```
# Data frame
# Друга структура за съхраняване на данни в R е data frame. Разликата с
матрицата е,
# че елементите в матрицата е вектор от вектори и всички елементи са от един
тип, докато
# data frame-а е лист от вектори и елементите на data frame-а са от един тип,
само в рамките
# на отделния вектор.


car_brand <- c('volkswagen', 'chevrolet', 'peugot', 'volvo', 'porsche',
'chevrolet', 'mitsubishi', 'mazda',
'toyota', 'subaru', 'isuzu', 'volvo', 'mercury', 'honda', 'peugot', 'porsche',
'dodge',
'volkswagen', 'bmw', 'mercedes-benz', 'toyota', 'isuzu', 'volkswagen', 'mazda',
'plymouth',
'honda', 'mitsubishi', 'plymouth', 'volkswagen', 'volkswagen', 'honda',
'toyota', 'nissan',
'nissan', 'mitsubishi', 'volvo', 'toyota', 'jaguar', 'volkswagen', 'dodge')
fuel_type <- c('gas', 'gas', 'gas', 'gas', 'gas', 'gas', 'gas', 'gas', 'gas',
'gas', 'gas', 'gas', 'gas', 'gas',
'gas', 'gas', 'gas', 'gas', 'gas', 'gas', 'diesel', 'gas', 'gas', 'diesel',
'gas', 'gas', 'gas',
'gas', 'diesel', 'gas', 'gas', 'gas', 'gas', 'diesel', 'gas', 'gas', 'gas',
'gas', 'gas', 'gas')
horsepower <- c(111, 154, 102, 115, 110, 140, 160, 101, 121, 182, 48, 70, 68,
102, 88, 145, 58, 76, 60, 86,
101, 100, 78, 70, 90, 176, 262, 68, 101, 135, 84, 64, 120, 72, 123, 155, 184,
175, 68, 102)
turbo_aspiration <- c(FALSE, TRUE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, TRUE,
FALSE, FALSE, FALSE, TRUE, FALSE,
FALSE, FALSE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, FALSE, FALSE, TRUE, FALSE,
FALSE,
FALSE, TRUE, TRUE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, TRUE, TRUE, FALSE,
TRUE)
date <- c('2019-01-09', '2019-03-01', '2019-01-16', '2018-12-13', '2019-02-16',
'2019-02-15', '2019-02-19',
'2019-02-27', '2019-01-19', '2019-01-22', '2019-03-05', '2018-12-19', '2019-01-
13', '2019-02-04',
'2019-01-30', '2019-01-12', '2019-02-28', '2018-12-15', '2018-12-31', '2018-12-
20', '2019-01-21',
'2019-01-24', '2019-02-05', '2019-02-17', '2018-12-19', '2018-12-26', '2019-02-
26', '2018-12-26',
'2018-12-19', '2018-12-17', '2019-03-04', '2019-03-07', '2019-01-26', '2019-02-
06', '2018-12-20',
'2018-12-13', '2018-12-15', '2018-12-24', '2018-12-30', '2019-03-03')
cars <- data.frame(
date = as.Date(date), # Дати
car_brand, fuel_type, # Стрингове
turbo_aspiration, # Булева променлива
horsepower # Числа
)


rm(list = c("car_brand", "fuel_type", "horsepower", "turbo_aspiration", "date"))
# функцията rm() премахва обектите, които описани в масива, присвоен на
параметъра "list"

str(cars)
