# Worksheet 1 : Introduction to R

## Contents



---

### Today's Agenda

- Using RStudio and RMarkdown
- R as a calculator
- Loading data into R
- Plotting data

---

During this worksheet you will explore R and RStudio, which you'll be using throughout the course both to learn the statistical concepts discussed in the course and to analyze real data and come to informed conclusions. To clarify: **R** is the name of the programming language itself and **RStudio** is a convenient interface.

A programming language is just the language for telling the computer what to do. The most frustrating thing about programming is that the computer will do exactly what you tell it to, so one tiny typo can make everything you are working on fail. But, the fact that computers do exactly what you tell them to is also what makes them great! The best thing you can do to avoid frustration is to test that the computer is doing what you're expecting often.

### RStudio Interface

Launch RStudio either from jupyter.davidson.edu or locally from your laptop. Once it has booted up you should see an interface with a panel on the left and two panels on the right.

The panel on the left is the most important one; this is where R is working. The > indicates the line you are on with R ready to run. Test that R is working by running the command 3+5. R can be used as a calculator. Now try to run 3+*5 and notice that R tells you that it doesn't understand what you're saying to it. Errors often contain information about how to fix the error. This panel is called the console, we will explore more of what we can do here later.

If we execute code line by line, as we did in the last example, our work is not being saved. You will often want to re-run large chunks of your code and it would be a large waste of time to retype all of your commands every time you want to re-run code. Therefore, we should always be running our code in an R Markdown file. In the very top left of R Studio find the icon that is a sheet of paper with a green plus symbol on it and select a new R Markdown file and give it the name Testing. Once this has been opened you should have a new panel in the top left and the console you were using before is in the bottom left. Most the work we do from now on will be in an R Markdown file, not in the console, let's see how it works.

Let's delete everything below the part that says ## R Markdown (you can read this later by opening a new R Markdown file the same way as before). Now in the top right of the R Markdown panel is a green C with a

plus over it. Click that and select R. This creates a little code area that will run R code. Try typing 3+5 inbetween the single quotes and the press the play button.

```
3+5
```

```
## [1] 8
```

If you look down to your console you will see that the code ran there. You can press play again to run the code again without having to retype anything. This is a great tool to avoid errors in your code because you can see how each piece is working individually. Doing your work in an R Markdown file is a best practice that you should adopt and will be required for the assignment submissions.

The panel in the upper right has information about data you have stored in your computer. Try to run the following command a<-3+5 by creating another R chunk. On the next line (still inside the same chunk type) a . You have told your computer to remember the calculation $3 + 5$ and store the sum in the variable a . Any time you want to access the sum again you can just type the name you gave it. Also notice that the value of the variable a is shown in the upper right panel.

In the lower right panel has a lot of useful information. You can see the file manager where you can load data into R and download updated files if you are using jupyter. This is also where any plots you make will be displayed. Another useful feature is that the current packages you have loaded will also be displayed here.

### R Packages

R is an open-source programming language, meaning that users can contribute packages that make our lives easier, and we can use them for free. Here are some common R packages we will use in this course:

- The suite of **tidyverse** packages: for data wrangling and data visualization
- **openintro**: for data and custom functions with the OpenIntro resources
- **gapminder**: for easy access to an excerpt of the Gapminder data on life expectancy, GDP per capita, and population by country

If these packages are not already available in your R environment, install them by typing the following lines of code into the console of your RStudio session, pressing the enter/return key after each one. Note that you can check to see which packages (and which versions) are installed by inspecting the Packages tab in the lower right panel of RStudio.

```
install.packages("tidyverse")
install.packages("openintro")
install.packages("gapminder")
```

You may need to select a server from which to download; any of them will work. Next, you need to load these packages in your working environment. We do this with the library function. Run the following lines in your console.

```
library(tidyverse)
library(openintro)
library(gapminder)
```

You only need to *install* packages once, but you need to *load* them each time you relaunch RStudio.

The Tidyverse packages share common philosophies and are designed to work together. You can find more about the packages in the tidyverse at https://www.tidyverse.org.

