Using the RStudio environment to solve several problems in R, developing competency with statements, variable assignments, expressions, vectors, functions, and packages.

Project Report Submitted By

Iti Rohilla

Northeastern University Boston Campus

ALY6000 70917 Introduction to Analytics SEC 19 Fall 2023 CPS [BOS-A-HY] by instructor

Roy Wada

Submitted On

Sept-26-2023

Summary

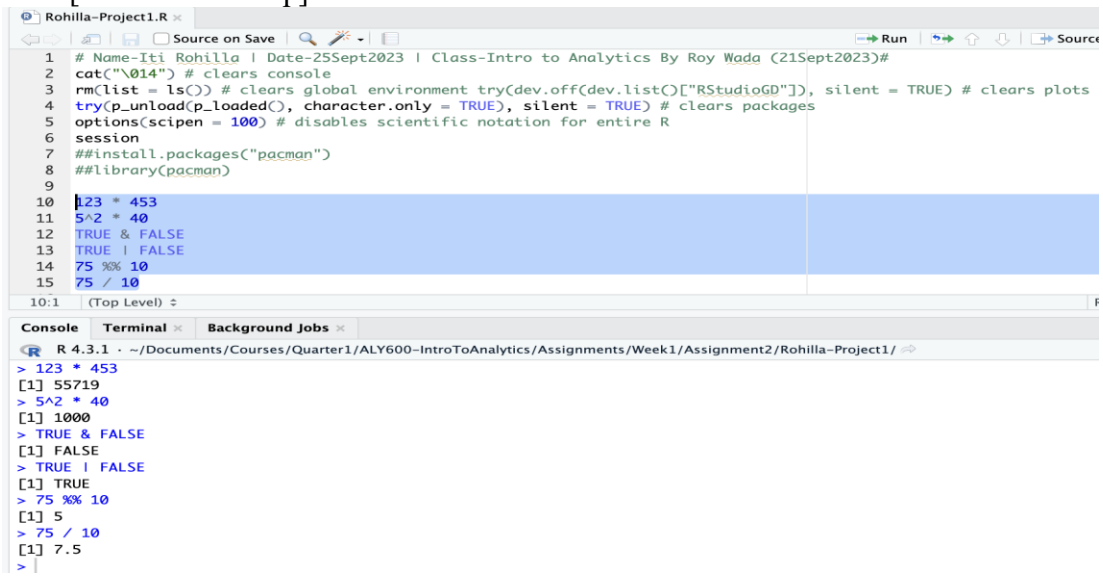
In this assignment we use the capabilities of RStudio in this assignment to use the R programming language to solve several issues. R is a flexible and popular language for statistical computation and data analysis, and RStudio is a strong integrated development environment (IDE) that makes it easier to write, run, and debug R code. We will gain expertise in basic ideas including statements, variable assignments, expressions, vectors, functions, and packages with this project while working in the RStudio environment.

Assignment Objectives:

RStudio Familiarization
Statements and Variable Assignments
Expressions and Operations
Working with Vectors
Packages and Libraries

Q1- Write lines of code to compute all the following. Include the answers in your written report.

Ans1[in the below snip]



Q2- Create a vector using the c function with the values 17, 12, -33, 5 and assign it to a variable called first_vector.

Q3- Create a vector using the c function with the values 5, 10, 15, 20, 25, 30, 35 and assign it to a variable called counting_by_fives.

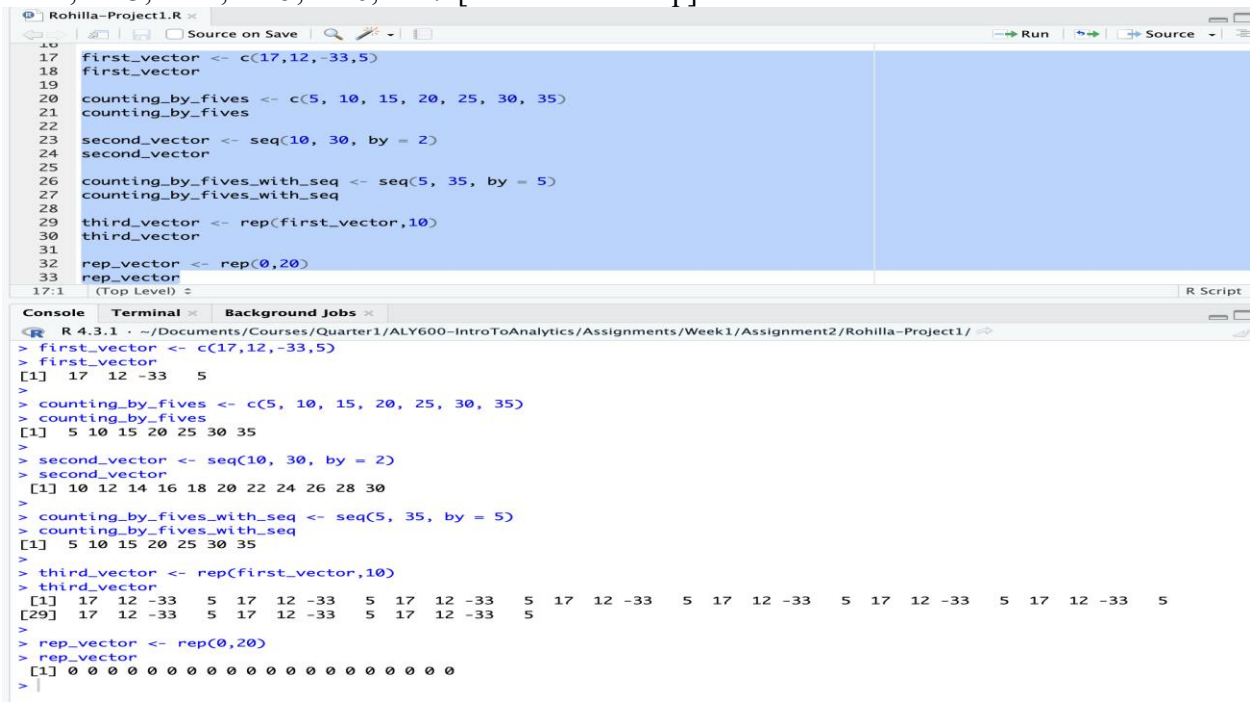
Q4- Create a vector using the seq function containing every even number between 10 and 30 inclusive and assign it to a variable called second_vector.

