Reem Alsharabi S20106353 CS3072 Lab 2

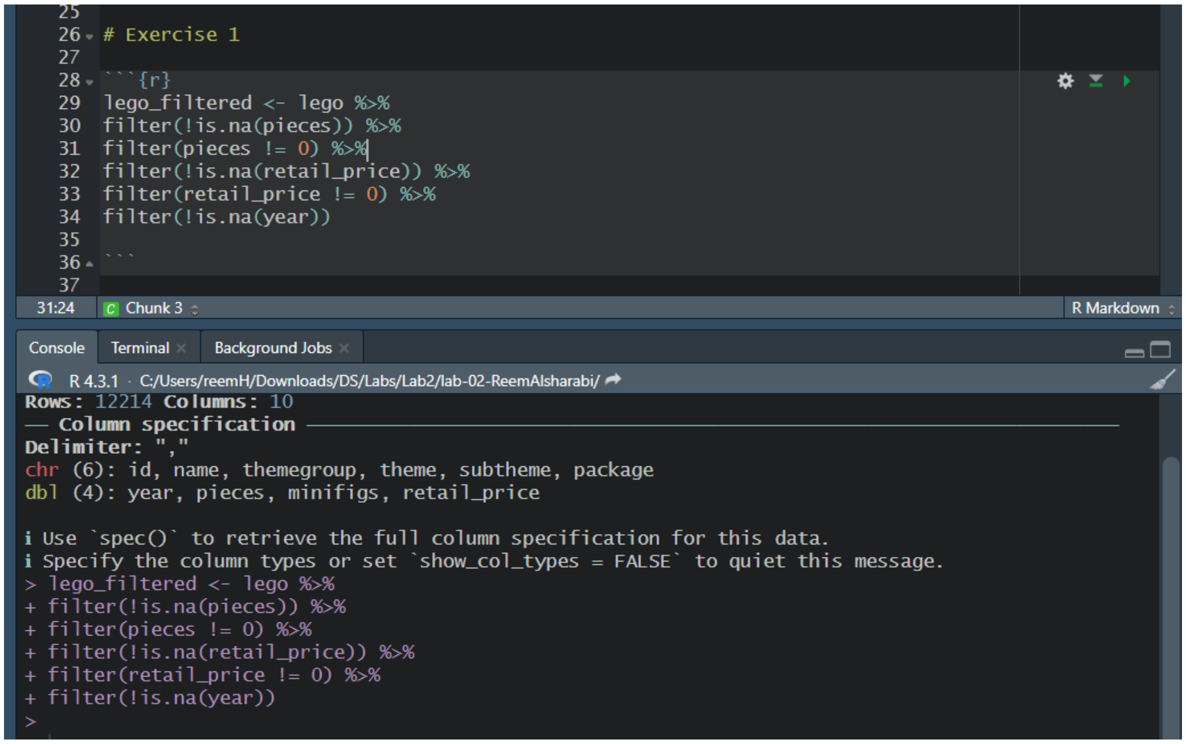
## Data Wrangling

**Exercises**

1. Some sets have missing information for retail_price or pieces or both. This could be because the sets are free (giveaways), they aren't traditional lego sets (comic books, etc) or just because the information is missing. Filter the lego dataset based on the specifications below and save the result as lego using <-. Hence, you will overwrite the original lego object. In addition, describe the implications of removing these sets.

   Your new lego tibble (data frame) should have:

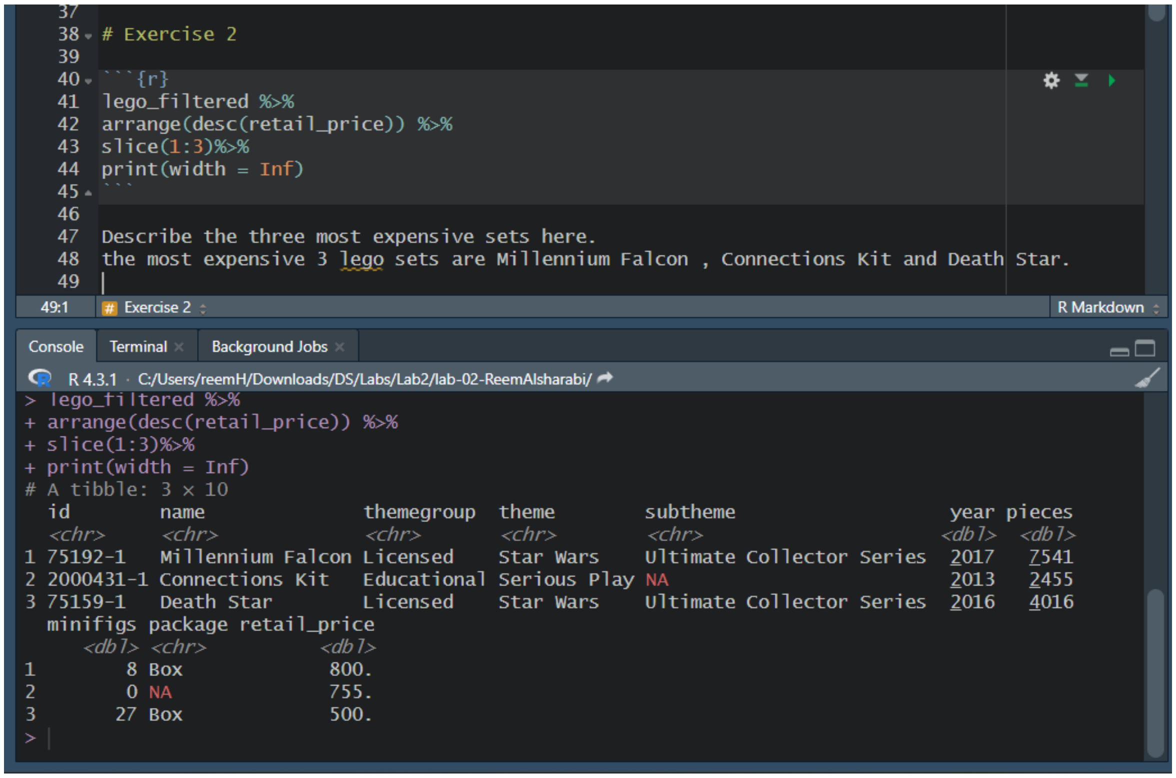
   - no missing pieces
   - only contain sets with a nonzero number of pieces
   - no missing retail_price
   - only contain sets with a nonzero retail_price
   - no missing year



~~~
# Exercise 1

