ECON20003 QM2

Tutorial 2 Semester 1, 2022

Chin Quek

Department of Economics

March 2022

Objectives

- Mathematical operators and functions
- Importing Excel data into RStudio Environment
- Graphical Descriptive Techniques

Mathematical operators and functions

Review Questions

- What is the difference between = and ==?
- What does != mean?

Suppose I have three variables: `x=2`, `y=16` and `z=8`. What are the following outputs?

- `x+y`
- `z^2`
- `x<z`
- `y==16`
- `0&y`

```r
x <- 2
y <- 16
z <- 8

x+y
```

```
## [1] 18
```

```r
z^2
```

```
## [1] 64
```

```r
x<z
```

```
## [1] TRUE
```

```r
y==16
```

```
## [1] TRUE
```

```r
0&y
```

```
## [1] FALSE
```

Review Questions

How would we use R to compute $\ln(\sqrt{100})$?

- Can we use square brackets instead of round brackets in R?

Review Questions

How would we use R to compute $\ln(\sqrt{100})$?

- Can we use square brackets instead of round brackets in R?

```
log(sqrt(100))
```

```
## [1] 2.302585
```

Exercise 1

With numbers, *RStudio* can be used like any ordinary calculator. To demonstrate this, launch *RStudio*, type `4*16` in the *Console* and hit *Enter*. IF you use the R Script to type `4*16`, then you will need to either click Run or hit *CTRL* + *Enter*.

```
4*16
```

```
## [1] 64
```

then try `(4*16/sqrt(256))^3` and hit enter again

```
(4*16/sqrt(256))^3
```

```
## [1] 64
```

last try `log(4*16/sqrt(256))^3`

```
log(4*16/sqrt(256))^3
```

```
## [1] 2.664197
```

Exercise 2

Retrieve the *t1e2* data set that you created and saved in Exercise 2 of Tutorial 1 and generate the following three new variables:

- $X = Height + Weight$
- $Y = Age^2$
- $Z = 3Age - 5$

Exercise 2

Retrieve the *t1e2* data set that you created and saved in Exercise 2 of Tutorial 1 and generate the following three new variables:

- $X = Height + Weight$
- $Y = Age^2$
- $Z = 3Age - 5$

Is there any issue with the following lines of code?

```
X <- height+Weight
Y <- Age^2
Z <- 3Age-5
```

Recall that R is case sensitive!

```
X <- Height+Weight
Y <- Age^2
Z <- 3*Age-5
```

Once the variables are created correctly, in the Environment pane, you should see `X`, `Y` and `Z` under values (each being numeric).

```
rm(list=ls())
```

Importing Data

- How do we import data from a MS Excel file?

- What are the two different ways of calling a variable from within a dataset?

For example, suppose we have a dataset named `t2e3` in the *Global Environment*. Within `t2e3`, there are two numeric variables `Height` and `Weight`.

```
## [1] "Height" "Weight"
```

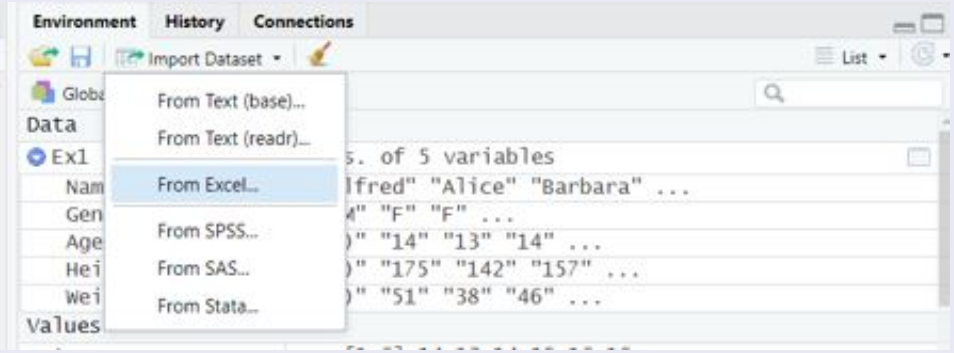
- How do we view the values of `Weight`?

