Introduction

The objective of this class is to provide students with some tools and ideas to design their bioinformatic tutorials. In this tutorial, students will have an overview of the tools implemented in the R package [*learnr*](https://rstudio.github.io/learnr/index.html), which was developed to produce interactive tutorials with R. Although developed in R, the interactive tutorial is usually executed in a web browser (but it could also be entirely conducted within RStudio).

The interactive tutorial presented here is subdivided into 4 topics:

1. **Exercise**: In this topic, the aim of the exercise is presented together with a pseudocode (with associated R functions) outlying steps to design and implement an R code. Students are asked to develop their R codes and execute them in the code chunk. You can type your code directly in the window/console and execute it using the R Code button.
2. **Questions**: In this topic, students are asked to answer two questions related to the outputs produced by their code.
3. **Solution**: In this topic, a solution proposed by the instructor is available in a code chunk. Line of code are also commented to allow better understanding of the approach and implementation.
4. **YouTube tutorial**: Here, a short YouTube video presents the procedure to launch a *learn* interactive tutorial and briefly presents the exercise. The main objective of this video is to show students that it is quite easy to integrate a video into their bioinformatic tutorials.

Finally, the instructor wants to stress that students are not obliged to design their bioinformatic tutorials using *learnr*. You can also use the R Markdown language/syntax and output tutorials in HTML or PDF formats (more on this subject in Chapter 1).

Exercise: Class meeting days

Aim of exercise

The aim of this exercise is to develop an R code to reproduce part of the timetable associated to the EEB603 course. Here, students are tasked to work on an R code, which will ultimately automatically produce a data frame with class meeting dates and their associated weekdays and week IDs. Although some R packages are specialized in handling date-time data (e.g. [*lubridate*](https://lubridate.tidyverse.org)), the instructor would like students to design code relying solely on basic R functions.

To design your code here are some key information:

* **Classes Begin:** August 24, 2020 (2020-08-24)
* **Classes End:** December 11, 2020 (2020-12-11)
* **Thanksgiving Holiday (no classes):** November 23-29, 2020
* **EEB603 class weekdays:** Tuesday and Thursday

More information about the Fall semester at BSU can be found [here](https://www.boisestate.edu/registrar/boise-state-academic-calendars/fall-2020/).

Pseudocode, associated functions & commenting

Usually, before delving into writing code, we are starting by producing a pseudocode, which breaks down the necessary steps involved in achieving our objective(s) as well as their related tasks. Once we have completed this latter task, we are adding information associated to best available functions to be applied to achieve each step/task. In cases where no existing functions exist for our task, users will have to design user-defined function(s) (this topic will be covered later in this course). Finally, it is always important to think about input and output formats and required arguments/objects to complete our tasks. We will be studying reproducible code in chapter 4.

For this exercise, here is a proposed pseudocode (with associated R basic functions):

1. Build a data frame with all dates from beginning to end of Fall semester. This can be done using seq.Date() and converting output into a data frame using data.frame(). Please use following date formatting: YYY-MM-DD (e.g. 2020-08-24).
2. Find weekdays associated to dates. This can be done using weekdays().
3. Classes only take place on Tuesday and Thursday. So, extract dates matching class weekdays. This can be done using subset().
4. Add week ID (from 1 to 16) to object from step 3.

* Start by creating a vector of week IDs using seq() and rep() functions.
* Add object that you just created in front (= first column) of object from step 3.

1. Exclude dates falling during holiday break.

* Start by establishing a vector with dates for that week. See step 1 for more details.
* Find and exclude holiday days. This can be done by using which() and indexing from object from step 4.

**Note:** If you want to learn more about the R functions described here, type the following syntax in the R console: e.g. ?seq(). This action will pull up the documentation associated with the function.

Finally, it is key to comment your R code using # to enable users understanding your thought process and assess code content and its requirements.

Your R code

Here’s an empty code chunk provided for entering your answer.

Use the pseudocode provided above to write the R code required to solve this exercise. The instructor has provided his solution in the next tab. You can look at it for inspiration, but remember that there are many ways to get to the same solution.

