

WEEK-1

- **Exercise 1**

Let's create the following vectors:

`u <- 4`

`v <- 8`

Use the elementary arithmetic operators `+`, `-`, `*`, `/`, and `^` to:

add `u` and `v`

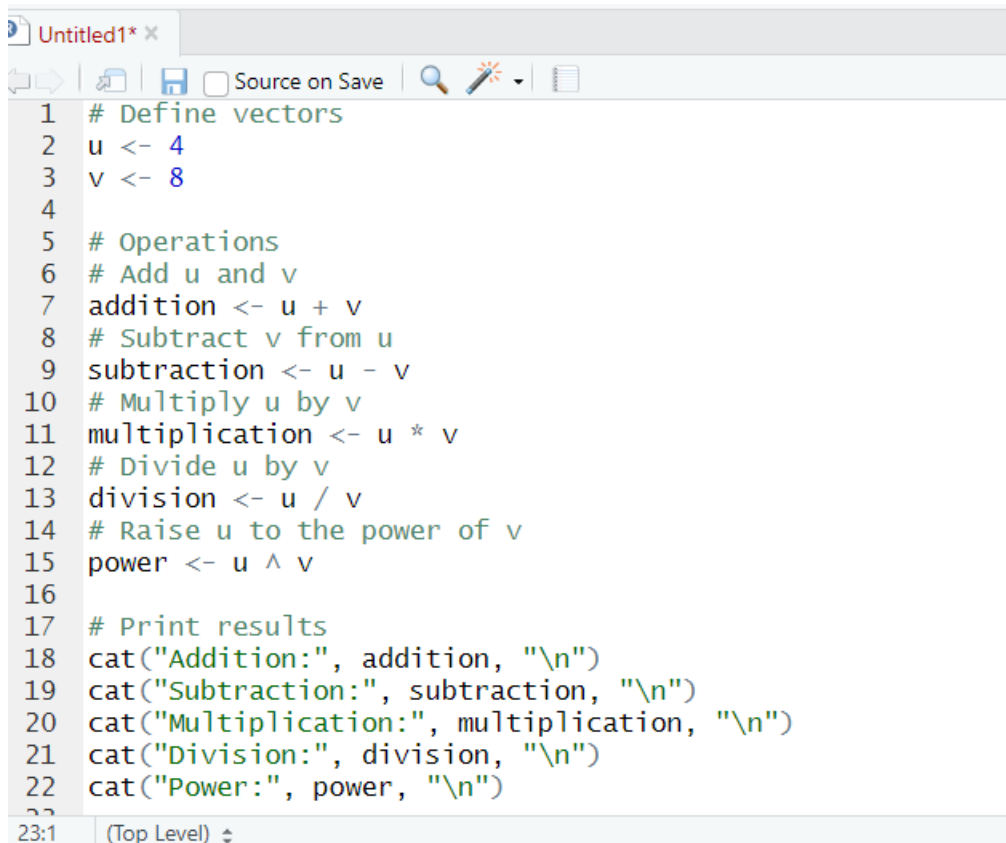
subtract `v` from `u`

multiply `u` by `v`

divide `u` by `v`

raise `u` to the power of `v`

ANSWER:

A screenshot of a code editor window titled 'Untitled1*'. The editor contains R code that defines two vectors, u and v, and then performs various arithmetic operations on them. The code is as follows:

```
1 # Define vectors
2 u <- 4
3 v <- 8
4
5 # Operations
6 # Add u and v
7 addition <- u + v
8 # Subtract v from u
9 subtraction <- u - v
10 # Multiply u by v
11 multiplication <- u * v
12 # Divide u by v
13 division <- u / v
14 # Raise u to the power of v
15 power <- u ^ v
16
17 # Print results
18 cat("Addition:", addition, "\n")
19 cat("Subtraction:", subtraction, "\n")
20 cat("Multiplication:", multiplication, "\n")
21 cat("Division:", division, "\n")
22 cat("Power:", power, "\n")
23
```

The editor interface includes a toolbar with icons for undo, redo, save, and search, as well as a 'Source on Save' checkbox. The status bar at the bottom indicates '23:1 (Top Level)'.