```{r}
lego_filtered <- lego %>%
filter(!is.na(pieces)) %>%
filter(pieces != 0) %>%
filter(!is.na(retail_price)) %>%
filter(retail_price != 0) %>%
filter(!is.na(year))

```
~~~

```
R 4.3.1 · C:/Users/reemH/Downloads/DS/Labs/Lab2/lab-02-ReemAlsharabi/
Rows: 12214 Columns: 10
── Column specification ────────────────────────────────────────
Delimiter: ","
chr (6): id, name, themegroup, theme, subtheme, package
dbl (4): year, pieces, minifigs, retail_price

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
> lego_filtered <- lego %>%
+ filter(!is.na(pieces)) %>%
+ filter(pieces != 0) %>%
+ filter(!is.na(retail_price)) %>%
+ filter(retail_price != 0) %>%
+ filter(!is.na(year))
>
```

2. Arrange the dataset in descending order of *retail_price* and print the first three rows. Report in words the names of the three most expensive lego sets, their prices, and how many pieces each has.



~~~
# Exercise 2

```{r}
lego_filtered %>%
arrange(desc(retail_price)) %>%
slice(1:3)%>%
print(width = Inf)
```

Describe the three most expensive sets here.
the most expensive 3 lego sets are Millennium Falcon , Connections Kit and Death Star.
~~~

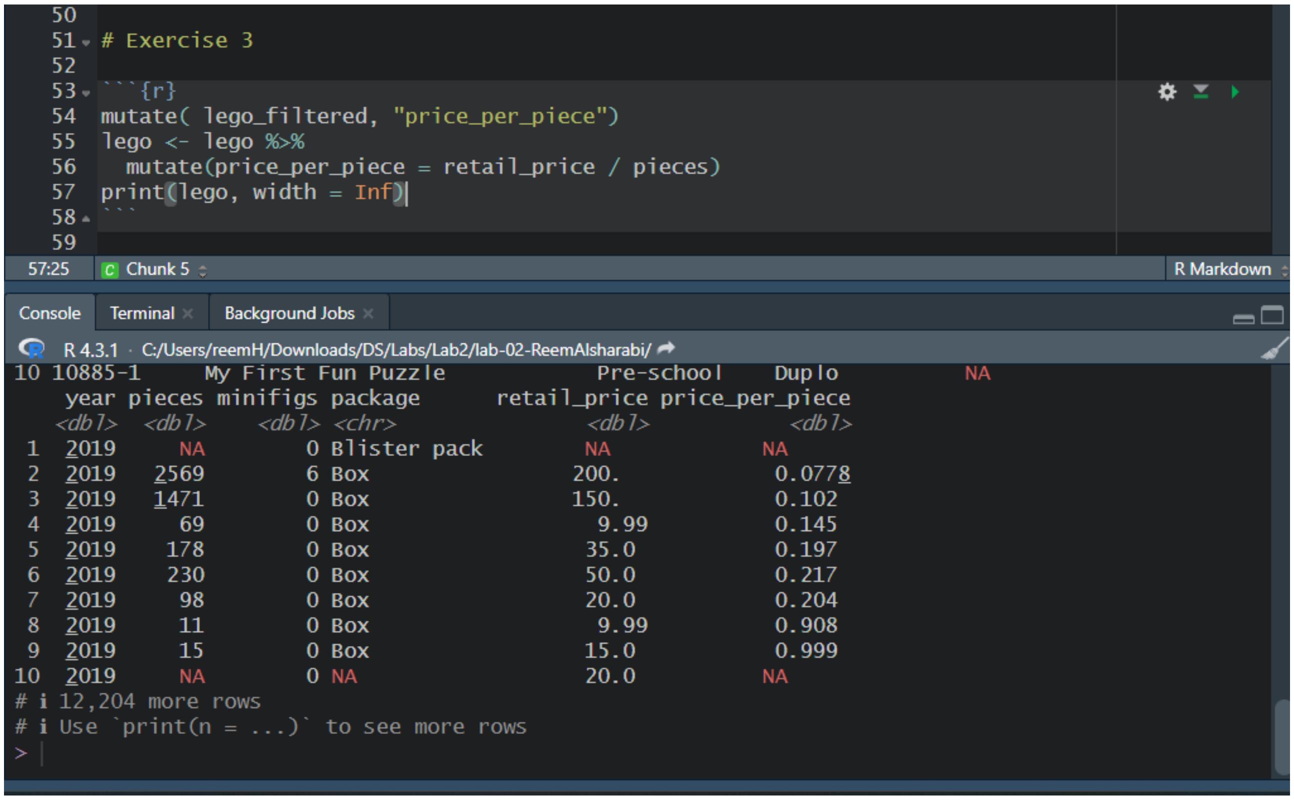
```
R 4.3.1 · C:/Users/reemH/Downloads/DS/Labs/Lab2/lab-02-ReemAlsharabi/
> lego_filtered %>%
+ arrange(desc(retail_price)) %>%
+ slice(1:3)%>%
+ print(width = Inf)
# A tibble: 3 × 10
  id        name              themegroup   theme        subtheme                   year pieces
  <chr>     <chr>             <chr>        <chr>        <chr>                     <dbl>  <dbl>
1 75192-1   Millennium Falcon Licensed     Star Wars    Ultimate Collector Series  2017   7541
2 2000431-1 Connections Kit   Educational  Serious Play NA                         2013   2455
3 75159-1   Death Star        Licensed     Star Wars    Ultimate Collector Series  2016   4016
  minifigs package retail_price
     <dbl> <chr>          <dbl>
1        8 Box             800.
2        0 NA              755.
3       27 Box             500.
>
```

