Sorting

Now that we have some basic R knowledge under our belt, let's try to gain some insights into the safety of different states in the context of gun murders.

`sort`

We want to rank the states from least to most gun murders. The function `sort` sorts a vector in increasing order. So we can see the number of gun murders by typing

```
library(dslabs)
```

```
## Warning: package 'dslabs' was built under R version 4.1.3
```

```
data(murders)
sort(murders$total)
```

```
##  [1]    2    4    5    5    7    8   11   12   12   16   19   21   22   27   32
## [16]   36   38   53   63   65   67   84   93   93   97   97   99  111  116  118
## [31]  120  135  142  207  219  232  246  250  286  293  310  321  351  364  376
## [46]  413  457  517  669  805 1257
```

However, this does not give us information about which states have which murder totals. For example, we don't know which state had 1257 murders in 2010.

`order`

The function `order` is closer to what we want. It takes a vector and returns the vector of indexes that sort the input vector. This may sound confusing so let's look at a simple example: we create a vector and sort it:

```
x <- c(31, 4, 15, 92, 65)
sort(x)
```

```
## [1]  4 15 31 65 92
```

Rather than sort the vector, the function `order` gives us back the index that, if used to index the vector, will sort it:

```
index <- order(x)
index
```

```
## [1] 2 3 1 5 4
```

```
x[index]
```

```
## [1]  4 15 31 65 92
```

If we look at this index we see why it works:

```
x
```

```
## [1] 31  4 15 92 65
```

```
order(x)
```

```
## [1] 2 3 1 5 4
```

Note that the second entry of `x` is the smallest so `order(x)` starts with `2`. The next smallest is the third entry so the second entry is `3` and so on.

How does this help us order the states by murders? First remember that the entries of vectors you access with `$` follow the same order as the rows in the table. So, for example, these two vectors, containing the state names and abbreviations respectively, are matched by their order:

```
## first 10 states and atheir abbreviations
murders$state[1:10]
```

```
##  [1] "Alabama"              "Alaska"               "Arizona"
##  [4] "Arkansas"             "California"           "Colorado"
##  [7] "Connecticut"          "Delaware"             "District of Columbia"
## [10] "Florida"
```

```
murders$abb[1:10]
```

```
##  [1] "AL" "AK" "AZ" "AR" "CA" "CO" "CT" "DE" "DC" "FL"
```

So this means we can now order the state names by their total murders by first obtaining the index that orders the vectors according to murder totals, and then indexing the state names or abbreviation vector:

```
ind <- order(murders$total)
murders$abb[ind]
```

```
##  [1] "VT" "ND" "NH" "WY" "HI" "SD" "ME" "ID" "MT" "RI" "AK" "IA" "UT" "WV" "NE"
## [16] "OR" "DE" "MN" "KS" "CO" "NM" "NV" "AR" "WA" "CT" "WI" "DC" "OK" "KY" "MA"
## [31] "MS" "AL" "IN" "SC" "TN" "AZ" "NJ" "VA" "NC" "MD" "OH" "MO" "LA" "IL" "GA"
## [46] "MI" "PA" "NY" "FL" "TX" "CA"
```

We see that California had the most murders.

`max` and `which.max`

If we are only interested in the entry with the largest value we can use `max` for the value

```
max(murders$total)
```

```
## [1] 1257
```

and `which.max` for the index of the largest value

```r
i_max <- which.max(murders$total)
murders$state[i_max]
```

```
## [1] "California"
```

For the minimum we can use `min` and `which.min` in the same way.

So is California the most dangerous state? In a next section we argue that we should be considering rates not totals. Before doing that we introduce one last order related function: `rank`

`rank`

Although less useful than `order` and `sort`, the function `rank` is also related to order. For any given list it gives you a vector with the rank of the first entry, second entry, etc... of the vector. Here is a simple example.

```r
x <- c(31, 4, 15, 92, 65)
rank(x)
```

```
## [1] 3 1 2 5 4
```

To summarize let's look at the results of the three functions we have introduced

original	sort	order	rank
31	4	2	3
4	15	3	1
15	31	1	2
92	65	5	5
65	92	4	4