# Както виждаме, data frame-а може да съдържа данни от различен тип - дата,
стринг,
# булев или числов.
```

```
#
# В R можем да вземем първите и последните няколко реда с помощта на функциите
head() и tail()
head(x = cars)
head(x = cars, n = 3) # n - колко наблюдения искаме да видим. По подразбиране,
стойността на n е 6
tail(x = cars, n = 3)



# Създаване на извадки
# Искаме да създадем вектор, който съдържа стойностите TRUE и FALSE
N <- 30
set.seed(1234)
sample(x = c(T, F), size = N, replace = TRUE, prob = c(0.9, 0.1))

set.seed(4321)
round(runif(n = N, min = 1, max = 10)) # Генериране на равномерно разпределена
величина с N на брой стойности в интервала 1, 10
# sample(x, size, replace, prob) - на случаен принцип се избират N на брой
елементи от предварително зададен вектор
# x - вектор, от който се избират елементите,
# size - размерът на желаната извадка (в нашия случай 30),
# replace - опцията дали да се преизбират наново елементи. Ако size > x, то
задължително трябва replace = TRUE
# prob - вектор с вероятностите за избиране на случаен елемент. Трябва дължината
на prob да бъде равна на дължината на x и сборът да бъде равен на 1

# round(x, digits = 0) - закръгля числата до digits цифри след десетичната
запетая.




# Обединяване на матрици и data frame-ове по редове и колони

# Ще покажем случая с матрици. С data frame е абсолютно същото.
# Създаваме две матрици
M1 <- matrix(data = 1:20, nrow = 10, byrow = T)
M2 <- matrix(data = 21:30, nrow = 2, byrow = T)
M1; M2


# Функциите за обединяване на две матрици/data frame-ове са:
# - rbind - обединение по редове - долепя втората матрица под първата
# - cbind - обединение по колони - долепя втората матрица след първата
rbind(M1, M2)
# Връща грешка, защото, когато обединяваме две матрици по редове, то размерът на
# колоните трябва да бъде еднакъв. Същото се отнася и при обединяване по колони
-
# искаме редовете да са еднавни на брой.
# Проверка може да се извърши с nrow(), ncol() и dim()

dim(M1)
dim(M2)

# Нека да транспонираме втората матрица (транспонирането е операция, за която
редовете се
# превръщат в колони). Функцията за транспониране в R е t()
M2.t <- t(M2)
dim(M2.t)

rbind(M1, M2.t)
```

```
# Пример за обединение по колони
M3 <- matrix(data = 21:40, nrow = 10, byrow = T)
cbind(M1, M3)


# Функциите apply, lapply, sapply и т.н. могат да се прилагат и върху data
frame-ове



# Взимане на ред, колона и елемент от data frame
# Всеки от начините, които споменахме при взиамне на елементи от матрица, са
приложими и
# за data frame, като отделно имаме и допълнителни.

# dataframe[избор на редове, избор на колони]

cars[1, ] # Първи ред и всички колони (индексът за колоните е празен)

cars[, 2] # Втора колона и всички редове (индексът за редовете е празен)
cars[1:nrow(cars), 2] # еквивалентен на горния ред

cars[5, c(2, 3)] # Пети ред, втора и трета колона
cars[5, 4] # 5 ред и 4-та колона


# С функцията row.names можем както да взимаме имената на редовете, така можем и
да променяме вече съществуващи имена

row.names(cars) <- paste("car", 1:nrow(cars), sep = "_")
# С paste() обединяваме вектори със стрингове (n-тия елемент на единия вектор с
n-тия елемент на друг вектор)
# Имената трябва да са уникални


# При data frame-а можем да вземем елементи и посредством техните имена
# - по колони
cars$car_brand # С "$" взимаме елементи от листа. Data frame-а е ЛИСТ от ВЕКТОРИ
=> взимаме вектора
cars[, "car_brand"]
# Еквивалентни са на cars[, 2]


# - по редове
cars["car_1", ]
# Еквивалентхно е на cars[1, ]

# - по редове и колони
cars["car_1", c("car_brand", "turbo_aspiration")]




# Следващата стъпка е филтрирането на данни и взимането на подгрупи по зададен
признак
# & - логическо "и", | - логическо "или", == - проверка за равенстов
# which() - Връща вектор с индекстите на елементите, за които булевото условие
връща TRUE.
