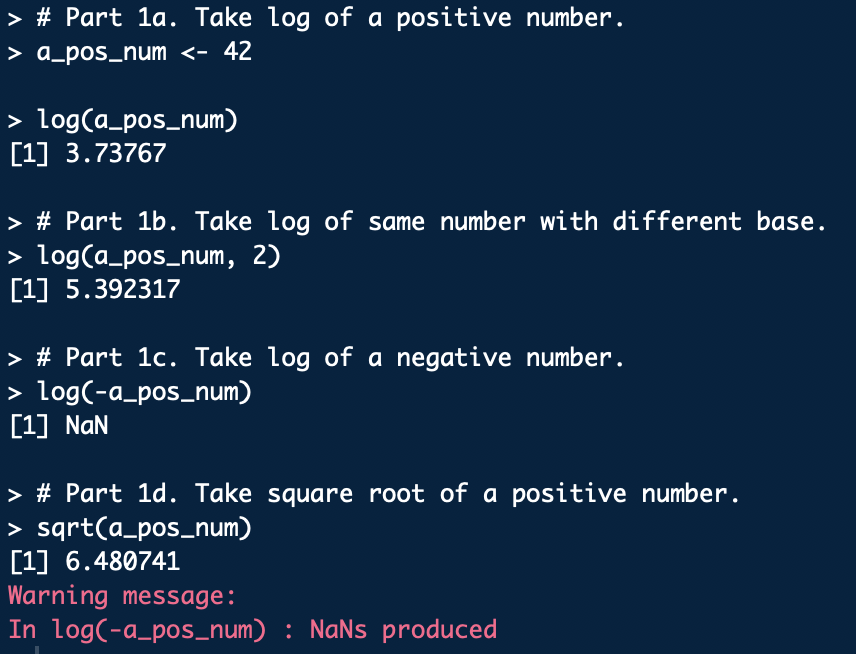
**Question 1**: Basic Math

# **Part 1a**. Take log of a positive number.

a\_pos\_num <- 42

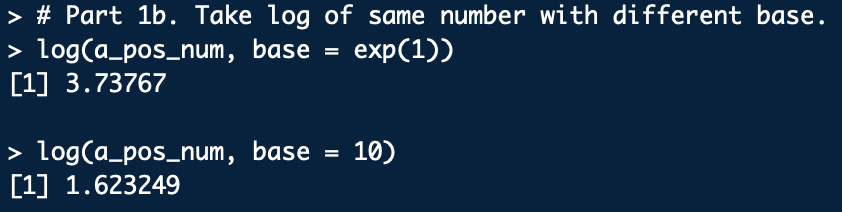
log(a\_pos\_num)



# **Part 1b**. Take log of same number with different base.

log(a\_pos\_num, base = exp(1))

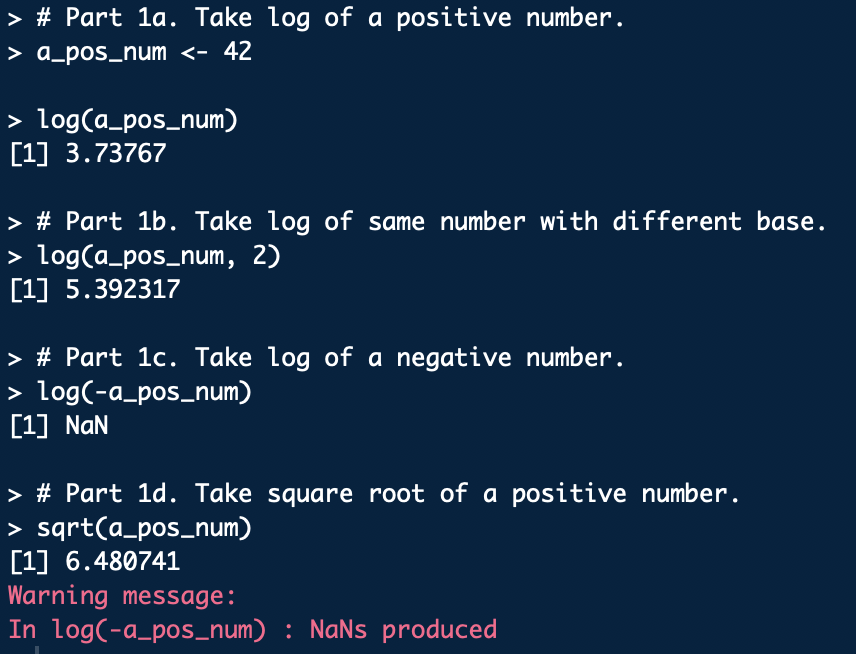
log(a\_pos\_num, base = 10)



Per the R documentation, the logarithmic function defaults to the natural logarithm whose base is Euler’s number, the mathematical constant . The above R code demonstrates that with no base declared (in Part 1a) and , where the base is explicitly declared as , produce the same result. Then, the common logarithm base 10 is demonstrated.