Q5- Create a vector using the seq function containing the values 5, 10, 15, 20, 25, 30 , 35 and assign it to a variable called counting_by_fives_with_seq.

Q6- Create a vector using the function rep and provide it with first_vector as its first argument and 10 as its second argument. Assign the result to a variable called third_vector.

Q7- Using the rep function, create a vector containing the number zero, 20 times. Store the result in a variable called rep_vector.

Ans2,Ans3,Ans4,Ans5,Ans6,Ans7-[in the below snip]



Q8-create a vector using the range operator (the colon), that contains the numbers from 10 to 1. Store the result in a variable called fourth_vector.

Q9- Create a vector using the range operator that contains the numbers from 5 to 15. Store the result in a variable called counting_vector.

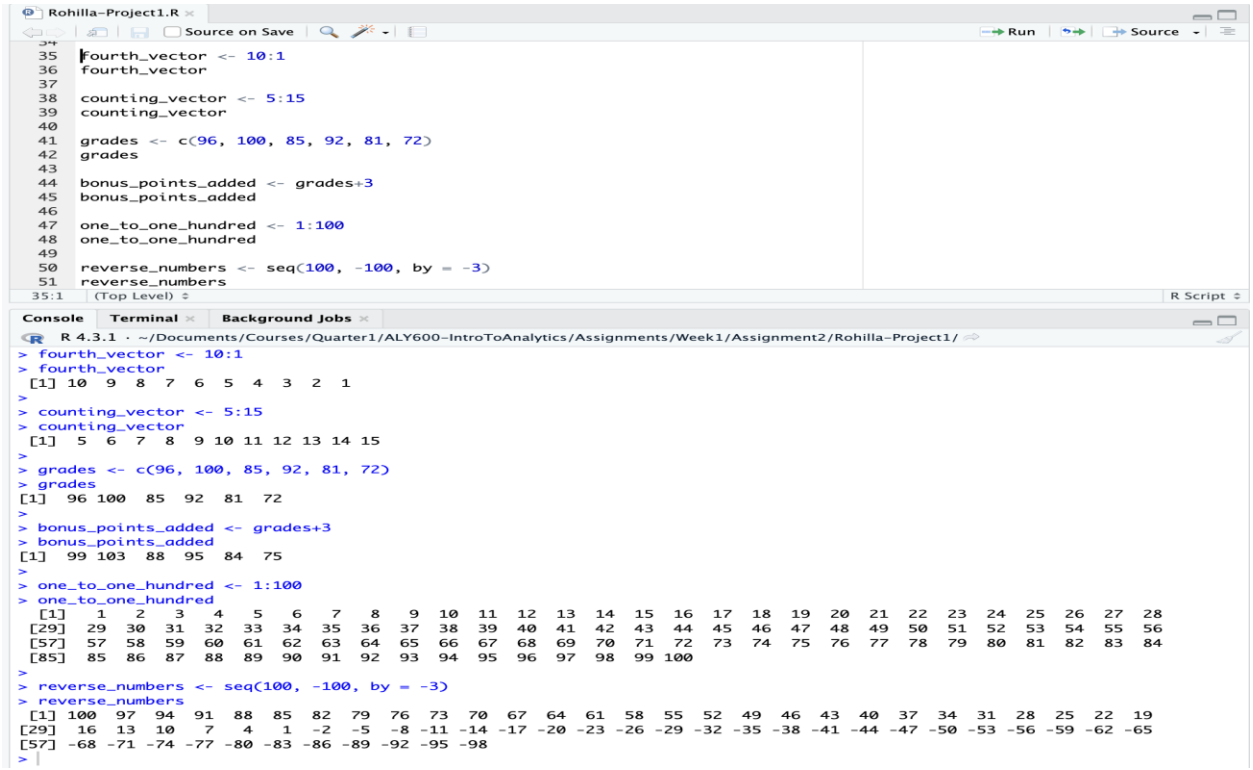
Q10- Create a vector with the values (96, 100, 85, 92, 81, 72) and store it in a variable called grades.

Q11- Add the number 3 to the vector grades. Store the result in a variable called bonus_points_added.

Q12- Create a vector with the values 1 – 100. Store it in a variable called one_to_one_hundred. Do not type out all 100 numbers.

Q13- Create a vector with values from 100 to -100 by 3s. Store the result in a variable called reverse_numbers. To clarify, the first 3 numbers in this vector will be (100, 97, 94...)

Ans8,Ans9,Ans10,Ans11,Ans12,Ans13: [in the below snip]



Q14- Write each of the following lines of code. Add a one-sentence comment above each line explaining what is happening. Include your comments in the written report.

