```
a2 <- 2
b2 <- 1.5
x2 <- seq(1, 8, 0.05)
e2 <- rnorm(length(x2),0,0.5)
y2 <- a2 + b2*sin(x2) + e2
y2_line <- a2 + b2*sin(x2)
plot(x2,y2)

lpr <- loess(y2~x2)                      # Run a local polynomial regression (nonparametric
                                         # regression).

lines(x2,y2_line,col = "blue",lwd=3)     # Add the population regression curve on the plot
lines(x2,fitted(lpr),col = "red",lwd=3)  # Add the fitted regression curve on the plot
colors()

rm(list=ls())                       # clear the workspace.

##############################################################################################

# A few of essential components to learn to use R

# Operations with R

# 1. Arithmetic operators

5 + 10                         # addition
5 - 10                         # subtraction
3 * 5                          # multiplication
(5 + 5)/2                      # division
-1/0                           # Inf: infinity
0/0                            # NaN: Not a number
2^5                            # Exponentiation (or 2**5)
26 %% 5                        # Remainder
26 %/% 5                       # quotient (integer part)
                               # You can also use floor() to have the
                               # integer part and remainder.
sqrt(9)                        # Or 9^(0.5)

# 2. Logical operators: TRUE or FALSE

17 > 10
17 < 10
7 >= 7                         # How about 7 => 7
17 <= 10
17 == 10
17 != 10
(17 > 10) + (17 == 10)         # TRUE is also recognized as 1 and FALSE as 0.
(7 >= 7) < (10 < 7)

(17 > 17) & (TRUE+FALSE == 1)  # "and" and "or" operators
(17 > 17) | (TRUE+FALSE == 1)

# Variable Assignment: We can give names to data objects and these give us variables.

r <- 3                         # Assign the value 3 to r  (r=3 also works),
r                              # print out the value of the variable r (print(x) also works)
R                              # Case sensitive
area <- pi * r^2               # "pi" is a built-in variable.

# Exercises ########################################################################################

# Assign the value 5 to the variable "my_apples"


# Print out the value of the variable "my_apples"


set.seed(1)

# Generate a random variable from the chi square distribution with degree of freedom 5 (use rchisq()),
# and assign this to B


# Assign the rounded integer value of my_apples*(B/3) to the variables my_oranges.: use round()


# Compare my_apples and my_oranges and assign TRUE to "Comp" if my_apples > my_oranges and assign
# FALSE otherwise.


# Add these two variables together and assign this value to the new variable "my_fruit"


# Print out the value of the variable my_fruit


#######################################################################################################

# Types of variables

my_numeric <- 2                 # numbers are called numeric
my_character <- "Father"        # Text or string values are called characters
my_logical <- 1 > 2             # TRUE and FALSE (or T and F) are called logical.

is.numeric(my_numeric)
class(my_numeric)               # How can you check the class of my_character and my_logical?

# Exercises ###########################################################################################

# Compare mother and father
my_character2 <- "Mother"

# Which one is TRUE?
my_character2 == my_character
my_character2 > my_character
my_character2 < my_character

# Check the class of my_character2


#  Create a variable cl that has 1 if my_character is character or logical, and -1 otherwise
cl <-
cl

#######################################################################################################
my_numeric + my_logical       # TRUE is recognized as 1 and FALSE as 0.

# change the class of a variable
my_numeric <- as.character(my_numeric)
class(my_numeric)

my_numeric + my_logical

```

```
# Exercise
##########################################################################################

# Generate a standard normal random. rnorm()
# Check whether this number is >1.96 or <-1.96
# Replicate this 1000 times and count the number of observations
# that are greater than 1.96 or less than -1.96.
# Calculate the ratio of this number to the the number of replications
# and see whether this is close to 0.05.

num <- 0
for (i in 1 : 1000) {
  a <-

  }
num/1000

# Simulation about the CLT.
# Generate 100 zero and one binary random variables and take the mean.
# calculate the t value and count if it is not between (-1.96,1.96)
# Repliate this calculation 1000 times and see how many you count.

reject <- 0
for (i in 1 : 1000) {
  x <- round(runif(100,0,1))
  x_bar <- mean(x)
  se_x <- sd(x)


  }
print(reject/1000)
```