# **Part 1c**. Take log of a negative number.

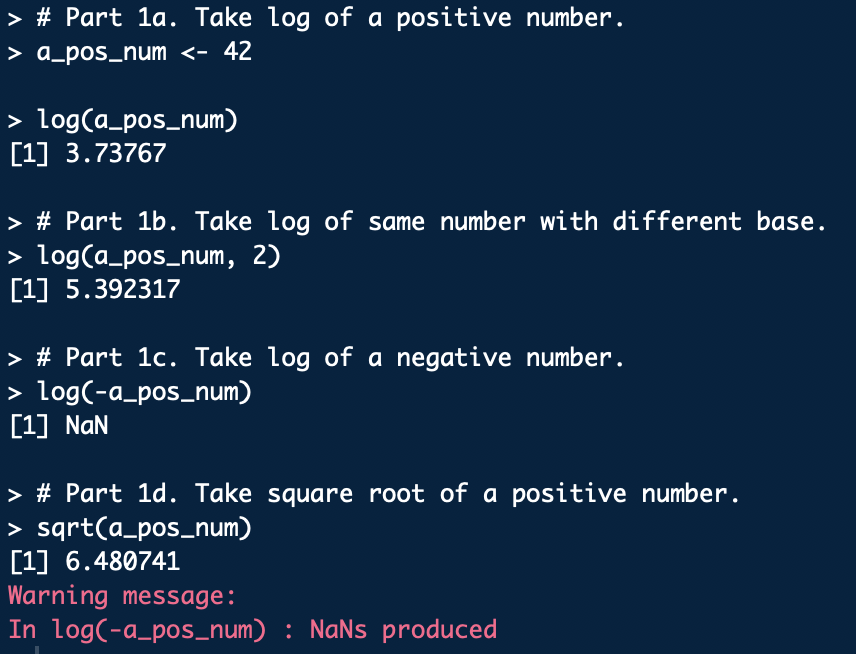
log(-a\_pos\_num)



The logarithmic function’s inverse, the exponential function , is defined for positive bases. Any positive raised to the power of any real number results in a positive . Therefore, the logarithmic function itself is undefined for . This is reflected by any attempts to take the logarithm of a negative number in R resulting in NaN, or not a number.

# **Part 1d**. Take square root of a positive number.

sqrt(a\_pos\_num)



**Question 2**: Random Number Generation

# **Part 2a**. Create vector of 15 standard normal random variables

# and calculate its mean and standard deviation.

norm\_rand\_vec\_1 <- rnorm(15)

mean(norm\_rand\_vec\_1)

sd(norm\_rand\_vec\_1)

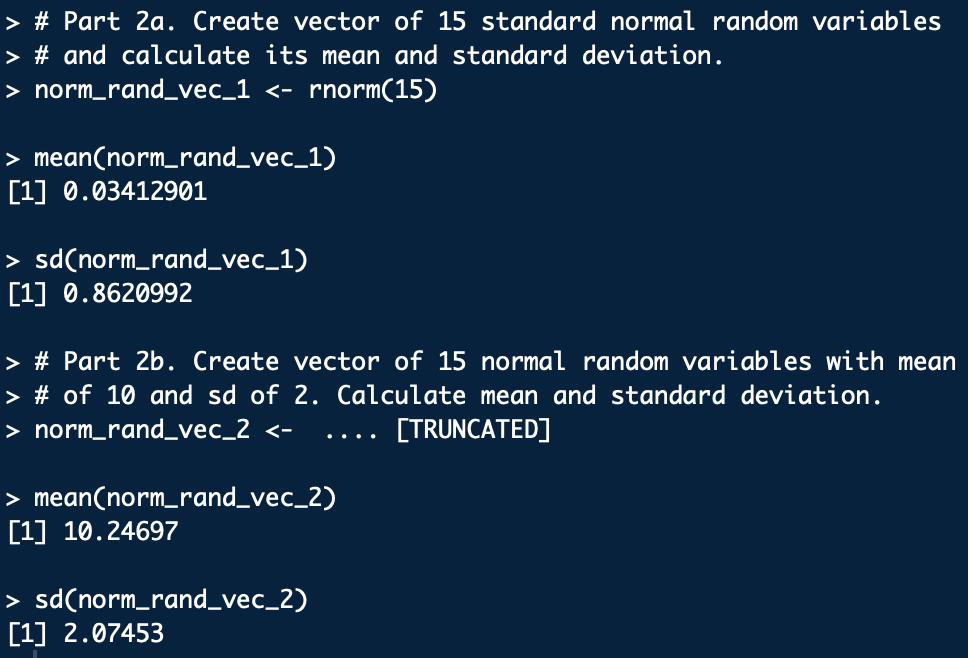
# **Part 2b**. Create vector of 15 normal random variables with mean