Exercise 3

If you want to import data from a MS Excel file, the easiest way to do this would be to first install the `readxl` packages.

How do we install packages?

Click on the *Import Dataset* button on the *Environment* tab and then select the *From Excel...* option.



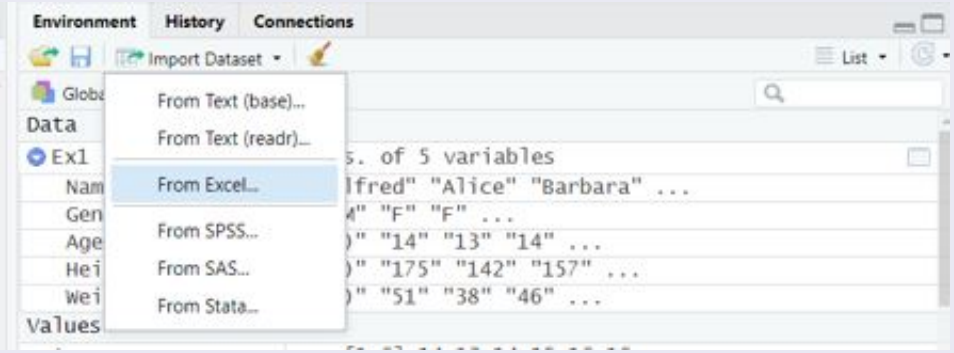
Then locate the Excel file *t2e3.xlsx* file on your hard drive and click *Open*. RStudio displays a preview of the data.
If you are satisfied with what you see click on the *Import* button.

Exercise 3

If you want to import data from a MS Excel file, the easiest way to do this would be to first install the `readxl` packages.

How do we install packages?

Click on the *Import Dataset* button on the *Environment* tab and then select the *From Excel...* option.



Then locate the Excel file *t2e3.xlsx* file on your hard drive and click *Open*. RStudio displays a preview of the data.
If you are satisfied with what you see click on the *Import* button.

Take a look at the Console. What do you observe?

How do we verify the data?

From the tutorial handout, what can we do within the RStudio data viewer? In other words, what features are there?

Calling a variable from a dataset

From the tutorial handout, it is mentioned that you can call a variable from an already loaded data set in two different ways.

First option:

Call a variable by the name of the dataset that includes it and the name of the variable joined by a `$` sign

```
t2e3$Height
```

Second option:

Attach the whole data set to the active project with the `attach()` function and then refer to the variables simply by their names.

```
attach(t2e3)
Weight
```

```
##  [1] 64 71 93 90 73 79 68 75 77 82 77 73 75 82 73 70 75 86 84 70
```

Review Questions

In R, graphs are typically created interactively by calling various graphics functions.
Suppose I have a dataset named `data` and it contains 3 variables: `systolic_bp`, `diastolic_bp` and `patient_id`.
How would you use R to plot a scatterplot of `systolic_bp` on the vertical axis and `diastolic_bp` on the horizontal axis?

What about plotting a histogram and box plot of `systolic_bp`?

Review Questions

What are the similarities/differences between the following three lines of code?

```
plot(weight)
plot(weight, height)
plot(y = height, x = weight)
```

Formatting plots

Explain the following lines of code:

```
plot(Height, Weight,
     main = "Scatterplot of Weight versus Height",
     col = "red", pch = 19,
     xlab = "Height in cm", ylab = " Weight in kg",
     xlim = c(0,200), ylim = c(0,100))
```

Takeaway notes:

- If you provide **two variables names**, then by default the `plot` function returns a scatterplot: x variable = first variable, and y variable = second variable
- If you specify **one variable name only**, the `plot` function treats this variable as the y variable and displays the observation numbers on the horizontal axis.

These functions can either create a complete plot or add an extra layer an existing plot. Basic plots can be produced by calling one of the following functions:

- `plot(x,y)` Scatterplot of two quantitative variables, x (horizontal axis) and y (vertical axis);
- `hist(x)` Histogram for a quantitative variable, x;
- `barplot(x)` Bar plot for a qualitative variable, x;
- `lines(x,y)` Line chart of two quantitative variables, x (horizontal axis) and y (vertical axis);
- `boxplot(x)` Boxplot for a quantitative variable, x.