- Set: Millennium Falcon
  Price: $800
  Pieces: 7,541

- Set: Connections Kit
  Price: $755
  Pieces: 2,455

- Set: Death Star
  Price: $500
  Pieces: 4,016

3. It appears that the most expensive sets generally have more pieces. Use *mutate()* to create a new variable *price_per_piece*, representing the price in dollars per piece for each of the sets. Save the result as *lego*. Hence, you will overwrite the current *lego* object.



~~~
# Exercise 3

```{r}
mutate( lego_filtered, "price_per_piece")
lego <- lego %>%
  mutate(price_per_piece = retail_price / pieces)
print(lego, width = Inf)
```
~~~

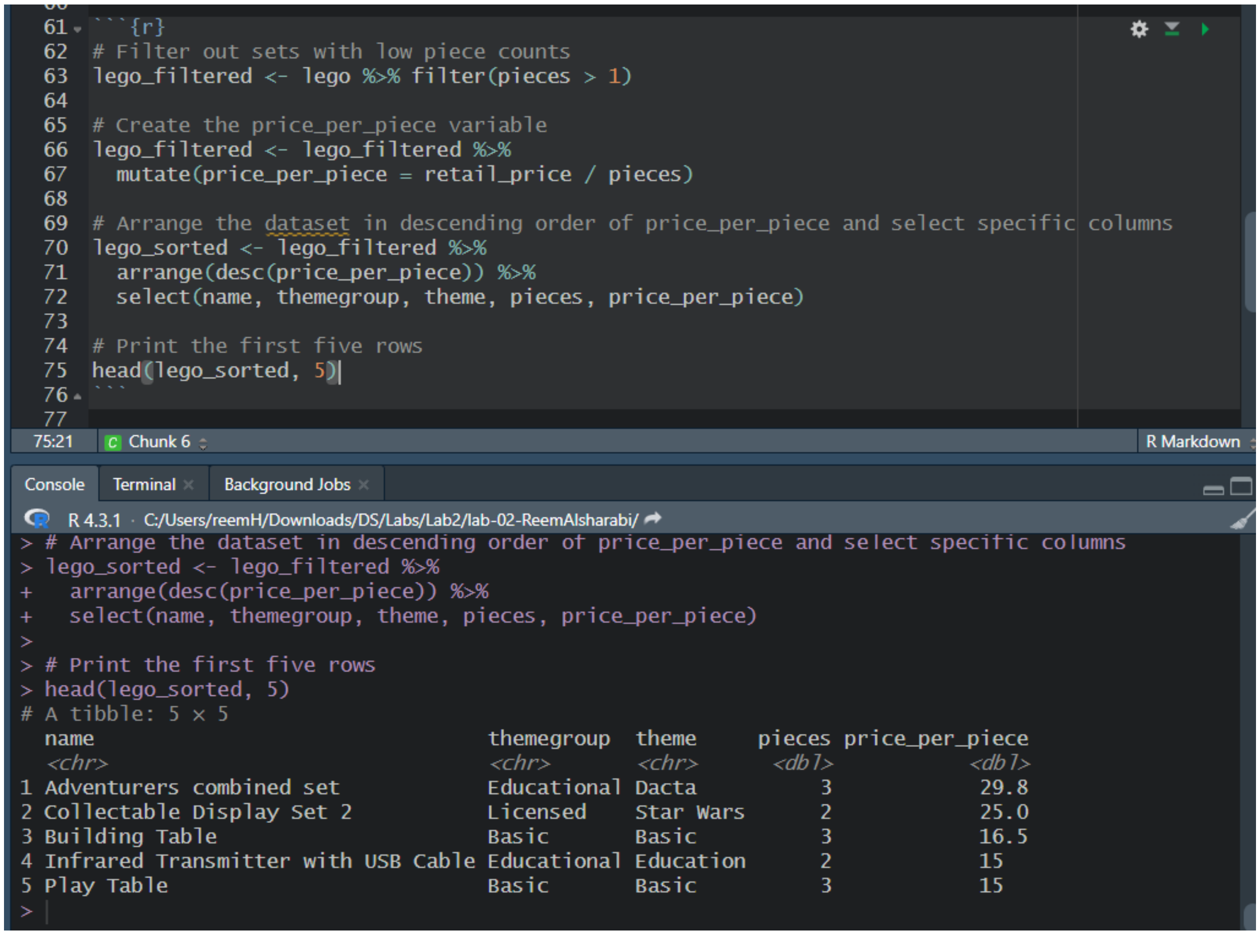
57:25 Chunk 5 R Markdown

Console Terminal Background Jobs

R 4.3.1 · C:/Users/reemH/Downloads/DS/Labs/Lab2/lab-02-ReemAlsharabi/

```
10 10885-1      My First Fun Puzzle             Pre-school     Duplo              NA
    year pieces minifigs package      retail_price price_per_piece
   <dbl>  <dbl>    <dbl> <chr>               <dbl>           <dbl>
 1  2019     NA        0 Blister pack           NA          NA
 2  2019   2569        6 Box                  200.           0.0778
 3  2019   1471        0 Box                  150.           0.102
 4  2019     69        0 Box                    9.99         0.145