**Note:** You can execute your code by pressing the Run Code button. You also have to call your objects to see their content.

#Start typing your code here!

###----  
#SOLUTION  
###----  
  
#Please only use the solution to help you design your code!   
  
###~~~  
# Build a data frame with all dates from beginning to end of Fall semester  
###~~~  
#From <- "2020/08/24"  
#To <- "2020/12/11"  
dates <- data.frame(seq.Date(from=as.Date("2020/08/24"), to=as.Date("2020/12/11"), by="day"))  
#Assign a name to the col (allows using $)  
colnames(dates) <- "Date"  
  
###~~~  
# Find weekdays associated to dates  
###~~~  
#Add a col to dataframe with weekday  
dates$Weekday <- weekdays(dates$Date)  
  
###~~~  
# Classes only take place on Tuesday and Thursday. So, extract dates matching class weekdays  
###~~~  
#Subset dates based on selected weedays  
datesclass <- subset(dates, dates$Weekday %in% c("Tuesday", "Thursday"))  
  
###~~~  
# Add week ID (from 1 to 16) to object from step 4  
###~~~  
#Create weekID vector and add it in front of dataframe  
datesclass <- data.frame(WeekID = sort(rep(seq(from=1, to=16, by=1), 2)), datesclass)  
  
###~~~  
# Exclude dates falling during holiday break  
###~~~  
#Establish vector of dates related to holidays  
#From <- "2020/11/23"  
#To <- "2020/11/29"  
holiday <- seq.Date(from=as.Date("2020/11/23"), to=as.Date("2020/11/29"), by="day")  
  
#Find and exclude holidays  
datesclass <- datesclass[-which(datesclass$Date %in% holiday),]  
  
#Call object  
datesclass

Questions

Use your R code to answer the following questions. Note that you might have to use other basic R functions to obtain the answers.

Solution: Class meeting days

The solution proposed by the instructor is available in the code chunk below. Line of code are also commented to allow better understanding of the approach and implementation (more on this subject in Chapter 4). When you execute the code, it will produce all the output tables for you to inspect. As shown in the Exercise: Class meeting days topic, you can directly embed the solution to your exercise in the code chunk and it will be accessible to users by pressing on the Solution button.

###~~~  
# Build a data frame with all dates from beginning to end of Fall semester  
###~~~  
#From <- "2020/08/24"  
#To <- "2020/12/11"  
dates <- data.frame(seq.Date(from=as.Date("2020/08/24"), to=as.Date("2020/12/11"), by="day"))  
#Assign a name to the col (allows using $)  
colnames(dates) <- "Date"  
  
#Call object  
dates  
  
###~~~  
# Find weekdays associated to dates  
###~~~  
#Add a col to dataframe with weekday  
dates$Weekday <- weekdays(dates$Date)  
  
#Call object  
dates  
  
###~~~  
# Classes only take place on Tuesday and Thursday. So, extract dates matching class weekdays  
###~~~  
#Subset dates based on selected weedays  
datesclass <- subset(dates, dates$Weekday %in% c("Tuesday", "Thursday"))  
  
#Call object  
datesclass  
  
###~~~  
# Add week ID (from 1 to 16) to object from step 4  
###~~~  
#Create weekID vector and add it in front of dataframe  
datesclass <- data.frame(WeekID = sort(rep(seq(from=1, to=16, by=1), 2)), datesclass)  
  
#Call object  
datesclass  
  
###~~~  
# Exclude dates falling during holiday break  
###~~~  
#Establish vector of dates related to holidays  
#From <- "2020/11/23"  
#To <- "2020/11/29"  
holiday <- seq.Date(from=as.Date("2020/11/23"), to=as.Date("2020/11/29"), by="day")  
  
#Find and exclude holidays  
datesclass <- datesclass[-which(datesclass$Date %in% holiday),]  
  
#Call object  
datesclass

YouTube tutorial

You can include YouTube and Vimeo videos into your interactive tutorials. Here, the instructor is including the video to launch *learnr* tutorials as an example.



Introduction to interactive bioinformatic tutorials