Lecture 7: Object-Oriented Programming

1. Create a `summary` function for the class `pixel`. Verify the method dispatch mechanism both before and after implementing the `summary` function.

2. Compare and describe the difference between `t.test()` and `t.data.frame()`. What would happen if you run the following code, and why?

```
x <- structure(1:10, class = "test")
t(x)
```

3. Read the documentation for `UseMethod()` and explain why the following code returns the results that it does.

```
g <- function(x) {
  x <- 10
  y <- 10
  UseMethod("g")
}
g.default <- function(x) c(x = x, y = y)

x <- 1
y <- 1
g(y)
```

4. What do you expect this code to return? What does it actually return, and why?

```
generic2 <- function(x) UseMethod("generic2")
generic2.a1 <- function(x) "a1"
generic2.a2 <- function(x) "a2"
generic2.b <- function(x) {
  class(x) <- "a1"
  NextMethod()
}

generic2(structure(list(), class = c("b", "a2")))
```

Lecture 8: Web Scraping

1. Use the web tool CSS Diner to familiarize yourself with CSS Selectors.

2. Follow the web scraping workflow available here, which utilizes the `SelectorGadget`. After completing the workflow, propose an alternative solution using CSS selectors. You will likely need to use your browser's developer tools to explore the DOM structure.

3. Repeat the exercise from step 2, but this time using `RSelenium` or `chromote` to automate the process.

4. Extract the necessary information from the World Bank data example using regular expressions to capture the specific content.

Lecture 9: Shiny Applications

1. Extend the Shiny app presented in class by adding a second tab containing a data table that reports summary statistics of the Old Faithful Geyser dataset. You can refer to the UI in the Buffon's needle example, which can be found here.

2. Draw the new reactive graph using the app. Can you find ways to make the app more efficient? Consider improvements such as optimizing reactive expressions or reducing the computational burden.

3. Experiment with different themes in Shiny using the following code. Observe how the output in your console changes as you modify the theme:

```
library(shiny)
library(bslib)
thematic::thematic_shiny(font = "auto")

ui <- fluidPage(
  theme = bs_theme(),
  ...
)

server <- function(input, output) {
  bs_themer()
  ...
}

shinyApp(ui, server)
```