 5  2019    178        0 Box                   35.0          0.197
 6  2019    230        0 Box                   50.0          0.217
 7  2019     98        0 Box                   20.0          0.204
 8  2019     11        0 Box                    9.99         0.908
 9  2019     15        0 Box                   15.0          0.999
10  2019     NA        0 NA                    20.0          NA
# ℹ 12,204 more rows
# ℹ Use `print(n = ...)` to see more rows
>
```

4. Arrange the *lego* dataset in descending order of *price_per_piece* and return only the columns *name, themegroup, theme, pieces, price_per_piece*, and the first five rows. What do you notice about these sets?



~~~
```{r}
# Filter out sets with low piece counts
lego_filtered <- lego %>% filter(pieces > 1)

# Create the price_per_piece variable
lego_filtered <- lego_filtered %>%
  mutate(price_per_piece = retail_price / pieces)

# Arrange the dataset in descending order of price_per_piece and select specific columns
lego_sorted <- lego_filtered %>%
  arrange(desc(price_per_piece)) %>%
  select(name, themegroup, theme, pieces, price_per_piece)

# Print the first five rows
head(lego_sorted, 5)
~~~

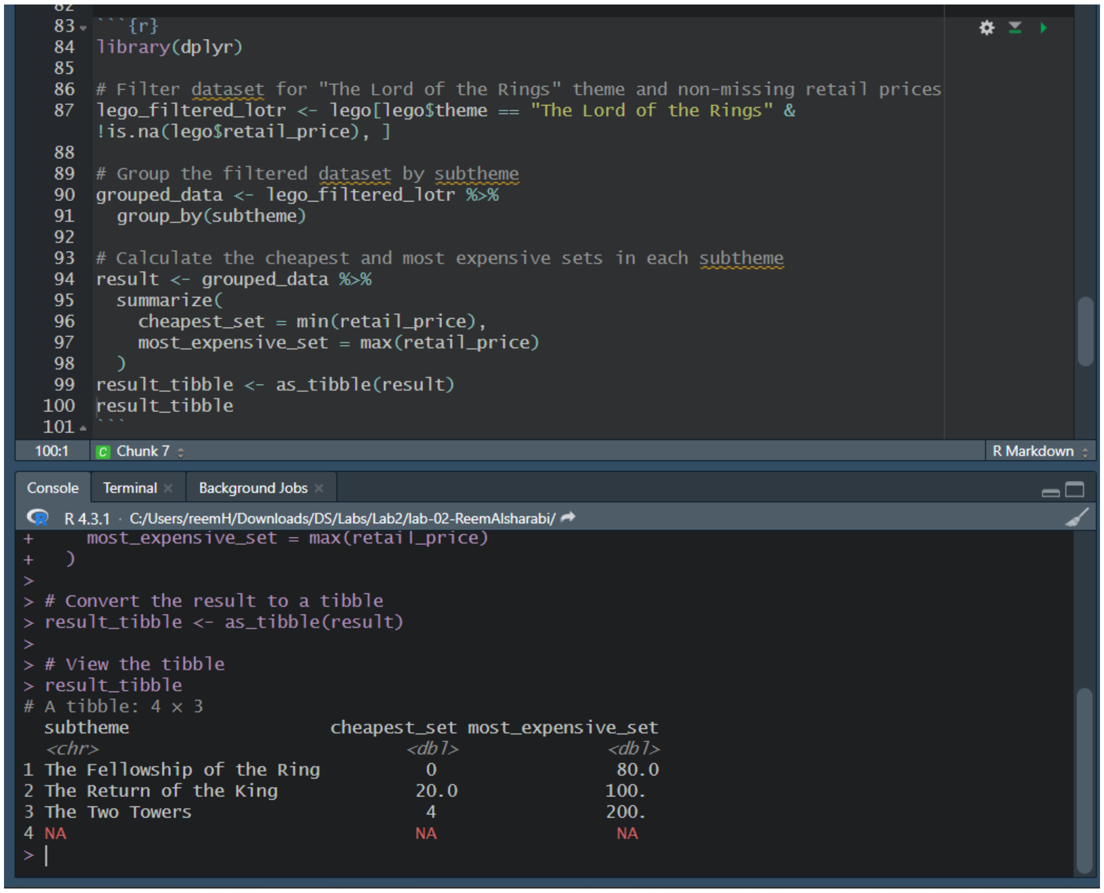
```
```
75:21 Chunk 6 R Markdown