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/1.R/ ↗
> power <- u ^ v
> # Print results
> cat("Addition:", addition, "\n")
Addition: 12
> cat("Subtraction:", subtraction, "\n")
Subtraction: -4
> cat("Multiplication:", multiplication, "\n")
Multiplication: 32
> cat("Division:", division, "\n")
Division: 0.5
> cat("Power:", power, "\n")
Power: 65536
>
```

- **Exercise 2**

Now, suppose u and v are not scalars, but vectors with multiple elements:

```
u <- c(4, 5, 6)
```

```
v <- c(1, 2, 3)
```

Without using R, write down what you expect as the result of the same operations as in the previous exercise:

add u and v

subtract v from u

multiply u by v

divide u by v

raise u to the power of v

ANSWER

```
u <- c(4, 5, 6)
```

```
v <- c(1, 2, 3)
```

Addition: add u and v

Result: $(4+1, 5+2, 6+3) = (5, 7, 9)$

Subtraction: subtract v from u

Result: $(4-1, 5-2, 6-3) = (3, 3, 3)$

Multiplication: multiply u by v element-wise

Result: $(41, 52, 6 \cdot 3) = (4, 10, 18)$

Division: divide u by v element-wise

Result: $(4/1, 5/2, 6/3) = (4, 2.5, 2)$

Power: raise u to the power of v element-wise

Result: $(4^1, 5^2, 6^3) = (4, 25, 216)$

- **Exercise 3**

When we want to carry out a series of arithmetic operations, we can either use a single expression, or a series of expressions. Consider two vectors u and v:

```
u <- c(8, 9, 10)
```

```
v <- c(1, 2, 3)
```

We can create a new vector w in a single line of code:

```
w <- (2 * u + v) / 10
```

or carry out each operation on a separate line:

```
w <- 2 * u
```

```
w <- w + v
```

```
w <- w / 10
```

Convert the following expressions to separate operations, and check that both approaches give the same

```
w <- (u + 0.5 * v) ^ 2
```

```
w <- (u + 2) * (u - 5) + v
```

```
w <- (u + 2) / ((u - 5) * v)
```

ANSWER


```
[1] 72.25 100.00 132.25
> w <- u + 2
> w <- w * (u - 5)
> w <- w + v
> w
[1] 31 46 63
> w <- u + 2
> w_denominator <- (u - 5) * v
> w <- w / w_denominator
> w
[1] 3.333333 1.375000 0.800000
>
> |
```

Exercise 4

We can do the reverse as well. Convert the following multi-line operations to a single expression. Check that both approaches give the same result.

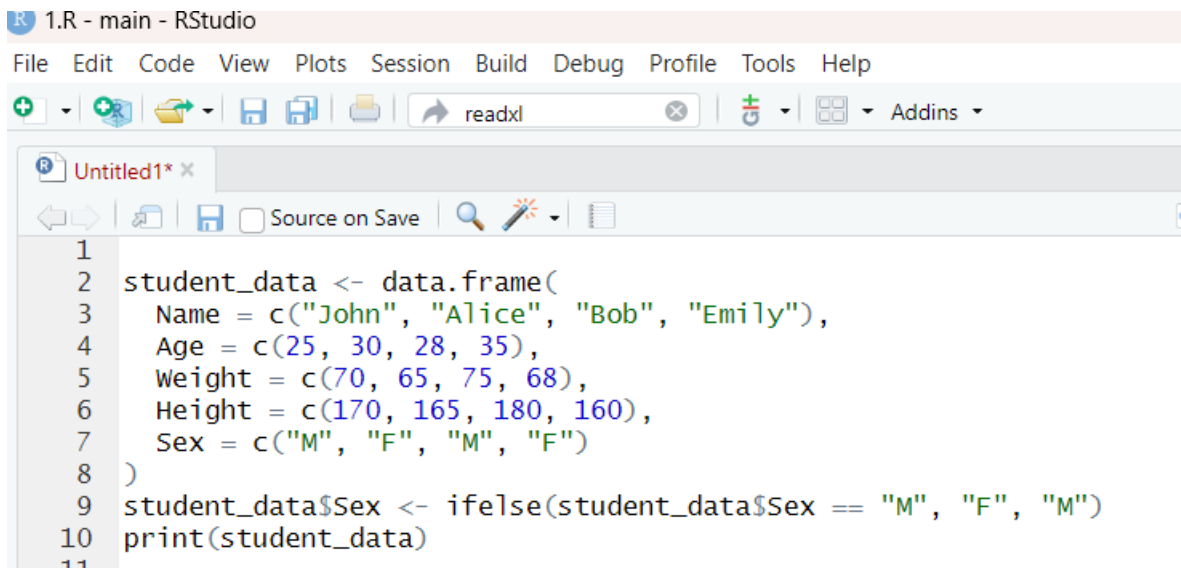
Part a:

