Selecting parts of objects

MEB433 and MEB434

Lukáš Lehotský and Petr Ocelík

https://campus.datacamp.com/courses/free-introduction-to-r

Functions: most useful basic functions

```
c() # combine two or more elements into an object

class() # explore elements' data class
length() # explore number of first dim. of object
dim() # explore dimensions of two-dimensional obj.
nrow() # number of rows
ncol() # number of columns

head() # first few rows of data
tail() # last few rows of data
str() # explore structure of object

names() # names in the named vector - one dimension
rownames() # names of rows - two dimensions
colnames() # names of columns - two dimensions
```

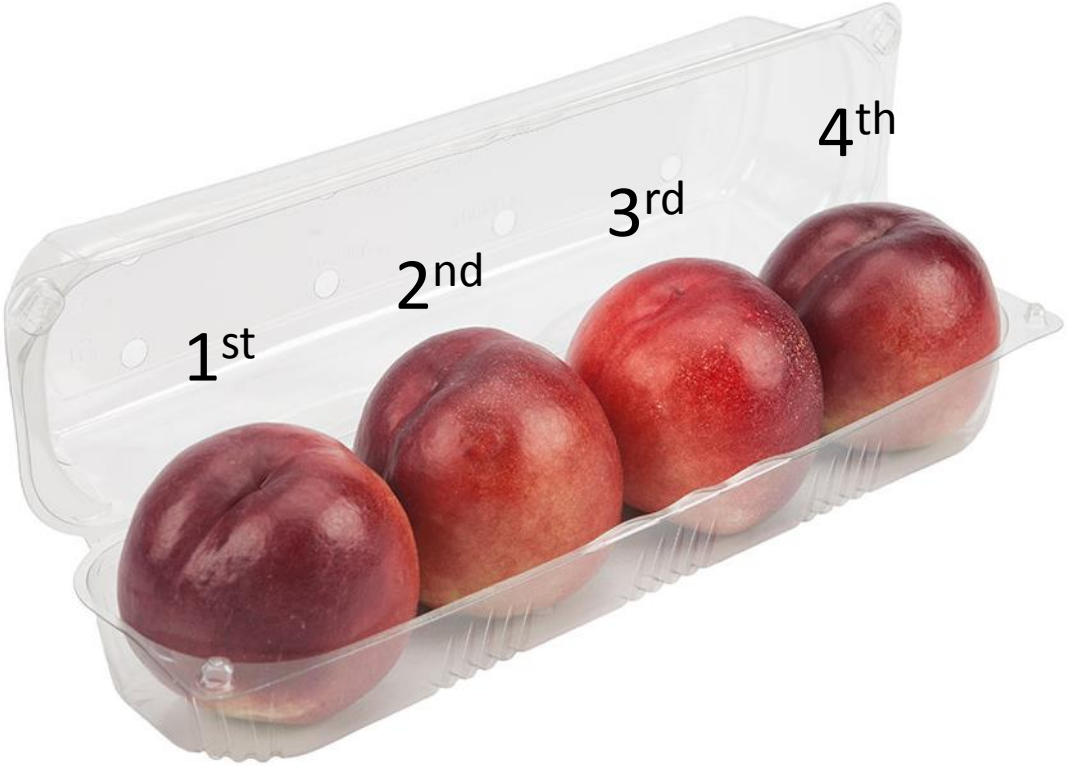
Selecting: logic of indexes

- Important to realize whole logic of R is relying on **position of objects**
- Any object consists of **N elements** (N = 1, 2, … , n)
- Any element/nested object might be selected based on its **position** (also called **index**)
- Selection has to **include all dimensions (always)**

```
object[dim 1,...,dim n]
```

Selecting: one dimension

```
peaches <- c("first","second","third","fourth")
```



Selecting: one dimension

```
peaches <- c("first","second","third","fourth")
```

1st 2nd 3rd 4th

```
peaches[3]
```

```
[1] "third"
```

Selecting: two dimensions

- Data frame/matrix
- Selection of rows and/or columns
- Always have to include **both dimensions**
- If dimension is empty, returns **all values** – think of it as a **restriction on certain dimmension**

```
object[ row , column ]
```

