# ds105-practice-fall22

## File: 6_subsettingpractice.org

1. Print the length of the pre-built data set `Nile` using an R function

```
length(Nile)
str(Nile)
```

```
[1] 100
 Time-Series [1:100] from 1871 to 1970: 1120 1160 963 1210 1160 1160 813 1230 1370 1140 ..
```

2. Retrieve the second to fifth value of the `Nile` data set

```
Nile[c(2,3,4,5)]
Nile[2:5]
```

```
[1] 1160  963 1210 1160
[1] 1160  963 1210 1160
```

3. Which data science question does the last output answer? Write a full sentence

   »What was the flow of the river Nile in the second through fifth year of observations?«

4. Extract the year that corresponds to the last value of `Nile`. Remember that the years are stored in `time(Nile)`.

```
t <- time(Nile)
time(Nile)[length(Nile)]
```

```
[1] 1970
```

5. What was the Nile flow in 1967?

```
Nile[97]
Nile[t==1967]
```

```
[1] 919
[1] 919
```

6. What is the index of the third-to-last element of `Nile`? Use `which` to answer this question.

```
which(Nile == Nile[length(Nile)-3])
```

```
[1] 97
```

7. How many values of `Nile` are larger than the third-to-last value of `Nile`?

```
length(which(Nile > Nile[t==1967]))
Nile > Nile[t==1967]
which(Nile > Nile[t==1967])
```

```
[1] 43
Time Series:
Start = 1871
End = 1970
Frequency = 1
  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
 [12]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
 [23]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE
 [34] FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE
 [45] FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [56] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE
 [67] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
 [78] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE  TRUE
 [89]  TRUE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
[100] FALSE
 [1]  1  2  3  4  5  6  8  9 10 11 12 13 14 15 16 17 19 20 21 22 23 24 25
[24] 26 27 28 33 38 39 40 46 47 59 64 65 68 76 84 86 88 89 91 94
```

8. How much water flowed down the Nile between 1871 and 1970?

```
sum(Nile)
paste("Nile flow 1871-1970:", sum(Nile), "million cubic metres.")
```

```
[1] 91935
[1] "Nile flow 1871-1970: 91935 million cubic metres."
```

9. In which year was the Nile at its lowest level? Use the `min` function for this task.

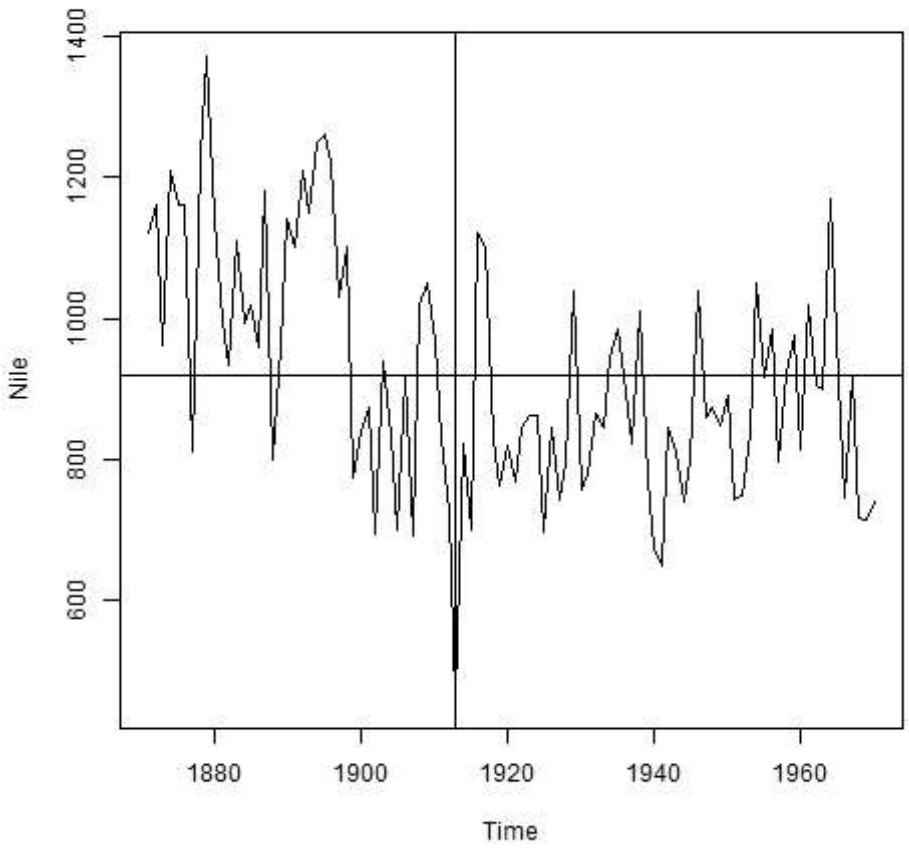
```
tmin <- t[Nile==min(Nile)]
tmin
```

```
[1] 1913
```