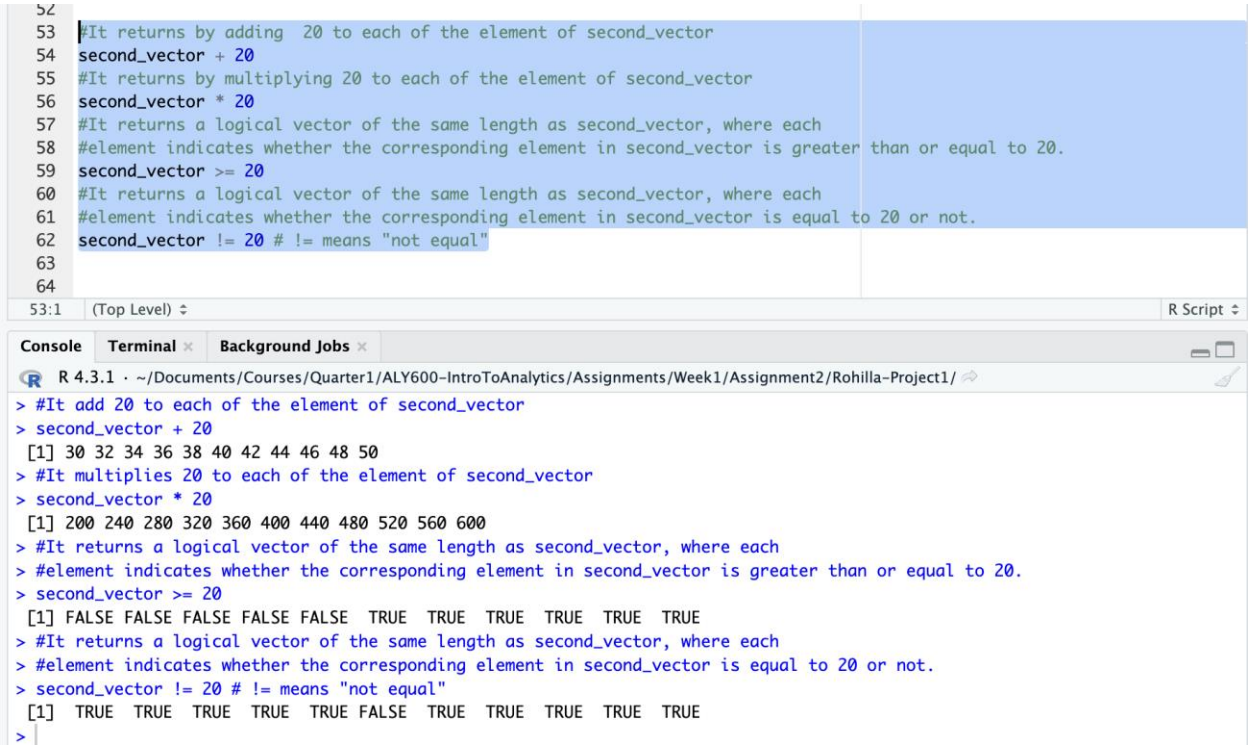
Ans14-EXPLAINATION

second_vector + 20-It returns by adding 20 to each of the element of second_vector

second_vector * 20-It returns by multiplying 20 to each of the element of second_vector

second_vector >= 20-It returns a logical vector of the same length as second_vector, where each element indicates whether the corresponding element in second_vector is greater than or equal to 20.

second_vector != 20 # != means "not equal"-It returns a logical vector of the same length as second_vector, where each element indicates whether the corresponding element in second_vector is equal to 20 or not.



Q15- Using the built in sum function, compute the sum of one_to_one_hundred and store it in a variable called total.

Q16- Using the built in mean function, compute the average of one_to_one_hundred and store the result in a variable called average_value.

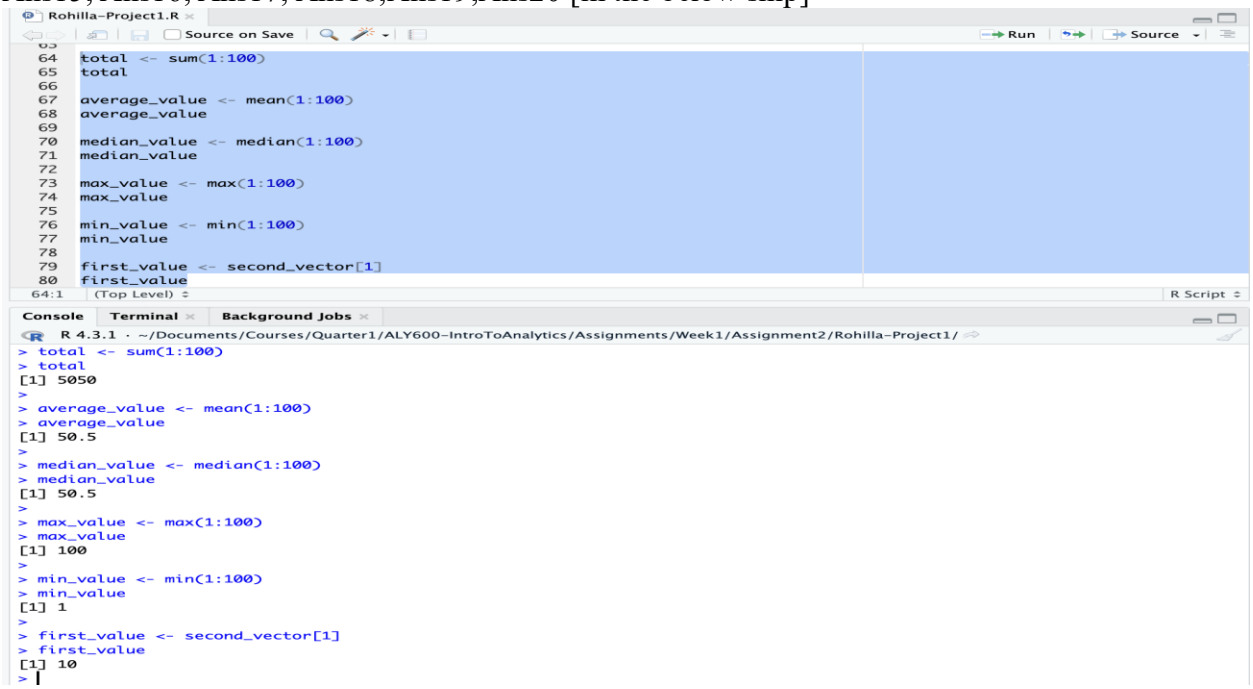
Q17- Using the built in median function, compute the average of one_to_one_hundred and store the result in a variable called median_value.

