HW1

Lubin

2024-10-19

Работа с данными

Загрузим данные в переменную (датафрейм).

```
data.df <-read.table("https://people.math.umass.edu/~anna/Stat597AFall2016/rnf6080.dat")
head(data.df)
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
## 1 60  4  1  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
## 2 60  4  2  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
## 3 60  4  3  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
## 4 60  4  4  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
## 5 60  4  5  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
## 6 60  4  6  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
##   V22 V23 V24 V25 V26 V27
## 1   0   0   0   0   0   0
## 2   0   0   0   0   0   0
## 3   0   0   0   0   0   0
## 4   0   0   0   0   0   0
## 5   0   0   0   0   0   0
## 6   0   0   0   0   0   0
```

Определим количество строк и столбцов в датафрейме.

```
cat("Количество столбцов -", ncol(data.df), "\n")
```

```
## Количество столбцов - 27
```

```
cat("Количество строк -", nrow(data.df))
```

```
## Количество строк - 5070
```

Выведем имена колонок в датафрейме.

```
cat(colnames(data.df))
```

```
## V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
V22 V23 V24 V25 V26 V27
```

Выведем значение из пятой строки седьмого столбца.

```
data.df[5, 7]
```

```
## [1] 0
```

Выведем вторую строку в датафрейме.

```
data.df[2,]
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
## 2 60  4  2  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
##   V22 V23 V24 V25 V26 V27
## 2   0   0   0   0   0   0
```

Команда “names(data.df) <- c(”year”, “month”, “day”, seq(0,23))” изменяет имена столбцов на “year”, “month”, “day”, “0” - “23”.

```
names(data.df) <- c("year", "month", "day", seq(0, 23))
```

Просмотрим таблицу с помощью “head” и “tail”.

```
head(data.df)
```

```
##   year month day 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## 1   60     4   1 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 2   60     4   2 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 3   60     4   3 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 4   60     4   4 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 5   60     4   5 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 6   60     4   6 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

```
tail(data.df)
```

```
##      year month day 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
## 5065   80    11  25 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 5066   80    11  26 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 5067   80    11  27 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 5068   80    11  28 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 5069   80    11  29 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 5070   80    11  30 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
##      23
## 5065  0
## 5066  0
## 5067  0
## 5068  0
## 5069  0
## 5070  0
```

Последние 24 столбца представляют собой часы, а первые 3 - год, месяц, день соответственно. Предположительно в этих столбцах находится информация о количестве атмосферных осадков, выпавших в течение часа (номер часа указан в названии колонки). Можно заметить, что в этом наборе данных для большинства годов отсутствует информация об осадках, выпавших в зимние месяца и в марте (исчисление месяцев начинается с 4 и заканчивается 11).

Добавим новый столбец “daily”, в который запишем сумму крайних правых 24 столбцов.

