



# Программирование в среде R

Шевцов Василий Викторович,  
директор ДИТ РУДН, [shevtsov\\_vv@rudn.university](mailto:shevtsov_vv@rudn.university)

# Ввод данных в R

## scan()

- Данные считываются в вектор или в список с консоли или из файла
- `scan(file = "", what = double(0), nmax = -1, n = -1, sep = "", quote = if(identical(sep, "\n")) "" else "\"", dec = ".", skip = 0, nlines = 0, na.strings = "NA", flush = FALSE, fill = FALSE, strip.white = FALSE, quiet = FALSE, blank.lines.skip = TRUE, multi.line = TRUE, comment.char = "", allowEscapes = FALSE, encoding = "unknown")`

# Параметры

- `file` — имя файла, откуда считываются данные; если `file=`, то производится ввод с клавиатуры; если задано только имя файла, то он (файл) ищется в текущей директории; иначе задаётся полный путь к файлу; может быть и URL.
- `what` — задаётся тип считываемых данных: `logical`, `integer`, `numeric`, `complex`, `character`, `list`; если считывается список, то строки в файле воспринимаются как поля списка указанных выше типов;
- `nmax` — целое положительное число — максимальное число данных для чтения или максимальное число записей в списке; при пропуске этого аргумента или при неправильном его задании файл считывается до конца;
- `n` — целое положительное число — максимальное число данных для чтения; неправильные значения или не типа `integer` игнорируются
- `sep` — разделитель полей; по умолчанию - пробел;
- `quote` — вид кавычек (двойные или одинарные);
- `dec` — десятичный разделитель (точка или запятая);
- `skip` — целое положительное число — число строк файла, которые следует пропустить перед чтением;
- `nlines` — целое положительное число — максимальное число строк для считывания;
- `na.strings` — символьный вектор — его элементы интерпретируются как пропущенные значения NA; пустые поля по умолчанию считываются как NA;

# Параметры

- `flush` — логический аргумент — значение `TRUE` позволяет добавлять комментарии к считываемым данным после последнего считанного поля (но не более одного);
- `fill` — логический аргумент — значение `TRUE` добавляются пустые поля к строкам, в которых количество полей данных меньше определённого параметром `what`;
- `strip.white` — логический вектор; используется только если задан параметр `sep`, удаляет пустое пространство (пробел) перед символьными переменными и после них;
- `quiet` — логический аргумент; при значении `FALSE` функция выведет сообщение о том, сколько элементов было прочитано;
- `blank.lines.skip` — логический аргумент; при значении `TRUE` пустые строки игнорируются (не считываются) (заметим, что параметры `skip` и `nlines` всё равно будут учитывать все пустые строки);
- `multi.line` — логический аргумент; используется если аргумент `what` принимает значение `list`; при значении `FALSE` все записи будут считаны в одну строку; если же и `fill=T`, то чтение при достижении конца строки будет прекращено;
- `comment.char` — символьный аргумент, определяет знак комментария;
- `allowEscapes` — логический аргумент — нужно ли следующие последовательности символов `\n`, `\a`, `\b`, `\f`, `\r`, `\t`, `\v` при чтении рассматривать как команды ( `TRUE`) или просто как символы ( `FALSE`);
- `encoding` — символьный аргумент, задаёт кодировку считываемого файла.

