hw01

Mosunov Rodion RI-411055

2024-10-18

Работа с данными

Загрузим набор данных в датафрейм с помощью функции read.table(), выведим первые 6 строк

```
data.df <- read.table("rnf6080.dat.txt", header=FALSE)
print(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
```

Посмотрим сколько у нас наблюдений и столбцов с помощью функции dim()

```
print(dim(data.df))
```

```
## [1] 5070   27
```

Получим имена колонок датафрейма с помощью функции colnames()

```
print(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"
```

Найдем значение 5 строки 7 столбца

```
print(data.df[5, 7])
```

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

Напечатаем 2 строку из датафрейма

```
print(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
```

Следующая команда names(data.df) <- c("year", "month", "day", seq(0,23)) присваивает название каждой колнке датафрейма: 1-3 колонка - это year, month, day, а оставшиеся имеют имя от 0 до 23

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

Функции head() и tail() позволяют просмотреть первые и последние 6 строк. Последние 24 колонки представляют значения осадков по часам, нумерация колонок соответствует своему часу

```
print(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
```

```
print(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
```

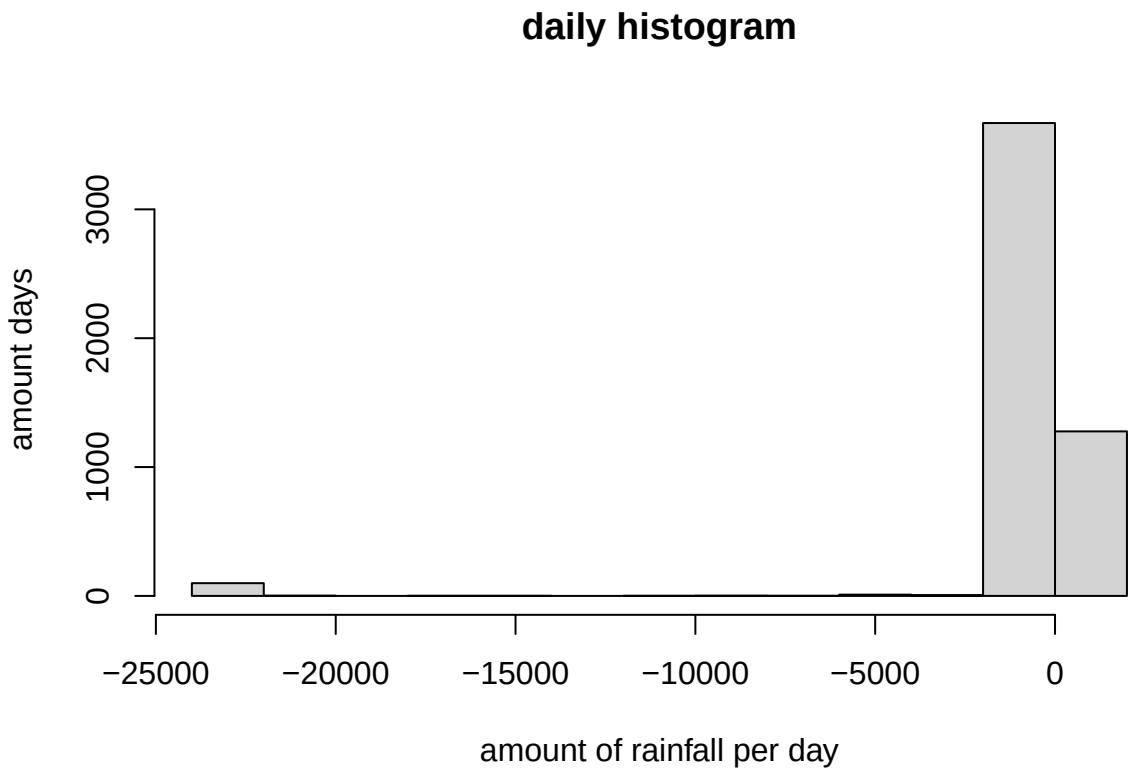
Добавим в таблицу новую колонку daily и запишем в нее суммы последних 24 колонок и выведем ее

```
data.df$daily <- rowSums(data.df[, (ncol(data.df)-23):ncol(data.df)], na.rm = TRUE)
print(head(data.df['daily']))
```

```
##   daily
## 1     0
## 2     0
## 3     0
## 4     0
## 5     0
## 6     0
```

Построим гистограмму по колонке daily.

```
hist(data.df$daily,
     main = "daily histogram",
     xlab = "amount of rainfall per day",
     ylab = "amount days")
```



