```
set.seed(123)

#problem 1
#i
N = 1000
K = 5
Xtilde = matrix(rnorm(N*K,0,1), N, K)
X1=Xtilde[,1]
X2=Xtilde[,2]+X1
X3=Xtilde[,3]+X2
X4=Xtilde[,4]+X3
X5=Xtilde[,5]+X4

X=matrix(c(X1,X2,X3,X4,X5),N,5)
E = apply(X,2,mean)
print(E)
V = var(X)
print(V)

#ii
first_funct <-function(N,B, X){

  x=X[1:N,]
  beta_0 = rep(1,5)
  beta_matrix = matrix(0, B, K)

  t1 <- rep(0, B)
  t2 <- rep(0, B)
  hat_s1 <- rep(0, B)
  hat_s2 <- rep(0, B)

  for(b in 1:B){

    #create error
    e_b <- rnorm(N,0, sqrt(2))
    y_b <- x %*% beta_0 + e_b

    #calc beta
    beta_b <- solve((t(x)%*%x))%*%t(x)%*%y_b
    beta_matrix[b,] <- beta_b

    #calc residuals
    u_b <- y_b - x%*%beta_b

    #calc var
    hat_s1[b] <- 1/N*sum(u_b^2)
    hat_s2[b]  <- 1/(N-K)*sum(u_b^2)

    #tstats
    t1[b] <- (beta_b[1]-beta_0[1])/sqrt(hat_s1[b]*solve(t(x)%*%x)[1,1])
    t2[b] <- (beta_b[1]-beta_0[1])/sqrt(hat_s2[b]*solve(t(x)%*%x)[1,1])
  }
  return(list(betas=beta_matrix, hat_s1=hat_s1, hat_s2=hat_s2, t1=t1, t2=t2))
}


#iii
N = c(10, 20, 100, 1000)
B=1000


#storing results
s1_means <- rep(0,length(N))
s2_means <- rep(0, length(N))
t1_stats <- matrix(0,nrow = B ,ncol =length(N))
t2_stats <- matrix(0,nrow = B, ncol= length(N))
#runs the function for each N value
for(i in 1:length(N)){
  reg <- first_funct(N[i], B, X)
  s1_means[i] <- mean(reg$hat_s1) #stores means
  s2_means[i] <- mean(reg$hat_s2)
  #stores t stats
  t1_stats[,i] <- reg$t1
  t2_stats[,i] <- reg$t2
}

var(reg$betas)
2*solve(t(X)%*%X)
print(s1_means)
print(s2_means)

#generates the norm
Z = rnorm(1000)
m = mean(Z)
for(i in 1:length(N)){
  hist(t1_stats[,i], prob=TRUE,xlab= "t1", main = "t1 dist against std normal") #plots vector of t1 stats
  curve(dnorm(x,mean=m, sd=sd(Z)), col="blue", lwd=2, add=TRUE, yaxt="n") #puts mean curve
  hist(t2_stats[,i], prob=TRUE,xlab= "t2", main = "t2 dist against std normal")
  curve(dnorm(x,mean=m, sd=sd(Z)), col="blue", lwd=2, add=TRUE, yaxt="n")
}


second_funct <-function(N,B, X){
  rejections = 0
  x=X[1:N,]
  beta_0 = rep(1,5)
  beta_matrix = matrix(0, B, K)

  t1 <- rep(0, B)
  t2 <- rep(0, B)
  hat_s1 <- rep(0, B)
  hat_s2 <- rep(0, B)

  for(b in 1:B){

    #create error
    e_b <- rnorm(N,0, sqrt(2))
    y_b <- x %*% beta_0 + e_b

    #calc beta
    beta_b <- solve((t(x)%*%x))%*%t(x)%*%y_b
    beta_matrix[b,] <- beta_b

    #calc residuals
    u_b <- y_b - x%*%beta_b

    #calc var
    hat_s1[b] <- 1/N*sum(u_b^2)
    hat_s2[b]  <- 1/(N-K)*sum(u_b^2)

    #tstats
    t1[b] <- (beta_b[1]-0)/sqrt(hat_s1[b]*solve(t(x)%*%x)[1,1])
    t2[b] <- (beta_b[1]-0)/sqrt(hat_s2[b]*solve(t(x)%*%x)[1,1])

  }
  #calcualtes percentage of rejections if abs of tvalue exceed barrier
  rejections = rejections + length(t1[abs(t1) > 1.96])
  rejections = rejections + length(t2[abs(t2) > 1.96])
  percent_rejections = rejections/(2*B)
  return(percent_rejections = percent_rejections)
}

for(i in 1:length(N)){
  reg <- second_funct(N[i], B, X)
  print(reg)
}


#problem 2
#i
install.packages('foreign')
library(foreign)
setwd('C:/Users/Ethan/Desktop/Columbia/Senior_Fall/Advanced_Econometrics/pset_1')
data = read.dta('TeachingRatings.dta')

#ii
y=data$course_eval
X=cbind(1, data$beauty, data$age)
colnames(X)=c('intercept', 'beauty', 'age')
plot(X[,'beauty'], y)
beta_hat <- solve(t(X)%*%X)%*%t(X)%*%y #computes beta_hat using formula
residual <- y - X%*%beta_hat
mean(residual)
var(residual)
sd(residual)

#makes residual maker and finds residuals
M <- diag(length(y))-X%*%solve(t(X)%*%X)%*%t(X)
residual_2 <- M%*%y
#reports values of residuals
mean(residual_2)
sd(residual_2)

#iii
X1 = rep(1, length(X[,'beauty']))
X0 = cbind(X[,'beauty'], X[,'age'])
I = diag(length(y))
M1 = I - X1%*%solve(t(X1)%*%X1)%*%t(X1)
beta_demean = solve(t(X0)%*%M1%*%X0)%*%t(X0)%*%(M1%*%y)
beta

beauty_x = X[,'beauty']
age_x = X[,'age']

#iv
s_squared = t(residual_2)%*%residual_2/(length(beauty_x)-3) #computes sample variance
sigma = s_squared[1]*solve(t(X)%*%(X)) #computes var covar matrix

#CIs based on formula
b1_upper_confidence = beta_demean[1,1]+1.96*sqrt(SE[2,2]) #2,2 is the variance entry in cov/var matrix
b1_lower_confidence = beta_demean[1,1]-1.96*sqrt(SE[2,2])

b2_upper_confidence = beta_demean[2,1]+1.96*sqrt(SE[3,3]) #3,3 is the variance entry in the cov/var matrix
b2_lower_confidence = beta_demean[2,1]-1.96*sqrt(SE[3,3])


#v
#demeans the y_hat
y_hat_demean = (M1%*%X0)%*%beta_demean
#computes normal yhat
y_hat = X%*%beta_hat
#computes centered r squared on demeaned y and demeaned yhat
centered_R_sqrd = t(y_hat_demean)%*%(y_hat_demean)/(t(M1%*%y)%*%(M1%*%y))
#computes uncentered r squared on original y and y_hat
uncentered_R_S = t(y_hat)%*%y_hat/(t(y)%*%y)

#vi
#computes mean of residual
m_resid = mean(residual_2)
#computes correlation b/w residuals and each regressor
cor(residual_2, X[,'beauty'])
cor(residual_2, X[,'age'])
```