## read.table() и read.csv()

- Если исходные данные представлены в виде таблицы и итоговый результат должен быть таблицей (фреймом данных), то удобнее воспользоваться функцией `read.table( )` либо `read.csv()`. Полная запись функции:
- `read.table(file, header = FALSE, sep = "", quote = "\"", dec = ".", row.names, col.names, as.is = !stringsAsFactors,`
- `na.strings = "NA", colClasses = NA, nrows = -1, skip = 0, check.names = TRUE, fill = !blank.lines.skip, strip.white = FALSE, blank.lines.skip = TRUE, comment.char = "#", allowEscapes = FALSE, flush = FALSE, stringsAsFactors = default.stringsAsFactors(), fileEncoding = "", encoding = "unknown")`
- Упрощённые варианты функции:
- `read.csv(file, header = TRUE, sep = ",", quote="\"", dec=".", fill = TRUE, comment.char="", ...)`
- `read.csv2(file, header = TRUE, sep = ";", quote="\"", dec=",", fill = TRUE, comment.char="", ...)` `read.delim(file, header = TRUE, sep = "\t", quote="\"", dec=".", fill = TRUE, comment.char="", ...)` `read.delim2(file, header = TRUE, sep = "\t", quote="\"", dec=",", fill = TRUE, comment.char="", ...)`

# Аргументы

- `file` — обязательный аргумент, имя файла;
- `header` — логический параметр; при значении `TRUE` считываются имена переменных из файла;
- `sep` — разделитель полей; по умолчанию - пробел;
- `quote` — вид кавычек (двойные или одинарные);
- `dec` — десятичный разделитель в числах (точка или запятая);
- `row.names` — вектор имён строк; представляет собой либо вектор с именами строк итоговой таблицы; либо число — номер столбца исходной таблицы с названиями строк; либо имя столбца считываемой таблицы, где приведены названия строк; если этот параметр не задан, то строки в итоговой таблице будут пронумерованы;
- `col.names` — вектор имён столбцов в итоговой таблице; по умолчанию — «V<номер столбца>»;
- `as.is` — нужно ли символьные переменные, не преобразованные в числовые или логические, переводить в факторы. `as.is` — либо логический, либо числовой вектор, определяющий столбцы, неконвертируемые в факторы.

# Аргументы

- `colClasses` — символьный вектор; определяет классы данных в столбцах (символьные, логические, числовые, даты). Возможные значения: `NA` — автоматическая конвертация типов данных, `NULL` — столбец пропускается (данные не преобразовываются), тип данных в который будут переведены элементы столбца, `factor`;
- `na.strings` — символьный вектор, элементы которого при чтении исходной таблицы в файле будут интерпретироваться как `NA`;
- `nrows` — целочисленный аргумент; определяет максимальное число считываемых строк;
- `skip` — положительный целочисленный аргумент; определяет число строк, пропускаемых перед чтением;
- `check.names` — логический аргумент; при значении `TRUE` имена переменных будут проверены на синтаксическую правильность и отсутствие дублирования;
- `fill` — логический аргумент; при значении `TRUE` строки разной длины будут приведены к единой (максимальной) добавлением пустых полей;
- `strip.white` — логический аргумент; используется только если определён разделитель `sep`, позволяет убирать пробелы перед и после символьных переменных;



## Примечания

- Функция `read.table()` является основной для считывания данных из таблиц.
- Поле таблицы считается пустым, если в нём ничего нет (до знака комментария или до символа окончания строки).
- Если параметр `row.names` не определён (т.е. не заданы имена строк результирующей таблицы), а длина заголовка на единицу меньше числа столбцов, то первый столбец будет рассматриваться как столбец с названиями строк.
- Число столбцов в считываемой таблице определяется автоматически после прочтения первых пяти строк.
- Всё, что находится в исходной таблице после знаков комментария, не считывается.
- Задание параметра `nrows` (пусть даже с очень избыточными значениями) позволяет уменьшить затраты памяти.
- Для чтения больших матриц предпочтительнее использовать функцию `scan()`.