```
data.df$daily <- rowSums(data.df[4:27])
head(data.df, n=20)
```

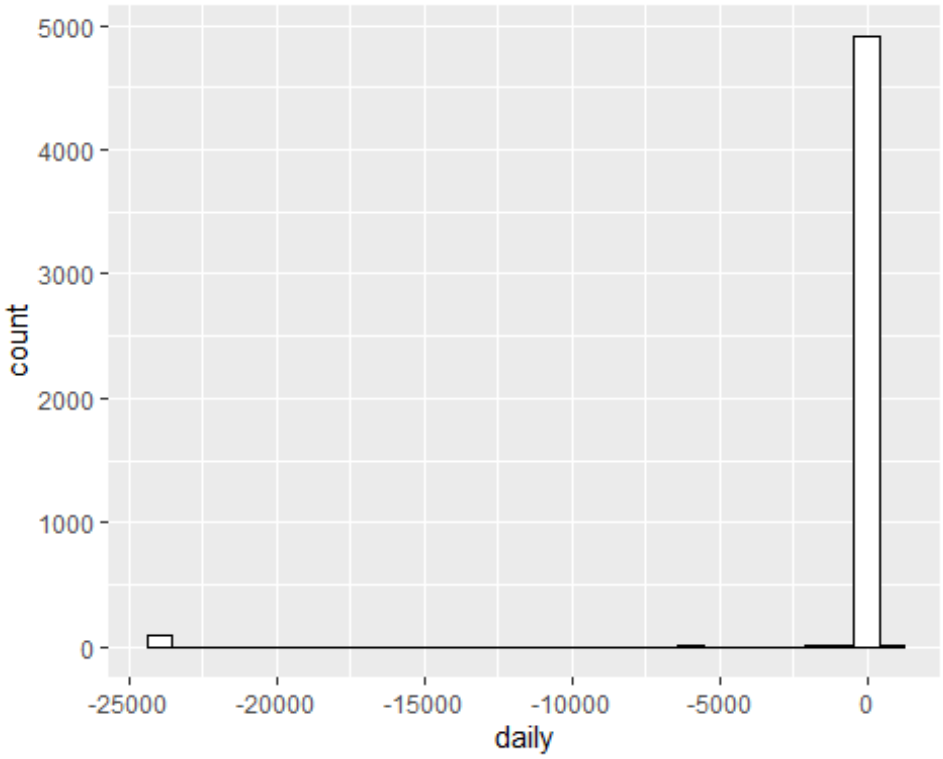
```
##    year month day 0 1 2 3 4 5  6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
## 1    60     4   1 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 2    60     4   2 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 3    60     4   3 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

```
## 4    60     4   4 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 5    60     4   5 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 6    60     4   6 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 7    60     4   7 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 8    60     4   8 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 9    60     4   9 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 10   60     4  10 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 11   60     4  11 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 12   60     4  12 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 13   60     4  13 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 14   60     4  14 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 15   60     4  15 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 16   60     4  16 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 17   60     4  17 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 18   60     4  18 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 19   60     4  19 0 0 0 0 0 0  0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
## 20   60     4  20 0 0 0 0 8 0 13 8 3 0  0  0  0  0  0  0  0  0  0  0  0  0  0
##    23 daily
## 1   0     0
## 2   0     0
## 3   0     0
## 4   0     0
## 5   0     0
## 6   0     0
## 7   0     0
## 8   0     0
## 9   0     0
## 10  0     0
## 11  0     0
## 12  0     0
## 13  0     0
## 14  0     0
## 15  0     0
## 16  0     0
## 17  0     0
## 18  0     0
## 19  0     0
## 20  0    32
```

Построим гистограмму по столбцу “daily”

```
library("ggplot2")
ggplot(data=data.df, aes(x=daily)) +
geom_histogram(color="black", fill="white")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



По полученной гистограмме можно сделать вывод, что с 1960 по 1980 год в Канаде во всех месяцах, кроме зимних и марта, практически не выпадало осадков (в сумме около 5000 дней не было осадков). Также можно заметить отрицательные значения количества осадков в столбце “daily”, что является не нормальным. Они портят восприятие гистограммы.

Выведем строки, в которых значение столбца “daily” отрицательно, чтобы посмотреть на значения в других столбцах. Также получим общую информацию о значениях в столбцах с 4 по 27 с помощью функции “summary”.

```
head(subset(data.df[4:28], daily < 0), n=20)
```

```
##         0    1    2    3    4    5    6    7    8    9   10   11   12   13   14
## 495     0    0    0    0    0    0    0    0    0    0 -999 -999 -999 -999    5
## 675     0    0    0    0    0    0    0    0    0    0 -999 -999   18   10   18
## 678     0    0    0    3   10   15   23    8   10    0    0    0    0    0 -999
## 718     0    0    5    0   18    0   13    3   13   18    5    8    3   10   25
## 1870    3    0    3   13    3    5   20   20   38   43   20   18   15 -999 -999
## 2119    0    0    0    0    0    0    0 -999    5    0    0    0    0    0    0
## 2244 -999    0    0    0    0    0    0    0    0    0    0    0    0    0 -999
## 2259 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2261    0    0    0    0    0    0    0    0    0 -999 -999 -999 -999 -999 -999
## 2262 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2263 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2264 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2265 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2266 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2267 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2268 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2269 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2270 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
## 2271 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
```