Q18- Using the built in max function, compute the average of one_to_one_hundred and store the result in a variable called max_value.

Q19- Using the built in min function, compute the average of one_to_one_hundred and store the result in a variable called min_value.

Q20- Using brackets, extract the first value from second_vector and store it in a variable called first_value.

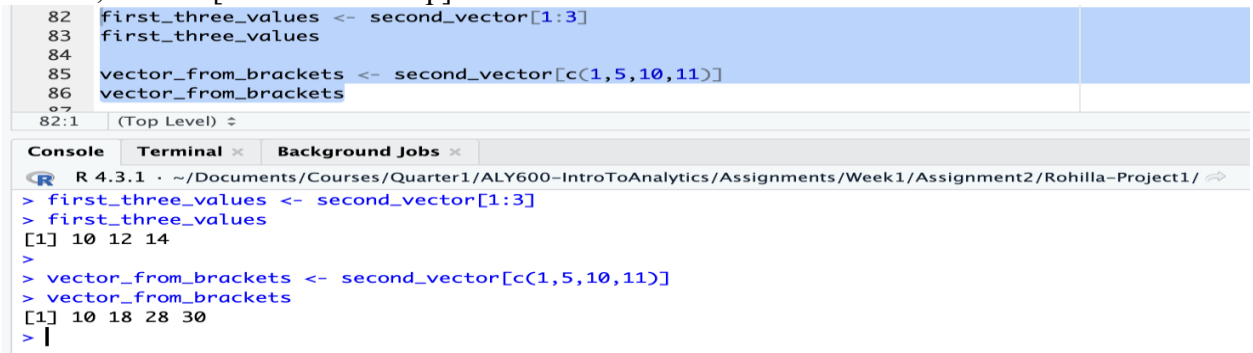
Ans15, Ans16, Ans17, Ans18,Ans19,Ans20 [in the below snip]



Q21- Using brackets, extract the first, second and third values from second_vector and store it in a variable called first_three_values.

Q22- Using brackets, extract the 1st, 5th, 10th, and 11th elements of second_vector. Store the resulting vector in a variable called vector_from_brackets.

Ans-21, Ans-22 [in the below snip]

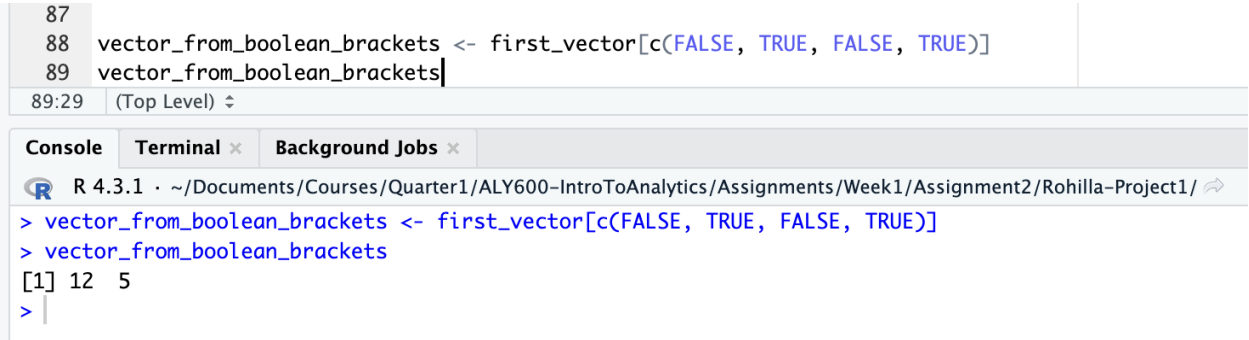


Q23 Use the brackets to extract elements from the first_vector using the following vector c(FALSE, TRUE, FALSE, TRUE). Store the result in a variable called vector_from_boolean_brackets. Explain in a comment what happens. Include the answer in your written report.

Ans23 EXPLAINATION

- The values of the logical_vector are c(FALSE, TRUE, FALSE, TRUE).
- The values of the first_vector are c(17,12,-33,5).
- Based on the values of logical_vector, entries from first_vector are extracted using square brackets, [logical_vector].
- It specifically includes elements whose corresponding logical_vector value is TRUE.
- logical_vector declares TRUE for the second and fourth elements and FALSE for the first and third ones.

- As a result, the 2nd and 4th components from first_vector, which are 12 and 5.



Q24- Examine the following piece of code and write a one-sentence comment explaining what is happening. Include the answer in your written report.

Ans24-EXPLAINATION

It performs a logical comparison on eachof the elements of second_vector, if it's greater or equal to 20 it returns TRUE otherwise FALSE.Here's the resulting logical vector:

second_vector:

```
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> second_vector <- seq(10, 30, by = 2)
> second_vector
 [1] 10 12 14 16 18 20 22 24 26 28 30
```

logical comparison:

```
91  second_vector >= 20
92
92:1  (Top Level)
Console  Terminal  Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> second_vector >= 20
 [1] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
>
```

Q25- Examine the following piece of code and write a one-sentence comment explaining what is happening.

A25-Here seq() function generates a sequence of values from 10 to 20 and in each step , it is incremented by 2 and stores the output in **ages_vector** variable

```
93  ages_vector <- seq(from = 10, to = 30, by = 2)
94  ages_vector
95
93:1  (Top Level)
Console  Terminal  Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> ages_vector <- seq(from = 10, to = 30, by = 2)
> ages_vector
 [1] 10 12 14 16 18 20 22 24 26 28 30
```

Q26- Examine the following piece of code and write a one-sentence comment explaining what is happening, assuming ages_vector was computed in the previous problem. Include the answers in your written report.

