Домашнее задание №1

## Работа с данными

1. Загрузите данные в датафрейм, который назовите data.df

data.df <- read.table("http://www.stats.uwo.ca/faculty/braun/data/rnf6080.dat")

1. Сколько строк и столбцов в data.df? Если получилось не 5070 наблюдений 27 переменных, то проверяйте аргументы

paste("data.df loaded: ", nrow(data.df), " observations and ", ncol(data.df), " variables")

## [1] "data.df loaded: 5070 observations and 27 variables"

Импорт данных прошел успешно!

1. Получите имена колонок из data.df

colnames(data.df)

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

1. Найдите значение из 5 строки седьмого столбца

data.df[5, 7]

## [1] 0

1. Напечатайте целиком 2 строку из data.df

data.df[2,]

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

1. Объясните, что делает следующая строка кода names(data.df) <- c("year", "month", "day", seq(0,23))

names(data.df)

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

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

Данная строка изменяет название колонок

1. Воспользуйтесь функциями head и tail, чтобы просмотреть таблицу. Что представляют собой последние 24 колонки?

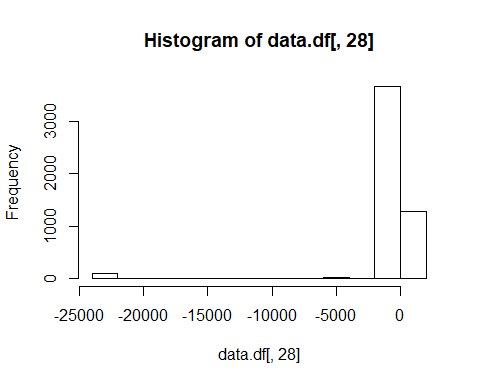
tail(data.df)

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

Последние 24 колонки показывают количество осадков по часам

1. Добавьте новую колонку с названием daily, в которую запишите сумму крайних правых 24 колонок. Постройте гистограмму по этой колонке. Какие выводы можно сделать?

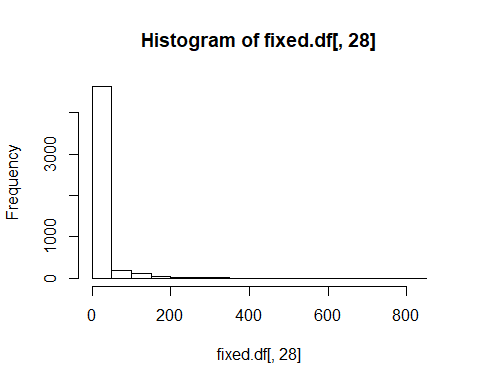
data.df$daily <- rowSums(data.df[, 4:27])  
hist(data.df[, 28])



В данных присутствуют ошибки (выбросы)

1. Создайте новый датафрейм fixed.df в котром исправьте замеченную ошибку. Постройте новую гистограмму, поясните почему она более корректна

fixed.df <- data.df  
fixed.df$daily[which(fixed.df$daily < 0)] = 0  
hist(fixed.df[, 28])



Некорректные данные были обнулены. На новой гистограмме отсутствуют некорректные (отрицательные) значения количества осадков

## Синтаксис и типизирование

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

v <- c("4", "8", "15", "16", "23", "42") #вектору присваиваются символы, а не числа, соответсвенно дальнейшие действия будут произведены с кодами соответствующих символов  
max(v)

## [1] "8"

sort(v)

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

sum(as.integer(v)) # посчитать сумму от символов невозможно

## [1] 108

Исправить проблему можно 2 способами: либо присвоить вектору числа, либо рассматривать каждый элемент вектора как число

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

v2 <- c("5", 7, 12)  
as.integer(v2[2]) + as.integer(v2[3]) # Все элементы вектора должны быть одного типа, поэтому в данном случае они инициализируются как \*char\*, а значит арифметические операции неприменимы

## [1] 19

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

## [1] 19

# С dataframe и list дела обстоят иначе: элементы разных типов можно встретить в пределах одного фрейма  
l4 <- list(z1 = "6", z2 = 42, z3 = "49", z4 = 126)  
l4[[2]] + l4[[4]]

## [1] 168

as.integer(l4[2]) + as.integer(l4[4])

## [1] 168

В списках *list* адресация [] - выводит список элементов по индексу, а [[]] - сам элемент

## Работа с функциями и операторами

1. выведите на экран:
   * Числа 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  
## [15] 5209 5581 5953 6325 6697 7069 7441 7813 8185 8557 8929 9301 9673

+ Числа 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

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

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

Очевидно что первый вариант повторяет всю последовательность 3 раза, а второй - повторяет каждый элемент последовательности