```
library(tidyverse)
library(dplyr)
library(ggplot2)
library(tidyr)
library(forcats)
library(psych)
library(moderndive)
library(broom)


mydf <- filter(Random, Username == 'n10599568')
mydf <- filter(Data, Origin == 'BOI')

mydf = subset(mydf, select = -c(Origin, OriginStateName) )
mydf

mydf <- left_join(mydf, AirportCodes,
                  by = c("Dest" = "Airport Code"))
mydf <- left_join(mydf, AirlineCodes,
                  by = c("Reporting_Airline" = "Airline code"))

sapply(mydf, typeof)
sapply(mydf, class)



#Section A - Data and Visualization
#A1 Data Structure

#Create a data dictionary
var_names<- c('FlightDate', 'Reporting_airline', 'Destination', 'DestStateName',
'DepDelay','ArrDelay', 'Airport Name', 'Airline name')

var_desc <- c('date when flight departs','name of airline carrier', 'Codename Of
destination','destination state name of USA', 'Departure Delay count','Arrival Delay count',
'name of the airport it arrived', 'name of the airline' )

var_type <- c('character', 'character', 'character', 'character', 'double', 'double',
'character', 'character')

var_class <- c('character', 'character', 'character', 'character', 'numeric', 'numeric',
'character', 'character')

mydf_data <- data.frame(Variable = var_names, description = var_desc, type = var_type, class =
var_class)

#change columns class from character to more appropriate

mydf$FlightDate <- as.Date(mydf$FlightDate, '%d/%m/%Y')
mydf$Reporting_Airline <- as.factor(mydf$Reporting_Airline)
mydf$Dest <- as.factor(mydf$Dest)
mydf$DestStateName <- as.factor(mydf$DestStateName)
mydf$`Airport Name` <- as.factor(mydf$`Airport Name`)
mydf$`Airline name` <- as.factor(mydf$`Airline name`)
mydf$DepDelay <- as.numeric(unlist(mydf$DepDelay))
mydf$ArrDelay <- as.numeric(unlist(mydf$ArrDelay))
sapply(mydf, class)




to <- c('Date', 'factor', 'factor', 'factor', 'numeric', 'numeric', 'factor', 'factor')
converted_mydf <- data.frame(Variable = var_names, from = var_type, to )

#Total no.of observations
NROW(mydf$DepDelay)
#answer is 1662 observations
#mean = 6.036, median = -4 and shows there are 7 missing valuES.
summary(mydf$DepDelay)
```

```
#standard deviation of departure delays = 40.58908
sd(mydf$DepDelay, na.rm = TRUE)


#report mean, median and standard deviation data for top 5 most popular airlines
head(n = 5, mydf %>%
       group_by(`Airline name`) %>%
       summarise(average = mean(DepDelay, na.rm = TRUE)))
head(n =5, mydf %>%
       group_by(`Airline name`) %>%
       summarise(median = median(DepDelay, na.rm = TRUE)))
sd <- head(n = 5 , mydf %>%
             group_by(`Airline name`) %>%
             summarise(sd = sd(DepDelay), na.rm = TRUE))
sd

#A2 Graphical Summaries
#A.1.Create a graphically excellent plot that shows the relationship between departure delay
and arrival delay.

scatPlot <- ggplot(mydf, aes(x = DepDelay,y = ArrDelay))
scatPlot + geom_point() +
  theme_light()+
  labs(x = 'Departure Delay',
       y = 'Arrival Delay',
       title = 'Relationship between departure and arrival delay ')+
  stat_smooth(se = FALSE)
#A.2.Create a graphically excellent plot that shows the distribution of departure delays.
histoGram1<- ggplot(data = mydf, aes(x = DepDelay))
histoGram1 + geom_histogram(binwidth = 5, color = 'darkslategray',
                            fill = 'darkslategray' )+
  scale_x_continuous(breaks = seq(-50, 300, 25))+
  scale_y_continuous(breaks = seq(0, 700, 100))+
  ggtitle("Distripution of departure delay")+
  labs(x = "departure delay", y = "scale count")+
  theme_minimal()


#A.3.Create a graphically excellent plot that shows the distribution of arrival delays.

histoGram1 <- ggplot(data = mydf, aes(x = ArrDelay))
histoGram1 + geom_histogram(binwidth = 5, color = 'darkslategray',
                            fill = 'darkslategray' )+
  scale_x_continuous(breaks = seq(-50, 300, 25))+
  scale_y_continuous(breaks = seq(0, 700, 100))+
  ggtitle("Distripution of arrival delay")+
  labs(x = "arrival delay", y = "scale count")+
  theme_minimal()


# Include only the three most popular airlines and combine the rest as 'other'.
mydf %>% summarise(Alinecnt = n_distinct(`Airline name`))
aline_group <- mydf %>% group_by(Reporting_Airline) %>%
  summarise(counts = n()) %>% arrange(desc(counts))
mydf = mydf %>% mutate(aline = fct_lump(`Airline name`, 3))
unique(mydf$aline)
#Hence, top 3 airlines are skywest airlines, southwest airlines and delta airlines.
#A.4.Create a plot that?sh?ws how departure delay varies by airline. Change your axis limits so
you only display
#times between -20 and 20 minutes.