## Data Basics

We can obtain data a number of ways. One way is to create data explicitly, we will learn more about that later. Another way is to load data from a library. Since we have already loaded the gapminder package, let's get some data from it.

```
view(gapminder)
```

```
head(gapminder)
```

```
## # A tibble: 6 x 6
##   country     continent  year lifeExp      pop gdpPercap
##   <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
## 1 Afghanistan Asia       1952    28.8  8425333      779.
## 2 Afghanistan Asia       1957    30.3  9240934      821.
## 3 Afghanistan Asia       1962    32.0 10267083      853.
## 4 Afghanistan Asia       1967    34.0 11537966      836.
## 5 Afghanistan Asia       1972    36.1 13079460      740.
## 6 Afghanistan Asia       1977    38.4 14880372      786.
```

Now let's load the data from last class. If you are using jupyter, the file should already be in your file manager. If so, right click to import the data. Otherwise, you will need to download it from Moodle and run the following:

```
questionnaire <- read.csv(file = "/Users/matthewbachmann/Documents/GitHub/MAT104-Fall23/Week_1/questionn
questionnaire
```

```
##        name student_id num_grapes car_color num_skittles dream num_pets dino
## 1   Matthew          1         12       red          734    13        0   no
## 2   Brianna          0          8     black          750     0        2   no
## 3  Jennifer          3          8      blue          412    10        2  yes
## 4   Jillian          9          5       red          264    10        2   no
## 5       Liv          0          6     white           60    60        2   no
## 6    Carter          4          4      grey         1104   420        2   no
## 7    Serena          2          5       red            5    10        1   no
## 8     Vinay          5         12     white          133    10        2   no
## 9     Mario          0          8       red          215   360        2  yes
## 10  Meghana          1          8       red          227   120        0   no
## 11      Chi          4          4       red           11    30        2  yes
## 12     Luke          8          6       red          343    30        2   no
## 13     Maya          8          5     black         2500    60        1   no
## 14   Annika          6          8 dark blue           52    60        4   no
## 15     Skye          9          8       red         2460    75        1   no
## 16      Ann          6          7      blue          567     2        2  yes
## 17   Brooke          8          6       red          156    20        2  yes
## 18      Ava          1          6       red          300    60        2  no
## 19    Erica          7          7     green          231    20        2  yes
## 20    Nahot         NA          8       red          437   180        0   no
##        dino_type pop.tart
## 1            N/A  neither
## 2            N/A  neither
## 3   brontosaurus  neither
## 4            N/A  ravioli
## 5            N/A  neither
## 6            N/A  ravioli
## 7            N/A sandwich
## 8            N/A  neither
## 9  brachiosaurus  ravioli
## 10           N/A  neither
## 11 pink dinosaur sandwich
## 12           N/A  ravioli
```

```
## 13           N/A  ravioli
## 14           N/A  neither
## 15           N/A  ravioli
## 16  velociraptor  neither
## 17    tiny t-rex  neither
## 18           N/A  neither
## 19     long neck sandwich
## 20           N/A  neither
```