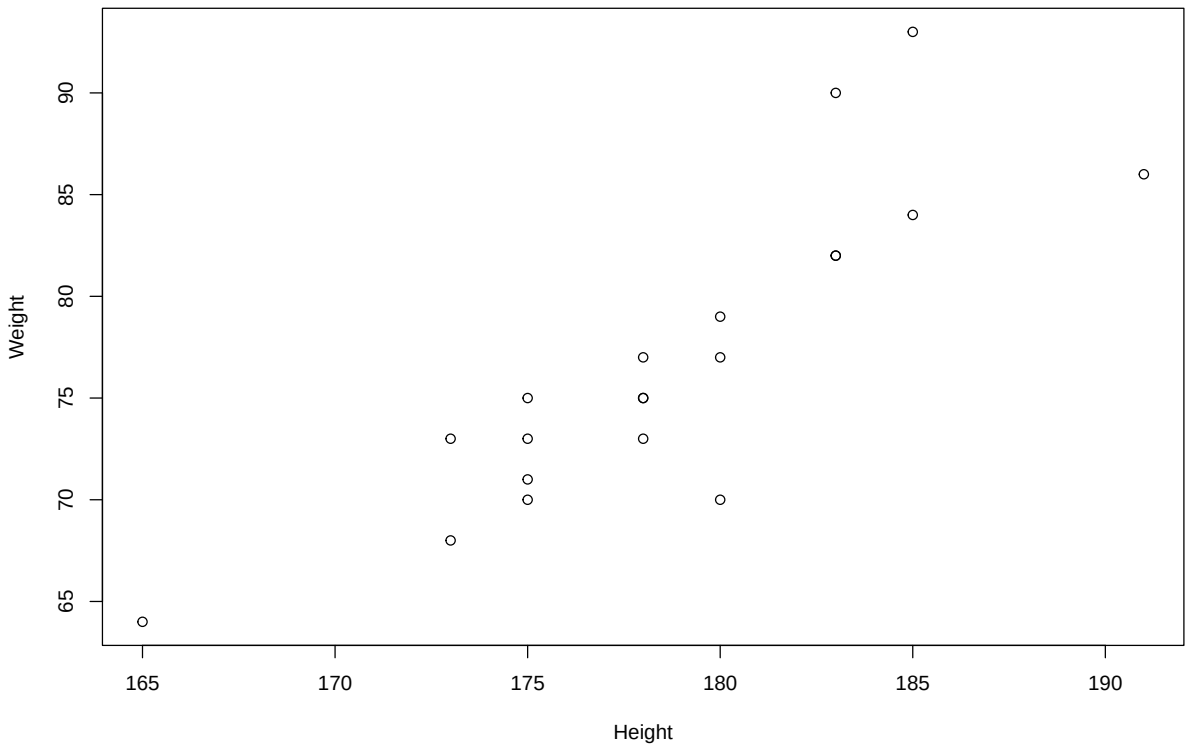
Additional arguments

These functions can have additional arguments, like e.g.

- `data`: data frame;
- `type`: type of the plot, e.g. "p" for points, "l" for lines, "b" for both points and lines, "c" for empty points joined by lines;
- `xlim, ylim`: two-element numeric vectors defined by the c() function14 to set the minimum and maximum values for the horizontal and vertical axes, respectively;
- `xlab, ylab`: labels for the horizontal and vertical axes, respectively;
- `main`: plot title (on top) and subtitle (at bottom), respectively.
- `legend`: placement of the legend, e.g. "left", "topleft";
- `pch`: plotting symbol;
- `col`: colour code or name for lines and symbols