# Тоест взимаме индексите на елементите, които ни вълнуват по някакъв начин.

# Нека да проверим кои от всички 40 модела коли имат turbo aspiration.
which(cars$turbo_aspiration)
cars[which(cars$turbo_aspiration), ] # Извеждаме самите наблюдения
# cars$turbo_aspiration е булева променлива => which(cars$turbo_aspiration) ще
```

```
ни върне индексите
# на наблюденията, които изпълняват това условие => имайки индексите с
cars[индексите, ],
# извеждаме и самите резултати
cars[which(cars$turbo_aspiration), "car_brand"] # Имената на момичетата, които
са приели

cars[which(cars$horsepower >= 102), ] # коли, които имат над 102 конски сили
# Векторът cars$horsepower съдържа стойностите 111, 154, 102, 115, ..., 184,
175, 68, 102.
# С условието cars$horsepower >= 102 ще върне T, T, T, T, ..., F, T, T, T, T, F,
T =>
# which(cars$horsepower >= 102) ще върне вектор с индекси 1, 2, 3, 4, 5, 6, ...,
36, 37 38, 40 =>
# cars[which(cars$horsepower >= 102), ] ще върне информацията за колите, които
имат конски сили поне 102


cars[which(cars$car_brand %in% c("chevrolet", "porsche")), ]
# първи/а масив/вектор/лист/стойност %in% втори/а масив/вектор/лист/стойност -
# проверяваме кои елементи от първата структура присъстват във втората.


# Как да вземем моделите на колите, които имат turbo aspiration и са продадени
# през последната седмица? Дали са има такива?




attach(cars)
car_brand
detach(cars)
car_brand



# С това приключваме с въведението за трите типа структури данни в R.






# О П И С А Т Е Л Н А С Т А Т И С Т И К А

# 1. Променливи
# Най-простата дефиниция за променливи е характеристика, която се варира от един
човек/обект
# до друг. Променливите са категоризирани в два типа - количествени или числови
(представят се
# с числа)и категорийни или качествени (нечислово изобразяване).

# Категорийните променливи се разбиват на два подтипа - номинални (без подредба)
и ординални
# (с подредба). Примери за стойности на НОМИНАЛНИ променливи са: видове
заведения, различни
# държави и организции, видове алкохол, раси и т.н. Примери за стойности на
ОРДИНАРНИ
# променливи са: степени на образование/квалификация, нива на социалните
общества и т.н.


# Количествените се разбиват на два подтипа - непрекъснати (стойностите могат да
варират
```

```
# в даден числов интервал) и дискретни (възможните стойности могат да бъдат
изброени).
# Примери за стойности на НЕПРЕКЪСНАТИ променливи са: възраст, различни
съотношения, тегло
# и т.н. Примери за стойности на ДИСКРЕТНИ променливи са: брой катастрофи на
шофьори, брой на
# деца и т.н.


# 2. Разпределения
# Разпределение - това е таблица, графика или формула, съдържаща стойностите на
наблюденията
# и колко често се случват. Разпределенията се характеризират с форма, модалност
(modality),
# локация, размах, симетричност/асиметричност и ексцес.

# На графиката по-долу са представени няколко от формите на плътността на
разпределенията
N <- 2*10^4
distributions <- c("Камбановидна", "J-", "Правоъгълна", "Дясно асиметрична",
"Двумодална")
par(mfrow = c(2, 3))
for(distr in distributions) {
distribution_values <- switch(distr, "Камбановидна" = rnorm(N), "J-" = rexp(N),
"Правоъгълна" = round(runif(N), 2),
"Дясно асиметрична" = rgamma(N, shape = 5),
"Двумодална" = c(rnorm(N, mean = -2, sd = 2), rnorm(N, mean = 3, sd = 1)))
distr_density <- density(distribution_values)
hist(distribution_values, col = "pink", prob = T, main = paste(distr, "форма на
разпределението"),
ylab = "Плътност", xlab = "Стойности", ylim = range(distr_density$y),
xlim = range(distr_density$x))
lines(distr_density, lwd = 3, col = "blue")
}
par(mfrow = c(1, 1))


# За видовете разпределения, симулациите, квантили и плътност ще говорим по-
нататъка.




