Intro to R Homework

Ai Yukino

HW link

Import packages

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.6     v purrr   0.3.4
## v tibble  3.1.7     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Creating vectors/factors and dataframes

We are performing RNA-Seq on cancer samples being treated with three different types of treatment (`A`, `B`, and `P`). You have 12 samples total, with 4 replicates per treatment. Write the R code you would use to construct your metadata table as described below.

	sex	stage	treatment	myc
sample1	M	I	A	2343
sample2	F	II	A	457
sample3	M	II	A	4593
sample4	F	I	A	9035
sample5	M	II	B	3450
sample6	F	II	B	3524
sample7	M	I	B	958
sample8	F	II	B	1053
sample9	M	II	P	8674
sample10	F	I	P	3424
sample11	M	II	P	463
sample12	F	II	P	5105

Create the vectors/factors for each column (Hint: you can type out each vector/factor, or if you want the process to go faster try exploring the `rep()` function)

```
sex <- rep(c("M", "F"), 6)
stage <- rep(c("I", "II", "III"), 4)
treatment <- rep(c("A", "B", "P"),
```

```
                as.integer(rep(4, 3)))
myc <- c(2343, 457, 4593, 9035, 3450, 3524,
         958, 1053, 8674, 3424, 463, 5105)
```

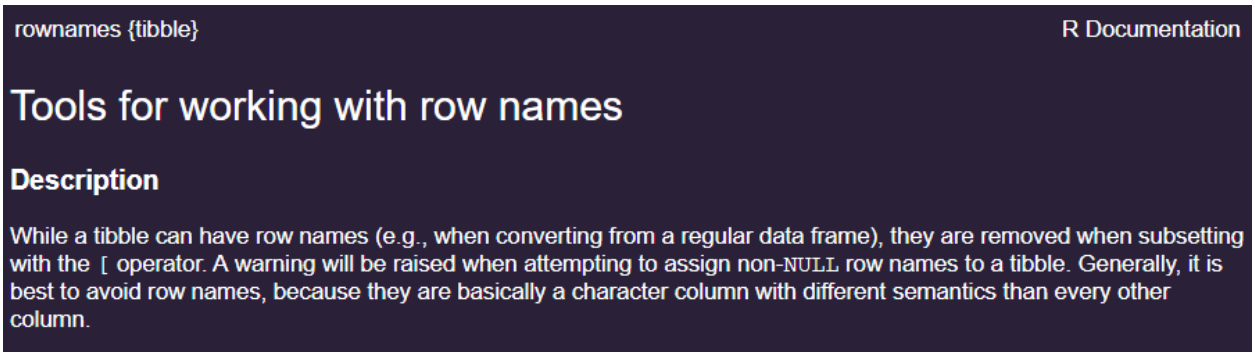
Put them together into a dataframe called `meta`

```
meta <- tibble(sex, stage, treatment, myc)

rm(sex, stage, treatment, myc)
```

Use the `rownames()` function to assign row names to the dataframe (Hint: you can type out the row names as a vector, or if you want the process to go gaster try exploring the paste() function).

The tibbles docs suggest that row names are bad practice:



So we will make an ordinary column called `rownames` instead.

```
meta <- meta %>%
  mutate(rownames = paste0("sample", 1:12)) %>%
  select(rownames, everything())
```

So our final `meta` tibble looks like

```
meta
```

```
## # A tibble: 12 x 5
##    rownames sex   stage treatment   myc
##    <chr>    <chr> <chr> <chr>     <dbl>
##  1 sample1  M     I     A          2343
##  2 sample2  F     II    A           457
##  3 sample3  M     III   A          4593
##  4 sample4  F     I     A          9035
##  5 sample5  M     II    B          3450
##  6 sample6  F     III   B          3524
##  7 sample7  M     I     B           958
##  8 sample8  F     II    B          1053
##  9 sample9  M     III   P          8674
## 10 sample10 F     I     P          3424
## 11 sample11 M     II    P           463
## 12 sample12 F     III   P          5105
```

Subsetting vectors/factors and dataframes

Using the `meta` dataframe from above, write out the R code you would use to perform the following operations (questions **DO NOT** buld upon each other):

return only the `treatment` and `sex` columns using `[]`:

```
meta[, c("treatment", "sex")]
```

```
## # A tibble: 12 x 2
##    treatment sex
##    <chr>     <chr>
##  1 A         M
##  2 A         F
##  3 A         M
##  4 A         F
##  5 B         M
##  6 B         F
##  7 B         M
##  8 B         F
##  9 P         M
## 10 P         F
## 11 P         M
## 12 P         F
```

return the `treatment` values for samples $5, 7, 9,$ and 10 using `[]`:

```
meta[c(5, 7, 9, 10), c("rownames", "treatment")]
```

```
## # A tibble: 4 x 2
##   rownames treatment
##   <chr>    <chr>
## 1 sample5  B
## 2 sample7  B
## 3 sample9  P
## 4 sample10 P
```