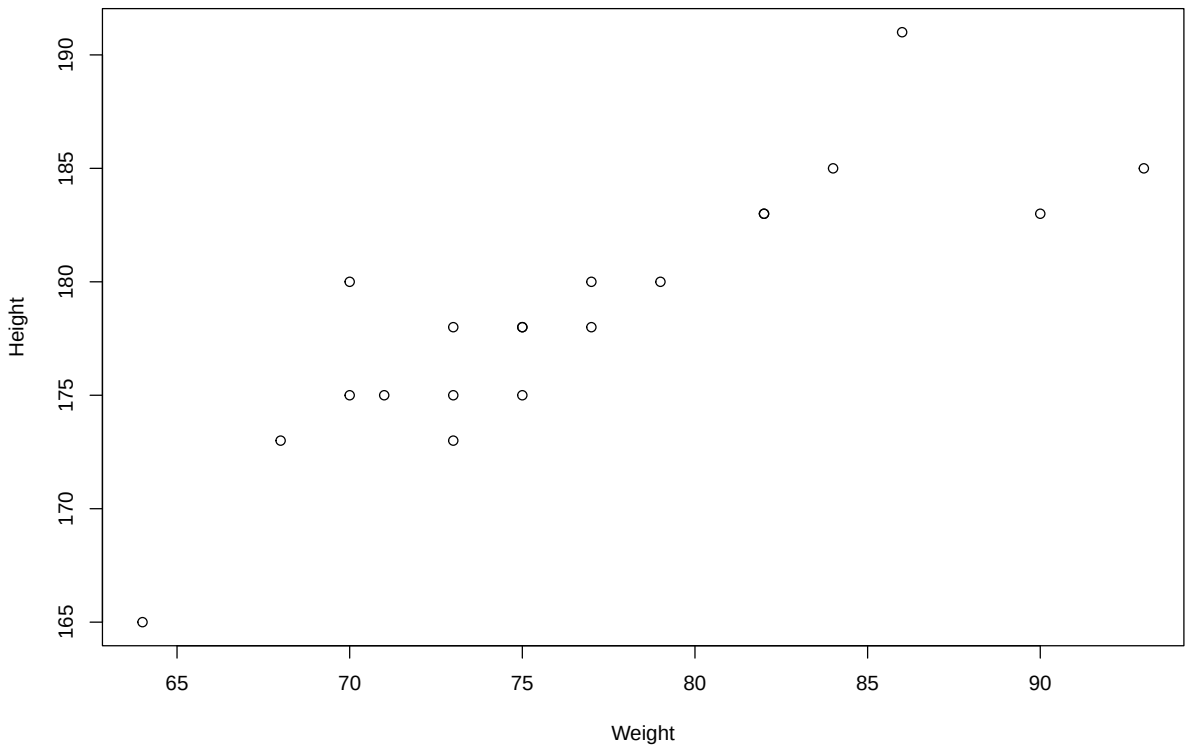
Relationship between Height and Weight

```
plot(Height, Weight)
```