Selecting: two dimensions

```
rack[ 2 , ]
```

Selecting: two dimensions

```
rack[ 2 , ]
```

2nd row

Selecting: two dimensions

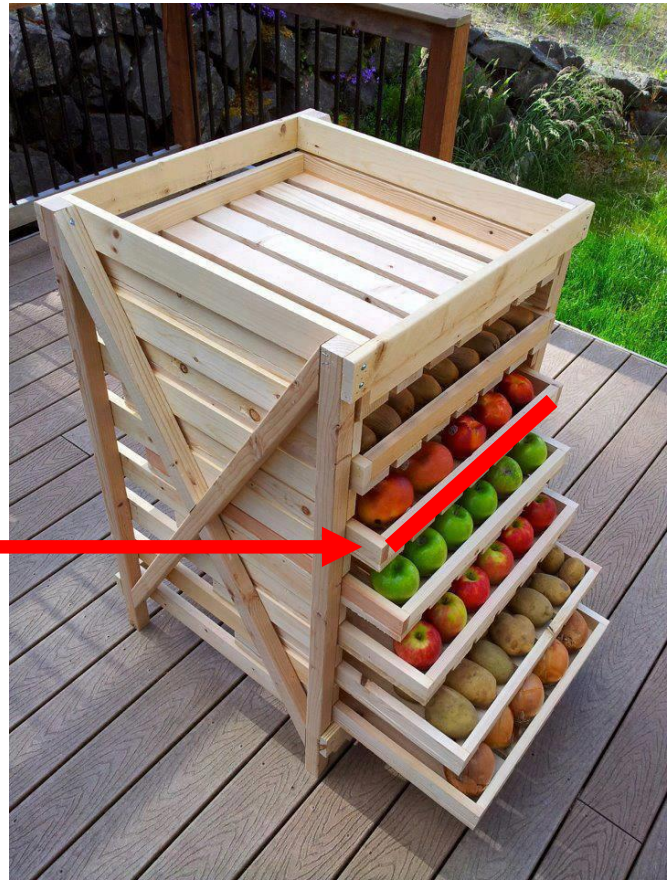
```
rack[ 2 , ]
```

2nd row

all columns

Selecting: two dimensions

```
rack[ 2 , ]
```

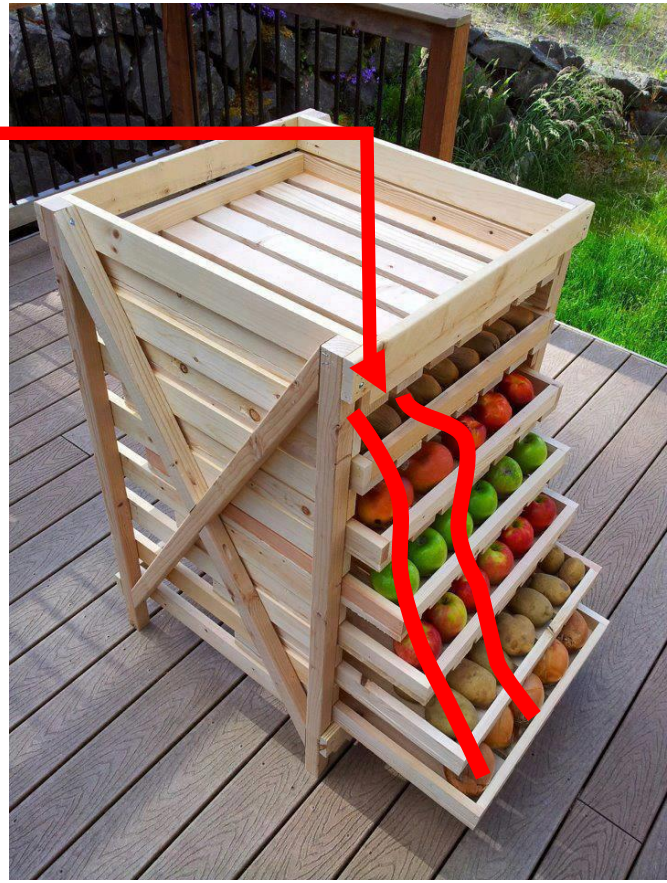


Selecting: two dimensions

```
rack[ 2 , ]
rack[ , 2]
```

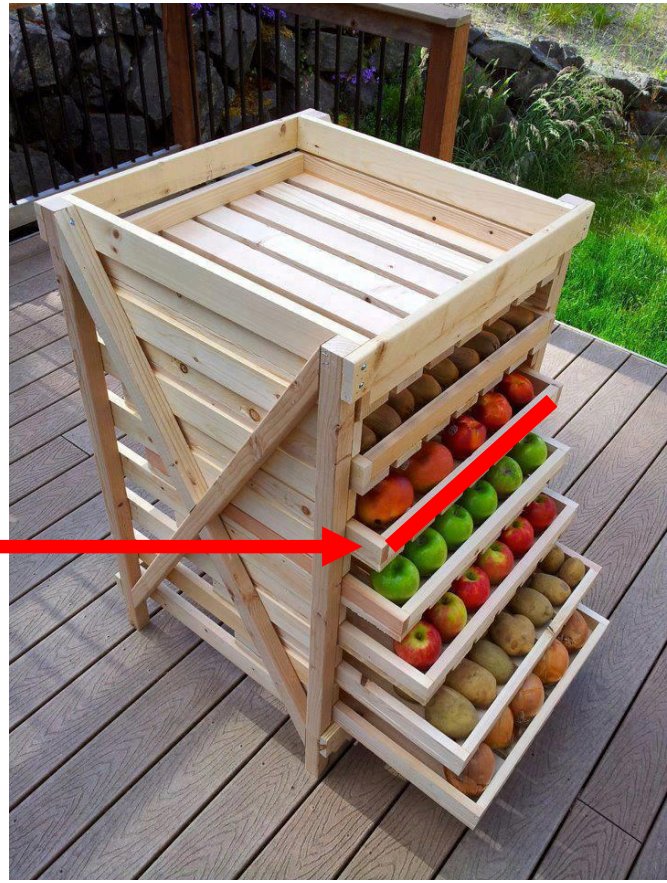
Selecting: two dimensions

```
rack[ 2 , ]
rack[ , 2]
```