Ans26-Using conditional indexing it extracts shows only the values grater than 20 from the seq stored in ages_vector variable on the console.

```
93  ages_vector <- seq(from = 10, to = 30, by = 2)
94  ages_vector
95
96  ages_vector [ages_vector >= 20]
97
93:1  (Top Level)
```

```
Console  Terminal  Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> ages_vector <- seq(from = 10, to = 30, by = 2)
> ages_vector
 [1] 10 12 14 16 18 20 22 24 26 28 30
>
> ages_vector [ages_vector >= 20]
[1] 20 22 24 26 28 30
>
```

Q27- Using the same approach as the previous question, create a new vector by removing from the grades vector all values lower than or equal to 85. Store the new vector in a variable called lowest_grades_removed.

Ans27-

```
98  lowest_grades_removed <- grades [grades >= 85]
99  lowest_grades_removed
100
98:1  (Top Level)
```

```
Console  Terminal  Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> lowest_grades_removed <- grades [grades >= 85]
> lowest_grades_removed
[1]  96 100  85  92
>
```

Q28- Use the grades vector to create a new vector with the 3rd and 4th elements of grades removed. Store this in a variable called middle_grades_removed. Try utilizing a vector of negative indexes to complete this task.

Ans28[in the below snip]

```
negative_index <- c(3,4)
middle_grades_removed <- grades[- negative_index]
middle_grades_removed
```

```
> negative_index <- c(3,4)
> middle_grades_removed <- grades[- negative_index]
> middle_grades_removed
[1]  96 100  81  72
>
```

Q29- Use bracket notation to remove the 5th and 10th elements of second_vector. Store the result in a variable called fifth_vector.

Ans29[in the below snip]

```
minus_index <- c(5,10)
fifth_vector <- second_vector[-minus_index]
fifth_vector
```

```
> minus_index <- c(5,10)
> fifth_vector <- second_vector[-minus_index]
> fifth_vector
[1] 10 12 14 16 20 22 24 26 30
>
```

Q30- Write the following code. Explain in a comment what you think the code is doing. Include the answer in your written report.

Ans30-

set. Seed (5)- setting the seed ensures that the same set of numbers are reproduced every time we run a code with seed. This helps us in creating a random number but still having a control over the result, which is useful for sharing the code, testing etc.

runif- It generates random values from a uniform distribution.

n = 10-It signifies 10 random values that will be generated.

min = 0: Sets the minimum value for the uniform distribution (inclusive) to 0.

max = 1000: Sets the maximum value for the uniform distribution (exclusive) to 1000. This means that the generated values will be between 0 (inclusive) and 1000 (exclusive).

```
109  set.seed(5)
110  random_vector <- runif(n=10, min = 0, max = 1000)
111  random_vector
109:1  (Top Level)

Console  Terminal  Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> set.seed(5)
> random_vector <- runif(n=10, min = 0, max = 1000)
> random_vector
 [1] 200.2145 685.2186 916.8758 284.3995 104.6501 701.0575 527.9600 807.9352 956.5001 110.4530
```

Q31- Use the sum function to compute the total of random_vector. Store the result in a variable called sum_vector.

Ans31[in the below snip]

```
113  sum_vector <- sum(random_vector)
114  sum_vector
115
113:1  (Top Level)

Console  Terminal  Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> sum_vector <- sum(random_vector)
> sum_vector
[1] 5295.264
>
```

Q32- Use the cumsum function to compute the cumulative sum of random_vector. Store the result in a variable called cumsum_vector.

Ans32[in the below snip]

```
116  cumsum_vector <- cumsum(random_vector)
117  random_vector
118
118:1  (Top Level)

Console  Terminal  Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> cumsum_vector <- cumsum(random_vector)
> random_vector
 [1] 200.2145 685.2186 916.8758 284.3995 104.6501 701.0575 527.9600 807.9352 956.5001 110.4530
>
```

Q33- Use the mean function to compute the mean of random_vector. Store the result in a variable called mean_vector.

Ans33[in the below snip]

```
119 mean_vector <- mean(random_vector)
120 mean_vector
121
121:1 (Top Level)
Console Terminal Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> mean_vector <- mean(random_vector)
> mean_vector
[1] 529.5264
>
```

Q34- Use the sd function to compute the standard deviation of random_vector. Store the result in a variable called sd_vector.

Ans34[in the below snip]

```
122 sd_vector <- sd(random_vector)
123 sd_vector
124
124:1 (Top Level)
Console Terminal Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> sd_vector <- sd(random_vector)
> sd_vector
[1] 331.3606
>
```

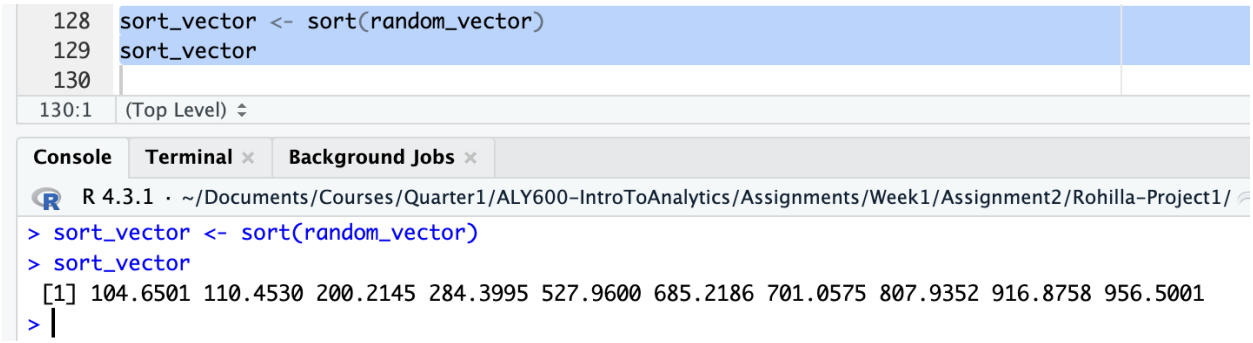
Q35- Use the round function to round the values of random_vector. Store the result in a variable called round_vector.