```
## 2272 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999 -999
##       15   16   17   18   19   20   21   22   23  daily
## 495  -999 -999    0    3    0    0    0    0    0  -5986
## 675     8    0    0    0    0    0    0    0    0  -1944
## 678  -999    0    0    0    0    0    0    0    0  -1929
## 718    18   23   25   10   13   23   15 -999    0   -751
## 1870 -999 -999    0    0    0    0    0    0    0  -3795
## 2119    0    0    0    0    0    0    0    0    0   -994
## 2244    3    0    0 -999    0    0    0    0    0  -2994
## 2259 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2261 -999 -999 -999 -999 -999 -999 -999 -999 -999 -14985
## 2262 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2263 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2264 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2265 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2266 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2267 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2268 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2269 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2270 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2271 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
## 2272 -999 -999 -999 -999 -999 -999 -999 -999 -999 -23976
```

```
summary(data.df[4:27])
```

```
##        0                  1                  2                  3          
##  Min.   :-999.00   Min.   :-999.00   Min.   :-999.00   Min.   :-999.00  
##  1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00  
##  Median :   0.00   Median :   0.00   Median :   0.00   Median :   0.00  
##  Mean   : -20.88   Mean   : -21.16   Mean   : -21.19   Mean   : -20.83  
##  3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00  
##  Max.   : 325.00   Max.   : 323.00   Max.   : 239.00   Max.   : 119.00  
##        4                  5                  6                  7          
##  Min.   :-999.00   Min.   :-999.00   Min.   :-999.00   Min.   :-999.00  
##  1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00  
##  Median :   0.00   Median :   0.00   Median :   0.00   Median :   0.00  
##  Mean   : -20.86   Mean   : -21.08   Mean   : -21.67   Mean   : -21.19  
##  3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00  
##  Max.   : 198.00   Max.   : 348.00   Max.   : 173.00   Max.   : 394.00  
##        8                  9                 10                 11          
##  Min.   :-999.0   Min.   :-999.00   Min.   :-999.00   Min.   :-999.00  
##  1st Qu.:   0.0   1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00  
##  Median :   0.0   Median :   0.00   Median :   0.00   Median :   0.00  
##  Mean   : -20.9   Mean   : -20.73   Mean   : -21.18   Mean   : -20.97  
##  3rd Qu.:   0.0   3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00  
##  Max.   : 320.0   Max.   : 259.00   Max.   : 109.00   Max.   : 130.00  
##        12                 13                 14                15          
##  Min.   :-999.00   Min.   :-999.00   Min.   :-999.0   Min.   :-999.0  
##  1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.0   1st Qu.:   0.0  
##  Median :   0.00   Median :   0.00   Median :   0.0   Median :   0.0  
##  Mean   : -19.89   Mean   : -20.56   Mean   : -21.4   Mean   : -20.9  
##  3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.0   3rd Qu.:   0.0  
##  Max.   : 203.00   Max.   : 218.00   Max.   : 297.0   Max.   : 183.0  
##        16                 17                 18                 19          
##  Min.   :-999.00   Min.   :-999.00   Min.   :-999.00   Min.   :-999.00
```