# Вывод данных в R

# write()

## ■ write(x, file, ncolumns, append, sep)

- x — данные, которые нужно записать;
- file — имя файла, куда будет записана информация;
- ncolumns — число столбцов, в которых будет записана информация;
- append — логический аргумент — если значение TRUE, то данные допишутся в исходный файл, если же FALSE, то файл будет переписан заново;
- sep — разделитель столбцов ( \t — табуляция)

```
t1 <- as.matrix(mtcars)  
write(t1,file = "test1.csv")
```

Имя	Дата изменения	Тип	Размер
 test1.csv	23.05.2018 20:02	Файл Microsoft Ex...	2 КБ

## cat()

- `cat(... , file = "", sep = " ", fill = FALSE, labels = NULL, append = FALSE)`
  - `...` — объекты R, которые будут записаны в файл.
  - `file` — имя файла, куда будет записана информация; если этот аргумент отсутствует, то данные будут выведены на экран;
  - `sep` — разделитель элементов записываемого объекта;
  - `fill` — логический или положительный целочисленный аргумент — контролирует создание новых строк в записываемом файле. Если `fill` — числовой, то задаётся длина строки (количество символов в строке). Если `fill` — логический и его значение `FALSE`, то новые строки создаются только при наличии в записываемых данных символа `"\n"`; если значение `TRUE`, то задаётся дополнительный аргумент `width`, определяющий длину создаваемой в файле строки;
  - `labels` — символьный вектор, задающий названия строк. Игнорируется, если аргумент `fill=FALSE`.
  - `append` — логический аргумент — используется, если задано имя файла. Если значение аргумента `TRUE`, то новые данные добавляются к исходному файлу, если `FALSE`, то записываются вместо старых.

## write.table(), write.csv(), write.csv()

- Функции write.table() записывает таблицу данных (или матрицу) в заданный файл. Если записываемый объект не является таблицей (фреймом данных), то он автоматически будет конвертирован.
- write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ", eol = "\n", na = "NA", dec = ".", row.names = TRUE, col.names = TRUE, qmethod = c("escape", "double"))

## write.table(), write.csv(), write.csv()

- `x` — записываемый объект. Предпочтительнее, чтобы это была матрица или таблица данных.
- `file` — имя файла, в который будут записаны данные.
- `append` — логический аргумент — используется, если только задано имя файла. Если значение аргумента `TRUE`, то новые данные добавляются к исходному файлу, если `FALSE` — записываются вместо старых.
- `quote` — логический или числовой аргумент. Если `quote` является логическим аргументом и его значение `TRUE`, то все символьные переменные и факторы будут записаны в файл в двойных кавычках. Если значение аргумента `FALSE` — символьные переменные и факторы записываются без кавычек. Если `quote` — числовой вектор, то его элементы задают номера столбцов, в которых данные должны быть записаны в кавычках. Названия столбцов и строк по умолчанию будут записаны в кавычках.
- `sep` — аргумент, определяющий разделитель полей; значения в каждой строке исходных данных разделяются этим символом.
- `eol` — символ окончания строки; для Windows и Unix - `"\n"`, `"\r"`; для MacOS - `"\r"`.
- `na` — символьный аргумент для обозначения отсутствующих элементов в исходных данных.
- `dec` — десятичный разделитель в числах (точка или запятая).
- `row.names` — аргумент, задающий названия строк в файле. Аргумент либо логический (если значение `TRUE`, то используются имена строк, указанные в исходных данных), либо представляет собой символьный вектор, непосредственно задающий имена строк.
- `col.names` — аналогичный аргумент, задающий имена столбцов.
- `qmethod` — символьный вектор, определяющий как поступать с вложенными друг в друга кавычками — `" "a" "`. Два значения — `escape` (внешние кавычки убираются - по умолчанию) и `double` (все кавычки остаются)

# Примечания

- Для функций `write.csv( )` и `write.csv2( )` аргументы `col.names`, `sep`, `dec` и `qmethod` не могут быть изменены.
- Если в исходной записываемой таблице данных нет столбцов, то при записи имена строк будут присвоены только в том случае, когда `row.names=TRUE`, и наоборот.
- Действительные и комплексные числа записываются с максимально возможной точностью.
- Если в исходной записываемой таблице данных имелись столбцы с матричной структурой, то при записи каждый из столбцов этой матрицы будет представлен в виде отдельного столбца; в связи с этим аргументы `col.names` и `quote` должны соответствовать числу столбцов результирующей таблицы, а не исходной.
- Каждый столбец в таблице данных, являющийся списком или датой, будет переконвертирован в символы.
- Только символьные столбцы (или столбцы, переконвертированные в символы) будут при записи заключены в кавычки (если это указано в аргументе `quote`)
- Аргумент `dec` применяется только к тем столбцам, которые не были переконвертированы в символьные (т.е. только к числовым данным).