Ans35[in the below snip]

```
125 round_vector <- round(random_vector)
126 round_vector
127
127:1 (Top Level)
Console Terminal Background Jobs
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
> round_vector <- round(random_vector)
> round_vector
 [1] 200 685 917 284 105 701 528 808 957 110
>
```

Q36- Use the sort function to sort the values of random_vector. Store the result in a variable called sort_vector.

Ans36[in the below snip]



Q37- Consider the following code. Explain in a comment what you think the code is doing. Include the answer in your written report.

A37-

set. Seed (5)- setting the seed ensures that the same set of numbers are reproduced every time we run a code with seed. This helps us in creating a random number but still having a control over the result, which is useful for sharing the code, testing etc.

- rnorm- It produces random numbers using a normal (Gaussian) distribution with a mean and standard deviation.
- n = 1000 - It signifies 10 random values that will be generated.
- mean = 50: The mean (average) of the distribution. It determines the center of the distribution..
- sd = 15: The standard deviation of the distribution. It controls the spread or variability of the values.

```
130 set.seed(5)
131 random_vector <- rnorm(n=1000, mean = 50, sd = 15)
132 random_vector
132:14 (Top Level)
```

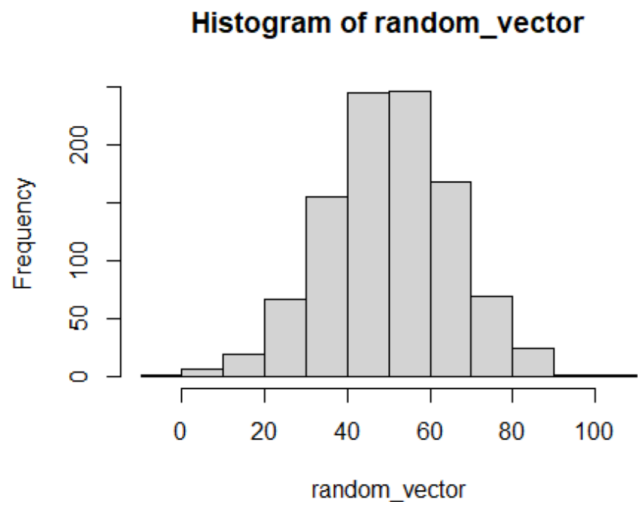
Console | Terminal | Tests | Background Jobs

R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/

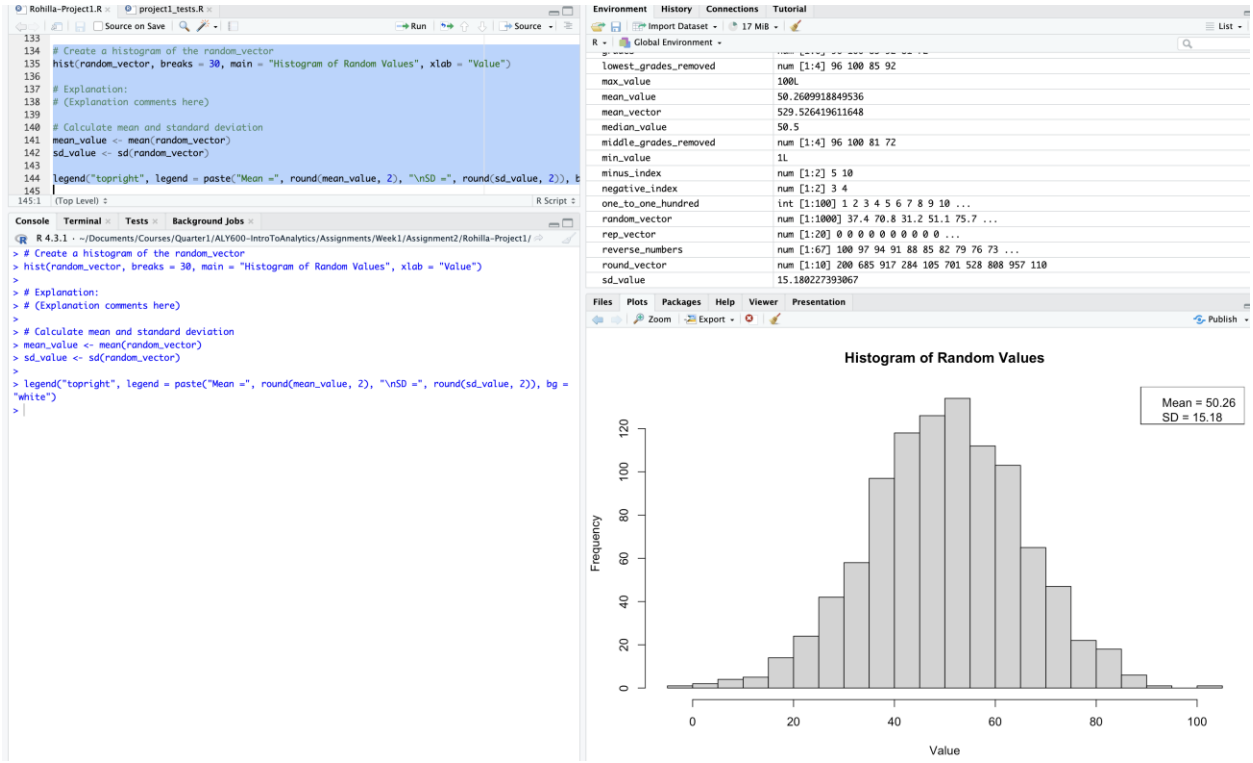
```
[621] 36.896642 51.679311 72.713471 55.648398 69.459222 88.435348 34.969630 62.010010
[631] 46.561684 45.982804 55.605366 64.708033 84.219371 57.984874 57.482462 52.311508
[641] 27.056669 75.862333 53.068540 68.567135 53.644456 65.203155 69.237345 54.338137
[651] 40.834400 43.047803 27.774328 45.437886 54.698978 46.024722 29.594923 55.916077
[661] 53.109473 30.850411 42.264958 48.816565 59.575227 62.193875 36.602228 43.976066
[671] 63.098992 60.345579 37.199083 59.812497 71.000132 65.036484  4.368345 46.365888
[681] 40.579735 53.800004 49.622168 47.200690 46.499304 90.863106  8.111105 74.771257
[691] 56.156468 58.278014 18.167799 48.291509 45.894928 53.125656 85.473019 74.083481
[701] 43.383201 58.450118 67.701342 35.993400 49.669894 46.501302 31.637808 58.464712
[711] 56.513500 54.442419 35.425284 40.386467 63.843831 33.705611 56.901360 46.727781
[721] 36.582930 49.733598 56.334545 51.850561 25.122842 48.082021 36.641295 63.454781
[731] 57.704820 55.095642 46.038382 56.676538 70.520290 37.677320 52.869780 71.702272
```

