**This is the instruction for run\_analysis.R to process the data**

1. **Download the dataset and check its existence**
2. **Assign each data to data frame**

**Read Feature Names**:

* + **features <- read.table("UCI HAR Dataset/features.txt", col.names = c("n","functions"))**: This line reads a file called "features.txt" located in the "UCI HAR Dataset" directory. It assigns column names "n" and "functions" to the columns of the resulting data frame. This file likely contains the names of features used in the dataset.

**Read Activity Labels**:

* + **activities <- read.table("UCI HAR Dataset/activity\_labels.txt", col.names = c("code", "activity"))**: This line reads a file called "activity\_labels.txt" located in the "UCI HAR Dataset" directory. It assigns column names "code" and "activity" to the columns of the resulting data frame. This file probably contains the mapping between activity codes and their corresponding labels.

**Read Test Data**:

* + **subject\_test <- read.table("UCI HAR Dataset/test/subject\_test.txt", col.names = "subject")**: Reads the subject IDs for the test set.
  + **x\_test <- read.table("UCI HAR Dataset/test/X\_test.txt", col.names = features$functions)**: Reads the test data features. It uses the feature names obtained in step 1 to assign column names to the test data.
  + **y\_test <- read.table("UCI HAR Dataset/test/y\_test.txt", col.names = "code")**: Reads the activity codes for the test set.

**Read Train Data**:

* + **subject\_train <- read.table("UCI HAR Dataset/train/subject\_train.txt", col.names = "subject")**: Reads the subject IDs for the training set.
  + **x\_train <- read.table("UCI HAR Dataset/train/X\_train.txt", col.names = features$functions)**: Reads the training data features. It also uses the feature names obtained in step 1 to assign column names to the training data.
  + **y\_train <- read.table("UCI HAR Dataset/train/y\_train.txt", col.names = "code")**: Reads the activity codes for the training set.

1. **Merges the training and the test to a new data set**

**Merge Subject IDs**:

* + **Subject <- rbind(subject\_train, subject\_test)**: Combines the subject IDs from both the training and test sets into a single data frame. **rbind()** function is used to combine the rows.

**Merge Feature Data**:

* + **X <- rbind(x\_train, x\_test)**: Combines the feature data from both the training and test sets into a single data frame. Again, **rbind()** function is used to combine the rows.

**Merge Activity Labels**:

* + **Y <- rbind(y\_train, y\_test)**: Combines the activity labels from both the training and test sets into a single data frame. **rbind()** function is used to combine the rows.

**Combine Subject IDs, Activity Labels, and Feature Data**:

* + **Merged\_Data <- cbind(Subject, Y, X)**: Combines the subject IDs, activity labels, and feature data horizontally into a single dataset. **cbind()** function is used to combine the data frames by column.

1. **Extracts only the measurements on the mean and standard deviation for each measurement**

**%>%** Operator:

* + This operator is known as the "pipe" operator. It takes the output from the left-hand side and passes it as the first argument to the function on the right-hand side. This allows for chaining operations together in a more readable and concise manner.

**select()** Function:

* + This function is used to select specific columns from a data frame.
  + In this case, it's being used to select columns based on certain criteria specified within the function.

**contains()** Function:

* + This function is used to filter column names that contain a specific pattern.
  + It's being used here to select columns that contain either "mean" or "std" in their names.

1. **Uses descriptive activity names to name the activities in the data set**

**TidyData$code**: Accesses the "code" column of the **TidyData** dataset.

**activities[TidyData$code, 2]**: This part of the code uses the numeric activity codes in the "code" column of **TidyData** to index the second column (containing activity labels) of the **activities** dataset.

**TidyData$code <- ...**: This assignment statement replaces the existing numeric activity codes in the "code" column of **TidyData** with their corresponding activity labels fetched from the **activities** d

1. **Appropriately labels the data set with descriptive variable names**

**names(TidyData)[2] = "activity"**: This line changes the name of the second column in **TidyData** to "activity", assuming that the second column contains activity labels.

**names(TidyData)<-gsub("Acc", "Accelerometer", names(TidyData))**: This line replaces "Acc" in column names with "Accelerometer".

**names(TidyData)<-gsub("Gyro", "Gyroscope", names(TidyData))**: This line replaces "Gyro" in column names with "Gyroscope".

**names(TidyData)<-gsub("BodyBody", "Body", names(TidyData))**: This line corrects duplicated "BodyBody" to "Body".

**names(TidyData)<-gsub("Mag", "Magnitude", names(TidyData))**: This line replaces "Mag" in column names with "Magnitude".

**names(TidyData)<-gsub("^t", "Time", names(TidyData))**: This line replaces "^t" at the beginning of column names with "Time".

**names(TidyData)<-gsub("^f", "Frequency", names(TidyData))**: This line replaces "^f" at the beginning of column names with "Frequency".

**names(TidyData)<-gsub("tBody", "TimeBody", names(TidyData))**: This line replaces "tBody" in column names with "TimeBody".

**names(TidyData)<-gsub("-mean()", "Mean", names(TidyData), ignore.case = TRUE)**: This line replaces "-mean()" in column names with "Mean", ignoring case.

**names(TidyData)<-gsub("-std()", "STD", names(TidyData), ignore.case = TRUE)**: This line replaces "-std()" in column names with "STD", ignoring case.

**names(TidyData)<-gsub("-freq()", "Frequency", names(TidyData), ignore.case = TRUE)**: This line replaces "-freq()" in column names with "Frequency", ignoring case.

**names(TidyData)<-gsub("angle", "Angle", names(TidyData))**: This line replaces "angle" in column names with "Angle".

**names(TidyData)<-gsub("gravity", "Gravity", names(TidyData))**: This line replaces "gravity" in column names with "Gravity".

1. **From the data set in step 4, creates a second, independent tidy data set with the average of each variable for each activity and each subject**

**%>%** Operator:

* + This operator, also known as the pipe operator, is used to chain together multiple operations. It passes the output from the left-hand side to the function on the right-hand side.

**group\_by(subject, activity)**:

* + This function groups the data in **TidyData** by the "subject" and "activity" columns. This means that the subsequent summarization will be performed within each unique combination of subject and activity.

**summarise\_all(funs(mean))**:

* + This function calculates the mean for each column within each group defined by **group\_by(subject, activity)**. **summarise\_all()** applies the function **mean()** to each column.
  + Since it's using **summarise\_all()**, it will calculate the mean for all columns in the grouped data frame.

**write.table(FinalData, "FinalData.txt", row.name=FALSE)**:

* + This function writes the **FinalData** data frame to a text file named "FinalData.txt".
  + **row.name=FALSE** argument ensures that row names are not written to the file.