Selecting: two dimensions

```
rack[ 2 , ]
rack[ , 2]
rack["redapples", ]
```

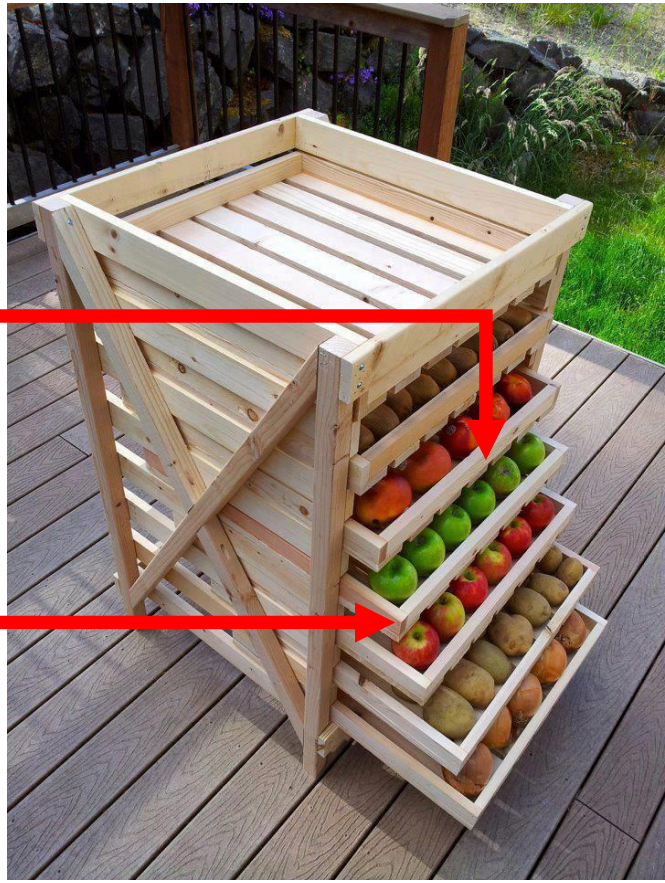


Selecting: two dimensions

```
rack[ 2 , ]
rack[ , 2]
rack["redapples", ]
rack["greenapples",4]
```

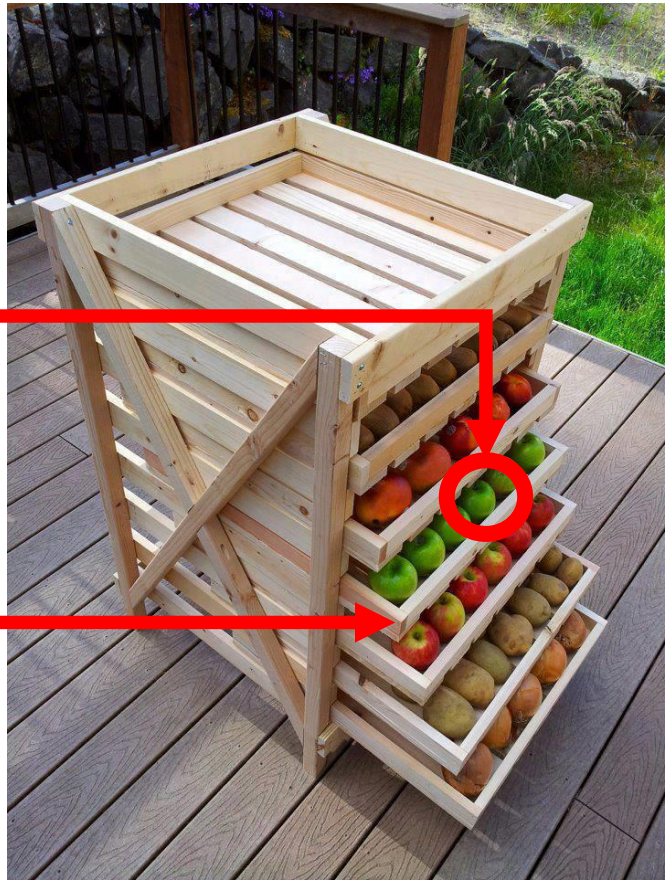
Selecting: two dimensions

```
rack[ 2 , ]
rack[ , 2]
rack["redapples", ]
rack["greenapples",4]
```