```
R 4.3.1 · C:/Users/reemH/Downloads/DS/Labs/Lab2/lab-02-ReemAlsharabi/
> # Arrange the dataset in descending order of price_per_piece and select specific columns
> lego_sorted <- lego_filtered %>%
+   arrange(desc(price_per_piece)) %>%
+   select(name, themegroup, theme, pieces, price_per_piece)
>
> # Print the first five rows
> head(lego_sorted, 5)
# A tibble: 5 × 5
  name                                themegroup  theme     pieces price_per_piece
  <chr>                               <chr>       <chr>      <dbl>           <dbl>
1 Adventurers combined set            Educational Dacta          3            29.8
2 Collectable Display Set 2           Licensed    Star Wars      2            25.0
3 Building Table                      Basic       Basic          3            16.5
4 Infrared Transmitter with USB Cable Educational Education      2            15
5 Play Table                          Basic       Basic          3            15
>
```

- The sets generally have a small number of pieces, ranging from 2 to 3.
- Despite their small piece counts, these sets have a higher price per piece, ranging from $15 to $29.8.
- The sets cover diverse themes, including Educational, Licensed (Star Wars), Basic, and Education.
- These sets may possess unique features, specialized designs, educational value, or collectible qualities that justify their higher price per piece.

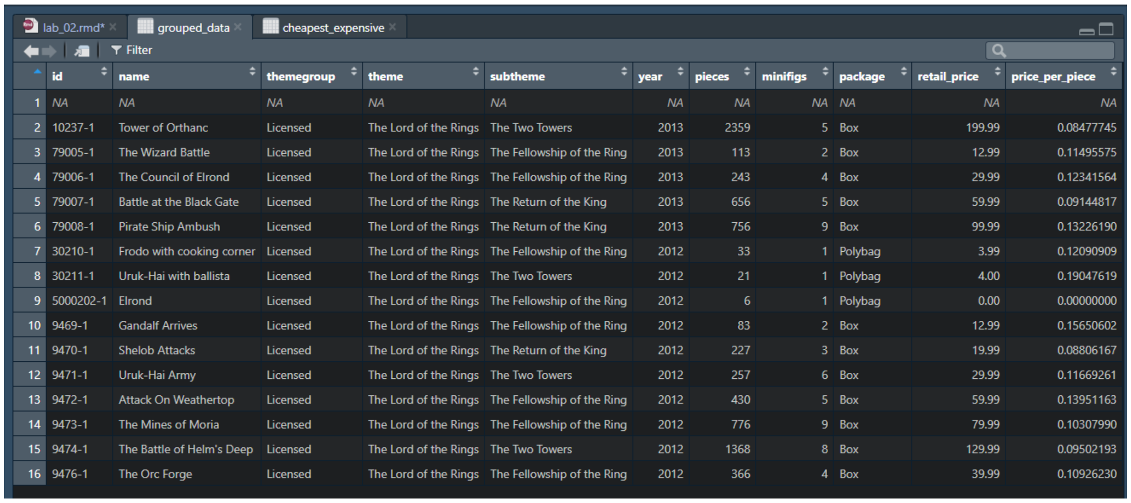
Overall, these observations suggest that these sets with small piece counts and higher prices per piece offer distinctive qualities or cater to specific purposes, making them stand out within the LEGO product range.

5. Return a tibble containing the cheapest and most expensive lego sets (based on *retail_price*) in each subtheme, considering only sets with the Lord of the Rings theme.



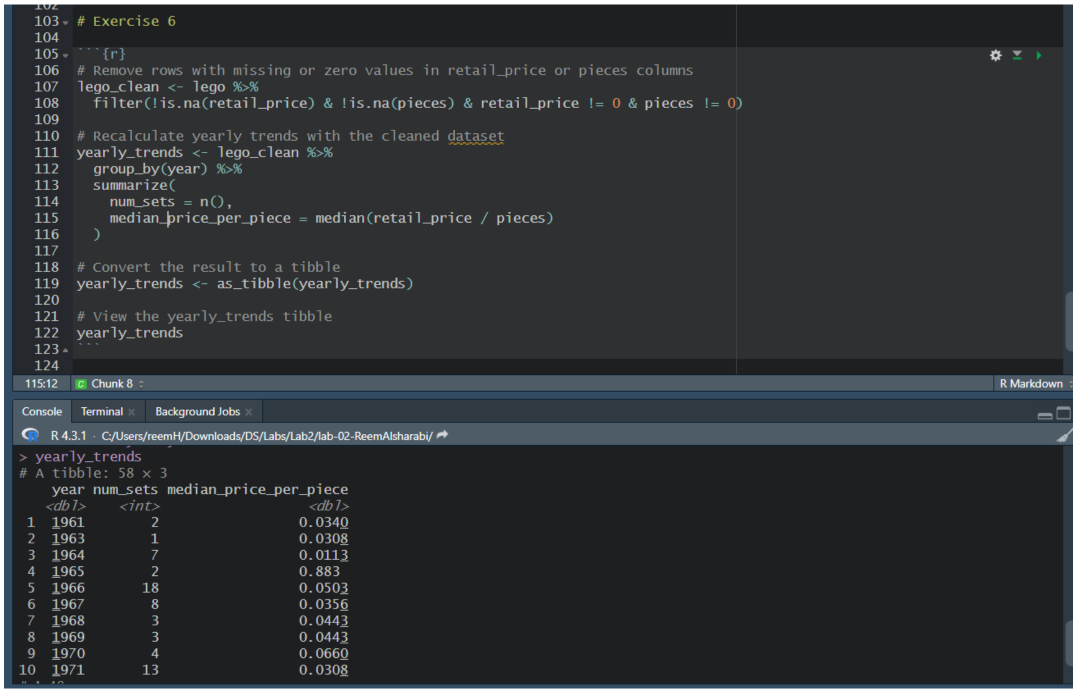
~~~
```{r}
library(dplyr)

# Filter dataset for "The Lord of the Rings" theme and non-missing retail prices
lego_filtered_lotr <- lego[lego$theme == "The Lord of the Rings" &
!is.na(lego$retail_price), ]

# Group the filtered dataset by subtheme
grouped_data <- lego_filtered_lotr %>%
  group_by(subtheme)

# Calculate the cheapest and most expensive sets in each subtheme
result <- grouped_data %>%
  summarize(
    cheapest_set = min(retail_price),
    most_expensive_set = max(retail_price)
  )
result_tibble <- as_tibble(result)
result_tibble
```
~~~

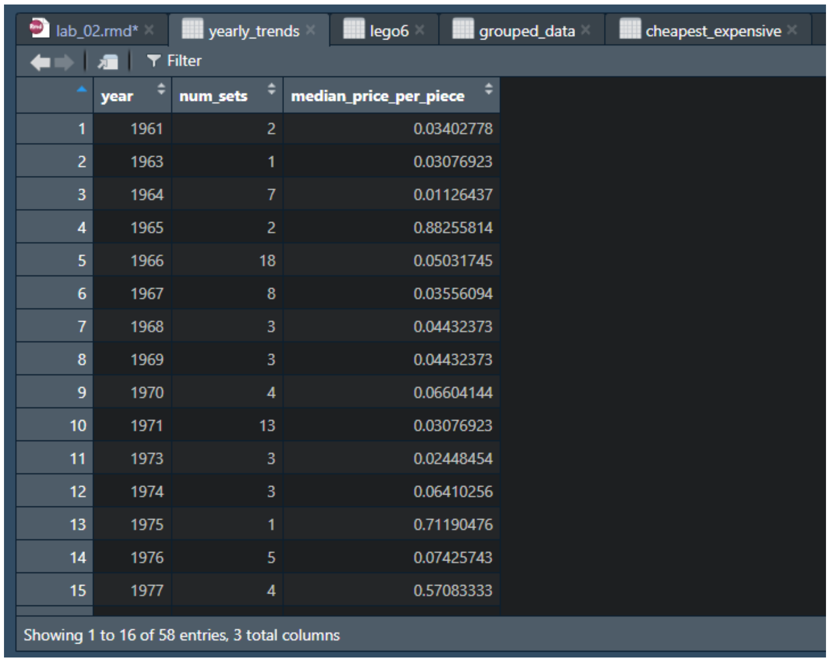
```
R 4.3.1 · C:/Users/reemH/Downloads/DS/Labs/Lab2/lab-02-ReemAlsharabi/
+     most_expensive_set = max(retail_price)
+   )
>
> # Convert the result to a tibble
> result_tibble <- as_tibble(result)
>
> # View the tibble
> result_tibble
# A tibble: 4 × 3
  subtheme                    cheapest_set most_expensive_set
  <chr>                              <dbl>              <dbl>