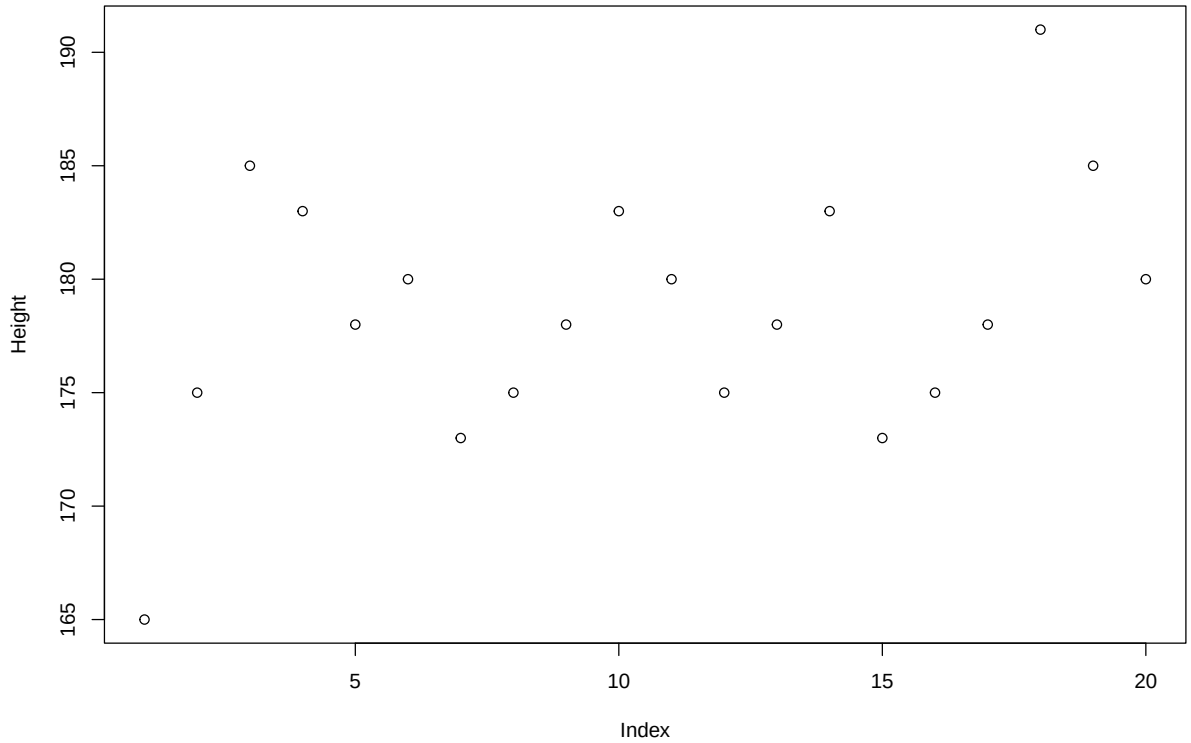
- How do we construct a scatterplot with Height on the horizontal axis and Weight on the vertical axis?

```
plot(Weight, Height)
```



What happens if you provide only ONE variable in the `plot` function?

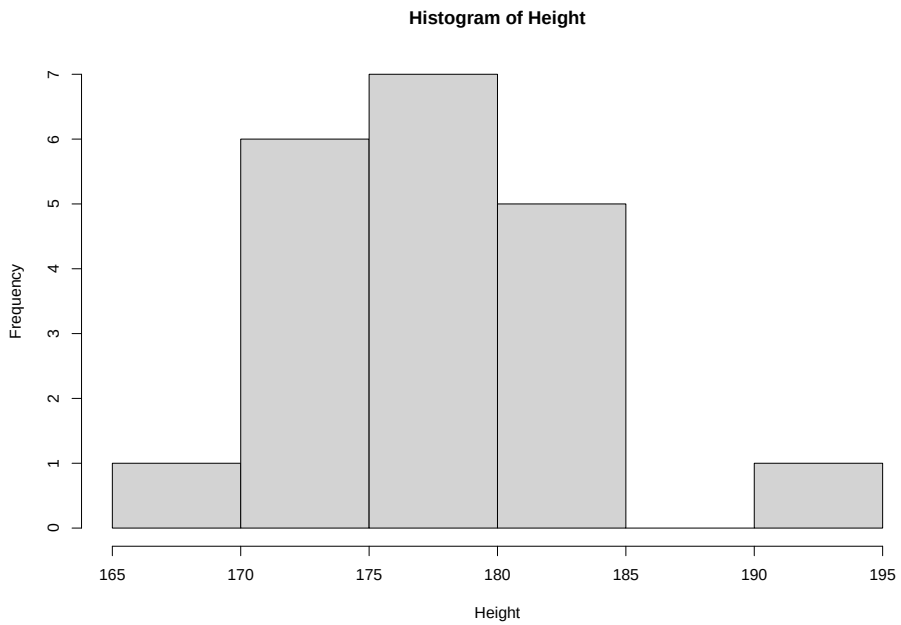
```
plot(Height)
```



How do we construct a histogram? What function do we use?