```
w <- u + v
w <- w / 2
w <- w + u
```

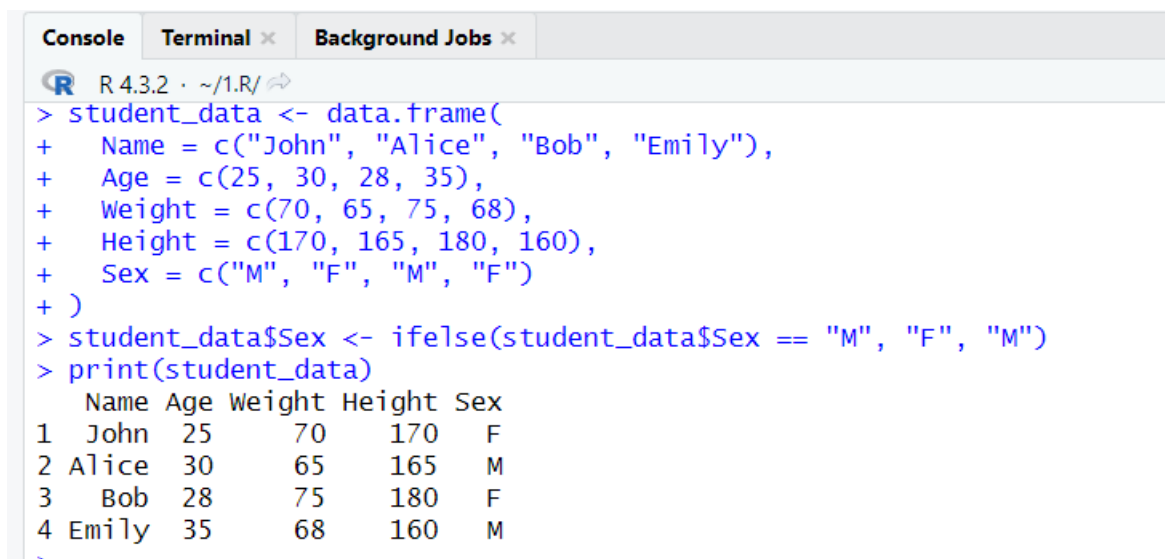
Part b:

```
w1 <- u^3
w2 <- u - v
w <- w1 / w2
```

ANSWER



OUTPUT:



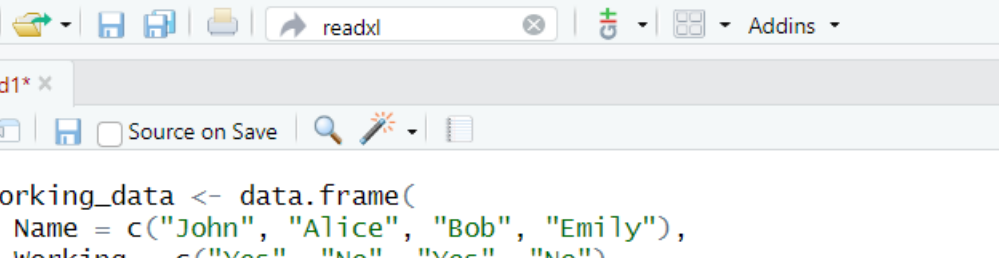
Exercise 2

Create this data frame (make sure you import the variable Working as character and not factor).

Name vs working status (Yes or No)

Add this data frame column-wise to the previous one.

- How many rows and columns does the new data frame have?
- What class of data is in each column?



The screenshot shows the RStudio interface with a script editor titled 'Untitled1'. The script contains the following R code:

```
1  
2 working_data <- data.frame(  
3   Name = c("John", "Alice", "Bob", "Emily"),  
4   Working = c("Yes", "No", "Yes", "No"),  
5   stringsAsFactors = FALSE # Ensure "Working" is imported as character  
6 )  
7 combined_data <- cbind(student_data, working_data)  
8 print(combined_data)  
9 num_rows <- nrow(combined_data)  
10 num_cols <- ncol(combined_data)  
11 cat("\nNumber of rows:", num_rows, "\n")  
12 cat("Number of columns:", num_cols, "\n")  
13 column_classes <- sapply(combined_data, class)  
14 print(column_classes)  
15  
16
```

OUTPUT:

```

R 4.3.2 · ~/1.R/ ↗
> combined_data <- cbind(student_data, working_data)
> print(combined_data)
  Name Age Weight Height Sex  Name Working
1 John  25    70    170   F  John     Yes
2 Alice 30    65    165   M Alice     No
3 Bob   28    75    180   F  Bob     Yes
4 Emily 35    68    160   M Emily     No
> num_rows <- nrow(combined_data)
> num_cols <- ncol(combined_data)
> cat("\nNumber of rows:", num_rows, "\n")