## \*.CSV файлы

- Функции `write.csv( )` и `write.csv2( )` используются для записи файлов в формате \*.csv.
- Если параметр `row.names=TRUE`, то значения параметров `qmethod` и `col.names` устанавливается NA; если же `row.names=FALSE`, то `qmethod=col.names=TRUE`.
- Для функции `write.csv( )` параметр `dec` принимает значение ".", а параметр `sep` — ",".
- Для функции `write.csv2( )` десятичный разделитель `dec=","`, разделитель полей `sep=";"`.
- Попытки изменить значения по умолчанию параметров `col.names`, `sep`, `dec` и `qmethod` игнорируются и будет выдано предупреждение.
- Если таблица данных содержит большое число столбцов (более ста), функция `write.table( )` работает медленно, так как каждый столбец может быть переменной своего класса и поэтому обрабатывается отдельно.



# read.table()

Аргумент	Назначение
file	Служит для указания пути к импортируемому файлу. Путь приводят либо в абсолютном виде (например, file = "C:/Temp/MyData.dat"), либо указывают только имя импортируемого файла (например, file = "MyData.txt"), но при условии, что последний хранится в рабочей папке программы (см. выше). В качестве имени можно также указывать полную URL-ссылку на файл, который предполагается загрузить из Internet (например: file = "http://somesite.net/YourData.csv"). Начиная с версии R 2.10, появилась возможность импортировать архивированные файлы в zip-формате.
header	Служит для сообщения программе о наличии в загружаемом файле строки с заголовками столбцов. По умолчанию принимает значение FALSE. Если строка с заголовками столбцов имеется, этому аргументу следует присвоить значение TRUE.
row.names	Служит для указания номера столбца, в котором содержатся имена строк (например, в рассмотренном выше примере это был первый столбец, поэтому row.names = 1). Важно помнить, что все имена строк должны быть уникальными, т.е. одинаковые имена для двух или более строк не допускаются.
sep	Служит для указания используемого в файле разделителя значений переменных ( <i>separator</i> – разделитель). По умолчанию предполагается, что значения переменных разделены "пустым пространством", например, в виде пробела или знака табуляции (sep = ""). В файлах формата csv значения переменных разделены запятыми, и поэтому для них sep = ",".
dec	Служит для указания знака, используемого в файле для отделения целой части числа от дроби. По умолчанию sep = ".". Однако во многих странах в качестве десятичного знака применяют запятую, о чем важно вспомнить перед загрузкой файла и, при необходимости, использовать dec = ",".
nrows	Выражается целым числом, указывающим количество строк, которое должно быть считано из загружаемой таблицы. Отрицательные и иные значения игнорируются. Пример: nrows = 100.
skip	Выражается целым числом, указывающим количество строк в файле, которое должно быть пропущено перед началом импортирования. Пример: skip = 5

# Data.frame

# Таблицы данных

- Таблицы данных `data.frame` – одномерный список из векторов одинаковой длины
- Внутри колонок должны быть данные одного типа
- Колонки могут быть разного типа

```
> x<-0:10  
> y<-20:30  
> z<-letters[1:11]  
> d<-data.frame(col1=x,col2=y,col3=z)  
> d
```

	col1	col2	col3
1	0	20	a
2	1	21	b
3	2	22	c
4	3	23	d
5	4	24	e
6	5	25	f
7	6	26	g
8	7	27	h
9	8	28	i
10	9	29	j
11	10	30	k

