Project 1 - R Practice

Project Instructions

In this project you will produce and submit two files. The first is an R script that you create by following the instructions below. The second is a report containing answers as specified in the the instructions below. Be sure to follow all formatting guidelines.

Setting up your project

When working in R Studio be careful with cloud drive pathnames. You may be best served using alocal drive whenever possible until you are comfortable troubleshooting technical issues.

1. Create a new project called "Lastname-Project1". For example, if your last name is Smith your project would be titled "Smith-Project1".
2. Create an R Script file within your project called "Lastname-Project1.R". As before, if your lastname is Smith, your file will be titled "Smith-Project1.R".
3. Include your name, the date, and the class in a comment as the first line of the script.

Testing your solution

The file project1_tests.R contains a set of test cases which will affirm that some of the problems you have solved are correct. To use this file, save it into your project folder.

Then, from the console line write the following code

```
# Do the install line a single time when you
# start. Then you will not need to do it again
# during this or future projects.
install.packages("testthat") # do this line

library(testthat)
test_file("project1_tests.R")
```

Tests can be run at any time. This code will report the total number of tested items that currently succeed and the errors preventing you from passing others. A single problem may have multiple tests while others have none. You should be able to complete this project with 100% accuracy on all tests. Use the exact names of variables stated in each problem, pay extra attention to matching any upper and lower case letters.

Project 1 Problems

Instructions:

Complete the following problems in the R script you created above. Be sure to name specified variables exactly as they are given, paying special attention to spellings and the use of upper and lower case letters.

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

```
123 * 453
5^2 * 40
TRUE & FALSE
TRUE | FALSE
75 %% 10
75 / 10
```

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

```
[1]  17  12 -33   5
```

3. 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.**

4. Create a vector using the range operator (the colon), that contains the numbers from 20 down to 1 . Store the result in a variable called **second_vector**.

```
 [1] 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1
```

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

```
 [1]  5  6  7  8  9 10 11 12 13 14 15
```

6. Create a vector with the values (96, 100, 85, 92, 81, 72). Store the result in a variable called **grades**

```
[1]  96 100  85  92  81  72
```

7. Add the number 3 to the vector **grades**. Store the result in a variable called **bonus_points_added**.

```
[1]  99 103  88  95  84  75
```

8. Create a vector with the values 1 – 100 and store it in a variable called one_to_one_hundred. Do not type out all 100 numbers.

```
  [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
17  18
```

```
 [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
35  36
 [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52
53  54
 [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70
71  72
 [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88
89  90
 [91]  91  92  93  94  95  96  97  98  99 100
```

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

```
second_vector + 20
second_vector * 20
second_vector >= 20
second_vector != 20  # != means "not equal"
```

10. Using the built in **sum** function, compute the sum of **one_to_one_hundred**. Store the result in a variable called **total**.

```
[1] 5050
```

11. Using the built in **mean** function, compute the average of **one_to_one_hundred**. Store the result in a variable called **average_value**

```
[1] 50.5
```

12. Using the built in **median** function, compute the average of **one_to_one_hundred**. Store the result in a variable called **median_value**

```
[1] 50.5
```

13. Using the built in **max** function, compute the max of **one_to_one_hundred**. Store the result in a variable called **max_value**

```
[1] 100
```

14. Using the built in **min** function, compute the min of **one_to_one_hundred**. Store the result in a variable called **min_value**

```
[1] 1
```

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

```
[1] 20
```

16. Using brackets, extract the first, second and third values from **second_vector**. Store the result in a variable called **first_three_values.**

```
[1] 20 19 18
```

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

```
[1] 20 16 11 10
```

18. Use the brackets to extract elements from **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.

```
[1] 12  5
```

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

```
second_vector >= 10
```

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

```
one_to_one_hundred[one_to_one_hundred >= 20]
```

21. Using the same approach as in the previous question, create a new vector from the **grades** vector with only values larger than 85. Store the result in a variable called **lowest_grades_removed**.

```
[1]  96 100  92
```

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

```
[1]  96 100  81  72
```

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

```
 [1] 20 19 18 17 15 14 13 12 10  9  8  7  6  5  4  3  2  1
```

24. Write the following code. This creates a variable called random_vector that will be utilized in problems 25 - 30.

```
set.seed(5)
random_vector <- runif(n=10, min = 0, max = 1000)
```

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

```
[1] 5295.264
```

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

```
 [1]  200.2145  885.4330 1802.3088 2086.7083 2191.3584 2892.4159
3420.3759
 [8] 4228.3111 5184.8112 5295.2642
```

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

```
[1] 529.5264
```

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

```
[1] 331.3606
```

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

```
 [1] 200 685 917 284 105 701 528 808 957 110
```

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

31. Download the datafile **ds_salaries.csv** from Canvas. Save it on your computer in the same folder (directory) where your .R file for this project is located.

32. Use the function **read.csv** to read the **ds_salaries.csv** file. Store the result of the read into a variable called **first_dataframe.**

33. Use the **summary** function with **first_dataframe** to produce summary statistics based on each column of the data frame.

Submitting to Canvas

When you are satisfied with your solution.

1. **Remove** any lines in your code that have "include.packages" in them or any lines with install.packages in them.

2. **Remove** any lines in your code that use the **view** function.

3. Submit 2 files under the assignment in Canvas. Your R script named **Lastname_Project1_Script.R** file and your report containing the information specified in the instruction as a pdf titled **Lastname_Project1_Report.pdf**. See The report formatting guide for information on proper formatting of the report.

Congratulations on completing your first R project!