Number of rows: 4
> cat("Number of columns:", num_cols, "\n")

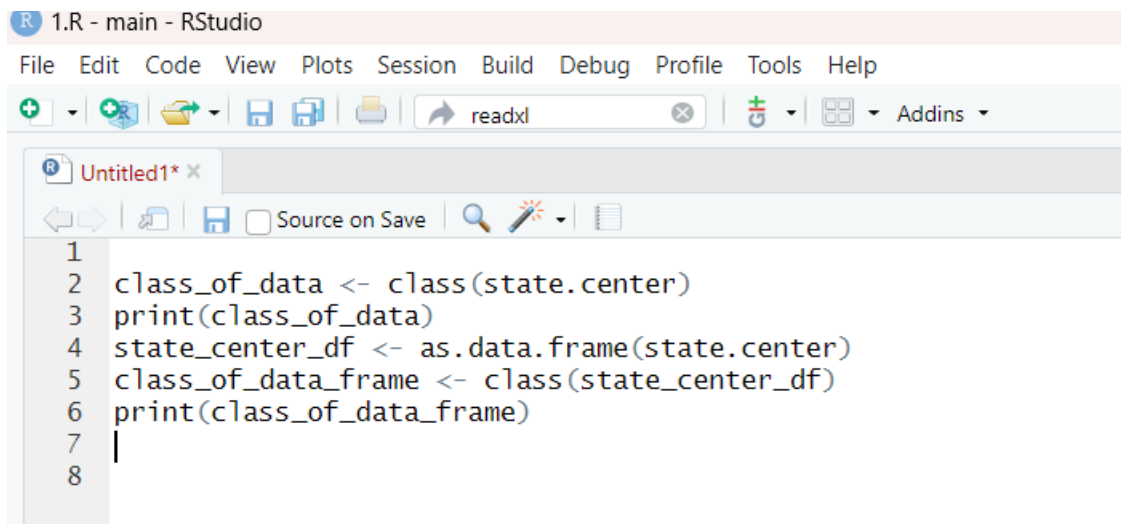
Number of columns: 7

> column_classes <- sapply(combined_data, class)
> print(column_classes)
      Name      Age      Weight      Height      Sex      Name      Working
"character" "numeric" "numeric" "numeric" "character" "character" "character"
>