# Операции с date.frame

# Вывод данных на экран

```
> head(mtcars)
```

	mpg	cyl	dis	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

```
> head(mtcars,10)
```

	mpg	cyl	dis	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

```
> tail(mtcars)
```

	mpg	cyl	dis	hp	drat	wt	qsec	vs	am	gear	carb
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.7	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.9	1	1	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.5	0	1	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.5	0	1	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.6	0	1	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.6	1	1	4	2

head(data\_frame,rows,addrownums=TRUE)

# Вывод данных на экран

	mpg	cyl	dis	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1

Showing 1 to 18 of 32 entries

View(mtcars)

# Статистики

str(mtcars)  
names(mtcars)  
summary(mtcars)

```
> str(mtcars)
'data.frame':   32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num   6  6  4  6  8  6  8  4  4  6 ...
 $ disp: num  160 160 108 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num   3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num   2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num   0  0  1  1  0  1  0  1  1  1 ...
 $ am  : num   1  1  1  0  0  0  0  0  0  0 ...
 $ gear: num   4  4  4  3  3  3  3  4  4  4 ...
 $ carb: num   4  4  1  1  2  1  4  2  2  4 ...

> names(mtcars)
[1] "mpg" "cyl" "disp" "hp" "drat" "wt" "qsec" "vs" "am" "gear" "carb"

> summary(mtcars)
      mpg          cyl          disp          hp          drat          wt          qsec
Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0   Min.   :2.760   Min.   :1.513   Min.   :14.50
1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5   1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89
Median :19.20   Median :6.000   Median :196.3   Median :123.0   Median :3.695   Median :3.325   Median :17.71
Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7   Mean   :3.597   Mean   :3.217   Mean   :17.85
3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0   3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90
Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0   Max.   :4.930   Max.   :5.424   Max.   :22.90

      vs          am          gear          carb
Min.   :0.0000   Min.   :0.0000   Min.   :3.000   Min.   :1.000
1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000
Median :0.0000   Median :0.0000   Median :4.000   Median :2.000
Mean   :0.4375   Mean   :0.4062   Mean   :3.688   Mean   :2.812
3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000
Max.   :1.0000   Max.   :1.0000   Max.   :5.000   Max.   :8.000
```

## Создание нового столбца

```
> mtcars$mpg2 <- mtcars$mpg*2
> mtcars$mpg2
 [1] 42.0 42.0 45.6 42.8 37.4 36.2 28.6 48.8 45.6 38.4 35.6 32.8 34.6 30.4 20.8 20.8 29.4
[18] 64.8 60.8 67.8 43.0 31.0 30.4 26.6 38.4 54.6 52.0 60.8 31.6 39.4 30.0 42.8
> summary(mtcars$mpg2)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 20.80  30.85   38.40   40.18   45.60   67.80

> mtcars$new_col <- 0
> mtcars$new_col
 [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 0 0 0 0 0 0 0 0

> mtcars$new_col1 <- 1:nrow(mtcars)
> summary(mtcars$new_col1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    8.75   16.50   16.50   24.25   32.00

> ncol(mtcars)
 [1] 14
```



# Адресация

```
> mtcars$mpg
[1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2 10.4 10.4 14.7
[18] 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4 15.8 19.7 15.0 21.4
> mtcars[1]
      mpg
Mazda RX4      21.0
Mazda RX4 Wag  21.0
Datsun 710     22.8
Hornet 4 Drive 21.4
Hornet Sportabout 18.7
Valiant        18.1
Duster 360     14.3
Merc 240D      24.4
Merc 230       22.8
Merc 280       19.2
Merc 280C      17.8
Merc 450SE     16.4
Merc 450SL     17.3
Merc 450SLC    15.2
Cadillac Fleetwood 10.4
Lincoln Continental 10.4
Chrysler Imperial 14.7
Fiat 128       32.4
Honda Civic    30.4

> mtcars[1,]
      mpg cyl disp  hp drat   wt  qsec vs am gear carb mpg2 new_col new_col1
Mazda RX4   21   6  160 110  3.9 2.62 16.46  0  1   4    4   42     0     1
.
```

```
> mtcars$mpg[2]
[1] 21
> mtcars[2,1]
[1] 21
> mtcars[,1]
[1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2 10.4 10.4 14.7
[18] 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4 15.8 19.7 15.0 21.4
.
```