To view all the responses for only one question we use $

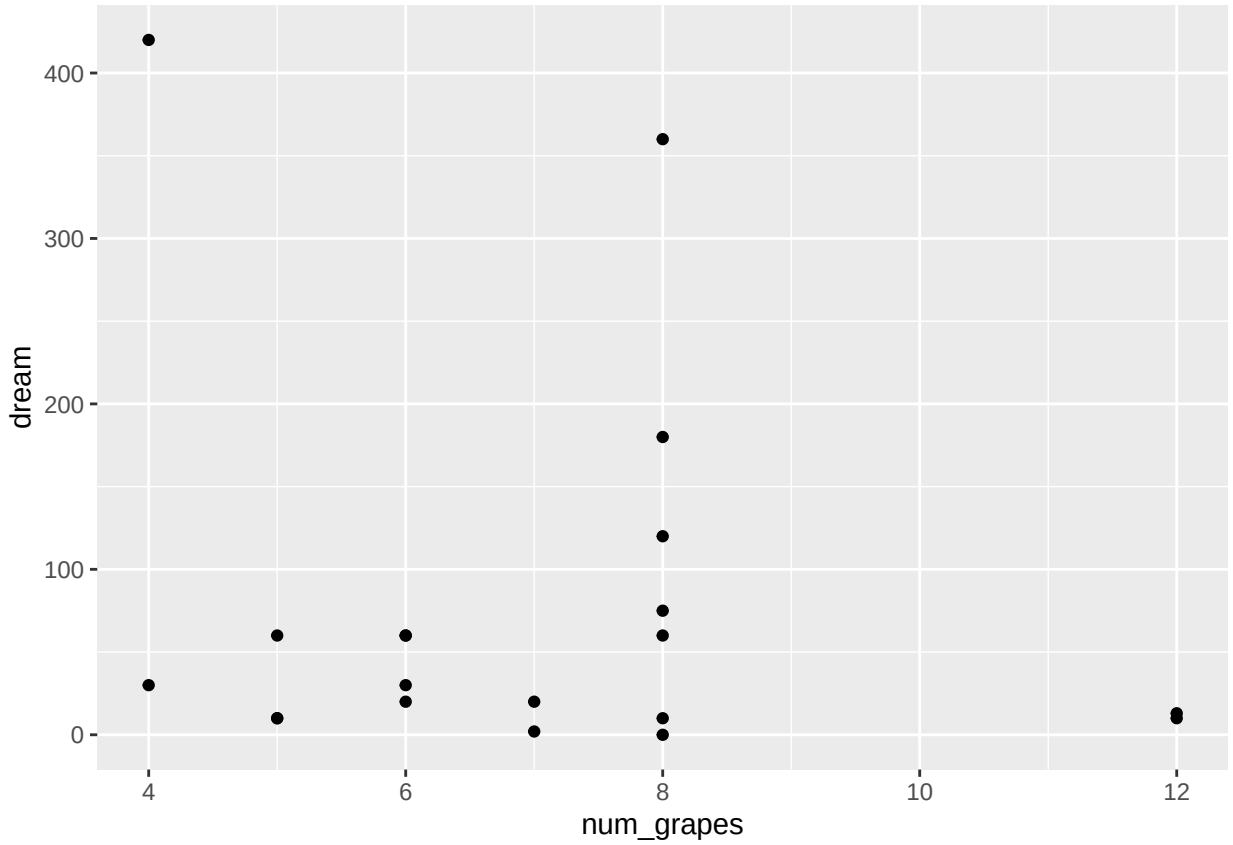
```
questionnaire$pop.tart
```

```
##  [1] "neither"  "neither"  "neither"  "ravioli"  "neither"  "ravioli"
##  [7] "sandwich" "neither"  "ravioli"  "neither"  "sandwich" "ravioli"
## [13] "ravioli"  "neither"  "ravioli"  "neither"  "neither"  "neither"
## [19] "sandwich" "neither"
```

### Plotting data

Now we will plot some data from the file:

```
ggplot(data = questionnaire, mapping = aes(x = num_grapes, y = dream)) + geom_point()
```