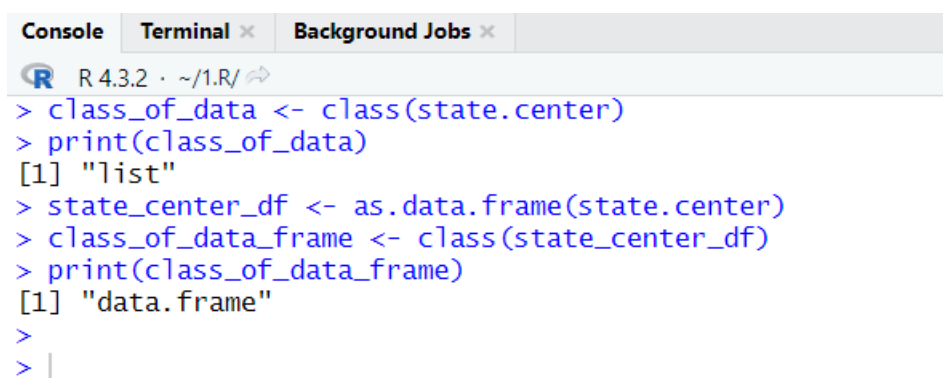
```

Exercise 3

Check what class of data is the (built-in data set) `state.center` and convert it to data frame.

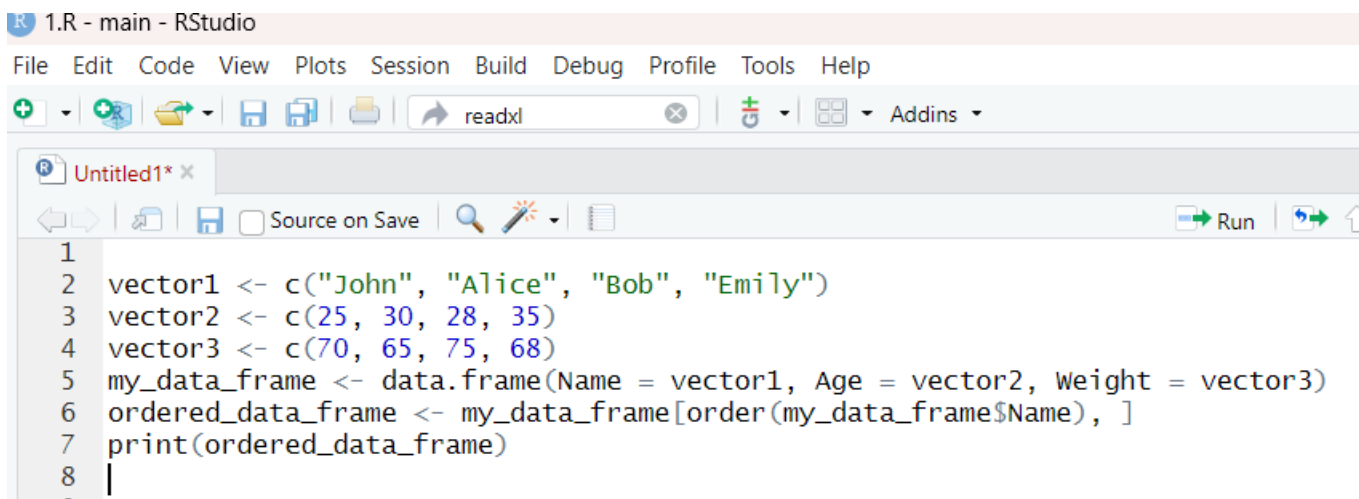


OUTPUT:



Exercise 4

Create a simple data frame from 3 vectors. Order the entire data frame by the first column..

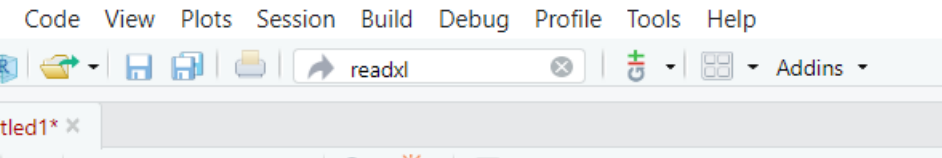


OUTPUT:

```
o:1 (top Level)
Console Terminal x Background Jobs x
R 4.3.2 · ~/1.R/ ↗
> vector1 <- c("John", "Alice", "Bob", "Emily")
> vector2 <- c(25, 30, 28, 35)
> vector3 <- c(70, 65, 75, 68)
> my_data_frame <- data.frame(Name = vector1, Age = vector2, Weight = vector3)
> ordered_data_frame <- my_data_frame[order(my_data_frame$Name), ]
> print(ordered_data_frame)
  Name Age Weight
2 Alice  30     65
3  Bob  28     75
4 Emily 35     68
1  John 25     70
>
```

Exercise 5

Create a data frame from a matrix of your choice, change the row names so every row says id_i (where i is the row number) and change the column names to variable_i (where i is the column number). I.e., for column 1 it will say variable_1, and for row 2 will say id_2 and so on.



The screenshot shows the RStudio application window. The title bar reads "R 1.R - main - RStudio". The menu bar includes "File", "Edit", "Code", "View", "Plots", "Session", "Build", "Debug", "Profile", "Tools", and "Help". The toolbar contains icons for adding files, saving, and other functions, with a search bar containing the text "readxl". Below the toolbar, the "Untitled1*" tab is active. The script editor shows the following R code:

```
1  
2 my_matrix <- matrix(1:12, nrow = 3, ncol = 4)  
3 my_dataframe <- as.data.frame(my_matrix)  
4 row_names <- paste("id", 1:nrow(my_dataframe), sep = "_")  
5 rownames(my_dataframe) <- row_names  
6 col_names <- paste("variable", 1:ncol(my_dataframe), sep = "_")  
7 colnames(my_dataframe) <- col_names  
8 print(my_dataframe)  
9
```

OUTPUT:

```
Console Terminal x Background Jobs x
R 4.3.2 · ~/1.R/ ↗
> my_matrix <- matrix(1:12, nrow = 3, ncol = 4)
> my_dataframe <- as.data.frame(my_matrix)
> row_names <- paste("id", 1:nrow(my_dataframe), sep = "_")
> rownames(my_dataframe) <- row_names
> col_names <- paste("variable", 1:ncol(my_dataframe), sep = "_")
> colnames(my_dataframe) <- col_names
> print(my_dataframe)
  variable_1 variable_2 variable_3 variable_4
id_1         1         4         7         10
id_2         2         5         8         11
id_3         3         6         9         12
>
> |
```

Exercise 6

For this exercise, we'll use the (built-in) dataset VADeaths.

- Make sure the object is a data frame, if not change it to a data frame.
- Create a new variable, named Total, which is the sum of each row.
- Change the order of the columns so total is the first variable.