10. Make a line plot of all observations in the data set `Nile` using `plot`, mark the year of the lowest level of the Nile with a vertical line, and the average flow through the Nile with a horizontal line. The result is stored in `nile.png`.

    You can draw a vertical line at point `x` with `abline(v=x)`, and a horizontal line at point `y` with `abline(h=y)`.

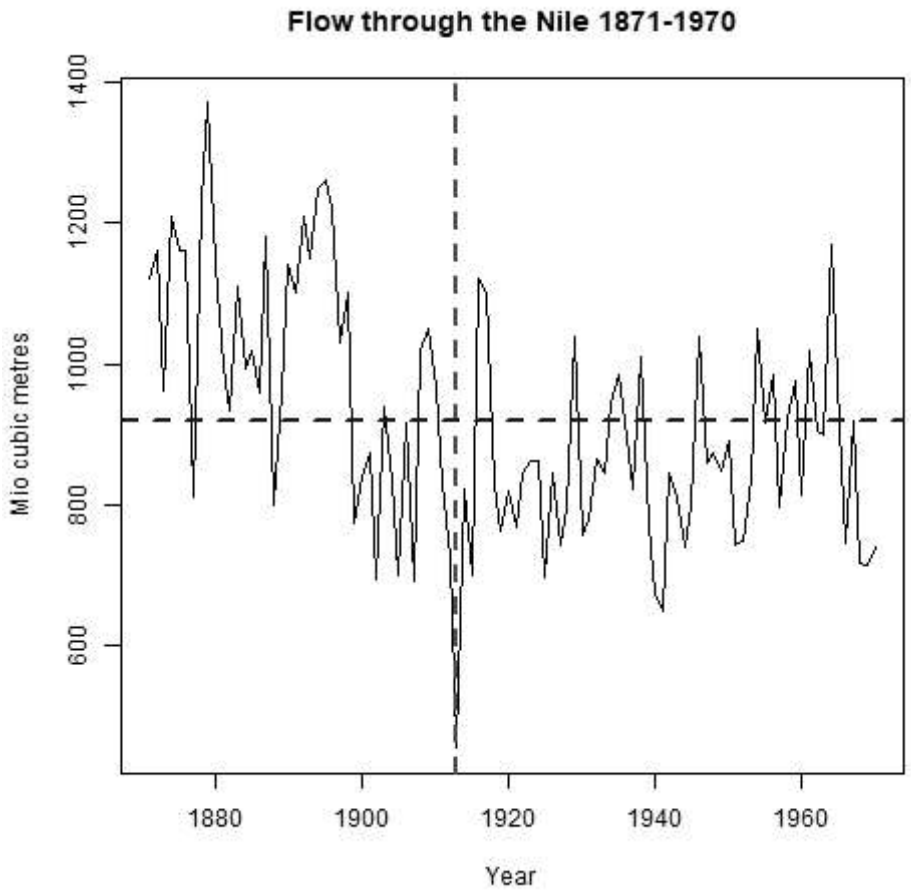
```
plot(Nile)
abline(v=tmin)
abline(h=mean(Nile))
```




Tip: You can change the appearance of lines with the parameters `col`, `lty`, `lwd`. E.g. `col="red, lty=2, lwd=2` for a red, dashed, thick line.

Customized with color, line, title, and label information:

```
plot(Nile,xlab="",ylab="")
abline(v=tmin, col="red", lty=2, lwd=2)
abline(h=mean(Nile), col="blue", lty=2, lwd=2)
title("Flow through the Nile 1871-1970",
      ylab="Mio cubic metres",
      xlab="Year")
```




# File: 6a$_{subsettingpractice.org}$

This exercise is very similar to the practice on the `Nile` data set. Here, we're using another built-in data set, `islands`. Instead of `time`, you need to use `names` to analyze the content of `islands`, and instead of a line plot, you will draw a dot plot.