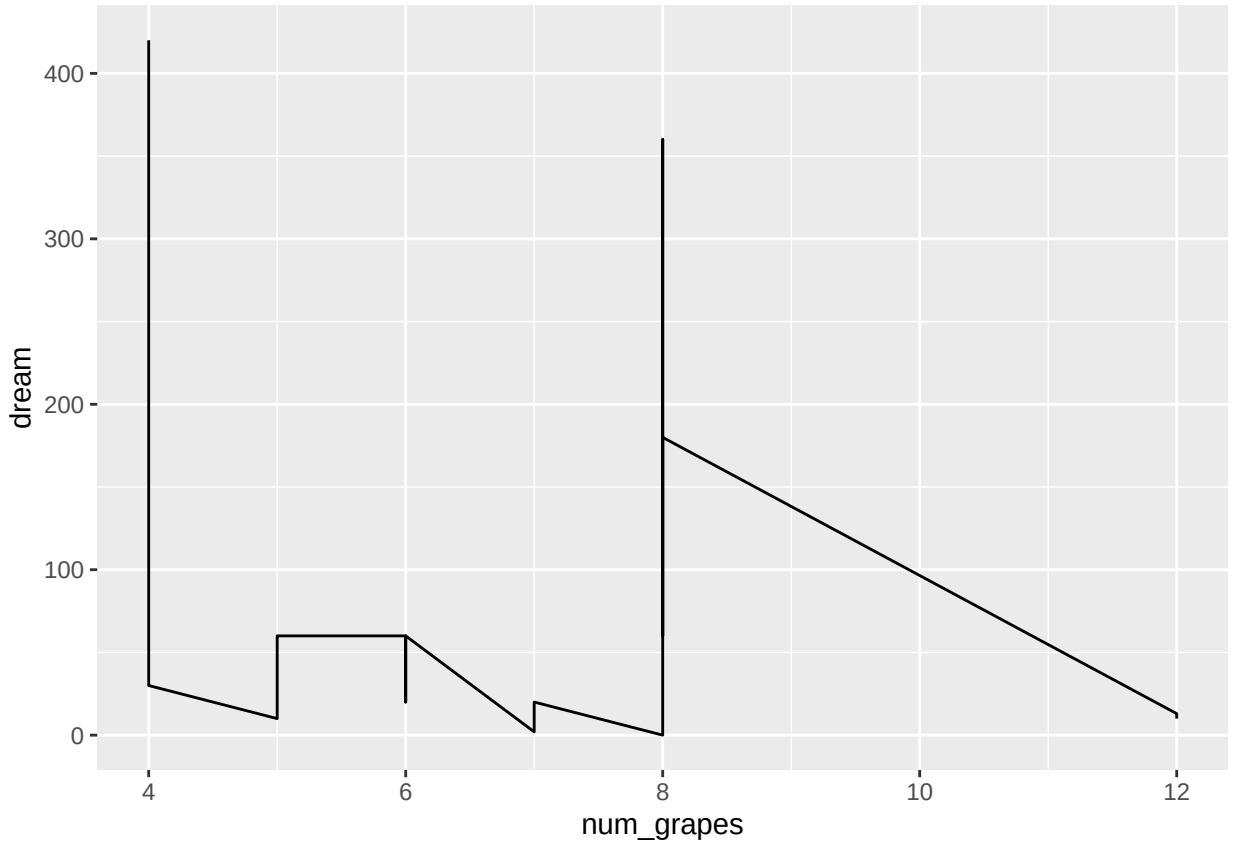



We use the ggplot() function to build plots. You should see the plot appear under the Plots tab of the lower right panel of RStudio. The code ggplot() follows a pattern:

-The first argument is always the dataset -Next, you provide the variables from the dataset to be assigned to aesthetic elements of the plot, e.g. the x and the y axes. -Finally, you use another layer, separated by a + to

specify the geometric object for the plot. Since we want points, we use geom_point(). For instance, if you wanted to visualize the above plot using a line graph, you would replace geom_point() with geom_line().

```
ggplot(data = questionnaire, mapping = aes(x = num_grapes, y = dream)) + geom_line()
```




You might wonder how you are supposed to know the syntax for the ggplot function. Thankfully, R documents all of its functions extensively. To learn what a function does and its arguments that are available to you, just type in a question mark followed by the name of the function that you're interested in. Try the following in your console:

```
?ggplot
```

---

## Class Activity/ Homework

---

In this activity you will practice the skills we learned today with a different dataset.

1. Let's begin by using R as a calculator: calculate 7 to the fifth power

```
##Type your code here##
```

2. Now install *and* load the package palmerpenguins

```
##Type your code here##
```

3. Packages have documentation just as functions did. Use the command that displays the documentation for the palmerpenguin package.

```
##Type your code here##
```

4. The package comes with a data table called penguins. The package tidyverse comes with a function glimpse() that lets you take a peek at the data. Add code that will only display the column of data corresponding to bill_depth_mm.

```
glimpse(penguins)
##Add code here##
```

5. Now plot the data where the x-axis is bill_length_mm and the y-axis is flipper_length_mm

```
##Type your code here##
```

When you have finished, Knit your project as an html document and submit the html document in Moodle with the following file name: "yourlastname_WS1.html"