```
R 1.R - main - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
+ readxl Addins
Untitled1* x
Source on Save Run
1
2 if (!is.data.frame(VADeaths)) {
3   VADeaths <- as.data.frame(VADeaths)
4 }
5
6 VADeaths$Total <- rowSums(VADeaths)
7 VADeaths <- VADeaths[, c("Total", names(VADeaths)[-ncol(VADeaths)])]
8 print(VADeaths)
9
```

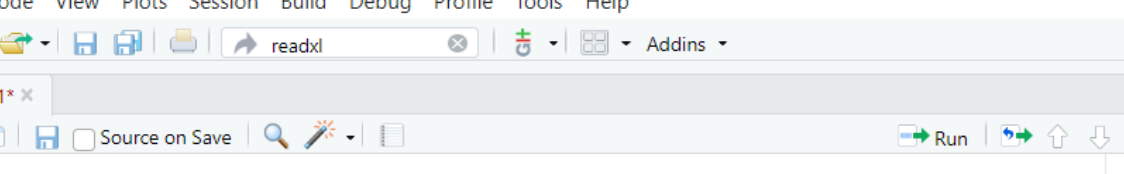
OUTPUT:

```
Console Terminal x Background Jobs x
R 4.3.2 ~ /1.R/
> if (!is.data.frame(VADeaths)) {
+   VADeaths <- as.data.frame(VADeaths)
+ }
> VADeaths$Total <- rowSums(VADeaths)
> VADeaths <- VADeaths[, c("Total", names(VADeaths)[-ncol(VADeaths)])]
> print(VADeaths)
  Total Rural Male Rural Female Urban Male Urban Female
50-54  44.2   11.7    8.7   15.4    8.4
55-59  67.7   18.1   11.7   24.3   13.6
60-64 103.5   26.9   20.3   37.0   19.3
65-69 161.6   41.0   30.9   54.6   35.1
70-74 241.4   66.0   54.3   71.1   50.0
>
>
```

Exercise 7

For this exercise we'll use the (built-in) dataset state.x77.

- Make sure the object is a data frame, if not change it to a data frame.
- Find out how many states have an income of less than 4300.
- Find out which is the state with the highest income.



The screenshot shows the RStudio interface with the following components:

- Menu Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help.
- Toolbar:** Includes icons for file operations (new, open, save, print), a search bar containing 'readxl', and a 'Run' button.
- Script Editor:** Contains the following R code:

```
1  
2 if (!is.data.frame(state.x77)) {  
3   state.x77 <- as.data.frame(state.x77)  
4 }  
5 states_less_than_4300 <- state.x77$Income < 4300  
6 num_states_less_than_4300 <- sum(states_less_than_4300)  
7 cat("Number of states with an income of less than 4300:", num_states_less_than_4300, "\n")  
8 highest_income_state <- state.x77[which.max(state.x77$Income), "State"]  
9 cat("State with the highest income:", highest_income_state, "\n")  
10  
11
```

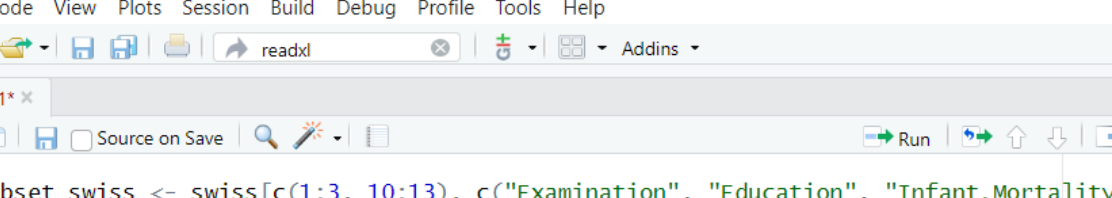
OUTPUT:

