Practical 1: Solutions

Jumping Rivers

First we must load the **tidyverse**

```
library("tidyverse")
```

Question 1 - lists

For this question, there is no **purrr**. It's a few questions to get you used to lists.

a) Create a list that contains a numeric, a logical, a character, a vector.

b) Name the elements of your list

c) The following code will load a list called `toy` into your global environment

```
data(toy, package = "jrTidyverse2")
```

Make sure to take a look at `toy` before you dive in. How many elements are in toy?

```
length(toy)
## [1] 5
```

d) Add an extra element called `z` onto `toy`. `z` can be whatever object you like.

```
toy$z = "fun with lists"
# or
toy[["z"]] = "fun with lists"
# or
toy["z"] = "fun with lists"
```

e) What is the average of the element `d` within `toy`?

```
mean(toy$d)
## [1] 6
```

f) What is the average of the column `f` within element `e` of `toy`?

```
mean(toy$e$f)
## [1] 0.5948947
```

g) What is the average of the column `f` in the element `e`, where the values of the column `g` are `"a"`?

```
toy$e %>%
  filter(g == "a") %>%
  summarise(mean(f))
## # A tibble: 1 x 1
##   `mean(f)`
##       <dbl>
## 1     0.112
```

Question 2 - purrr beginnings

a) If we have a vector `x`, we can square root it using the `sqrt()` function

```
x = c(1,4,9,25)
sqrt(x)
## [1] 1 2 3 5
```

Can you do the same but using the `map` functions? Make sure your output is a vector.

```
map_dbl(x, sqrt)
## [1] 1 2 3 5
```

Question 3 - Happiness

Now we're going to look at a list containing information such as happiness and economy rankings for countries around the globe in the years 2015, 2016 and 2017.

```
data(happiness, package = "jrTidyverse2")
```

a) Using `str()` to investigate the list and determine:

- How long is the list?

```
str(happiness, max.level = 0)
## List of 146
# 146 element in the list
```

- Is the list a recursive list?

```
str(happiness, max.level = 1, list.len = 3)
## List of 146
##  $ :List of 12
##  $ :List of 12
##  $ :List of 12
##   [list output truncated]
# Yes, recursive list
```

- How many countries does the list contain information on?
- For each country, how many piece of information do we have?

```
str(happiness, max.level =2, list.len = 3)
## List of 146
##  $ :List of 12
##   ..$ Country                    : chr "Switzerland"
##   ..$ Region                     : chr "Western Europe"
##   ..$ Year                       : num [1:3] 2015 2016 2017
##   .. [list output truncated]
##  $ :List of 12
##   ..$ Country                    : chr "Iceland"
##   ..$ Region                     : chr "Western Europe"
##   ..$ Year                       : num [1:3] 2015 2016 2017
##   .. [list output truncated]
##  $ :List of 12
##   ..$ Country                    : chr "Denmark"
##   ..$ Region                     : chr "Western Europe"
##   ..$ Year                       : num [1:3] 2015 2016 2017
##   .. [list output truncated]
##   [list output truncated]
# Each element of the list is another list representing a country.
```

```
# Each list contains elements representative of happiness information
# on that country for three successive years. Therefore there is
# 146 countries and 12 pieces of information on each.
```

b) Extract the name of each country using the `map` functions. To make it a bit easier to read, return the output as a character vector.

```
country_names = map_chr(happiness, "Country")
```

c) Try `names(happiness)`, what happens? Use the answer to `b)` to rename each element of the list after it's representative country.

```
names(happiness) = country_names
```

d) What has the UKs happiness rank been over the last 3 years? (You don't have to use **purrr** for this one.)

```
happiness[["United Kingdom"]]$`Happiness Rank`
## [1] 21 23 19
```

e) Which country has had the highest average happiness score?

```
happiness %>%
  map_dbl(~ mean(.x[["Happiness Score"]])) %>%
  sort(decreasing = TRUE) %>%
  head()
## Switzerland     Denmark     Iceland      Norway     Finland      Canada
##    7.530000    7.525000    7.522000    7.519000    7.429333    7.382333
```

f) Which country has had the largest increase in happiness score from 2015 - 2017?

```
happiness %>%
  map_dbl( ~ .x$`Happiness Score`[3] - .x$`Happiness Score`[1]) %>%
  sort(decreasing = TRUE) %>%
  head()
##    Latvia   Romania      Togo   Senegal     Gabon     Egypt
## 0.7519999 0.7009998 0.6559999 0.6309998 0.5690002 0.5410001
```

g) Which country has had the largest decrease in life expectency?

```
map_dbl(happiness, ~.x$`Health (Life Expectancy)`[3] - .x$`Health (Life Expectancy)`[1]) %>%
  sort() %>%
  head()
##        Syria        Libya     Cambodia    Indonesia     Zimbabwe
##   -0.2213967   -0.1836310   -0.1813566   -0.1455843   -0.1379862
## Saudi Arabia
##   -0.1301017
```

Question 4 - Happiness - Advanced

The following two questions are intended to be a bit trickier. Don't worry if you get stuck on them. Just ask!

a) How many countries economies have shrunk from 2015 - 2017?

```
map_lgl(happiness, ~(.x$`Economy (GDP per Capita)`[3] - .x$`Economy (GDP per Capita)`[1]) < 0) %>%
  sum()
## [1] 1
```

b) On average, which region of the world is the most "generous"?
Hint: store the region for each country in a vector, combine it into a data frame with the average generosity score for each country then use **dplyr**.

```
region = happiness %>%
  map_chr("Region")

av_gen = happiness %>%
  map_dbl(~mean(.x$Generosity))

region_gen = tibble(region = region, av_gen = av_gen)

region_gen %>%
  group_by(region) %>%
  summarise(av_region_gen = mean(av_gen)) %>%
  arrange(av_region_gen)
## # A tibble: 10 x 2
##    region                          av_region_gen
##    <chr>                                   <dbl>
##  1 Central and Eastern Europe              0.170
##  2 Eastern Asia                            0.174
##  3 Middle East and Northern Africa         0.192
##  4 Latin America and Caribbean             0.213
##  5 Sub-Saharan Africa                      0.222
##  6 Western Europe                          0.303
##  7 Southern Asia                           0.342
##  8 North America                           0.424
##  9 Southeastern Asia                       0.439
## 10 Australia and New Zealand               0.476
```

c) Using **ggplot2** and `geom_col()`, plot the answer to b) as a bar chart

```
region_gen %>%
  group_by(region) %>%
  summarise(av_region_gen = mean(av_gen)) %>%
  arrange(av_region_gen)%>%
  ggplot() +
  geom_col(aes(x = region, y = av_region_gen)) +
  coord_flip()
```