1 The Fellowship of the Ring             0               80.0
2 The Return of the King              20.0              100.
3 The Two Towers                         4              200.
4 NA                                    NA                NA
> |
```



| | id | name | themegroup | theme | subtheme | year | pieces | minifigs | package | retail_price | price_per_piece |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | NA | NA | NA | NA | NA | NA | NA | NA | NA | NA | NA |
| 2 | 10237-1 | Tower of Orthanc | Licensed | The Lord of the Rings | The Two Towers | 2013 | 2359 | 5 | Box | 199.99 | 0.08477745 |
| 3 | 79005-1 | The Wizard Battle | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2013 | 113 | 2 | Box | 12.99 | 0.11495575 |
| 4 | 79006-1 | The Council of Elrond | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2013 | 243 | 4 | Box | 29.99 | 0.12341564 |
| 5 | 79007-1 | Battle at the Black Gate | Licensed | The Lord of the Rings | The Return of the King | 2013 | 656 | 5 | Box | 59.99 | 0.09144817 |
| 6 | 79008-1 | Pirate Ship Ambush | Licensed | The Lord of the Rings | The Return of the King | 2013 | 756 | 9 | Box | 99.99 | 0.13226190 |
| 7 | 30210-1 | Frodo with cooking corner | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2012 | 33 | 1 | Polybag | 3.99 | 0.12090909 |
| 8 | 30211-1 | Uruk-Hai with ballista | Licensed | The Lord of the Rings | The Two Towers | 2012 | 21 | 1 | Polybag | 4.00 | 0.19047619 |
| 9 | 5000202-1 | Elrond | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2012 | 6 | 1 | Polybag | 0.00 | 0.00000000 |
| 10 | 9469-1 | Gandalf Arrives | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2012 | 83 | 2 | Box | 12.99 | 0.15650602 |
| 11 | 9470-1 | Shelob Attacks | Licensed | The Lord of the Rings | The Return of the King | 2012 | 227 | 3 | Box | 19.99 | 0.08806167 |
| 12 | 9471-1 | Uruk-Hai Army | Licensed | The Lord of the Rings | The Two Towers | 2012 | 257 | 6 | Box | 29.99 | 0.11669261 |
| 13 | 9472-1 | Attack On Weathertop | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2012 | 430 | 5 | Box | 59.99 | 0.13951163 |
| 14 | 9473-1 | The Mines of Moria | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2012 | 776 | 9 | Box | 79.99 | 0.10307990 |
| 15 | 9474-1 | The Battle of Helm's Deep | Licensed | The Lord of the Rings | The Two Towers | 2012 | 1368 | 8 | Box | 129.99 | 0.09502193 |
| 16 | 9476-1 | The Orc Forge | Licensed | The Lord of the Rings | The Fellowship of the Ring | 2012 | 366 | 4 | Box | 39.99 | 0.10926230 |

6. Use *group_by()* and *summarize()* to create a new tibble with one row for each year, and columns for the year, the number of sets released in that year, and the median price per piece for sets from that year. Save this resulting tibble as an object named *yearly_trends*.