Find out what `islands` contains and what its structure is. Take a look at `example(islands)`, too. Submit the completed file in Canvas for bonus points.

```
str(islands)
```

```
Named num [1:48] 11506 5500 16988 2968 16 ...
- attr(*, "names")= chr [1:48] "Africa" "Antarctica" "Asia" "Australia" ...
```

1. How many elements does the vector `islands` have?

```
length(islands)
```

```
[1] 48
```

2. Retrieve the third to fourth, the twelfth to thirty-fifth, and the 48th element of `islands` with one command, and check that you retrieved 28 elements altogether (with another command)

```
islands[c(3:5,12:35,48)]
length(islands[c(3:5,12:35,48)])
```

```
          Asia       Australia    Axel Heiberg            Cuba
         16988            2968              16              43
         Devon       Ellesmere          Europe       Greenland
            21              82            3745             840
       Hainan      Hispaniola        Hokkaido          Honshu
           13              30              30              89
      Iceland         Ireland            Java          Kyushu
           40              33              49              14
        Luzon      Madagascar        Melville        Mindanao
           42             227              16              36
     Moluccas     New Britain      New Guinea New Zealand (N)
           29              15             306              44
New Zealand (S)    Newfoundland   North America        Victoria
           58              43            9390              82
[1] 28
```

3. Create a vector `islands_unnamed` from `islands` that is not named and show its structure. Tip: to remove the `names` of a vector, assign `NULL` to it.

```
islands_unnamed <- islands
names(islands_unnamed) <- NULL
str(islands_unnamed)
```

```
num [1:48] 11506 5500 16988 2968 16 ...
```

4. Print the value of `islands` that belongs to the `names(islands)` element `"Iceland"`- which data science question could this answer?

```
islands[names(islands)=="Iceland"]
```

```
Iceland
     40
```

»What is the area of Iceland in thousands of square miles?«

5. Which element of `islands` corresponds to the greatest area? (Put differently: what is the greatest landmass on Earth?)

```
names(islands)[islands == max(islands)]
```

```
[1] "Asia"
```

6. What is the index of the next-to-last element of `islands`? Use `which` to answer this question and save the result as `index`.

```
index <- which(islands == islands[length(islands)-1])
index
```

```
Vancouver
       47
```

7. Check your answer to the last question by finding the value of `islands` that belongs to `index`.

```
islands[index]
```

```
Vancouver
       12
```

8. How many values of `islands` are larger than the next-to-last value of `islands`?

```
length(islands[islands > islands[index]])
```

```
[1] 47
```

9. What is the landmass of all areas listed in `islands`?

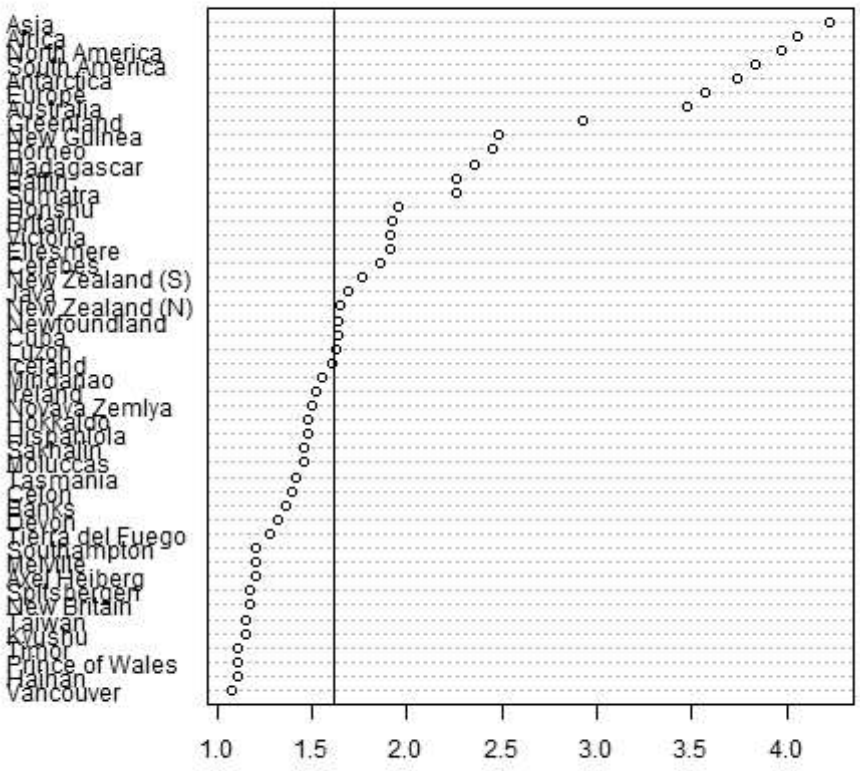
```
paste("Total landmass:", sum(islands), "thousands of square miles.")
```

```
[1] "Total landmass: 60131 thousands of square miles."
```