Q38- Use the hist function and provide it with random_vector. Explain the result in a comment.

Include both the explanation and visualization in your report.



Ans38[in the below snip]



It creates a histogram of the random_vector using the hist function.

-random_vector -It is a numeric vector that contains random data.

-**breaks parameter** -It is set to 30, which divides the data into 30 bins (bars) in the histogram.

-**main parameter** -It provides a **title** for the histogram.

- **xlab parameter**-It labels the x-axis as "Value."

It calculates the mean and standard deviation of random_vector using the mean and sd functions, respectively, and stores these values in mean_value and sd_value variables.

It adds a legend to the histogram using the legend function which is positioned in the top-right corner.

The legend text displays the calculated mean and standard deviation values, which are rounded to two decimal places.

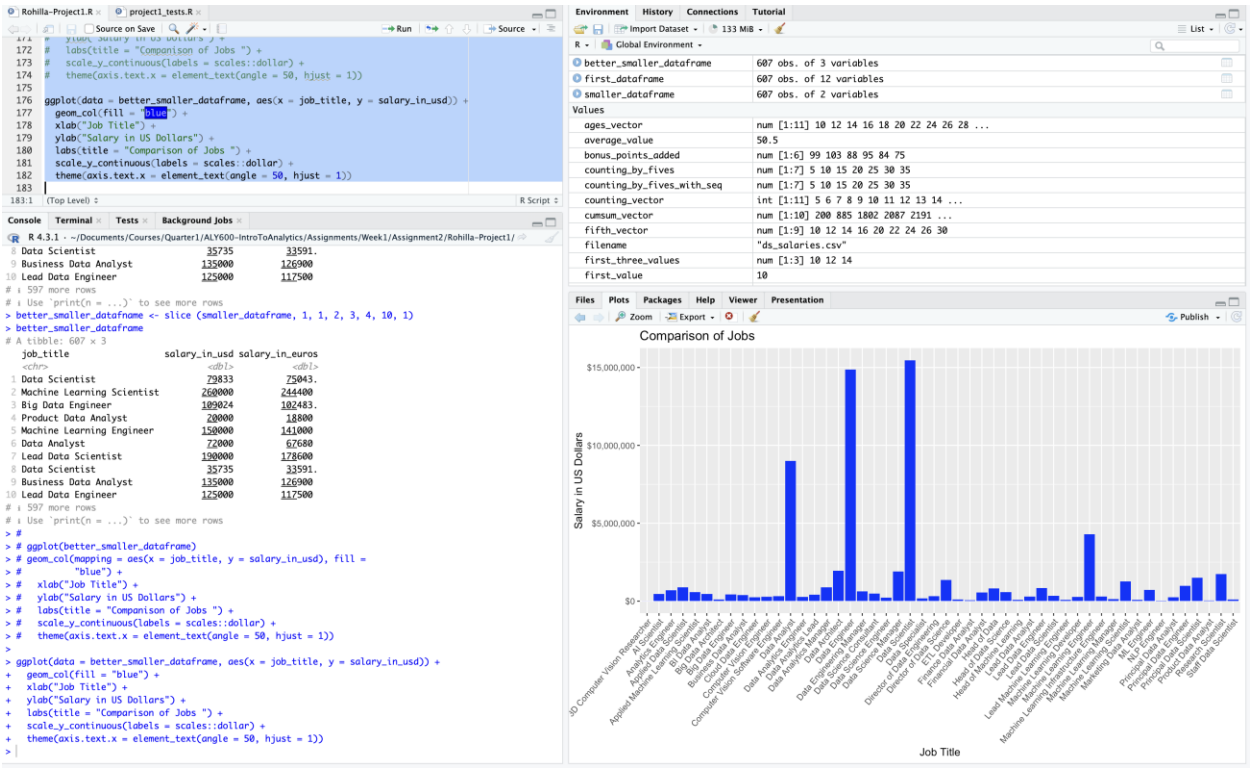
The background color of the legend is set to white. (Using bg=white)

Q42- Try each of the following blocks of code. Add a one-sentence comment describing what you believe is happening. Include your answers in your written report.

```
head(first_dataframe)

head(first_dataframe, n = 7)

names(first_dataframe)

smaller_dataframe <- select(first_dataframe, job_title, salary_in_usd)

smaller_dataframe
```

```r
better_smaller_dataframe <- arrange(smaller_dataframe,
desc(salary_in_usd))

better_smaller_dataframe

better_smaller_dataframe <- filter(smaller_dataframe, salary_in_usd >
80000)

better_smaller_dataframe

better_smaller_dataframe <-
  mutate(smaller_dataframe, salary_in_euros = salary_in_usd * .94)

better_smaller_dataframe

better_smaller_dataframe <- slice(smaller_dataframe, 1, 1, 2, 3, 4, 10,
1)

better_smaller_dataframe

ggplot(better_smaller_dataframe) +
  geom_col(mapping = aes(x = job_title, y = salary_in_usd), fill =
"blue") +
  xlab("Job Title") +
  ylab("Salary in US Dollars") +
  labs(title = "Comparison of Jobs ") +
  scale_y_continuous(labels = scales::dollar) +
  theme(axis.text.x = element_text(angle = 50, hjust = 1))
```

Ans42.