country = c('Switzerland', 'Norway', 'Iceland', 'Denmark', 'Finland',
'Netherlands', 'Germany',
'Ireland', 'Sweden', 'United Kingdom', 'Austria', 'Belgium', 'France', 'Italy',
'Spain', 'Malta', 'Slovenia', 'Estonia', 'Czech Republic', 'Portugal',
'Slovakia',
'Poland', 'Croatia', 'Lithuania', 'Greece', 'Latvia', 'Hungary', 'Romania',
'Bulgaria', 'Russia', 'Bosnia And Herzegovina', 'Serbia', 'Belarus', 'Albania',
'Macedonia', 'Ukraine', 'Moldova')

avg_sal_in_USD = c(4959.48, 3416.1, 3136.95, 2975.8, 2583.43, 2575.2, 2538.92,
2521.51, 2494.39,
2394.78, 2219.23, 2162.23, 2140.64, 1675.32, 1480.64, 1278.04, 1180.66,
1158.76, 1069.03, 925.84, 894.2, 884.81, 852.23, 804.68, 795.34, 771.91,
708.91, 649.68, 611.41, 558.01, 481.17, 412.22, 409.62, 348.56, 346.15,
293.26, 284.94)

salaryDF <- data.frame(country, avg_sal_in_USD)


# Проучването е правено за 2018-та година за европейските страни
Country <- c("Austria", "Belgium", "Croatia", "Czechia", "Denmark", "Estonia",
```

```
"Finland",
"France", "Germany", "Greece", "Hungary", "Ireland", "Italy", "Latvia",
"Lithuania", "Luxembourg", "Netherlands", "Norway", "Poland", "Portugal",
"Romania", "Slovakia", "Slovenia", "Spain", "Sweden", "Switzerland", "UK")
Car <- c("Volkswagen", "Volkswagen", "Skoda", "Skoda", "Peugeot", "Toyota",
"Nissan",
"Renault", "Volkswagen", "Toyota", "Suzuki", "Hyundai", "Fiat", "Volkswagen",
"Fiat", "Volkswagen", "Volkswagen", "Nissan", "Skoda", "Renault", "Dacia",
"Skoda", "Renault", "Seat", "Volvo", "Skoda", "Ford")
carsDF <- data.frame(Country, Car)



# А Н А Л И З Н А Е Д Н О М Е Р Н А П Р О М Е Н Л И В А
# (UNIVARIATE ANALYSIS)


# 3. Категорийни променливи
# Категорийните проенливи като цяло носят по-малко информация отколкото
числовите променливи.
# Често обаче, те носят по-голяма стабилност за прогнозните моделите отколкото
числовите.

# Освен графично представяне на данните (ще го разгледаме по-късно), най-добре
честотата се
# вижда посредством таблици. Функцията, която ни трябва е table()


head(carsDF)
tt <- table(carsDF$Car)
tt # Взимаме честотното разпределение

round(prop.table(tt)*100, 2) # Процентното разпределение
# С prop.table() взимаме процентите от вече направена таблица

# Volkswagen държи най-голям пазарен дял по страни, следван от Skoda




# 4. Числови променливи
# 4.1. Оценка на центъра (локацията) на разпределение
# 4.1.1. Средна стойност (Очакване)

meanFunction <- function(x) {
sum(x)/length(x)
}

meanFunction(salaryDF$avg_sal_in_USD)
mean(salaryDF$avg_sal_in_USD)


# 4.1.2. Медиана
# Подреждаме данните във вариационен ред. Взимаме средната стойност (при нечетен
брой)