```
##  1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00
##  Median :   0.00   Median :   0.00   Median :   0.00   Median :   0.00
##  Mean   : -20.83   Mean   : -19.97   Mean   : -20.18   Mean   : -19.81
##  3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00
##  Max.   : 183.00   Max.   : 345.00   Max.   : 393.00   Max.   : 208.00
##        20                21                22                23
##  Min.   :-999.00   Min.   :-999.00   Min.   :-999.00   Min.   :-999.00
##  1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00   1st Qu.:   0.00
##  Median :   0.00   Median :   0.00   Median :   0.00   Median :   0.00
##  Mean   : -19.64   Mean   : -19.74   Mean   : -19.54   Mean   : -19.32
##  3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00   3rd Qu.:   0.00
##  Max.   : 155.00   Max.   : 178.00   Max.   : 381.00   Max.   : 279.00
```

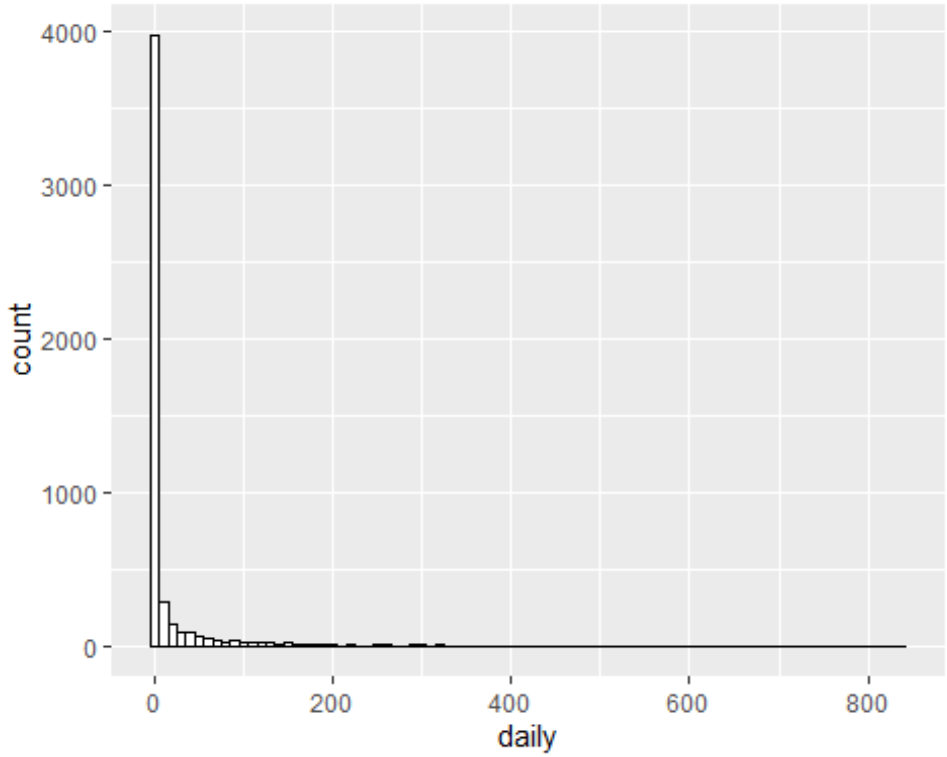
Можно сделать вывод, что некоторые столбцы содержат странное значение “-999”, из-за которого и получается отрицательное значение в столбце “daily”. Скорее всего “-999” ставится, когда данные о количестве осадков за данный час отсутствуют.

Создадим новый датафрейм из старого, заменив все значения “-999” на 0. Далее пересчитаем значения в столбце “daily”.

```
fixed.df <- data.df[1:27]
fixed.df[fixed.df == -999] <- 0
fixed.df$daily <- rowSums(fixed.df[4:27])
```

Теперь построим новую гистограмму

```
library("ggplot2")
ggplot(data=fixed.df, aes(x=daily)) +
geom_histogram(binwidth=10, color="black", fill="white")
```



Новая гистограмма более коректна, так как в ней отсутствуют отрицательные значения количества осадков (количество осадков не может быть отрицательным).

Синтаксис и типизирование

Для каждой строки кода поясните полученный результат, либо объясните почему она ошибочна.

Инициализируем переменную “v”.

```
v <- c("4", "8", "15", "16", "23", "42")
```

Здесь функция “max()”, которая находит максимальное значение в векторе, вывела 8, а не 42, так как в качестве аргумента был задан вектор символов. В случае с вектором символов “max()” сравнивает ASCII коды элементов и выводит элемент с максимальным ASCII кодом. В нашем случае это “8”. “8” больше “42”, потому что “8” больше “4” (строки сравниваются посимвольно, если ASCII код n-го символа первой строки больше кода n-го символа второй строки, то первая строка считается больше второй, то есть “8” < “81”, “4” > “39”, “abc” < “adc” и т. д.).

