```
########################################
############ Assignment  2 #############
########################################

###Instructor: Prof. Gunnar Heins
###Student:    C.H.(Chenhui Lu)
###UFID:       76982846

#=============================================================================

##1 Writing Functions [30 points]

#1. Write a function that tests if a vector x is longer than a vector y or equally long.

Test_longer <- function(x,y){
  length1 <- length(x) # get the length of vector x
  length2 <- length(y) # get the length of vector y
  z <- length1 >= length2 # campare the length
  return(z)
}

#Example
x <- c(1,8,3,4)
y <- c(1,3,9,2,3)
Test_longer(x,y)
#[1] FALSE.
#works



#2. Write a function that tests if a vector x has more elements that are greater than 5
than a vector y.

Test_bigger_element <- function(x,y){    # set up function
  h <- 0                                 # h is counting number
  for ( i in 1:length(x)) {              # count the number of elements of x which > 5
    if (x[i] > 5) {
      h <- h + 1
    }
  }

 k <- 0                                  # h is counting number
 for (i in 1:length(y)) {                # similarly
   if (y[i] > 5) {
     k <- k + 1
   }
 }

 return(h > k)                           # compare and return
}

#Example 1.
Test_bigger_element(x,y)
# [1] FALSE
#Example 2.
x2 <- c(8,6,5,8,1)
y2 <- c(2,4,5,9)
Test_bigger_element(x2,y2)
# [1] TRUE
# Works



#3. Write a function with three arguments (x,y, and c, where c is a scalar) that tests if
```

a vector x has more elements that are smaller than c than vector y.

```
Test_smaller_element <- function(x,y,c){ # set up function
  h <- 0                                 # h is counting factor
  for ( i in 1:length(x)) {              # count the number of elements of x which < c
    if (x[i] < c) {
      h <- h + 1
    }
  }

  k <- 0                                 # h is counting factor
  for (i in 1:length(y)) {               # similarly
    if (y[i] < c) {
      k <- k + 1
    }
  }

  return(h > k)                          # compare and return
}

#Example 1.
Test_smaller_element(x,y,3)
# [1] FALSE
#Example 2.
Test_bigger_element(x2,y2)
# [1] TRUE
# Works



#4. Suppose you have two vectors, x and y, each with 3 distinct elements.
# Write a function that returns the number of elements that x and y have in common
# (e.g. if the vectors are c(1,2,3) and c(3,6,2), it should return 2).

Com_element <- function(x,y){       #set up funciton
  h <- 0                            #set up counting factor
  for (i in 1:length(x)) {          #loop in x
    x_i <- x[i]
    for (j in 1:length(y)) {        #loop in y which under x's loop
      y_j <- y[j]
      if (x_i == y_j) {             #count the number of x_i = y_j,
        h <- h + 1
      }
    }
  }
  h                                 #return the counting number
}

#Example 1
x <- c(1,2,3)
y <- c(3,6,2)
Com_element(x, y)
#[1] 2
#Example 2
h <- c(5,3,6)
k <- c(5,6,3)
Com_element(h, k)
#[1] 3
#works



##2 Writing Loops [40 points]

#1.Use a for-loop to compute: sum(i=1 to 100) [i / (i + 1)]
```

```
z <- 0                  # initial sum = 0
for (i in 1:100) {      # set up the loop
  t <- i / (i + 1)      # t is the scalar of every components
  z <- z + t
}
z                       # return the sum
#[1] 95.80272



#2 Euler's number e
z <- 0                      # initial sum = 0
for (i in 0:100) {          # set up the loop
  z = z + 1/(factorial(i)) # calculate the sum of every components
}
z                           # return the sum
#[1] 2.718282



#3.Use 2 for-loops to compute
z <- 0                      # initial sum = 0
for (i in 1:6) {            # set up the loop of i
  for (j in 1:6) {          # set up the loop of j which is under loop i
    z = z + (i - j)^2       # calculate the sum of every components
  }
}
z                           # return the sum
# [1] 210



#4.Suppose vector x is given by x = {2, 4, 3, 5, 1, 7}.
# Compute the following summationusing loops
sum_sum <- function(x){             # set up the function which can slove the problem
similar to above
  z <- 0                            # initial sum = 0
  for (i in 1:length(x)) {          # set up the double loops
    for (j in 1:length(x)) {
      z <- z + (x[i] - x[j])^2      # calculate the sum of every components
    }
  }
  z                                 # return the sum
}

x = c(2, 4, 3, 5, 1, 7)             # calculate with the function
sum_sum(x)
```