# или средната стойност на средните два елемента (при четен брой елементи).

medianFunction <- function(x) {
x_sorted <- sort(x)
nn <- length(x_sorted)

if(nn %% 2 == 0) {
return(mean(x_sorted[nn/2 + c(0, 1)]))
} else {
```

```
return(x_sorted[round(nn/2 + 0.25)])
}
}

# %% == mod
# %/% == div


medianFunction(salaryDF$avg_sal_in_USD)
median(salaryDF$avg_sal_in_USD)


medianFunction(1:10)
median(1:10)



# 4.1.3. Мода - най-често срещаната стойност
modeFunction <- function(x) {
tt <- table(x)
return(names(tt)[tt == max(tt)])
}

# Модата може да има повече от една стойност

# table() - честотоното разпределение на променлива (коя стойност колко пъти се
с)
# which.max() - връща индекса на първото число, което приема МАКСИМАЛНА стойност
# which.min() - връща индекса на първото число, което приема МИНИМАЛНА стойност
# names() - връща имената на стойностите


modeFunction(round(salaryDF$avg_sal_in_USD/100))



summary(salaryDF$avg_sal_in_USD) # Описателна статистика за центъра на
разпределението
quantile(salaryDF$avg_sal_in_USD, prob = seq(0.1, 0.9, by = 0.1))
# квантили




# Задача 1 - Ежедневни инциденти с мотоциклети
# Шотландският изпълнителен директор в отдел "Аналитични услуги" на Транспортна
# статистика събира данни за произшествията с мотоциклети. В таблицата по-долу
са
# представени, броят на инцидентите с мотоциклети в Шотландия по пътища с
ограничение
# до 30 и над 30 мили в час, случили се по дни от седмицата.

Day <- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday",
"Sunday")
Built_up <- c(88, 100, 76, 98, 103, 85, 69)
Non_built_up <- c(70, 58, 59, 53, 56, 94, 102)

motorcycleAccidentsDF <- data.frame(Day, Built_up, Non_built_up)


# а. Каква е средната стойност и медианата на броя на произшествията за двата
вида пътища
# б. Каква е формата на разпределението по различните пътища. Интересува ни само
```

модалността

```
# Задача 2 - Инвестиции в акциите "LMT" и "FB"
Date <- c('2015-09', '2015-10', '2015-11', '2015-12', '2016-01', '2016-02',
'2016-03', '2016-04',
'2016-05', '2016-06', '2016-07', '2016-08', '2016-09', '2016-10', '2016-11',
'2016-12',
'2017-01', '2017-02', '2017-03', '2017-04', '2017-05', '2017-06', '2017-07',
'2017-08',
'2017-09', '2017-10', '2017-11', '2017-12', '2018-01', '2018-02', '2018-03',
'2018-04',
'2018-05', '2018-06', '2018-07', '2018-08', '2018-09')
FB <- c(89.9, 101.97, 104.24,
104.66, 112.21, 106.92, 114.1, 117.58, 118.81, 114.28, 123.94, 126.12, 128.27,
130.99,
118.42, 115.05, 130.32, 135.54, 142.05, 150.25, 151.46, 150.98, 169.25, 171.97,
170.87,
180.06, 177.18, 176.46, 186.89, 178.32, 159.79, 172, 191.78, 194.32, 172.58,
175.73,
164.46)
LMT <- c(207.31, 219.83, 219.16, 217.15, 211, 215.79, 221.5, 232.38, 236.23,
248.17, 252.73,
242.97, 239.72, 246.38, 265.25, 249.94, 251.33, 266.58, 267.6, 269.45, 281.13,
277.61,
292.13, 305.39, 310.29, 308.16, 319.12, 321.05, 354.85, 352.44, 337.93, 320.84,
314.54,
295.43, 326.1, 320.41, 345.96)
stocksDF <- data.frame(Date, LMT, FB)

# Анализът трябва да се извърши върху възвръщаемостите на цените, получени по
diff(log(x)),
# където "x" е цената на актива

# а. Какви са средните стойности и медианите. Според вас как се интерпретират
тези числа
# б. Можете ли да кажете в кой актив бихте инвестирали при наличието само на
тази информация
```