Problem Set 1

Nilson Palma

2025-09-06

- Submission process: Please submit your assignment directly to Gradescope. You can do this by knitting your file and downloading the PDF to your computer. Then navigate to Gradescope.com or via the link on BCourses to submit your assignment.

Helpful hints:

- Render your file early and often to minimize rendering errors! If you copy and paste code from the slides, you are bound to get an error that is hard to diagnose. Typing out the code is the way to smooth rendering. We recommend rendering your file each time after you write a few sentences/add a new code chunk, so you can detect the source of the rendering error more easily. You can render by clicking the blue "Render" arrow in the top menu bar in R studio. This will save you and the teaching team time!
- To render as PDF, you will need to have LaTeX installed. If you are using the datahub **this should already be there and you don't need to install or update**. If you are not using datahub, there is an R package to handle this! If you don't have LaTeX, you can un-comment and run the following code (note: you only need to do this once, and don't need to load this like other packages).

```
# If you don't have LaTeX installed at all,
# uncomment and run the line below
#install.packages("tinytex")

# If you need to update LaTeX or run into rendering issues,
# uncomment and run the line below
#tinytex::reinstall_tinytex()
```

- Please make sure that your code does not run off the page of the rendered PDF. If it does, we can't see your work. To avoid this, have a look at your rendered PDF and ensure all the code fits in the file. When it doesn't, go back to your .qmd file and add spaces (new lines) using the return or enter key so that the code runs onto the next line.

Question 1

Un-comment and fix the following code to make it run. Note: You are only using numeric values in this question.

```
my_variable <- 2

my_other_variable <- 8

my_variable + my_other_variable
```

```
[1] 10
```

Question 2

Create four variables with the following data types:

1. character
2. numeric
3. integer
4. logical

```
word <- "Yes"
x <- 4
y <- as.integer(5)
Dog <- TRUE

class(word)
```

```
[1] "character"
```

```
class(x)
```

```
[1] "numeric"
```

```
class(y)
```

```
[1] "integer"
```

```
class(Dog)
```

```
[1] "logical"
```

Question 3

Using the variables x and y below, perform the following comparisons.

```
x <- 10; y <- 11

### is x equal to y?

x == y
```

```
[1] FALSE
```

```
### does x not equal y?

x != y
```

```
[1] TRUE
```

```
### is x greater than or equal to y?

x >= y
```

```
[1] FALSE
```

Question 4

R is a powerful calculator that can help us become more efficient epidemiologist.

Recall that an odds ratio is calculated by the following: (a / c)/(b / d) or (a * d)/(b * c).

Suppose a number of people became ill after exposure to cheesecake. Our two levels of exposure to cheesecake are (1) those who ate cheesecake and (2) those who did not eat cheesecake.

```
# run the following code to view our 2x2 table
# notice how we used one of R's base function called "matrix"
# we directly inputted our values in a list format c("","",...)
# added the argument ncol = 2 to split the list into two columns
# added the argument byrow = TRUE to first complete the rows then the columns

cheesecake_exposure <- matrix(c(15, 36, 18, 25), ncol = 2, byrow = TRUE)
# directly named the two columns
colnames(cheesecake_exposure) <- c("Cases", "Controls")
# directly named the two rows
rownames(cheesecake_exposure) <- c("Exposed", "Not Exposed")
# executed our variable to view the output
cheesecake_exposure
```

```
            Cases Controls
Exposed        15       36
Not Exposed    18       25
```

Calculate the odds ratio of becoming ill due to cheesecake. Save the odds ratio as an object in your environment.

```
ill_or <- (cheesecake_exposure[1] * cheesecake_exposure[4]) /
  (cheesecake_exposure[3] * cheesecake_exposure[2])
```

Question 5

Create two vectors with the following numeric values in the presented order. Then, add them together.

- Use the c() function for the first vector: 1, 2, 3, 4, 5
- Use the colon (":") operator for the second vector: 51, 52, 53, 54, 55

As a logic check, you should expect an output with one vector: 52, 54, 56, 58, 60

```
v1 <- c(1,2,3,4,5)
v2 <- 51:55
v1 + v2
```

```
[1] 52 54 56 58 60
```

Question 6

(a) Write a line of code that will pull up the documentation for the base R function called "round".
(b) What arguments does the function require?
(c) What is the default value for the second argument?

```
?round
```

b) x: a numeric vector. Or, for round and signif, a complex vector.

Digits: integer indicating the number of decimal places (round) or significant digits (signif) to be used.

c) Default value for digits is 0.

Question 7

Write a single line of code to divide 377 by 120 and round the result to three decimal places. Save this rounded result to a variable called "ptolemy_pi" in your environment. Test if ptolemy_pi is equal to R's built-in approximation of pi (stored by default as the object "pi").

```
pi
```

```
[1] 3.141593
```

```
ptolemy_pi <- round(377/120, 3)
pi == ptolemy_pi
```

```
[1] FALSE
```

Question 8

Use an if/else statement to print out "greater than 1" if the odds of becoming ill due to cheesecake (from question 4) is higher than 1.0. Print "not greater than 1" if the odds are not higher than 1.0.

```
if(ill_or > 1) {
  print("greater than 1")
} else {
  print("not greater than 1")
  }
```

```
[1] "not greater than 1"
```

You did it! Please knit to a pdf, download, and submit to Gradescope.