Selecting: two dimensions

```
rack[ 2 , ]
rack[ , 2]
rack["redapples", ]
rack["greenapples",4]
```



Selecting: two dimensions

- Data frame/matrix `df` with car information below
- Selection of rows and/or columns
  - Always have to include **both dimensions**
  - If dimension is empty, returns **all values**

```
     cars  type     price      consumption
1    BMW   3        1200000    6.2
2    Audi  A4       1164000    5.9
3    VW    Passat   950500     5.9
```

Selecting: two dimensions, single row

- Output of **row-wise selection** is always **data frame**

```
df[ 2 , ]
      cars  type  price       consumption
2     Audi  A4    1164000     5.9

      cars  type          price       consumption
1     BMW   3             1200000     6.2
2     Audi  A4            1164000     5.9
3     VW    Passat        950500      5.9
```

Selecting: two dimensions, single column

- Output of **column-wise selection**
  - Is a **vector** if **one** column is selected
  - Is a **data frame** if **two or more** columns selected

```
df[  , 1 ]
[1] "BMW" "Audi" "VW"
```

	cars	type	price	consumption
1	BMW	3	1200000	6.2
2	Audi	A4	1164000	5.9
3	VW	Passat	950500	5.9

Selecting: two dimensions, row and column

- Output of selection on **both dimensions**
  - Is **vector** if **one column** is selected
  - Is **data frame** otherwise

```
df[ 2 , 1 ]
[1] "Audi"
```

	cars	type	price	consumption
1	BMW	3	1200000	6.2
2	Audi	A4	1164000	5.9
3	VW	Passat	950500	5.9

Selecting: two dimensions, column by name

- If columns have names, there is way of **selecting column by its name**
  - **Column names** have to be **defined** (check with `colnames()`)
  - Column name should be **in parentheses** and in place of **column dimension**

```
df[ , "cars" ]
[1] "BMW" "Audi" "VW"
```

	cars	type	price	consumption
1	BMW	3	1200000	6.2
2	Audi	A4	1164000	5.9
3	VW	Passat	950500	5.9

Selecting: two dimensions, column by name

- Selection of one column returns **vector**
- The transformation might be undesirable
  - Additional argument `drop=FALSE`
  - Should be included **after the last dimension**
  - Preserves the object type as **data frame**

```
df[ , "cars" , drop=FALSE ]
```

```
      cars
1     "BMW"
2     "Audi"
3     "VW"
```

Selecting: two dim, multiple rows/columns

- Selection of multiple columns/rows possible through **nesting** – using **functions to construct the index**
  - Most commonly, function `c()` will allow to select **more than one** column or row (e.g. `c(1, 3)`)
  - Uninterrupted **range** of columns can be selected using **colon sign** ":" (e.g. `1:100`)

```
df[  , c(1 , 3) ]
```

	cars	*type*	*price*	*consumption*
1	*BMW*	*3*	*1200000*	*6.2*
2	*Audi*	*A4*	*1164000*	*5.9*
3	*VW*	*Passat*	*950500*	*5.9*

Selecting: two dimensions, direct col. name

- A `$` sign allows to call variable by its **name** directly
  - Applies to **data frame only**, allows to select **only one** column
  - No square brackets needed
  - Output is **one-dimensional vector only** (can't preserve the data frame structure)

```
df$cars
[1] "BMW" "Audi" "VW"

df$price
[1] 1200000 1164000 950500

class(df$price)
[1] "numeric"
```

Selecting: operators

Operator	Description
[[[	Selection by index
$ @	Selection by variable/object name
:	Sequence (from:to) operator

Source: Adler, J. (2012). R in a nutshell. Pp. 86.

Practice 1

- Create an object from the R dataset “eurodist”
- Explore the data source and transform it to the most suitable format for data manipulation (matrix)
- Extract distances from Stockholm into separate vector
- Find shortest and longest distance from Stockholm (google appropriate function)
- Figure out what is the distance between Cologne and Lisbon

Practice 2

- Install and load the package “poliscidata”
- Create a new object in your environment using the dataset “world”
- Explore variables in the dataset (use `colnames()` function)
- Extract countries into separate vector
- Extract country IDs into separate data frame
- Find the Czech Republic in the data frame
  - Extract the appropriate row from the data frame
- Find all V4 countries (CZ, SK, HU, PL)
  - Extract them into a separate data set
  - Extract only freedom indicators for these countries (all column names starting with “free_”)