# Адресация

```
> mtcars[c(1,5,9)]
```

	mpg	drat	am
Mazda RX4	21.0	3.90	1
Mazda RX4 Wag	21.0	3.90	1
Datsun 710	22.8	3.85	1
Hornet 4 Drive	21.4	3.08	0
Hornet Sportabout	18.7	3.15	0
Valiant	18.1	2.76	0
Duster 360	14.3	3.21	0
Merc 240D	24.4	3.69	0
Merc 230	22.8	3.92	0
Merc 280	19.2	3.92	0
Merc 280C	17.8	3.92	0
Merc 450SE	16.4	3.07	0
Merc 450SL	17.3	3.07	0

```
> mtcars[c(1,5,9),]
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Mazda RX4	21.0	6	160.0	110	3.90	2.62	16.46	0	1	4	4	42.0	0	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.44	17.02	0	0	3	2	37.4	0	5
Merc 230	22.8	4	140.8	95	3.92	3.15	22.90	1	0	4	2	45.6	0	9

```
> mtcars[1,c(1,5,9)]
```

	mpg	drat	am
Mazda RX4	21	3.9	1

```
> mtcars[1:10,]
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	42.0	0	1
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	42.0	0	2
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	45.6	0	3
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	42.8	0	4
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2	37.4	0	5
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	36.2	0	6
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	28.6	0	7
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	48.8	0	8
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2	45.6	0	9
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	38.4	0	10

# Выбор части данных

```
> mtcars$am == 1
[1] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[18] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
> mtcars$mpg > 18
[1] TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[18] TRUE TRUE TRUE TRUE FALSE FALSE TRUE TRUE TRUE TRUE FALSE TRUE FALSE TRUE
> mtcars$am == 1 & mtcars$mpg > 18
[1] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[18] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE FALSE TRUE FALSE TRUE
```

```
> mtcars[mtcars$cyl==6,]
      mpg cyl  disp  hp drat   wt  qsec vs  am gear carb mpg2 new_col new_col1
Mazda RX4    21.0   6 160.0 110 3.90 2.620 16.46 0   1    4    4 42.0     0       1
Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02 0   1    4    4 42.0     0       2
Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44 1   0    3    1 42.8     0       4
Valiant      18.1   6 225.0 105 2.76 3.460 20.22 1   0    3    1 36.2     0       6
Merc 280      19.2   6 167.6 123 3.92 3.440 18.30 1   0    4    4 38.4     0      10
Merc 280C     17.8   6 167.6 123 3.92 3.440 18.90 1   0    4    4 35.6     0      11
Ferrari Dino  19.7   6 145.0 175 3.62 2.770 15.50 0   1    5    6 39.4     0      30
> mtcars[mtcars$cyl==6,1:5]
      mpg cyl  disp  hp drat
Mazda RX4    21.0   6 160.0 110 3.90
Mazda RX4 Wag 21.0   6 160.0 110 3.90
Hornet 4 Drive 21.4   6 258.0 110 3.08
Valiant      18.1   6 225.0 105 2.76
Merc 280      19.2   6 167.6 123 3.92
Merc 280C     17.8   6 167.6 123 3.92
Ferrari Dino  19.7   6 145.0 175 3.62
```

# subset

`subset(x, ...)`

отфильтровывает и возвращает ту часть объекта `x`, которая соответствует определенному условию; например, команда