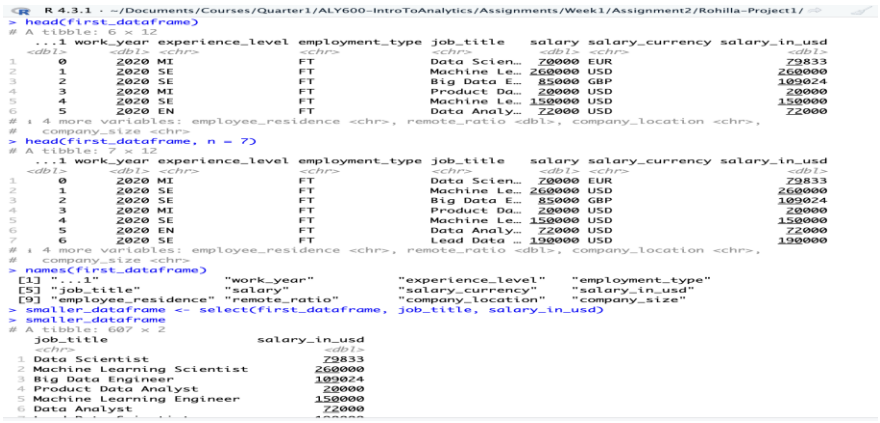
EXPLAINATION:

head(first_dataframe): Displays the first 6 rows of the first_dataframe dataset.

head(first_dataframe, n = 7): Displays the first 7 rows of the first_dataframe dataset, overriding the default of 6 rows.

names(first_dataframe): Retrieves and displays the column names (variable names) of the first_dataframe.

smaller_dataframe <- select(first_dataframe, job_title, salary_in_usd): It will create a new dataframe [smaller_dataframe]by selecting only the "job_title" and "salary_in_usd" columns from first_dataframe.



better_smaller_dataframe <- arrange(smaller_dataframe, desc(salary_in_usd)): It will Create a new dataframe [better_smaller_dataframe]by arranging smaller_dataframe in descending order which is based on the "salary_in_usd" column.

better_smaller_dataframe <- filter(smaller_dataframe, salary_in_usd > 80000): It will Create a new dataframe [better_smaller_dataframe] by filtering rows in smaller_dataframe where the "salary_in_usd" is greater than 80000.

better_smaller_dataframe <- mutate(smaller_dataframe, salary_in_euros = salary_in_usd * .94): It will Create a new column "salary_in_euros" in smaller_dataframe by multiplying the "salary_in_usd" column by 0.94.

better_smaller_datafname <- slice(smaller_dataframe, 1, 1, 2, 3, 4, 10, 1): It will create a new dataframe [better_smaller_datafname] by selecting specific rows (1, 1, 2, 3, 4, 10, 1) from smaller_dataframe.

The final block of code uses ggplot2 to create a plot (ggplot) of the data in better_smaller_dataframe. It creates a bar plot by comparing job titles and their salaries in US Dollars, customizes axis labels and titles, and specifies other visual properties for the plot.

```
R 4.3.1 · ~/Documents/Courses/Quarter1/ALY600-IntroToAnalytics/Assignments/Week1/Assignment2/Rohilla-Project1/
 8 Data Scientist                        35735          33591.
 9 Business Data Analyst                135000         126900
10 Lead Data Engineer                   125000         117500
# ℹ 597 more rows
# ℹ Use `print(n = ...)` to see more rows
> better_smaller_dataframe <- slice (smaller_dataframe, 1, 1, 2, 3, 4, 10, 1)
> better_smaller_dataframe
# A tibble: 607 × 3
   job_title                  salary_in_usd salary_in_euros
   <chr>                          <dbl>          <dbl>
 1 Data Scientist                  79833          75043.
 2 Machine Learning Scientist     260000         244400
 3 Big Data Engineer              109024         102483.
 4 Product Data Analyst            20000          18800
 5 Machine Learning Engineer      150000         141000
 6 Data Analyst                    72000          67680
 7 Lead Data Scientist            190000         178600
 8 Data Scientist                  35735          33591.
 9 Business Data Analyst          135000         126900
10 Lead Data Engineer             125000         117500
# ℹ 597 more rows
# ℹ Use `print(n = ...)` to see more rows
> #
> # ggplot(better_smaller_dataframe)
> # geom_col(mapping = aes(x = job_title, y = salary_in_usd), fill =
> #            "blue") +
> #   xlab("Job Title") +
> #   ylab("Salary in US Dollars") +
> #   labs(title = "Companison of Jobs ") +
> #   scale_y_continuous(labels = scales::dollar) +
> #   theme(axis.text.x = element_text(angle = 50, hjust = 1))
>
> ggplot(data = better_smaller_dataframe, aes(x = job_title, y = salary_in_usd)) +
+   geom_col(fill = "blue") +
+   xlab("Job Title") +
+   ylab("Salary in US Dollars") +
+   labs(title = "Comparison of Jobs ") +
+   scale_y_continuous(labels = scales::dollar) +
+   theme(axis.text.x = element_text(angle = 50, hjust = 1))
> smaller_dataframe
```

----------Work Citations----------

- Understanding the base concept of RStudio environment, developing competency with statements, variable assignments, expressions, vectors, functions, data frames and packages. https://northeastern.instructure.com/courses/160343/pages/module-1-%7C-resources?module_item_id=9214295
- For creating and understanding Histogram function https://www.youtube.com/watch?v=tp_BG5wDeVU
- Trouble shooting Errors in R studio. https://warin.ca/posts/rcourse-howto-interpretcommonerrors/
- Ensuring the Project Report is in APA Format https://writingcenter.uagc.edu/apa-formatting-microsoft-word