use `subset()` to return all data for those samples receiving treatment `P`:

```
subset(meta, treatment == "P")
```

```
## # A tibble: 4 x 5
##   rownames sex   stage treatment   myc
##   <chr>    <chr> <chr> <chr>     <dbl>
## 1 sample9  M     III   P          8674
## 2 sample10 F     I     P          3424
## 3 sample11 M     II    P           463
## 4 sample12 F     III   P          5105
```

use `filter()`/`select()` to return only the `stage` and `treatment` columns for those samples `myc` > 5000:

```
meta %>%
  select(rownames, stage, treatment, myc) %>%
```

```
  filter(myc > 5000) %>%
  select(rownames, stage, treatment)
```

```
## # A tibble: 3 x 3
##   rownames stage treatment
##   <chr>    <chr> <chr>
## 1 sample4  I     A
## 2 sample9  III   P
## 3 sample12 III   P
```

remove the `treatment` column from the dataset using `[]`:

```
treatment_index <- which(colnames(meta) == "treatment")
meta[, -treatment_index]
```

```
## # A tibble: 12 x 4
##    rownames sex   stage   myc
##    <chr>    <chr> <chr> <dbl>
##  1 sample1  M     I      2343
##  2 sample2  F     II      457
##  3 sample3  M     III    4593
##  4 sample4  F     I      9035
##  5 sample5  M     II     3450
##  6 sample6  F     III    3524
##  7 sample7  M     I       958
##  8 sample8  F     II     1053
##  9 sample9  M     III    8674
## 10 sample10 F     I      3424
## 11 sample11 M     II      463
## 12 sample12 F     III    5105
```

remove samples $7, 8,$ and 9 from the dataset using `[]`:

```
indices <- which(deframe(meta[, "rownames"]) %in% paste0("sample", 7:9))
meta[-indices, ]
```

```
## # A tibble: 9 x 5
##   rownames sex   stage treatment   myc
##   <chr>    <chr> <chr> <chr>     <dbl>
## 1 sample1  M     I     A          2343
## 2 sample2  F     II    A           457
## 3 sample3  M     III   A          4593
## 4 sample4  F     I     A          9035
## 5 sample5  M     II    B          3450
## 6 sample6  F     III   B          3524
## 7 sample10 F     I     P          3424
## 8 sample11 M     II    P           463
## 9 sample12 F     III   P          5105
```

keep only samples $1 - 6$ using `[]`:

```
indices <- which(deframe(meta[, "rownames"]) %in% paste0("sample", 1:6))
meta[indices, ]
```

```
## # A tibble: 6 x 5
##   rownames sex   stage treatment   myc
##   <chr>    <chr> <chr> <chr>     <dbl>
## 1 sample1  M     I     A          2343
## 2 sample2  F     II    A           457
## 3 sample3  M     III   A          4593
## 4 sample4  F     I     A          9035
## 5 sample5  M     II    B          3450
## 6 sample6  F     III   B          3524
```

add a column called `pre_treatment` to the beginning of the dataframe with values `T, F, F, F, T, T, F, T, F, F, T, T` (Hint: use `cbind()`):

```
pre_treatment <- c("T", "F", "F", "F", "T", "T",
                   "F", "T", "F", "F", "T", "T")
cbind(meta[, "rownames"], pre_treatment, meta[, -1])
```

```
##    rownames pre_treatment sex stage treatment  myc
## 1   sample1             T   M     I         A 2343
## 2   sample2             F   F    II         A  457
## 3   sample3             F   M   III         A 4593
## 4   sample4             F   F     I         A 9035
## 5   sample5             T   M    II         B 3450
## 6   sample6             T   F   III         B 3524
## 7   sample7             F   M     I         B  958
## 8   sample8             T   F    II         B 1053
## 9   sample9             F   M   III         P 8674
## 10 sample10             F   F     I         P 3424
## 11 sample11             T   M    II         P  463
## 12 sample12             T   F   III         P 5105
```

change the names of the columns to `A`, `B`, `C`, `D`:

```
original_col <- colnames(meta)
colnames(meta)[-1] <- c("A", "B", "C", "D")
meta
```

```
## # A tibble: 12 x 5
##    rownames A     B     C         D
##    <chr>    <chr> <chr> <chr> <dbl>
##  1 sample1  M     I     A      2343
##  2 sample2  F     II    A       457
##  3 sample3  M     III   A      4593
##  4 sample4  F     I     A      9035
##  5 sample5  M     II    B      3450
##  6 sample6  F     III   B      3524
##  7 sample7  M     I     B       958
##  8 sample8  F     II    B      1053
##  9 sample9  M     III   P      8674
## 10 sample10 F     I     P      3424
## 11 sample11 M     II    P       463
## 12 sample12 F     III   P      5105
```