Homework 3

Andrew Shao

2024-09-25

2.16.3 Q1

```
sum(!is.na(as.Date(paste0(1900:2021, '-02-29'), format = '%Y-%m-%d')))
```

```
## [1] 30
```

2.16.3 Q2

-10 years in days; R assumes 2 digit years to be in the 2000's if between 00 and 68 otherwise 1900's so `x` is really January 1, 2069 while `y` is December 31, 1968.

```
x <- as.Date("69-01-01", format = "%y-%m-%d")
y <- as.Date("68-12-31", format = "%y-%m-%d")
x - y
```

```
## Time difference of -36524 days
```

2.16.3 Q3

```
q3 <- as.Date('21-02-14', format = '%y-%m-%d')
q3 + 1000
```

```
## [1] "2023-11-11"
```

2.16.3 Q4

```
q4 <- as.POSIXct('21-07-04 08:15:00', format = '%y-%m-%d %H:%M:%S')
q4 +
  as.difftime(365, units = 'days') +
  as.difftime(2, units = 'days') +
  as.difftime(3, units = 'hours') +
  as.difftime(4, units = 'mins') +
  as.difftime(5, units = 'secs')
```

```
## [1] "2022-07-06 11:19:05 EDT"
```

3.2.5 Q1

```
a <- array(1:24, c(2, 3, 4))
apply(a, 2, mean)
```

```
## [1] 10.5 12.5 14.5
```

3.2.5 Q2

```
apply(a, c(1, 3), quantile, 0.25)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    8   14   20
## [2,]    3    9   15   21
```

3.2.5 Q3

```
apply(a, c(1, 3), quantile, c(0.25, 0.75))
```

```
## , , 1
##
##     [,1] [,2]
## 25%    2    3
## 75%    4    5
##
## , , 2
##
##     [,1] [,2]
## 25%    8    9
## 75%   10   11
##
## , , 3
##
##     [,1] [,2]
## 25%   14   15
## 75%   16   17
##
## , , 4
##
##     [,1] [,2]
## 25%   20   21
## 75%   22   23
```

3.4.4 Q1

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
animal <- rep(c("sheep", "pig"), c(3,3))
weight <- c(110, NA, 140, NA, 300, 800)
condition <- c("excellent", "good", NA, "excellent", "good", "average")
healthy <- c(rep(TRUE, 5), FALSE)
my_tibble <- tibble(animal, weight, condition, healthy)
my_data_frame <- data.frame(animal, weight, condition, healthy)
my_tibble
```

```
## # A tibble: 6 x 4
##   animal weight condition healthy
##   <chr>   <dbl> <chr>     <lgl>
## 1 sheep     110 excellent TRUE
## 2 sheep      NA good      TRUE
## 3 sheep     140 <NA>      TRUE
## 4 pig        NA excellent TRUE
## 5 pig       300 good      TRUE
## 6 pig       800 average   FALSE
```

```r
add_row(my_tibble, animal = "pig", weight = 900, condition = 'average', healthy = FALSE)
```

```
## # A tibble: 7 x 4
##   animal weight condition healthy
##   <chr>   <dbl> <chr>     <lgl>
## 1 sheep     110 excellent TRUE
## 2 sheep      NA good      TRUE
## 3 sheep     140 <NA>      TRUE
## 4 pig        NA excellent TRUE
## 5 pig       300 good      TRUE
## 6 pig       800 average   FALSE
## 7 pig       900 average   FALSE
```

3.4.4 Q2

`my_tibble[, 1]` should return a `tibble` of size 6×1 while `my_data_frame[, 1]` should return a vector. To reproduce `my_data_frame[, 1]` you should add `drop = TRUE` to `my_tibble[, 1]`.

3.5.7 Q1

```
dig_num <- 1:6
ani_char <- c("sheep", "pig", "monkey", "pig", "monkey")
x_mat <- matrix(1:12, nrow = 3, ncol = 4)
my_list <- list(num = dig_num, char = ani_char, mat = x_mat)
my_list
```

```
## $num
## [1] 1 2 3 4 5 6
##
## $char
## [1] "sheep"  "pig"    "monkey" "pig"    "monkey"
##
## $mat
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

`my_list[2]` returns the sublist of element 2 of `my_list` while `my_list[3]` returns the sublist of element 3.

3.5.7 Q2

`my_list[2:3]` returns the sublist containing elements 2 and 3 of `my_list` while `my_list[[2:3]]` returns the third element of the second element of `my_list`.

3.5.7 Q3

```
sapply(my_list, length)
```

```
##  num char  mat
##    6    5   12
```

3.5.7 Q4

```
my_list$mat[3, ] * 10
```

```
## [1]  30  60  90 120
```

```
my_list
```

```
## $num
## [1] 1 2 3 4 5 6
```

```
##
## $char
## [1] "sheep"  "pig"    "monkey" "pig"    "monkey"
##
## $mat
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

3.6.5 Q1

```
x <- c(NA, NULL, Inf, NaN)
```

3

```
length(x)
```

```
## [1] 3
```

3.6.5 Q2

class is `numeric` and storage type is `double`

```
class(x)
```

```
## [1] "numeric"
```

```
typeof(x)
```

```
## [1] "double"
```

3.6.5 Q3

`x + 1` returns `c(NA, Inf, NaN)`.
Adding 1 to the first element returns `NA` because adding to a `NA` returns `NA` because the values is not available.
Adding 1 to the second element effectively returns nothing because `NULL` has length 0 so adding to a length 0 object returns another object of length 0.
Adding 1 to the third element returns `Inf` because `Inf` represents infinity and mathematically, adding 1 to infinity is still infinity.
Adding 1 to the fourth element returns `NaN` because `NaN` is not a number and thus adding 1 is still not a number.

```
x + 1
```

```
## [1]  NA Inf NaN
```

3.6.5 Q4

`c(NA, TRUE, NA)`
The result of the operation on the first element is `NA` because it is not available and comparing two not available objects is also not available.
The result of the operation on the second element is nothing because comparing two objects of length 0 returns a logical vector of length 0.
The result of the operation on the third element is`TRUE` because infinity is equal to infinity in R.
The result of the operation on the fourth element is `NA` because comparing two not a number values is not available.

```
x == x
```

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