Lesson 01
Introduction to R
Variables & Data Types

Outline

- Getting R
  - Downloading and installing R
- The R Environment
  - Working Command Line Interface and RStudio
- Basics of R
  - Math, variables, data types, vectors, …
- R Packages
  - Locating, installing and loading packages
- Advanced Data Structures
  - Data frames, lists, matrices and arrays

Getting R

- Downloading R
  - https://cran.r-project.org/
  - Download R for Windows or (Mac) OS X
  - Choose base distribution / install R for the first time
  - Download R executable (current version 4.0.0)
- Installing R
  - Assumes Windows installation
  - If you can, install R on C drive in directory without spaces
    - For example create R directory directly on the C drive and install there
  - Run the executable and follow instructions
    - Example path would then be C:\R\R-4.0.0
    - If you are sure on 32 vs 64-bit, uncheck the one you don’t need
  - Recommend to pin R to either Start menu or Taskbar

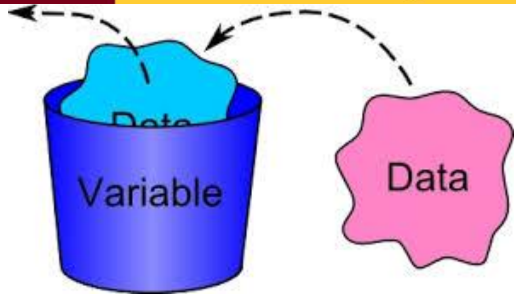
The R Environment

- Command Line Interface
  - Start R to display R console
  - `print("Welcome to R!")`
- RStudio
  - https://www.rstudio.com/products/rstudio/download/
  - RStudio Desktop Free (Open Source License)
  - Operating system (Windows, macOS)
  - Run the executable and follow instructions
    - Don't worry about the path to RStudio, just accept all defaults
  - Recommend to pin RStudio to either Start menu or Taskbar

The R Environment

- Download and unzip class files
  - `IDSC_4110_Files` folder is the class folder
- RStudio Projects
  - Tools -> Options -> Leave most defaults
  - Do not restore or save .Rdata
    - General -> Workspace -> Uncheck box and specify Never
  - File -> Open Project
    - Browse for `IDSC_4110_Files/01_Intro_R/Lectures/Intro_R` folder
    - Open the already created `Intro_R.Rproj` file
  - Use the console to test individual commands
    - Use Ctrl+l (lowercase L) to clear the console
  - Create R code in the file and run appropriate portions
  - Switch to already created **`Intro1_R_Basics.r`** R script file

Variables and Assignments



- Variable names
  - Any combination of alphanumeric characters including
    - Underscores (_) typically used for connecting different words like `mth_pmt` in many programming languages
    - Periods (.) which are used for the same purpose but are not typically used for different purpose especially in object-oriented programming languages
- The assignment operator: <-
  - Example: assign the value of 0.05 to the rate variable

    ```
    rate <- 0.05
    ```
  - The traditional = operator is typically not used

R Statements

- R comments begin with #
- Algebraic operations follow math order of precedence

```
# Using math to calculate monthly payment
15000 * (0.05/12) / (1 - (1 + 0.05/12)^(-5*12))
```

- Executing R statements
  - Type statement at the prompt and hit Enter
  - Highlight the statement(s) in the code editor and Run

Numeric Data Types

- Most common R data type is `numeric`
  - Typically known as `decimal`, `double` or `float` data types

```
rate <- 0.05
```

`is.numeric(rate)` will return `TRUE`

`class(rate)` will return `“numeric”`

```
princ <- 15000
```

`typeof(princ)` will return `“double”`

- Append `L` to make a number of `integer` data type

```
term <- 5L  # Less frequently used
```

`is.integer(term)` will return `TRUE`

`class(term)` or `typeof(term)` will both return `“integer”`

- Create a formula to calculate the monthly payment

```
mth_pmt <- princ * (rate/12) / (1 - (1 + rate/12)^(-term*12))
```

Character Data Type

- `character` data type for simple text data
  - Enclosed in double-quotes

    `pmt_msg <- "Your monthly payment is:“`

    `class(pmt_msg)` **returns** `“character”`
- Character or string data
  - Case sensitive
  - Character functions

    `nchar(pmt_msg)` **returns** `24`

    `substr(pmt_msg, 14, 20)` **returns** `payment`

    `paste(pmt_msg, format(round(mth_pmt,2)))`

Dates Data Types

- `Date` **data type handles dates only**

  `fst_pmt_dt <- as.Date("2030-05-10")`

  `class(fst_pmt_dt)` **returns** `“Date”`

  – Number of days since January 1, 1970

  `as.numeric(fst_pmt_dt)` **returns** `22044`

- `POSIXct` **data type handles dates and times**

  `act_pmt_dtm <- as.POSIXct("2030-05-02 10:34:52")`

  `class(act_pmt_dtm)` **returns** `“POSIXct”`

  – Number of seconds since January 1, 1970

  `as.numeric(act_pmt_dtm)` **returns** `1903966492`

Logical Data Type

True ☑ False ☐

- Logical data type assumes two values
  - Typically known as `Boolean` data type

```
pmt_made <- TRUE
```

`is.logical(pmt_made)` returns `TRUE`

- Relational operators: ==, !=, >, <, >=, <=

```
act_pmt <- 250
```

`act_pmt != mth_pmt` returns `TRUE`

- Logical operators: &&, ||, !, …

```
act_pmt <- 300
fst_pmt_rec <- as.Date("2030-05-05")
(act_pmt >= mth_pmt) && (fst_pmt_rec <= fst_pmt_dt)
```

R Packages – Installing

- This is where the power of R comes from
  - There are thousands of packages out there
- A package is a library of prewritten code designed to accomplish some task
  - We will use several established packages later in the class
  - Initial demonstration will use `optiRum` financial package
  - **NOTE: YOU DO NOT HAVE TO INSTALL THIS PACKAGE!**
  - One-time limited use is not worth actually doing it
  - Enough to just view the process of working with R packages
- Use R Studio to install and uninstall packages
  - Use **Packages** tab in the bottom-right pane
  - Click the **Install** button and type **optiRum**
    - Make sure Install dependencies is checked

R Packages – Loading

- Loading R packages
  - Check the name of the package (like **`optiRum`**) check box
  - This will execute the `library` command
    - `require` is an alternative
  - Usually placed at the top of the R file
- Using R packages
  - Find and briefly review the documentation first
    - https://cran.r-project.org/web/packages/optiRum/optiRum.pdf
  - Locate the functionality you need for the task at hand
    - Find PMT function documentation in the PDF

```
pmt_fnc <- -PMT(rate/12, term*12, princ)
```

R Packages – Unloading

- Unloading R packages
  - Uncheck the package name (like **`optiRum`**) check box
  - This will execute the `detach` command
- Uninstalling R packages
  - Click the white **x** inside the gray circle to the right of the package
  - This will execute the `remove.packages` command
  - The package no longer appears on the list
    - This does not uninstall dependencies
    - Dependent packages may be used in other packages

Summary

- Downloaded and installed R & RStudio
- Got familiar with RStudio environment
  - Customized few aspects of RStudio
- Opened your first RStudio project
- Worked with simple R variables
  - Assignment with <- operator
- Described basic R data types
  - `numeric`, `character`, `Date`, `POSIXct` and `logical`
- Introduced R packages