Data Coding for R studio

Variables:

- Similar_study: If they have taken part in a study similar to the current one (2 = no, 1 = yes)
- Protagonist: if they have been the protagonist from this situation in their real life
- Antagonist: if they have had an experience of being the antagonist from this situation
- Guilt_prone: to what extent they would say they are prone to experiencing guilt
- Guilt: to what extent does the participant perceives guilt in the transgressor
- Shame: to what extent does the participant perceives shame in the transgressor
- Regret: to what extent does the participant perceives regret in the transgressor
- Future: to what extent they believe the transgressor will reoffend
- Punishment: the extent the participant thinks the punishment was sufficient
- Forgiveness: how likely they believe the transgressor should be forgiven
- Moral character: the extent the participant believes the transgressor has a moral character
- Trust: the extent participants believe the transgressor to be trustworthy
- Condition: condition that the participant was allocated (1 = correct self-punishment, 0 = low self-punishment, 2 = high self-punishment)

```
#Correlational Analyses#
variable_data <- select(.data = dissertation_guilt, Future, Forgiveness, Punishment, Trust,
MoralCharacter, Shame, Guilt, Regret)
head(variable_data, 6)
x <- as.numeric(variable_data$Condition)
res <- cor(variable_data)
round(res, 2)
res2 <- rcorr(as.matrix(variable_data))
res2
flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
  )
}
flattenCorrMatrix(res2$r, res2$P)

install.packages("corrplot")
library(corrplot)
corrplot(res, method = 'number')

corrplot(res, method = 'number', type = "lower", order = "hclust",
         tl.col = "black", tl.srt = 45)

corrplot(res2$r, type="upper", order="hclust",
         p.mat = res2$P, sig.level = 0.01, insig = "blank")

col<- colorRampPalette(c("blue", "white", "red"))(20)
heatmap(x = res, col = col, symm = TRUE)

install.packages("GGally")
library(GGally)

panel.cor <- function(x, y, digits=2, prefix="", cex.cor)
{
  usr <- par("usr"); on.exit(par(usr))
  par(usr = c(0, 1, 0, 1))
  r = (cor(x, y))
  txt <- format(c(r, 0.123456789), digits=digits)[1]
  txt <- paste(prefix, txt, sep="")
  if(missing(cex.cor)) cex <- 0.8/strwidth(txt)
  text(0.5, 0.5, txt, cex = cex * abs(r))
}

corstarsl <- function(x){
  require(Hmisc)
  x <- as.matrix(x)
  R <- rcorr(x)$r
  p <- rcorr(x)$P

  ## define notions for significance levels; spacing is important.
  mystars <- ifelse(p < .001, "***", ifelse(p < .01, "** ", ifelse(p < .05, "* ", " ")))

  ## trunctuate the matrix that holds the correlations to two decimal
  R <- format(round(cbind(rep(-1.11, ncol(x)), R), 2))[,-1]

  ## build a new matrix that includes the correlations with their apropriate stars
  Rnew <- matrix(paste(R, mystars, sep=""), ncol=ncol(x))
  diag(Rnew) <- paste(diag(R), " ", sep="")
  rownames(Rnew) <- colnames(x)
  colnames(Rnew) <- paste(colnames(x), "", sep="")

  ## remove upper triangle
  Rnew <- as.matrix(Rnew)
  Rnew[upper.tri(Rnew, diag = TRUE)] <- ""
  Rnew <- as.data.frame(Rnew)

  ## remove last column and return the matrix (which is now a data frame)
  Rnew <- cbind(Rnew[1:length(Rnew)-1])
  return(Rnew)
}

correlation <- corstarsl(variable_data)
correlation

mydata.rcorr = rcorr(as.matrix(variable_data))
mydata.rcorr
mydata.coeff = mydata.rcorr$r
mydata.p = mydata.rcorr$P

#Linear Regression#
library(tidyverse)
library(broom)
# Guilt #

guilt.lm <- lm(Guilt ~ Condition, data = dissertation_guilt)
summary(guilt.lm)
hist(resid(guilt.lm),main='Histogram of residuals',xlab='Standardised
Residuals',ylab='Frequency')
plot(guilt.lm)
qqPlot(guilt.lm, simulate=TRUE, main="Q-Q Plot")

residplot <- function(fit, nbreaks=10) { z <- rstudent(fit)

hist(z, breaks=nbreaks, freq=FALSE,
    xlab="Studentized Residual",
    main="Distribution of Errors")

rug(jitter(z), col="brown")

curve(dnorm(x, mean=mean(z), sd=sd(z)),
     add=TRUE, col="blue", lwd=2)

lines(density(z)$x, density(z)$y,
     col="red", lwd=2, lty=2)

legend("topright", legend = c( "Normal Curve", "Kernel Density Curve"), lty=1:2,
col=c("blue","red"), cex=.7)
```

```
}

residplot(guilt.lm)

pguilt <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$Guilt, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Guilt", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")
pguilt

#Regret#
regret.lm <- lm(Regret ~ Condition, data = dissertation_guilt)
summary(regret.lm)

qqPlot(regret.lm, simulate=TRUE, main="Q-Q Plot")
residplot(regret.lm)

pregret <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$Regret, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Regret", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")

#Shame#
shame.lm <- lm(Shame ~ Condition, data = dissertation_guilt)
summary(shame.lm)

qqPlot(shame.lm, simulate=TRUE, main="Q-Q Plot")
residplot(shame.lm)

pshame <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$Shame, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Shame", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")

#Future#
future.lm <- lm(Future ~ Condition, data = dissertation_guilt)
summary(future.lm)

qqPlot(future.lm, simulate=TRUE, main="Q-Q Plot")
residplot(future.lm)

pfuture <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$Future, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Future", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")

#Punishment#
punishment.lm <- lm(Punishment ~ Condition, data = dissertation_guilt)
summary(punishment.lm)
```

```
qqPlot(punishment.lm, simulate=TRUE, main="Q-Q Plot")
residplot(punishment.lm)

ppunish <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$Punishment, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Punishment", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")

#Forgiveness#
forgiveness.lm <- lm(Forgiveness ~ Condition, data = dissertation_guilt)
summary(forgiveness.lm)

qqPlot(forgiveness.lm, simulate=TRUE, main="Q-Q Plot")
residplot(forgiveness.lm)

pforgive <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$Forgiveness, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Forgiveness", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")

#Moral Character#
moral.lm <- lm(MoralCharacter ~ Condition, data = dissertation_guilt)
summary(moral.lm)

qqPlot(moral.lm, simulate=TRUE, main="Q-Q Plot")
residplot(moral.lm)

pmoral <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$MoralCharacter, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Moral Character", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")

#Trust#
trust.lm <- lm(Trust ~ Condition, data = dissertation_guilt)
summary(trust.lm)

qqPlot(trust.lm, simulate=TRUE, main="Q-Q Plot")
residplot(trust.lm)

ptrust <- ggplot() + geom_boxplot(aes(x = dissertation_guilt_words$Condition, y =
dissertation_guilt_words$Trust, colour = dissertation_guilt_words$Condition)) +
  labs(title = "Trust", x = "Condition", y = "Likert Rating", colour = "Punishment
Conditions")
ptrust

grid.arrange(pguilt, pregret, pshame, pfuture, ppunish, pforgive, pmoral, ptrust,
             ncol = 2, nrow = 4)
```