# of 10 and sd of 2. Calculate mean and standard deviation.

norm\_rand\_vec\_2 <- rnorm(15, mean = 10, sd = 2)

mean(norm\_rand\_vec\_2)

sd(norm\_rand\_vec\_2)



**Part 2c**:

The mean and standard deviation implicitly used in the first normal random call (0, 1) and explicitly specified in the second one (10, 2) are used to randomly generate the 15 variables. Because there is some inherent variability in what random variables are generated, calculating the means and standard deviations of those random variables will result in values that are close to (but unlikely to be exactly the same as) the original means and standard deviations. Additionally, 15 is a relatively small sample size. As the random variable sample size increases, the calculated mean and standard deviation is likely to better approximate the original mean and standard deviation.

**Question 3**: Vector Operations

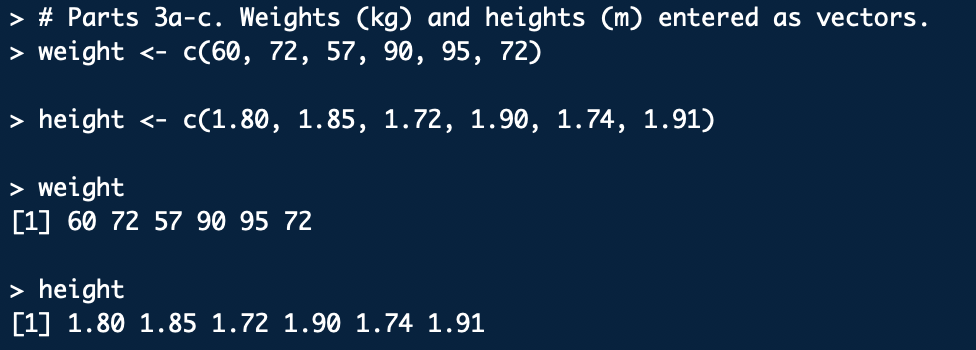
# **Parts 3a-c**. Weights (kg) and heights (m) entered as vectors.

weight <- c(60, 72, 57, 90, 95, 72)

height <- c(1.80, 1.85, 1.72, 1.90, 1.74, 1.91)

weight

height



# **Part 3d**. Scatterplot of weight vs height.

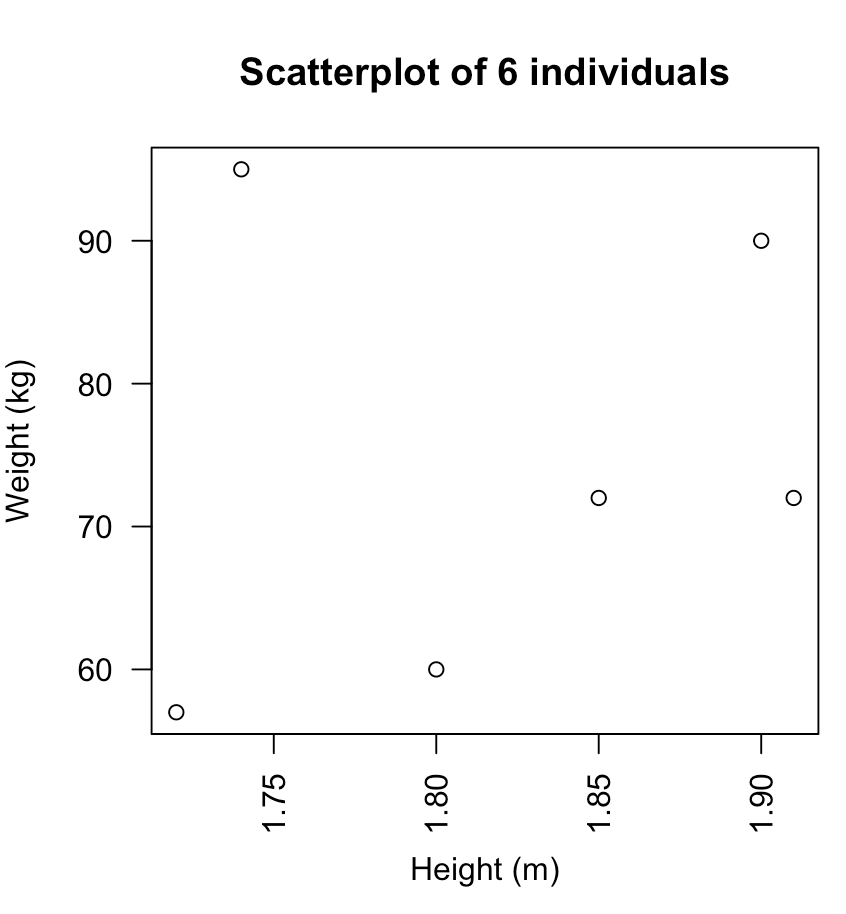
par(mar=c(5,4,4,4))

plot(height, weight,

main = 'Scatterplot of 6 individuals',

xlab = 'Height (m)',

ylab = 'Weight (kg)', las = 2)