10. Make a dot plot of all entries in the data set `islands` using the `dotchart` function. As its only argument, use the `log10`-transformed, ordered data set `areas <- log10(sort(islands))`.

    Add a line for the median of this data set, `median(areas)` and add it to the plot with `abline(v=median(areas)`.

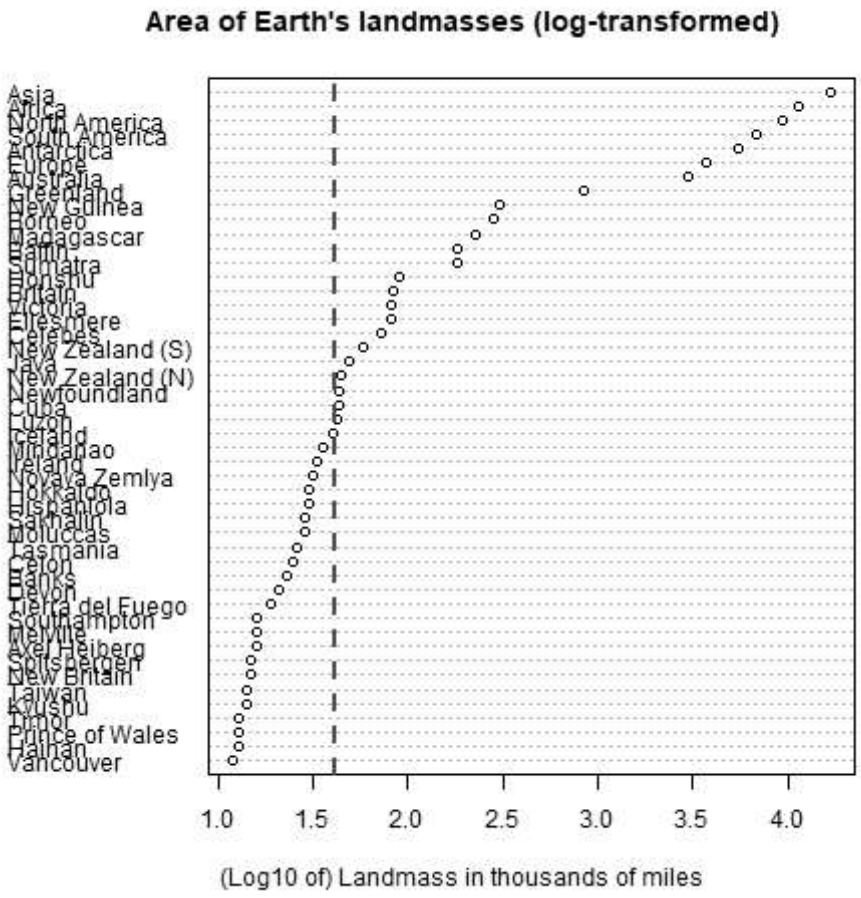
```
areas <- log10(sort(islands))
md_areas <- median(areas)
dotchart(areas)
abline(v=md_areas)
```




Tip: You can change the appearance of lines with the parameters `col`, `lty`, `lwd`. E.g. `col="red, lty=2, lwd=2` for a red, dashed, thick line.

Customized with line type, line width, title, labels and color:

```
dotchart(areas)
abline(v=md_areas, col="red",lty=2,lwd=2)
title("Area of Earth's landmasses (log-transformed)",
      xlab="(Log10 of) Landmass in thousands of miles")
```




Created: 2023-10-23 Mon 10:38