bxPlt1 <- ggplot(mydf, aes(x = DepDelay, y = aline))
bxPlt1 + geom_boxplot()+
  labs(title = "departure delay varies by airlines")+
  theme_light()+ xlim(-20,20)

# Include only the three most popular airlines and combine the rest as 'other'.
mydf %>% summarise(destcnt = n_distinct(DestStateName))
dest_group <- mydf %>% group_by(DestStateName) %>%
```

```
  summarise(counts = n()) %>% arrange(desc(counts))
mydf = mydf %>% mutate(destination = fct_lump(DestStateName, 3))
unique(mydf$destination)

#A.5.Create a plot that shows how arrival delay varies by airline. Change your axis limits so
you only display
#times between -40 and 40 minutes.

bxPlt2<- ggplot(mydf, aes(x = ArrDelay, y = destination))
bxPlt2 + geom_boxplot()+
  labs(title = "arrival delay varies by airlines")+
  theme_light()+ xlim(-40,40)

#B.1. Based on aline_group data we know that skywest and southwest airlines are top two most
popular airlines in my origin.

hypo_table <- mydf %>%
  na.omit() %>%
  filter(Reporting_Airline %in% c("OO", "WN")) %>%
  group_by(Reporting_Airline)

hypo_table = select(hypo_table, 2,5)
hypo_table$Status <- ifelse(hypo_table$DepDelay >0, 'Late', "Early/On Time")

summary(hypo_table)
hypo_table$Reporting_Airline <- factor(hypo_table$Reporting_Airline,
                                       levels = c("OO", "WN"))
hypo_table$Status <- factor(hypo_table$Status,
                            levels = c("Late", "Early/On Time"))
summary(hypo_table)

ggplot(data = hypo_table, aes(x = Status,
                              fill = factor(Reporting_Airline)))+
  geom_bar(position = "dodge")+
  scale_x_discrete(name = "Status",
                   labels = c("Late", "Early/On Time"))+
  scale_fill_discrete(name = "Late/Early Status")+
  theme_bw()

observed <- table(hypo_table$Reporting_Airline, hypo_table$Status)
colnames(observed) <- c("Late", "Early/On Time")
observed

total_aline <- summarise(group_by(hypo_table,Reporting_Airline),
                         count = n())
total_stat <- summarise(group_by(hypo_table,Status),
                        count = n())
total_aline <- matrix(total_aline$count)
total_stat <- matrix(total_stat$count)

expected <- data.frame(total_aline%*%t(total_stat)/sum(total_aline))
contribution <- (observed - expected)^2/expected
#B.2. Hypothesis Test and Interpretation
test_stat <- sum(contribution)
test_stat

deg_ofreedom <- (nrow(observed)-1)*(ncol(observed)-1)
deg_ofreedom

pchisq(q = test_stat, df = deg_ofreedom, lower.tail= F)


chisq <- chisq.test(observed)
chisq

#C. Linear Regression Model
#C.1. Create Linear Model (10%)
#fit a linear model to your chosen variables
#Produce a captioned, well-formatted table below that includes a descriptive parameter name,
```

```
estimate and 95% confidence interval.

lr_df <- select(mydf, 5,6)
describe(lr_df)

linmod <- lm(ArrDelay ~ DepDelay, lr_df)
linmod
ggplot(lr_df , aes(DepDelay, ArrDelay))+
  geom_point()+
  theme_light()+
  labs(x = "Departure delay",
       y = "Arrival delay",
       title = "Relationship between Departure and Arrival delay")+
  stat_smooth(method = 'lm', se = FALSE)

lr_table <- summary(linmod)
lr_table


get_regression_table(linmod)


#How much variability in the observed data does your model explain?

round(glance(linmod)$r.squared,4)

#C.2 Regression Assumptions
#create dataframe to analyse residuals

lm.for <- fortify(linmod)
head(lm.for)

#C.1. a plot that shows how the residuals vary with the values fitted through your regression
model.

ggplot(data = lm.for, aes(x = .fitted, y = .resid))+
  geom_point()+
  theme_bw()+
  geom_smooth()+
  labs(x = expression(paste("Fitted (",hat(y[i]), ")")), y = expression(paste("Residual
(",epsilon[i],")")))

#normality of residuals and standardised residuals

ggplot(data = lm.for, aes(x = .resid))+
  geom_histogram(colour = "grey", fill = "coral", aes(y = ..density..))+
  theme_bw()+
  stat_function(fun = "dnorm", args = list(mean = mean(lm.for$.resid), sd =
sd(lm.for$.resid)))+
  labs(x = "Residual", y = "Density")

ggplot(data = lm.for, aes(x = .stdresid))+
  geom_histogram(colour = "grey", fill = "coral", aes(y = ..density..))+
  theme_bw()+
  stat_function(fun = "dnorm", args = list(mean = 0, sd = 1))+
  labs(x = "Standardised Residual", y = "Density")


#C.2. QQ plot that compares the standardised residuals to a standard normal distribution.

ggplot(data=lm.for, aes(sample=.stdresid)) +
  stat_qq() +
  geom_abline(intercept=0, slope=1) +
  coord_equal()+
  theme_bw()+
  labs(x = "q values from standard normal",
       y = "q values from standardised residuals")

IQ <- read_csv('IQ.csv')
```

```
#ecdf
IQ_ecdf <- ecdf(IQ$IQ)

ggplot(data = data.frame(x = c(80,120)), aes(x = x))+
  stat_function(fun = IQ_ecdf)+
  theme_bw()

#define fucntion so not need to enter parameters
#cdf
p_cdf <- function(x){
  return(pnorm(q = x, mean = 100, sd = 20))
}

ks.test(x = IQ$IQ, y = "p_cdf")

#we reject the null hypothesis. Standard residuals does not follow normal distribution.

#to check if it's normal
mean(IQ$IQ)
sd(IQ$IQ)
```