HW-1

Mineev

2024-10-14

Работа с данными

Загрузка данных:

```
data.df <- read.table("https://people.math.umass.edu/~anna/Stat597AFall2016/rnf6080.dat")
cat("Количество строк: ", ncol(data.df))
```

```
## Количество строк:  27
```

```
cat("\nКоличество столбцов: ", nrow(data.df))
```

```
##
## Количество столбцов:  5070
```

Данные загружены правильно

Имена колонок:

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

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

Значение 5 строки 7 столбца:

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

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

2 строка:

```
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
```

Замена заголовков

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

Данная строка заменяет заголовки столбцов на “year”, “month”, “day”, 0, 1, 2, … 23

Начало таблицы:

```
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
## 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
```

Конец таблицы:

```
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
```

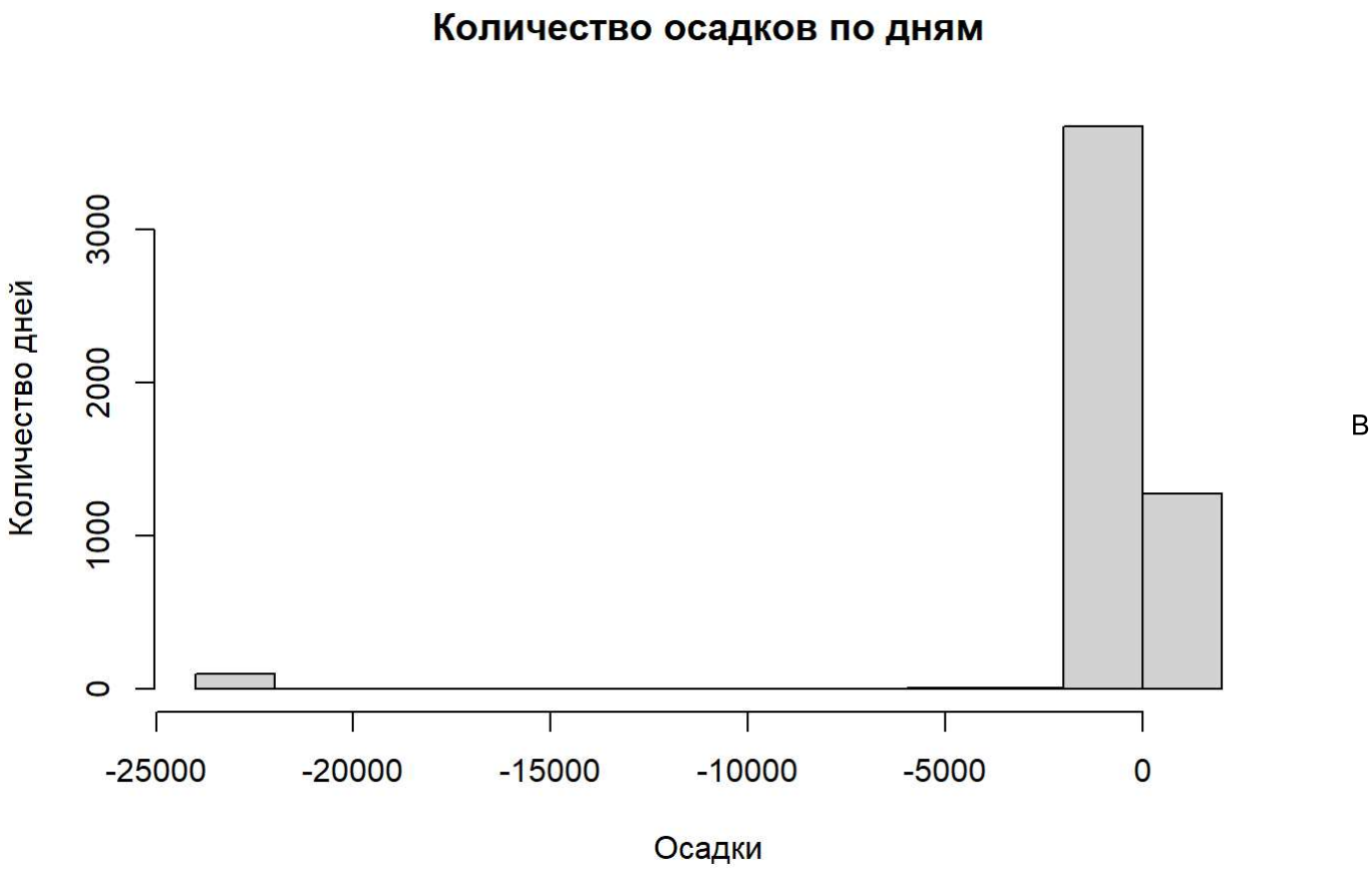
Последние 24 колонки - количество осадков по часам в течение дня.