~~~r
# Exercise 6

```{r}
# Remove rows with missing or zero values in retail_price or pieces columns
lego_clean <- lego %>%
  filter(!is.na(retail_price) & !is.na(pieces) & retail_price != 0 & pieces != 0)

# Recalculate yearly trends with the cleaned dataset
yearly_trends <- lego_clean %>%
  group_by(year) %>%
  summarize(
    num_sets = n(),
    median_price_per_piece = median(retail_price / pieces)
  )

# Convert the result to a tibble
yearly_trends <- as_tibble(yearly_trends)

# View the yearly_trends tibble
yearly_trends
~~~
```

```
> yearly_trends
# A tibble: 58 × 3
    year num_sets median_price_per_piece
   <dbl>    <int>                  <dbl>
 1  1961        2                 0.0340
 2  1963        1                 0.0308
 3  1964        7                 0.0113
 4  1965        2                 0.883
 5  1966       18                 0.0503
 6  1967        8                 0.0356
 7  1968        3                 0.0443
 8  1969        3                 0.0443
 9  1970        4                 0.0660
10  1971       13                 0.0308
```



| | year | num_sets | median_price_per_piece |
|---|---|---|---|
| 1 | 1961 | 2 | 0.03402778 |
| 2 | 1963 | 1 | 0.03076923 |
| 3 | 1964 | 7 | 0.01126437 |
| 4 | 1965 | 2 | 0.88255814 |
| 5 | 1966 | 18 | 0.05031745 |
| 6 | 1967 | 8 | 0.03556094 |
| 7 | 1968 | 3 | 0.04432373 |
| 8 | 1969 | 3 | 0.04432373 |
| 9 | 1970 | 4 | 0.06604144 |
| 10 | 1971 | 13 | 0.03076923 |
| 11 | 1973 | 3 | 0.02448454 |
| 12 | 1974 | 3 | 0.06410256 |
| 13 | 1975 | 1 | 0.71190476 |
| 14 | 1976 | 5 | 0.07425743 |
| 15 | 1977 | 4 | 0.57083333 |

Showing 1 to 16 of 58 entries, 3 total columns
```

