat("(b) Z-Score Normalization:", normalized_z_score, "\n")

cat("(c) Z-Score Normalization (MAD):", normalized_mad, "\n")

cat("(d) Normalization by Decimal Scaling:", normalized_decimal_scaling, "\n")
```

a) Calculate the standard deviation of AvgSpeed and TotalTime.

b) Calculate the Variance of AvgSpeed and TotalTime for the above dataset.

Input:

```
avg_speed <- c(78, 81, 82, 74, 83, 82, 77, 80, 70)

total_time <- c(39, 37, 36, 42, 35, 36, 40, 38, 46)

sd_avg_speed <- sd(avg_speed)

sd_total_time <- sd(total_time)

var_avg_speed <- var(avg_speed)

var_total_time <- var(total_time)

cat("Standard Deviation of AvgSpeed:", sd_avg_speed, "\n")

cat("Standard Deviation of TotalTime:", sd_total_time, "\n\n")

cat("Variance of AvgSpeed:", var_avg_speed, "\n")

cat("Variance of TotalTime:", var_total_time, "\n")
```

mobile phones problem:

R programming:

```
x_values <- c(4, 1, 5, 7, 10, 2, 50, 25, 90, 36)

y_values <- c(12, 5, 13, 19, 31, 7, 153, 72, 275, 110)plot(x_values, y_values, main="Scatter Plot of Mobile Phones Sold",

xlab="Number of Mobile Phones Sold", ylab="Money",

pch=16, col="blue")

grid()
```

Output: