**Data engineering syntax assessment (14 pts total)**

**Question 1 (8 points)**

**Answer the questions only in Python and upload your code file (Jupyter Notebook or \*.py file) to a repo on github and share the link**

A U.S graduate school has students from Asia, Europe, and America. The students' continents of origin and colleges are stored in a data frame STUDENTS. You can assume that there may be duplicate records, as shown below:

| name | continent | college |

|--------|-----------|-------------|

| Andy | America | Lib Arts |

| Jose | Europe | Business |

| Dieu | Asia | Engineering |

| Chloe | America | Engineering |

| Jose | Europe | Business |

Please write R or Pandas code to create a new data frame COUNTS that counts the number of unique student records corresponding to each continent, college pair. Lay out the data in a 3x3 grid with colleges sorted reverse alphabetically on the vertical axis. The horizontal axis does not need to be sorted. Null values should be zero-filled. For the sample input, the desired output would be:

| continent | America | Asia | Europe |

| college |---------|------|--------|

| Lib Arts | 1 | 0 | 0 |

| Engineering | 1 | 1 | 0 |

| Business | 0 | 0 | 1 |

**A) Write R or Pandas code to de-dup STUDENTS (1pt):**

Remarks : We can reduce duplicated records by using ‘drop\_duplicates()’ of Pandas.

**B) Create a data frame COUNTS storing the count of students for each continent, college pair (2pts):**

Remarks : We can use ‘groupby’ for counting the number of students for each continent, college pair. Then, we can formulate dataframe from the pair and the number of students (counter) by using ‘to\_frame’ with argument ‘name =’. The argument becomes column name of counter.

**C) Pivot COUNTS so that continent values become column names (2pts)**

Remarks : We can pivot the continent values using ‘pivot\_table’ with argument ‘index =’, ‘columns =’, ‘values’, which are corresponding to component of dataframe.

**D) If applicable, remove any unnecessary index levels created in part C (1pts):**

Remarks : Since we used ‘pivot\_table’, there exists no redundant index level.

**E) Zero-fill missing values in COUNTS (1pt):**

Remarks : We can replace missing values by using ‘fillna’ with the desired substitutes. Argument 0 means the missing value will be filled with zero.

**F) Sort COUNTS reverse alphabetically by college name (1pt):**

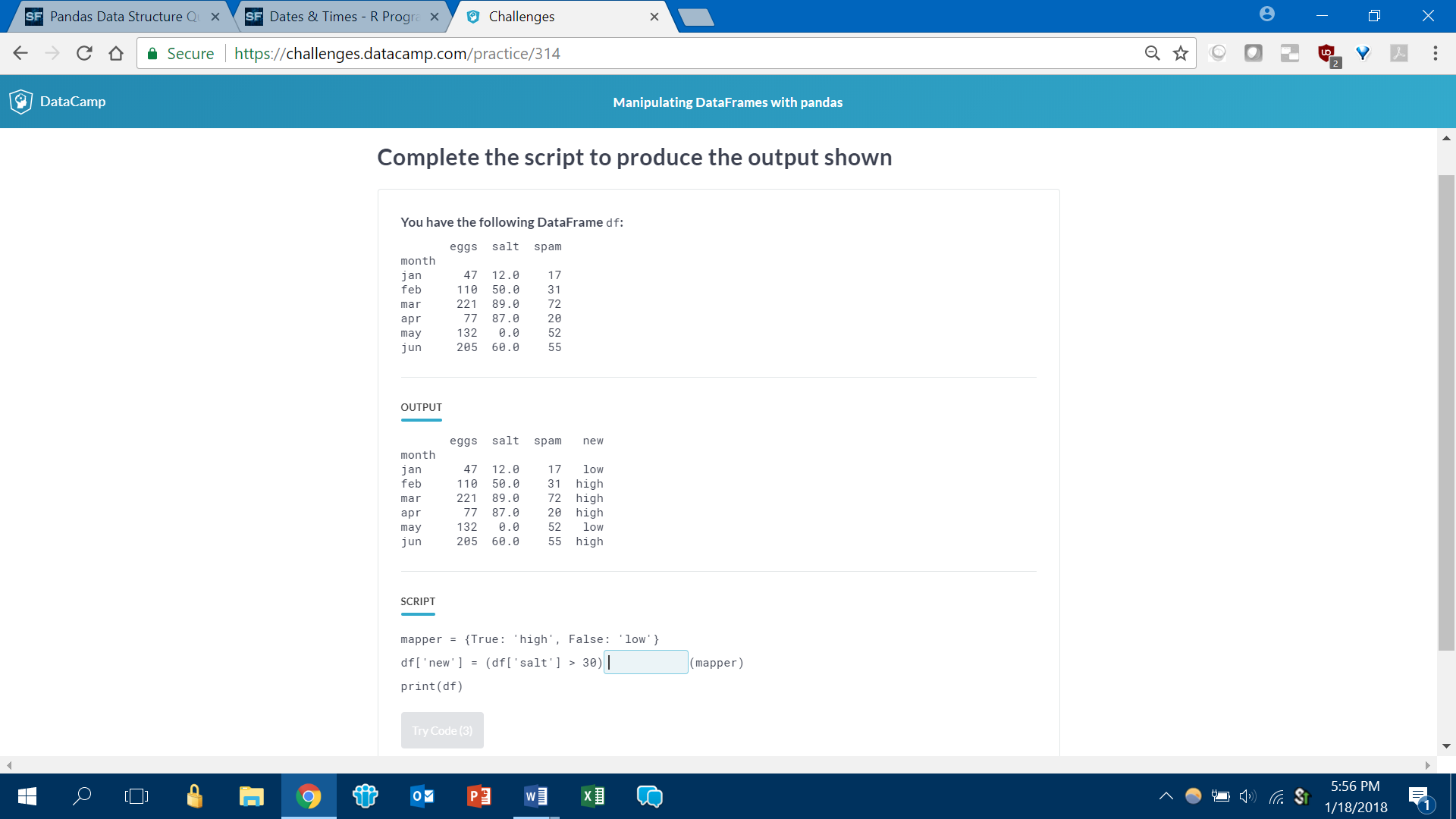
Remarks : We can sort dataframe by using ‘sort\_values’ with argument ‘by=’ for pointing out target column and ‘ascending=False’ in order to sort descending order.

**Question 2 (6 points)**

**Choose either Python or R for the entirety of this question.**

**Python:**

1. **Select the answer to complete the assigned task (2pts)**



Of the months April, May, and June, calculate how many had eggs>100

**A) (df.loc[[‘apr’, ‘may’, ‘jun’],][‘eggs’] > 100).sum()**

B) (df[[‘apr’, ‘may’, ‘jun’],][‘eggs’] > 100).sum()

C) (df[[‘apr’,’may’,’jun’]][‘eggs’] > 100).sum()

D) (df[(‘apr’, ‘may’, ‘jun’),][‘eggs’] > 100).count()

1. **Complete the script to produce the output shown (2pts)**

Output

datetime.date(2018, 1, 1)

Script

import datetime

datetime.datetime.\_\_\_\_\_\_\_(‘01012018’, “%d%m%Y”)\_\_\_\_\_\_\_\_

**A) strptime, .date()**

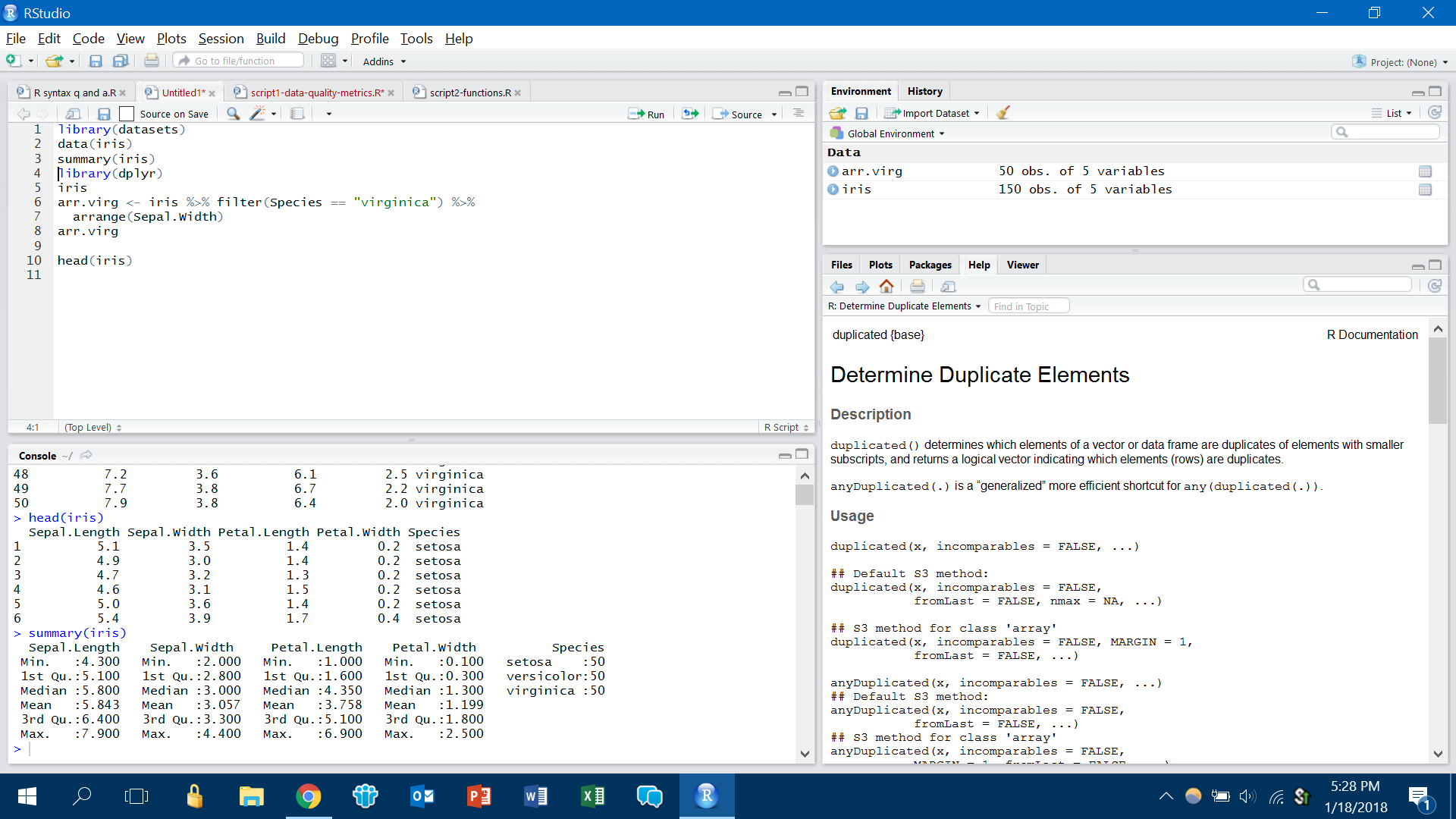
B) strptime, .as.date()

C) to\_date, [Nothing]

D) to\_datetime, [Nothing]

**C) Select the correct script to accomplish the task (2pts)**

Following is a summary of the iris dataset:



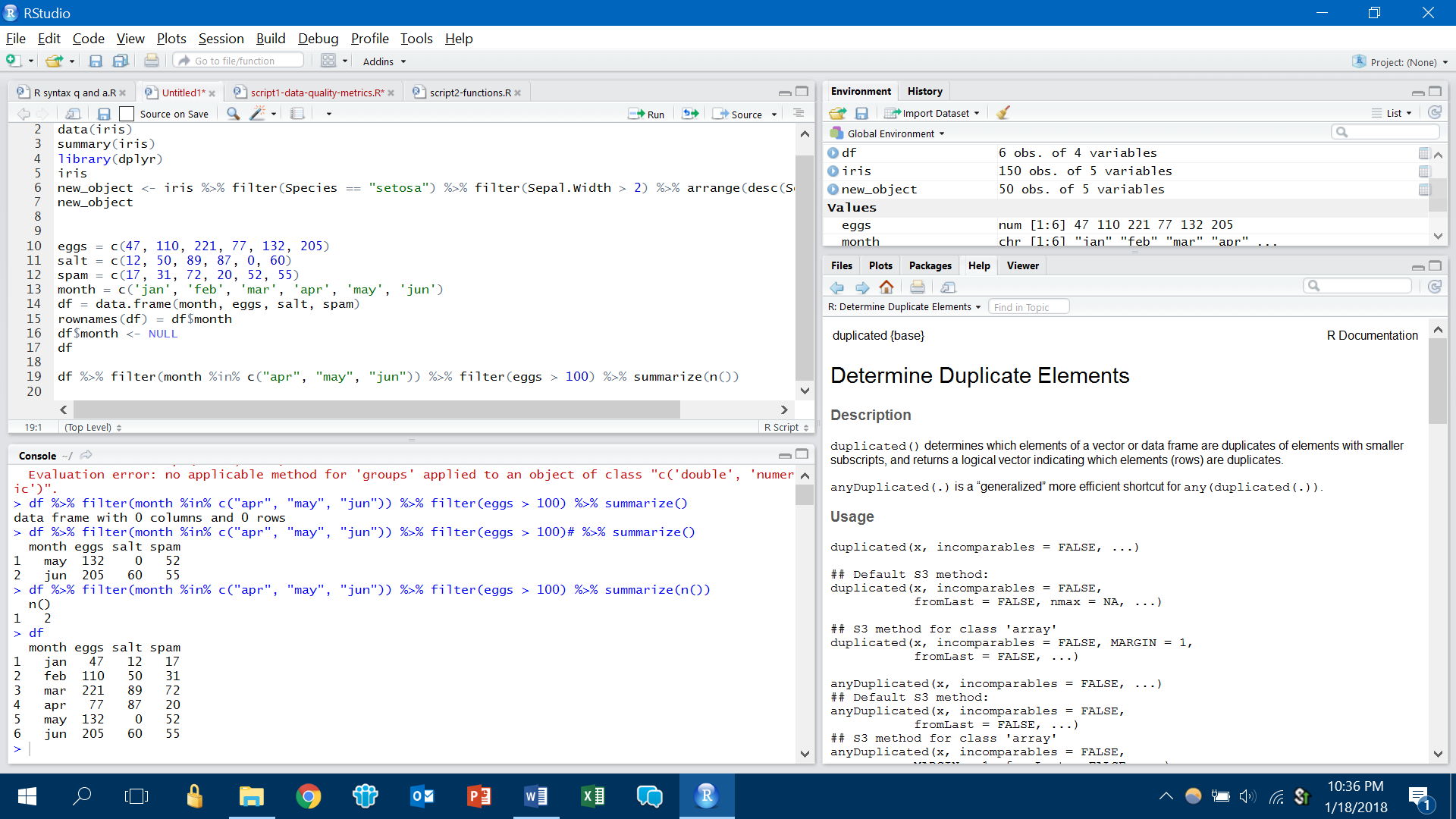
How would you create a new object with:

* only the “setosa” species
* Petal.Width greater than 2
* Sorted by Sepal.Length in descending order

1. iris[(iris[‘Species’] == ‘setosa’) && (iris[‘Petal.Width’] > 2)].sort\_values(‘Sepal.Length’, ascending = False)
2. **iris.loc[(iris[‘Species’] == ‘setosa’) & (iris[‘Petal.Width’] > 2)].sort\_values(‘Sepal.Length’, ascending = False) 🡪 Answer**
3. iris.loc[(iris[‘Species’] == ‘setosa’) & (iris[‘Petal.Width’] > 2)].sort\_values(‘Sepal.Length’, desc)
4. iris[(iris[‘Species’] == ‘setosa’) & (iris[‘Petal.Width’] > 2)].sort\_descending(‘Sepal.Length’)

**R:**

1. **Select the answer to complete the assigned task (2pts)**



A) df %>% select(month is.in("apr", "may", "jun")) %>% filter(eggs > 100) %>% summarize(count())

B) df %>% filter(month is.in("apr", "may", "jun")) %>% filter(eggs > 100) %>% summarize(n())

C) df %>% filter(month %in% c("apr", "may", "jun")) %>% filter(eggs > 100) %>% summarize(n())

D) df %>% filter(month %in% c("apr", "may", "jun")) %>% filter(eggs > 100) %>% summarize(count())

1. **Complete the script to produce the output shown (2pts)**

Output

[1] “2000-06-09”

Script

x <- “9\_June\_2000”

\_\_\_\_\_\_\_(x, format = “%d\_%B\_%Y”)

A) Date

B) as.POSIXct

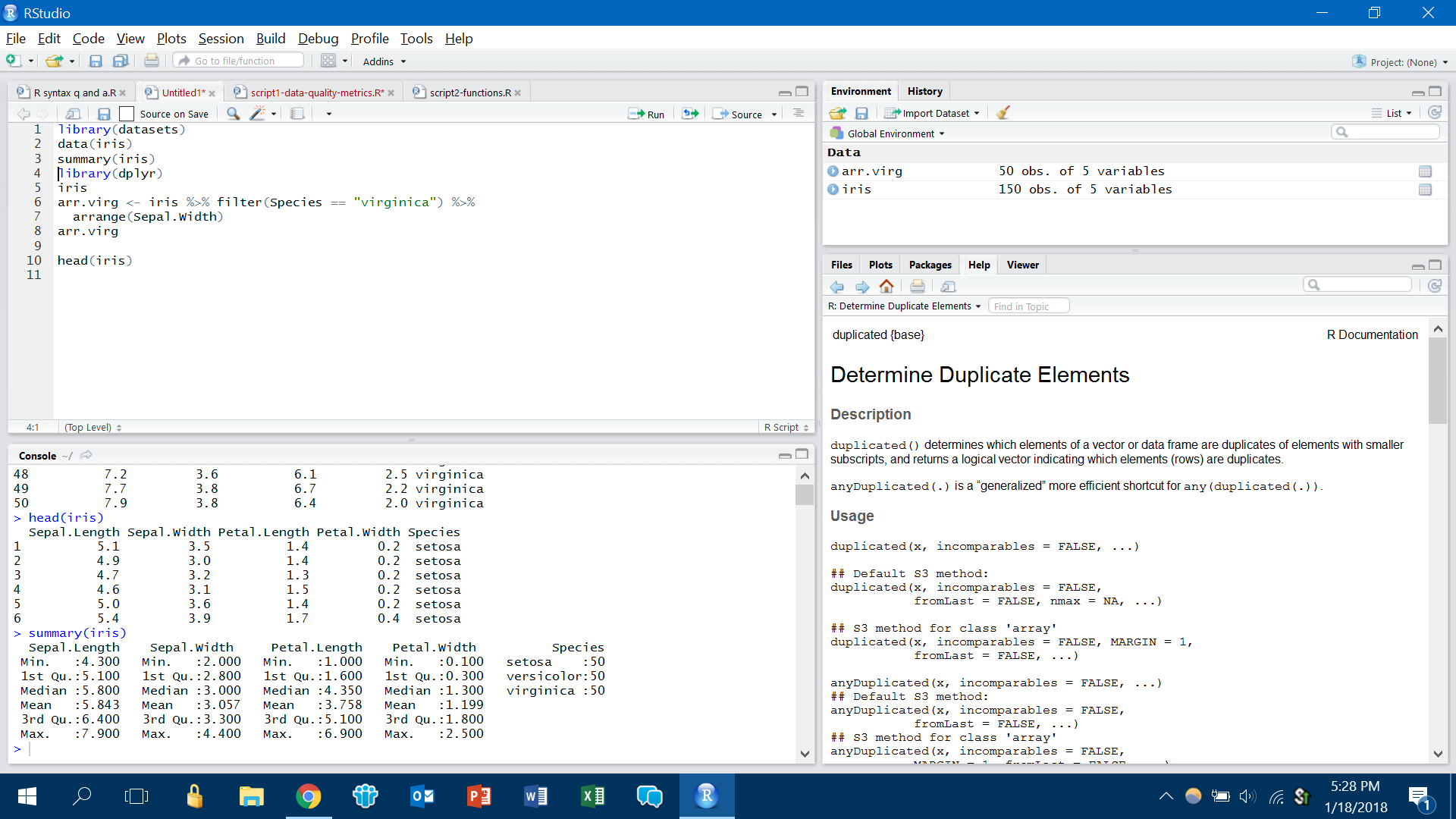
C) as.Date

D) as.date

E) as.DateTime

**C) Select the correct script to accomplish the task (2pts)**

Following is a summary of the iris dataset:



How would you create a new object with:

* only the “setosa” species
* Petal.Width greater than 2
* Sorted by Sepal.Length in descending order

A) new\_object <- filter(iris, Species == “setosa”) %>% filter(Petal.Width > 2) %>% sort(Sepal.Length, TRUE)

B) new\_object <- select(iris, Species == “setosa”) %>% filter(Petal.Width > 2) %>% sort(Sepal.Length, desc)

C) new\_object <- iris %>% filter(Species == "setosa") %>% filter(Petal.Width > 2) %>% arrange(desc(Sepal.Length))

D) new\_object <- iris %>% select(Species == "setosa") %>% filter(Petal.Width > 2) %>% arrange(Sepal.Length, desc)