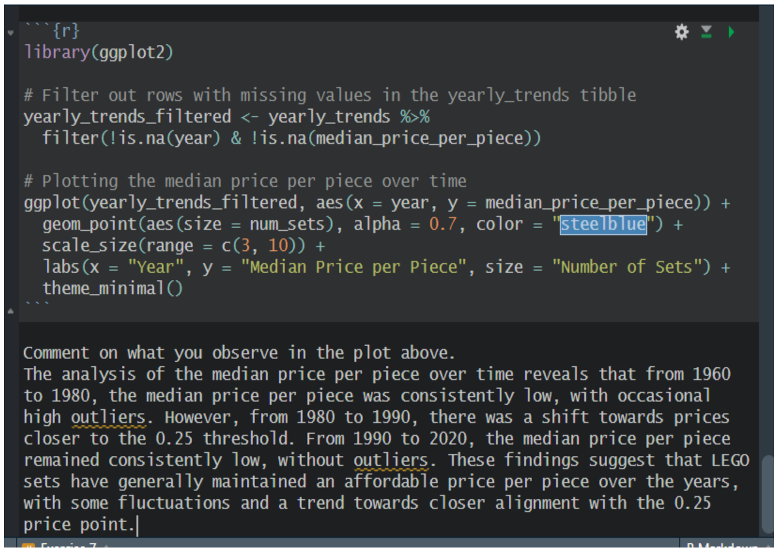
7. Create a plot of the median price per piece over time using the *yearly_trends* tibble. Size points according to the number of sets produced in that year. Adjust transparency, color, etc as appropriate and remember the principles of effective data visualization. Comment on what you observe.

   - From 1960 to 1980: The median price per piece was consistently low, below 0.25, with a few high outliers indicating occasional sets with higher prices per piece.

   - From 1980 to 1990: The median price per piece showed an increase and approached the 0.25 threshold, suggesting that the average price per piece became more aligned with this value during this decade.

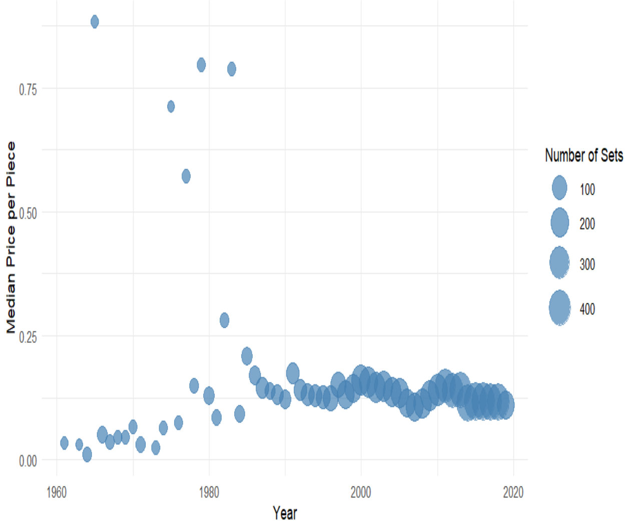
   - From 1990 to 2020: The median price per piece remained relatively low and did not exceed 0.25. There were no outliers during this period, indicating a consistent price per piece for LEGO sets without extremely high or low values.

These findings suggest that LEGO sets have generally maintained an affordable price per piece over the years, with some fluctuations and a trend toward closer alignment with the 0.25 price point.



~~~
```{r}
library(ggplot2)

# Filter out rows with missing values in the yearly_trends tibble
yearly_trends_filtered <- yearly_trends %>%
  filter(!is.na(year) & !is.na(median_price_per_piece))

# Plotting the median price per piece over time
ggplot(yearly_trends_filtered, aes(x = year, y = median_price_per_piece)) +
  geom_point(aes(size = num_sets), alpha = 0.7, color = "steelblue") +
  scale_size(range = c(3, 10)) +
  labs(x = "Year", y = "Median Price per Piece", size = "Number of Sets") +
  theme_minimal()
```
~~~

Comment on what you observe in the plot above.
The analysis of the median price per piece over time reveals that from 1960 to 1980, the median price per piece was consistently low, with occasional high outliers. However, from 1980 to 1990, there was a shift towards prices closer to the 0.25 threshold. From 1990 to 2020, the median price per piece remained consistently low, without outliers. These findings suggest that LEGO sets have generally maintained an affordable price per piece over the years, with some fluctuations and a trend towards closer alignment with the 0.25 price point.



## Submission

Knit to PDF to create a PDF document. Stage and commit all remaining changes, and push your work to GitHub. Make sure all files are updated on your GitHub repo. Only upload your PDF document to Blackboard. Before you submit the uploaded document, mark where each answer is to the exercises.