Day 1 - R intro exercises

2024-09-06

Workflow
Create a new emtpy R script. Solve the exercises one by one. Here you just type in the R-commands on empty line one by one. To execute your code place the cursor at the specified line and use the keyboard shortcuts:

- Run code/selection and move to next line: `Ctrl + Enter` (Ctrl + Enter) and Mac: `Command + Return` (Command + Return)
- Run current line/selection (retain cursor position): `Alt + Enter` (Alt + Enter) and Mac: `Option + Return` (Option + Return)

Once you are happy with the solution, you can **compile** the document as a stand-alone html-file. You do this by either

- Go 'File > Knit Document' in the drop-down menu,
- click on the small *notebook* icon in the editor window, or
- use the keyboard shortcut `Ctrl + Shift + K` (Ctrl + Shift + K) and on Mac: `Shift + Command + K`) (Shift + Ctrl + K).

Exercises

Create the following two vectors:

```
v1 <- c(1, -5, 3, -7, 12, -9)
v2 <- c(30, 50, 10)
```

A) Simple arithmetic

A.1) Add 10 to each element of `v1`.

```
v1 + 10
```

```
[1] 11  5 13  3 22  1
```

A.2) Multiply each element of `v1` by 10.

```
10 * v1
```

```
[1]  10 -50  30 -70 120 -90
```

A.3) Add `v1` to `v2`. Make sure you understand the result.

```
v1+v2
```

```
[1] 31 45 13 23 62  1
```

B) Vector operations

B.1) Select the 2nd and 5th element of `v1` (from above)

```
v1[c(2,5)]
```

```
[1] -5 12
```

B.2) Select all except the 2nd and 5th element from `v1`.

```
v1[-c(2,5)]
```

```
[1]  1  3 -7 -9
```

B.3) Create the logical vector `ww` which elements are `TRUE` if the elements of `v1` are larger or equal to 3 and `FALSE` otherwise.

```
ww <- v1 >= 3
```

B.4) Concatenate `v1` and `v2` and call this vector `v12`.

```
v12 <- c(v1, v2)
```

B.5) Find out whether any element of `v12` lies between 2 and 4. *Hint: `any()` is your friend.*

```
any(v12 > 2 & v12 < 4)
```

```
[1] TRUE
```

B.6) How many negative elements does `v12` contain? *Hint: Make a logical vector telling which elements are negative and then use the function `sum()` to count the TRUE values.*

```
neg <- v12 < 0
sum(neg)
```

```
[1] 3
```

B.7) Generate the vector `g` which contains first 3 times the character value `'weak'` followed by 2 times the character value `'strong'`. *Hint: Use the `rep()` function!*

```
g <- rep(c("weak", "strong"), times = c(3,2))
```

C) Basic statistics

Run the following commands:

```
set.seed(1234)
x <- round(runif(100, 0, 10))
y <- round(runif(100, 0, 10))
```

C.1) Find the smallest element in `x`.

```
min(x)
```

```
[1] 0
```

C.2) Find the largest element in `x`.

```
max(x)
```

```
[1] 10
```

C.3) Find the range of `x`.

```
range(x)
```

```
[1]  0 10
```

C.4) Find the sum of all the elements in `x`.

```
sum(x)
```

```
[1] 435
```

C.5) Find the mean of all the elements in `x`. (Does it seem reasonable?)

```
mean(x)
```

```
[1] 4.35
```

C.6) Find the standard deviation of all the elements in `x`. (Does it seem reasonable?)

```
sd(x)
```

```
[1] 2.804668
```

C.7) Make a vector from `x` where the elements no. $11 - 20$ And element no. 51 are removed. *Hint: Use indices with negative sign.*

```
x[-c(11:20, 51)]
```

```
 [1]  1  6  6  6  9  6  0  2  7  5  3  3  2  0  2  8  5  9  8  0  5  3  3  5  2
[26]  8  2  3 10  8  6  6  3  6  3  5  7  5  2  8  3  7  5  2  5  5  8  2  8  9
[51]  0  3  0  2  7  3  5  1  6  1  9  0  8  1  5  4  1  3  7  9  5  1  5  2  9
[76]  4  3  2  9  2  9  1  1  1  5  3  0  3  7
```

C.8) Make a vector from `x` where all the elements with value less than 5 are removed.

```
x[x>=5]
```

```
 [1]  6  6  6  9  6  7  5  7  5  9  8  8  5  9  8  5  5  8 10  8  6  6  6  5  7
[26]  5  8  7  5  5  5  8  8  9  7  5  6  9  8  5  7  9  5  5  9  9  9  5  7
```

C.9) Determine the index of those elements in `x` which have value either 0 or 10.

```
x0 <- x == 0
x10 <- x == 10
which(x0 | x10)
```

```
[1]  7 24 30 39 62 64 73 98
```

C.10) Determine the index of those elements in `x` which have the same value as the preceding one. *Hint: use the command* ***`diff()`***.

```
i <- which(diff(x) == 0)
i+1
```

```
 [1]  3  4 18 20 22 33 42 57 94 95
```

C.11) Consider `x` and `y` as paired observations. Make a vector of those elements of `x` for which `y` takes the value 5.

```
x[y==5]
```

```
[1] 2 9 8 8 6 6 9
```

C.12) Determine the indices for which `x` and `y` have the same value.

```
which(x==y)
```

```
[1]  2 11 18 43 47 55 77 93
```

C.13) Make a vector which from every pair of `x` and `y` chooses the largest of the two values. *Hint: use the command* ***`ifelse()`***.

```
ifelse(x>y, x, y)
```

```
 [1]  1  6  6  6  9  6  2  2  7  5  7  5 10  9  3  9  9  3  2  8  7  9 10  9  5
[26]  8  5  9  8  3 10  6  3  5  9  8  9  6 10  9  6 10  3  6  4  6  7  6 10  8
```

```
 [51]  4  3  7  9  2 10  6 10  4  8  9  6  4  2  2  7  4  5  8  6  8  9  7  8  3
 [76]  8  4  6  3  7  9  5  6  5 10  9  8  6  3  9  6 10  1  9  8  8  8  7 10  7
```

C.14) Solve the exercise above even more easily with the command `pmax()`.

```
pmax(x,y)
```

```
  [1]  1  6  6  6  9  6  2  2  7  5  7  5 10  9  3  9  9  3  2  8  7  9 10  9  5
 [26]  8  5  9  8  3 10  6  3  5  9  8  9  6 10  9  6 10  3  6  4  6  7  6 10  8
 [51]  4  3  7  9  2 10  6 10  4  8  9  6  4  2  2  7  4  5  8  6  8  9  7  8  3
 [76]  8  4  6  3  7  9  5  6  5 10  9  8  6  3  9  6 10  1  9  8  8  8  7 10  7
```