Добавление колонки и построение гистограммы

```
data.df$daily <- rowSums(data.df[, 4:27])
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
##   daily
## 1     0
## 2     0
## 3     0
## 4     0
## 5     0
## 6     0
```

```
hist(data.df$daily, main = "Количество осадков по дням", xlab = "Осадки", ylab = "Количество дней")
```

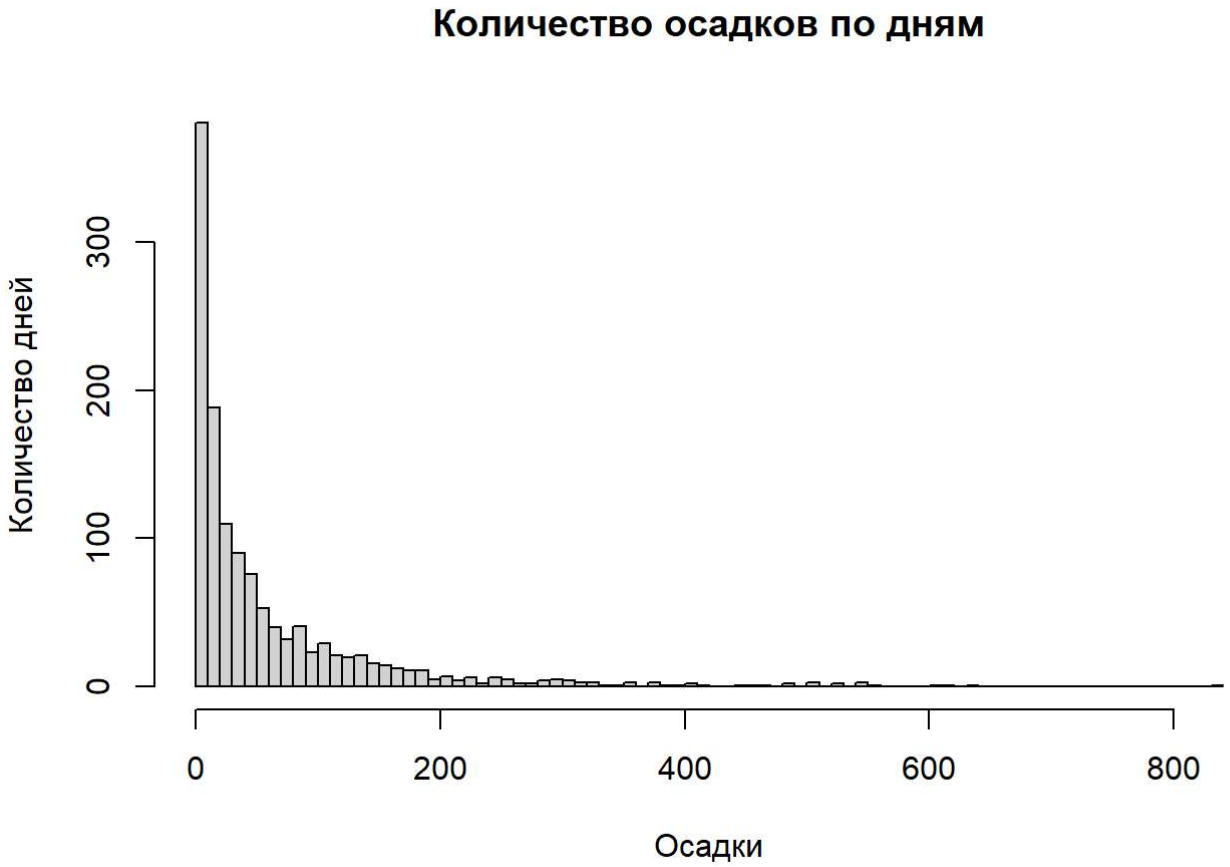


В

датафрейме присутствуют некорректные значения (-999)

Новый датафрейм

```
fixed.df <- data.df[data.df$daily > 0, ]
hist(fixed.df$daily, main = "Количество осадков по дням", xlab = "Осадки", ylab = "Количество дней", breaks = 80)
```



Новый датафрейм не содержит отрицательных значений.Также не учитываются дни без осадков. Новая гистограмма более корректна, так как она построена по адекватным значениям.

