

Coding Principles Exercises

Juan Carlos Villaseñor-Derbez (JC)

Exercise 1: Object Classes and Assignment

Part A: Set up

1. Open R Studio and navigate to your EVR628 project
2. Set up a code snippet for a header

```
#####
# title
#####
#
# Your Name Here
# Your email here
# date
#
# Description
#
#####
```

3. Create a new R script and save it as `objects_and_classes.R`
4. Use your new code snippet to add the header

Part B: Create and Check Object Classes

In your R Script, create objects of different classes and check their types:

1. A character object called `my_name` with your name
2. A numeric object called `my_age` with your age (or any number, it doesn't have to be your age)
3. A logical object called `is_student` with TRUE or FALSE
4. Check the class of each object using `class()`

```
my_name <- "JC" # character object called `my_name` with your name
my_age <- 32 # numeric object called `my_age` with your age
is_student <- FALSE # logical object called `is_student` with TRUE or FALSE

# Check the class of each object using `class()`
class(my_name)
```

```
[1] "character"
class(my_age)

[1] "numeric"
class(is_student)

[1] "logical"
```

Part C: Object Coercion

Let's see what happens when you try converting objects. In the R script, write and then execute code that will:

1. Convert your age to character using `as.character(my_age)`
2. Convert TRUE to numeric using `as.numeric()`
3. Try to convert "hello" to numeric - what happens?
4. Advanced: Use `|>` to build a code pipeline to:
 - a. Take your age
 - b. Convert it to character
 - c. Get its class

```
# Note that I am overwriting the object. If I had only done `as.character(my_age)` ,
# then the coerced output would have been printed to the console.
my_age <- as.character(my_age) # Convert your age to `character`#
# Check that it worked
class(my_age)
```

```
[1] "character"
as.numeric(TRUE) # Convert `TRUE` to numeric using `as.numeric()``

[1] 1
as.numeric("hello") # Try to convert `"hello"` to `numeric` - what happens?
```

Warning: NAs introduced by coercion

```
[1] NA
# Build a pipeline
my_age <- 32 # Start with my_age as numeric again
# And now build the pipeline
my_age |>
  as.character() |>
  class()
```

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

Exercise 2: Vectors and Operations

Part A: Create and Manipulate Vectors

1. Clean your environment (use the broom icon)
2. Create a numeric vector called `length_m` with values: 6, 4.1, 2.8, 5.5, 3.9, 5.8
3. Create a character vector called `shark_species` with: Great White Shark, Lemon Shark, Bull Shark, Hammerhead Shark, Mako Shark, and Great White Shark (yes, white shark again)
4. How many *variables* do you have in your environment?
5. How many length *observations* do we have? Find the length of both vectors using `length()`
6. How many unique *species* do we have? (Hint, use `|>` to build a pipeline)
7. Calculate the mean length of all sharks using `mean()`
8. Find the maximum length using `max()`

```
# My code will be here
```

! Important

- When passing arguments to functions, use `=` not `<-`
- When creating objects, use `<-` not `=`

Part B: Vector Operations and Indexing

1. Extract the first 3 shark species using indexing with `[]` and save them to an object called `first_3`
2. Extract shark species where maximum length is greater than 4 meters
3. Create a logical vector showing which sharks have max length above 4 meters
4. Assuming the values in `length_m` and `sharks_species` are ordered so that they match each other, find the shark species that is the largest
5. Calculate the mean length for all great whites sharks

```
# My code will be here
```

Likely pause here

Exercise 3: Data Frames and Tibbles

Part A: Estimate the effect of a Marine Protected Area on Biomass

1. Install and load an updated version of `EVR628tools`:

```
remotes::install_github("jcvdav/EVR628tools")
```

2. Start a new script called `MPA_analysis`, add a comment outline and then load the `EVR628tools` and `tidyverse` packages.
3. Load and inspect the new `?data_MPA`
 - a. What are the dimensions of the data?
 - b. What are the column names? col
 - c. How many `unique()` sites are there?
 - d. How many `unique()` years?
4. Create four objects containing:
 - a. Mean biomass inside the MPA before it was protected
 - b. Mean biomass inside the MPA after it was protected
 - c. Mean biomass outside the MPA before the MPA was created
 - d. Mean biomass outside the MPA after the MPA was created

Hint: Use a combination of subsetting (`[]`), relational (`==`), and logical operators (`&`)

5. Then, calculate:
 - a. Change after vs before for the protected site
 - b. Change after vs before for the unprotected site
6. Finally, calculate the difference between these two values. This is called the *naive* difference-in-differences estimate. You are calculating the differences across treatments and across time. See Villasenor-Derbez et al. (2018) and Lynham and Villaseñor-Derbez (2024) for details.

```
# My code will be here
```

Exercise 4: Code Style and Documentation

Part A: Fix Code Style

Fix the style issues in this code:

```
mydataframe=data.frame(species=c("Great White","Tiger", "Bull"),
length=c(4.5,3.2, NA))
mean(mydataframe$length,na.rm=TRUE)

# My improved code will be here
```

Note: What's with than `na.rm = TRUE`?

Part B: Add Comments and Section Headers

1. Add meaningful comments to your R script

Lynham, John, and Juan Carlos Villaseñor-Derbez. 2024. “Evidence of Spillover Benefits from Large-Scale Marine Protected Areas to Purse Seine Fisheries.” *Science* 386 (6727): 1276–81.

Villasenor-Derbez, Juan Carlos, Caio Faro, Melaina Wright, Jael Martinez, Sean Fitzgerald, Stuart Fulton, Maria del Mar Mancha-Cisneros, et al. 2018. “A User-Friendly Tool to Evaluate the Effectiveness of No-Take Marine Reserves.” *PLoS One* 13 (1): e0191821.