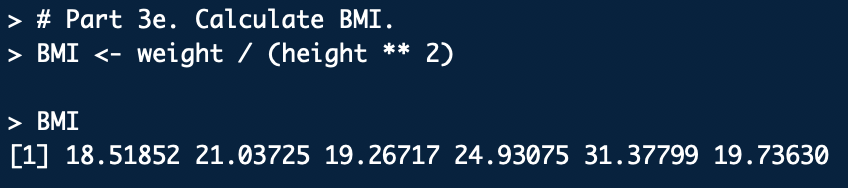


There seems to be a general positive correlation between height and weight in this sample. A taller person will tend to weigh more. This makes sense, as the extra body mass a taller person carries, like connective and muscular tissue, has its own weight (supported by Newton’s law, ). The relationship is a trend and not necessarily linear, as showed by the two individuals both weighing 72 kg having a 0.6 m height difference, or the high-BMI individual who is the second shortest yet heaviest of the group. The latter exemplifies that additional mass does not necessarily contribute to height.

# **Part 3e**. Calculate BMI.

BMI <- weight / (height \*\* 2)

BMI



# **Parts 3f-h**.

mean\_weight <- mean(weight)

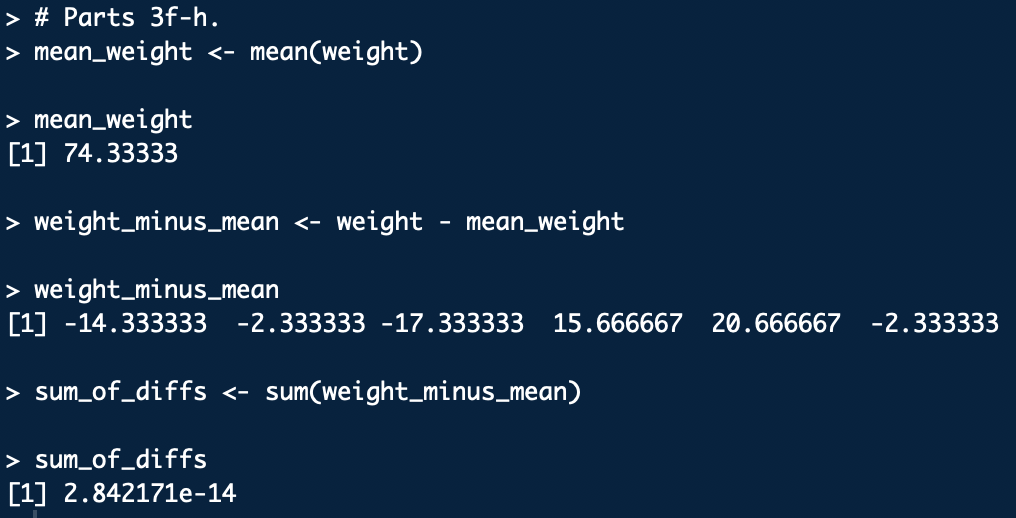
mean\_weight

weight\_minus\_mean <- weight - mean\_weight

weight\_minus\_mean

sum\_of\_diffs <- sum(weight\_minus\_mean)

sum\_of\_diffs



**Question 4**: Data Science Profile

# Create data science profile categories and rankings as vectors

# and combine them as columns of a data frame.

categories <- c('computer programming',

'math',

'statistics',

'machine learning',

'domain expertise (biology)',

'communication and presentation skills',

'data visualization')

rankings <- c(2, 3, 2, 1, 3, 3, 2)

Joseph <- data.frame(categories, rankings)

# Create bar graph of the data frame. To avoid long x-axis labels,

# the categories are placed in a legend.

par(mar=c(5,4,4,4))

Joseph\_bar <- barplot(Joseph$rankings,

main = 'Joseph Caguioa\'s Data Science Profile',

ylab = 'Rankings', ylim = c(0,5), las = 2,

col = rainbow(7),

legend = Joseph$categories,

args.legend = list(title = "Categories",

x = "topright",

cex = .65)

)