```
10:1 | (Top Level) ⌵  
Console | Terminal x | Background Jobs x  
R 4.3.2 · ~/1.R/ ↗  
> if (!is.data.frame(state.x77)) {  
+   state.x77 <- as.data.frame(state.x77)  
+ }  
> states_less_than_4300 <- state.x77$Income < 4300  
> num_states_less_than_4300 <- sum(states_less_than_4300)  
> cat("Number of states with an income of less than 4300:", num_states_less_than_4300, "\n")  
Number of states with an income of less than 4300: 20  
> highest_income_state <- state.x77[which.max(state.x77$Income), "State"]  
> cat("State with the highest income:", highest_income_state, "\n")  
State with the highest income:
```

Exercise 8

With the dataset `swiss`, create a data frame of only the rows 1, 2, 3, 10, 11, 12 and 13, and only the variables `Examination`, `Education` and `Infant.Mortality`.

- The infant mortality of Sarine is wrong, it should be a NA, change it.
- Create a row that will be the total sum of the column, name it Total.
- Create a new variable that will be the proportion of Examination ($\text{Examination} / \text{Total}$)



The screenshot shows the RStudio interface with the following components:

- Title Bar:** 1.R - main - RStudio
- Menu Bar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help
- Toolbar:** Includes icons for saving, running, and other standard RStudio functions. The text "readxl" is visible in the toolbar area.
- Source Editor:** Contains the following R code:


```
1 subset_swiss <- swiss[c(1:3, 10:13), c("Examination", "Education", "Infant.Mortality")]
2 subset_swiss[subset_swiss$Infant.Mortality == 9.7, "Infant.Mortality"] <- NA
3
4 total_row <- colSums(subset_swiss, na.rm = TRUE)
5 total_row <- c("Total", total_row)
6 subset_swiss <- rbind(subset_swiss, total_row)
7 subset_swiss$Proportion_Examination <- subset_swiss$Examination / subset_swiss$Examination["Tot
8
9 print(subset_swiss)
10
```

OUTPUT:

```
> print(subset_swiss)
```

| | Examination | Education | Infant.Mortality |
|--------------|-------------|-----------|------------------|
| Courtelay | 15 | 12 | 22.2 |
| Delemont | 6 | 9 | 22.2 |
| Franches-Mnt | 5 | 5 | 20.2 |
| Sarine | 16 | 13 | 24.4 |
| Veveyse | 14 | 6 | 24.5 |
| Aigle | 21 | 12 | 16.5 |
| Aubonne | 14 | 7 | 19.1 |
| 8 | Total | 91 | 64 |

```
> |
```

Exercise 9

Create a data frame with the datasets `state.abb`, `state.area`, `state.division`, `state.name`, `state.region`. The row names should be the names of the states.

- a) Rename the column names so only the first 3 letters after the full stop appear (e.g. States.abb will be abb).