For e.g. Height

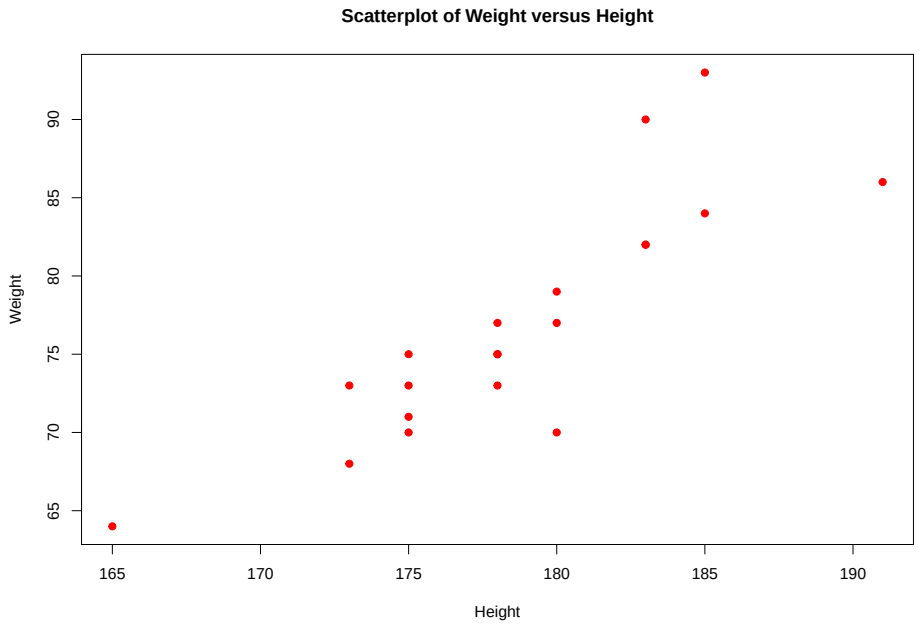
```
hist(Height)
```



- This plot is pretty basic, and can be made to look a bit fancier by the additional arguments mentioned earlier.

How do we add a title to the first scatterplot and to represent the pairs of observations on this scatterplot with red dots (shape #19 filled by "red")?

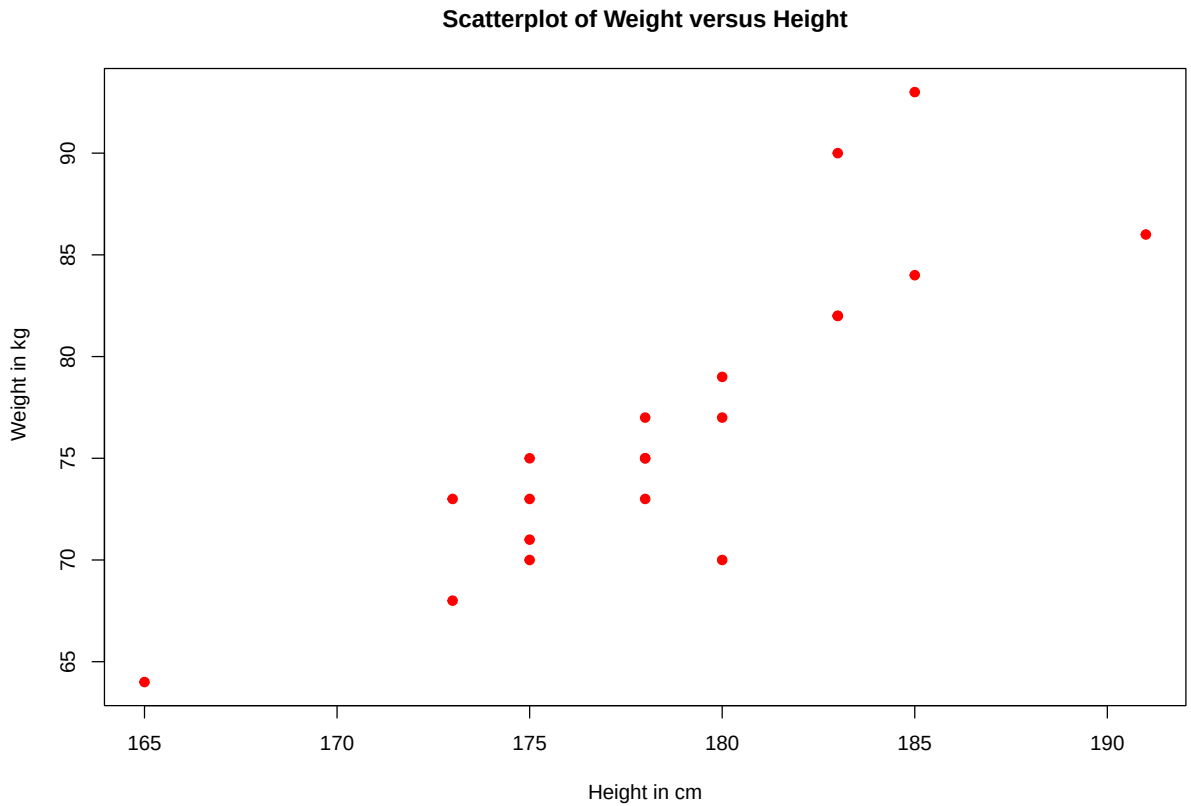
```
plot(Height, Weight,
     main = "Scatterplot of Weight versus Height",
     col = "red", pch = 19)
```