Синтаксис и типизирование

1

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

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

Данная функция ищет максимальный элемент вектора. Так как элементы вектора являются строками, сравнение элементов осуществляется по первому символу.

```
sort(v)
```

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

Элементы вектора были отсотртирвоаны по возрастанию. Так как элементы вектора являются строками, сравнение элементов осуществляется по первому символу, а вслучае их равенства по второму, …

```
#sum(v)
```

Функция sum() не может использоваться для строк.

2

```
v2 <- c("5", 7, 12)
#v2[2] + v2[3]
```

Вектор хранит данные одного типа, поэтому при наличии нескольких типов данных используется самый сильный из них (в данном случае - строковый)

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

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

Данная строка выводит сумму элемента в первой строке, втором столбце и элемента в первой строке, третьем столбце

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

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

```
#l4[2] + l4[4]
```

В первом случае: выводится сумма значений второго и четвёртого элементов списка Во второс случае: выводится сумма второго и четвёртого элемента списка, что невозможно, так как элементы списка не являются числами

Работа с функциями и операторами

1

Числа от 1 до 10000 с инкрементом 372:

```
seq(1, 100000, by=372)
```

```
##   [1]     1   373   745  1117  1489  1861  2233  2605  2977  3349  3721  4093
##  [13]  4465  4837  5209  5581  5953  6325  6697  7069  7441  7813  8185  8557
##  [25]  8929  9301  9673 10045 10417 10789 11161 11533 11905 12277 12649 13021
##  [37] 13393 13765 14137 14509 14881 15253 15625 15997 16369 16741 17113 17485
##  [49] 17857 18229 18601 18973 19345 19717 20089 20461 20833 21205 21577 21949
##  [61] 22321 22693 23065 23437 23809 24181 24553 24925 25297 25669 26041 26413
##  [73] 26785 27157 27529 27901 28273 28645 29017 29389 29761 30133 30505 30877
##  [85] 31249 31621 31993 32365 32737 33109 33481 33853 34225 34597 34969 35341
##  [97] 35713 36085 36457 36829 37201 37573 37945 38317 38689 39061 39433 39805
## [109] 40177 40549 40921 41293 41665 42037 42409 42781 43153 43525 43897 44269
## [121] 44641 45013 45385 45757 46129 46501 46873 47245 47617 47989 48361 48733
## [133] 49105 49477 49849 50221 50593 50965 51337 51709 52081 52453 52825 53197
## [145] 53569 53941 54313 54685 55057 55429 55801 56173 56545 56917 57289 57661
## [157] 58033 58405 58777 59149 59521 59893 60265 60637 61009 61381 61753 62125
## [169] 62497 62869 63241 63613 63985 64357 64729 65101 65473 65845 66217 66589
## [181] 66961 67333 67705 68077 68449 68821 69193 69565 69937 70309 70681 71053
## [193] 71425 71797 72169 72541 72913 73285 73657 74029 74401 74773 75145 75517
## [205] 75889 76261 76633 77005 77377 77749 78121 78493 78865 79237 79609 79981
## [217] 80353 80725 81097 81469 81841 82213 82585 82957 83329 83701 84073 84445
## [229] 84817 85189 85561 85933 86305 86677 87049 87421 87793 88165 88537 88909
## [241] 89281 89653 90025 90397 90769 91141 91513 91885 92257 92629 93001 93373
## [253] 93745 94117 94489 94861 95233 95605 95977 96349 96721 97093 97465 97837
## [265] 98209 98581 98953 99325 99697
```

Числа от 1 до 10000 длиной 50

```
seq(1, 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
```

2

```
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
```

В первом случае вектор целиком повторяется 3 раза, во втором - каждый элемент вектора повторяется по 3 раза