```

1
2 data(state.abb)
3 data(state.area)
4 data(state.division)
5 data(state.name)
6 data(state.region)
7 state_data <- data.frame(
8   abb = state.abb,
9   area = state.area,
10  division = state.division,
11  name = state.name,
12  region = state.region
13 )
14 rownames(state_data) <- state_data$name
15 colnames(state_data) <- gsub("\\..*", "", colnames(state_data))
16 print(state_data)

```

OUTPUT:

Console

Terminal

Background Jobs

R 4.3.2 · ~/1.R/ ↗

print(state_data)

| | abb | area | division | name | region |
|---------------|-----|--------|--------------------|---------------|---------------|
| Alabama | AL | 51609 | East South Central | Alabama | South |
| Alaska | AK | 589757 | Pacific | Alaska | West |
| Arizona | AZ | 113909 | Mountain | Arizona | West |
| Arkansas | AR | 53104 | West South Central | Arkansas | South |
| California | CA | 158693 | Pacific | California | West |
| Colorado | CO | 104247 | Mountain | Colorado | West |
| Connecticut | CT | 5009 | New England | Connecticut | Northeast |
| Delaware | DE | 2057 | South Atlantic | Delaware | South |
| Florida | FL | 58560 | South Atlantic | Florida | South |
| Georgia | GA | 58876 | South Atlantic | Georgia | South |
| Hawaii | HI | 6450 | Pacific | Hawaii | West |
| Idaho | ID | 83557 | Mountain | Idaho | West |
| Illinois | IL | 56400 | East North Central | Illinois | North Central |
| Indiana | IN | 36291 | East North Central | Indiana | North Central |
| Iowa | IA | 56290 | West North Central | Iowa | North Central |
| Kansas | KS | 82264 | West North Central | Kansas | North Central |
| Kentucky | KY | 40395 | East South Central | Kentucky | South |
| Louisiana | LA | 48523 | West South Central | Louisiana | South |
| Maine | ME | 33215 | New England | Maine | Northeast |
| Maryland | MD | 10577 | South Atlantic | Maryland | South |
| Massachusetts | MA | 8257 | New England | Massachusetts | Northeast |
| Michigan | MI | 58216 | East North Central | Michigan | North Central |
| Minnesota | MN | 84068 | West North Central | Minnesota | North Central |
| Mississippi | MS | 47716 | East South Central | Mississippi | South |
| Missouri | MO | 69686 | West North Central | Missouri | North Central |
| Montana | MT | 147138 | Mountain | Montana | West |
| Nebraska | NE | 77227 | West North Central | Nebraska | North Central |
| Nevada | NV | 110540 | Mountain | Nevada | West |
| New Hampshire | NH | 9304 | New England | New Hampshire | Northeast |
| New Jersey | NJ | 7836 | Middle Atlantic | New Jersey | Northeast |
| New Mexico | NM | 121666 | Mountain | New Mexico | West |

| | | | | | |
|----------------|----|--------|--------------------|----------------|---------------|
| New York | NY | 49576 | Middle Atlantic | New York | Northeast |
| North Carolina | NC | 52586 | South Atlantic | North Carolina | South |
| North Dakota | ND | 70665 | West North Central | North Dakota | North Central |
| Ohio | OH | 41222 | East North Central | Ohio | North Central |
| Oklahoma | OK | 69919 | West South Central | Oklahoma | South |
| Oregon | OR | 96981 | Pacific | Oregon | West |
| Pennsylvania | PA | 45333 | Middle Atlantic | Pennsylvania | Northeast |
| Rhode Island | RI | 1214 | New England | Rhode Island | Northeast |
| South Carolina | SC | 31055 | South Atlantic | South Carolina | South |
| South Dakota | SD | 77047 | West North Central | South Dakota | North Central |
| Tennessee | TN | 42244 | East South Central | Tennessee | South |
| Texas | TX | 267339 | West South Central | Texas | South |
| Utah | UT | 84916 | Mountain | Utah | West |
| Vermont | VT | 9609 | New England | Vermont | Northeast |
| Virginia | VA | 40815 | South Atlantic | Virginia | South |
| Washington | WA | 68192 | Pacific | Washington | West |
| West Virginia | WV | 24181 | South Atlantic | West Virginia | South |
| Wisconsin | WI | 56154 | East North Central | Wisconsin | North Central |
| Wyoming | WY | 97914 | Mountain | Wyoming | West |

>

Exercise 10

Add the previous data frame column-wise to state.x77

a) Remove the variable div.

b) Also remove the variables Life Exp, HS Grad, Frost, abb, and are.

c) Add a variable to the data frame which should categorize the level of illiteracy:

[0,1) is low, [1,2) is some, [2, inf) is high.

d) Find out which state from the west, with low illiteracy, has the highest income, and what that income is

```
combined_data <- cbind(state.x77, state_data)
```

```
combined_data <- combined_data[, !names(combined_data) %in% "div"]
```

```
combined_data <- combined_data[, !names(combined_data) %in% c("Life Exp", "HS Grad", "Frost", "abb", "area")]
```

```
combined_data$Illiteracy_Level <- cut(combined_data$Illiteracy, breaks = c(0, 1, 2, Inf), labels = c("low", "some", "high"), right = FALSE)
```

```
state_with_highest_income <- combined_data[combined_data$Illiteracy_Level == "low" & combined_data$Region == "West", ]
```

```
state_with_highest_income <- state_with_highest_income[state_with_highest_income$Income == max(state_with_highest_income$Income), c("State", "Income")]
```

```
print(state_with_highest_income)
```

```
1.R - main - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
+ + + + + readxl + + Addins
Untitled1* x
Source on Save Run Source
1
2 combined_data <- cbind(state.x77, state_data)
3
4 combined_data <- combined_data[, !names(combined_data) %in% "div"]
5
6
7 combined_data <- combined_data[, !names(combined_data) %in% c("Life Exp", "HS Grad", "Frost", "
8
9 combined_data$Illiteracy_Level <- cut(combined_data$Illiteracy, breaks = c(0, 1, 2, Inf), label
10
11 # Find the state from the west with low illiteracy and highest income
12 state_with_highest_income <- combined_data[combined_data$Illiteracy_Level == "low" & combined_d
13 state_with_highest_income <- state_with_highest_income[state_with_highest_income$Income == max(
14 print(state_with_highest_income)
15 |
16
```

OUTPUT:

```
> print(state_with_highest_income)
[1] Population      Income      Illiteracy      Murder      Area
[6] division        name        region          Illiteracy_Level
<0 rows> (or 0-length row.names)
>
~
```