We can provide more informative labels of the variables using the `xlab` and `ylab` arguments of plot.

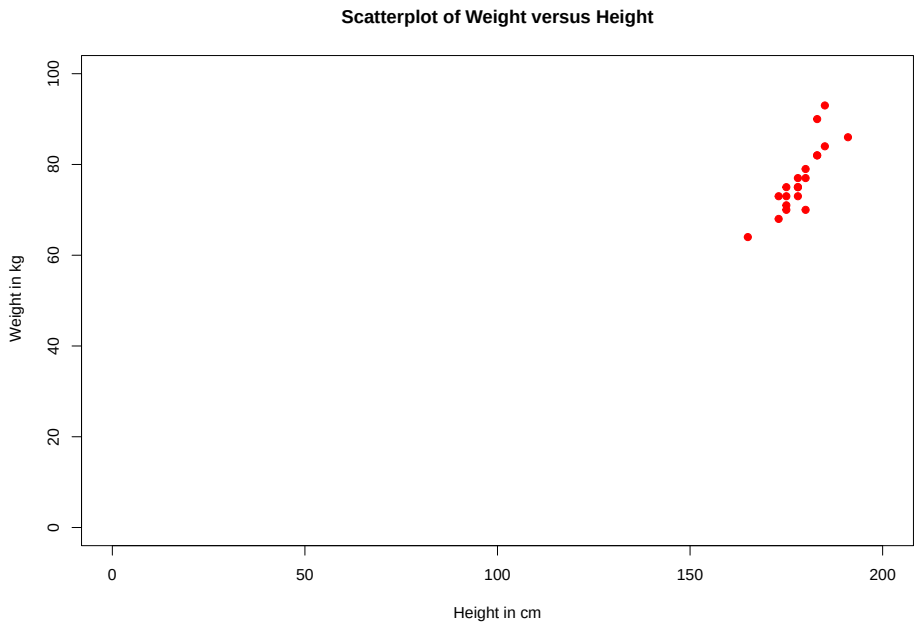
How do we modify the code from before to include more informative labels?

```
plot(Height, Weight,
     main = "Scatterplot of Weight versus Height",
     col = "red", pch = 19,
     xlab = "Height in cm", ylab = " Weight in kg")
```



Now, suppose we want to specify the minimum and maximum values for the horizontal and vertical axes with the xlim and ylim arguments of plot. For instance, to set the scale on the horizontal axis from 0 to 200 and the scale on the vertical axis from 0 to 100.

```
plot(Height, Weight,
     main = "Scatterplot of Weight versus Height",
     col = "red", pch = 19,
     xlab = "Height in cm", ylab = " Weight in kg",
     xlim = c(0,200), ylim = c(0,100))
```



Looking at histograms again, by default, they are black and white, and the number of class intervals, called bins, are determined by R using an algorithm. These features can be overwritten by the `col` and `breaks` arguments.

```
hist(Weight, breaks = 20, col = "blue")
```