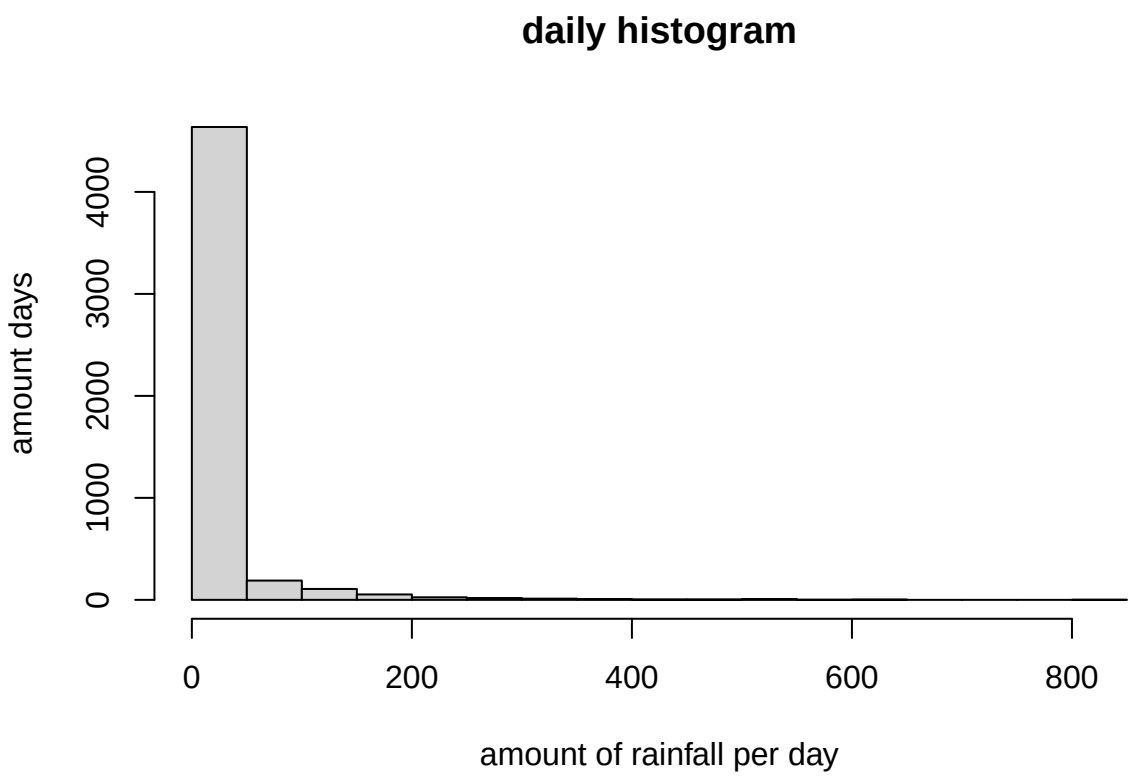
Поскольку в daily содержится сумма осадков в течении дня, то мы должны получить гистограмму, которая показывает по оси X - колчиство осадков в течении дня, а по оси Y - количество дней с соответствующим количеством осадков.
Просмотрев набор данных, было замечено, что в нем содержатся значения -999, что является некорректными данными. Их необходимо исправить.
Не зная причины возникновения таких значений, я предполагаю, что -999 означает очень маленькое

значения, поэтому я заменю -999 на 0, что будет является корректным минимальным значением для осадков.

```
fixed.df <- data.df
fixed.df[fixed.df < 0] <- 0
```

Создадим гистограмму с исправленными данными

```
hist(fixed.df$daily,
     main = "daily histogram",
     xlab = "amount of rainfall per day",
     ylab = "amount days")
```



Синтаксис и типизирование

Следующая команда создает массив символов

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

Следущая команда найдет максимальный символ в массиве, сравнение символов завсит от используемой кодировки и установленного локаля

```
max(v)
```

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

Следующая команда отсортирует массив символов, также с учетом кодировки и локаля

```
sort(v)
```

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

Следующая команда выдаст ошибку, так как она попытаемся просумировать элементы массива, однако суммировать символы нельзя

```
sum(v)
```

Следущая команда создает вектора, содержащий элементы символьного и числового типа.
Первая команда корректна, однако вторая - нет. Ошибка вызвана попыткой взять обратится к индексу целого числа, что является некорректной операцией.

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

Следующий набор команд создает датафрейм со столбцами z1, z2, z3 и присваивает им значения “5”, 7, 12 соответственно. Далее идет сложение элементов 2 и 3 столбца первой строки.

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

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

В следующем наборе команд происходит создание списка с элементами z1, z2, z3, z4 и суммирование 2 и 4 элемента. Последняя команда является некорректной, так как обращение типа l14[2] возвращает не значение 2 элемента, а подсписок вида list(z2=42)

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

Работа с функциями и операторами

Создадим последовательность числе от 1 до 10000 с шагом 372 с помощью функции seq()

```
print(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
```

Создадим последовательность чисел от 1 до 10000 содержащая 50 симовлов, используя аргумент length.out, который равномерно распределит числа в последовательности

```
print(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
```

Разница между двумя приведенными ниже командами заключается в том, что первая команда повторяет заданную последовательность 3 раза, а вторая команда повторяет каждый элемент последовательности 3 раза, прежде чем выведет следующий элемент

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

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

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

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