```
max(v)
```

```
## [1] "8"
```

Фукнция “sort()” работает с векторами символов (строк) аналогично “max()”. В нашем случае элементы вектора были отсортированы в порядке возрастания.

```
sort(v)
```

```
## [1] "15" "16" "23" "4"  "42" "8"
```

sum(v) - ошибка

Функция “sum()” не работает с векторами, элементы которых имеют тип данных “character” (“sum()” принимает на вход вектора с элементами типов “numeric”, “complex”, “logical”). Поэтому выскочила ошибка.

Для следующих наборов команд поясните полученный результат, либо объясните почему они ошибочна.

Инициализируем переменную v2

```
v2 <- c("5", 7, 12)
```

v2[2] + v[3] - ошибка

Вектор может содержать данные только одного типа. В результате создания вектора из переменных “5”, 7, 12 с помощью функции “c()” все его элементы стали иметь тип “character”. Оператор “+” не может работать с переменными типа “character”, из-за чего выскочила ошибка.

В отличие от вектора датафрейм может содержать элементы разных типов. Поэтому при его создании числа 7, 12 не изменили тип на “character”.

```
df3 <- data.frame(z1="5",z2=7,z3=12)
df3[1,2] + df3[1,3]
```

```
## [1] 19
```

В данном случае были выбраны и сложены элементы списка (список также может содержать элементы разных типов).

```
l4 <- list(z1="6", z2=42, z3="49", z4=126)
l4[[2]] + l4[[4]]
```

```
## [1] 168
```

l4[2] + l4[4] - ошибка

При использовании “[]” возвращается элемент списка, как список длиной один. Поэтому при попытке сложения выскакивает ошибка. Для того, чтобы добраться до самого элемента нужно использовать “[[]]” (как это было сделано в случае выше).

```
is.numeric(l4[2])
```

```
## [1] FALSE
```

```
is.numeric(l4[[2]])
```

```
## [1] TRUE
```

Работа с функциями и операторами

С помощью функции “seq()” вывести:

Последовательность чисел от 1 до 10000 с инкрементом 372

```
seq(from=1, to=10000, by=372)
```

```
##  [1]    1  373  745 1117 1489 1861 2233 2605 2977 3349 3721 4093 4465 4837
5209
## [16] 5581 5953 6325 6697 7069 7441 7813 8185 8557 8929 9301 9673
```

Последовательность чисел от 1 до 10000 длиной 50

```
seq(from=1, to=10000, length.out=50)
```

```
##  [1]     1.0000   205.0612   409.1224   613.1837   817.2449  1021.3061
##  [7]  1225.3673  1429.4286  1633.4898  1837.5510  2041.6122  2245.6735
## [13]  2449.7347  2653.7959  2857.8571  3061.9184  3265.9796  3470.0408
## [19]  3674.1020  3878.1633  4082.2245  4286.2857  4490.3469  4694.4082
## [25]  4898.4694  5102.5306  5306.5918  5510.6531  5714.7143  5918.7755
## [31]  6122.8367  6326.8980  6530.9592  6735.0204  6939.0816  7143.1429
## [37]  7347.2041  7551.2653  7755.3265  7959.3878  8163.4490  8367.5102
## [43]  8571.5714  8775.6327  8979.6939  9183.7551  9387.8163  9591.8776
## [49]  9795.9388 10000.0000
```

Объяснить разницу между rep(1:5,times=3) и rep(1:5, each=3)

Разница между rep(1:5,times=3) и rep(1:5, each=3) заключается в том, что в первом случае вся последовательность (1, 2, 3, 4, 5) повторяется три раза, а во втором случае каждый элемент последовательности повторяется по три раза.

```
rep(1:5, times=3)
```

```
##  [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
```

```
rep(1:5, each=3)
```

```
##  [1] 1 1 1 2 2 2 3 3 3 4 4 4 5 5 5
```