`a = subset(x < 5)`

создаст вектор `a`, который будет содержать только те значения `x`, которые не превышают 5; если `x` - таблица данных, то можно использовать аргумент `select` для указания столбцов, которые необходимо извлечь из этой таблицы

```
> subset(mtcars, cyl==6)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	42.0	0	1
Mazda RX4 wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	42.0	0	2
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	42.8	0	4
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	36.2	0	6
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	38.4	0	10
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4	35.6	0	11
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6	39.4	0	30

```
> subset(mtcars, cyl==6 & mpg>20)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4	42.0	0	1
Mazda RX4 wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4	42.0	0	2
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1	42.8	0	4

# rbind

```
> df1 <- subset(mtcars,cyl==6)
> df2 <- subset(mtcars,cyl==4)
> df3 <- rbind(df1,df2)
> df3
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	42.0	0	1
Mazda RX4 wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	42.0	0	2
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	42.8	0	4
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	36.2	0	6
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	38.4	0	10
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4	35.6	0	11
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6	39.4	0	30
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	45.6	0	3
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	48.8	0	8
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2	45.6	0	9
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1	64.8	0	18
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2	60.8	0	19
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1	67.8	0	20
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1	43.0	0	21
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1	54.6	0	26
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2	52.0	0	27
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2	60.8	0	28
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2	42.8	0	32

# cbind

```
> df1 <- mtcars[,1:3]
> df2 <- mtcars[,7:9]
> df3 <- cbind(df1,df2)
> df3
```

	mpg	cyl	disp	qsec	vs	am
Mazda RX4	21.0	6	160.0	16.46	0	1
Mazda RX4 Wag	21.0	6	160.0	17.02	0	1
Datsun 710	22.8	4	108.0	18.61	1	1
Hornet 4 Drive	21.4	6	258.0	19.44	1	0
Hornet Sportabout	18.7	8	360.0	17.02	0	0
Valiant	18.1	6	225.0	20.22	1	0
Duster 360	14.3	8	360.0	15.84	0	0
Merc 240D	24.4	4	146.7	20.00	1	0
Merc 230	22.8	4	140.8	22.90	1	0
Merc 280	19.2	6	167.6	18.30	1	0
Merc 280C	17.8	6	167.6	18.90	1	0
Merc 450SE	16.4	8	275.8	17.40	0	0
Merc 450SL	17.3	8	275.8	17.60	0	0



# Сортировка

```
> mtcars[order(mtcars$mpg),]
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4	20.8	0	15
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4	20.8	0	16
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4	26.6	0	24
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	28.6	0	7
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4	29.4	0	17
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8	30.0	0	31
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3	30.4	0	14
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2	30.4	0	23

```
> mtcars[order(mtcars$mpg,decreasing = TRUE),]
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1	67.8	0	20
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1	64.8	0	18
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2	60.8	0	19
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2	60.8	0	28
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1	54.6	0	26
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2	52.0	0	27
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	48.8	0	8
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	45.6	0	3

```
> mtcars[order(mtcars$carb,mtcars$mpg,decreasing = TRUE),]
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	mpg2	new_col	new_col1
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8	30.0	0	31
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6	39.4	0	30
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	42.0	0	1
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	42.0	0	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	38.4	0	10
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4	35.6	0	11
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4	31.6	0	29
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4	29.4	0	17
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	28.6	0	7
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4	26.6	0	24
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4	20.8	0	15
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4	20.8	0	16

## Пути поиска

- Функция **attach()** добавляет указанную таблицу данных к пути поиска R. Когда указывается имя переменной, программа ищет эту переменную в таблицах данных, включенных в траекторию поиска.
- Функция **detach()** удаляет таблицу данных из пути поиска.
- Функции **attach()** и **detach()** лучше всего использовать, когда вы работаете с одной таблицей данных
- Альтернатива - **with(объект,{команды})**

```
> attach(mtcars)
> mpg
[1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2 10.4 10.4 14.7
[21] 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4 15.8 19.7 15.0 21.4

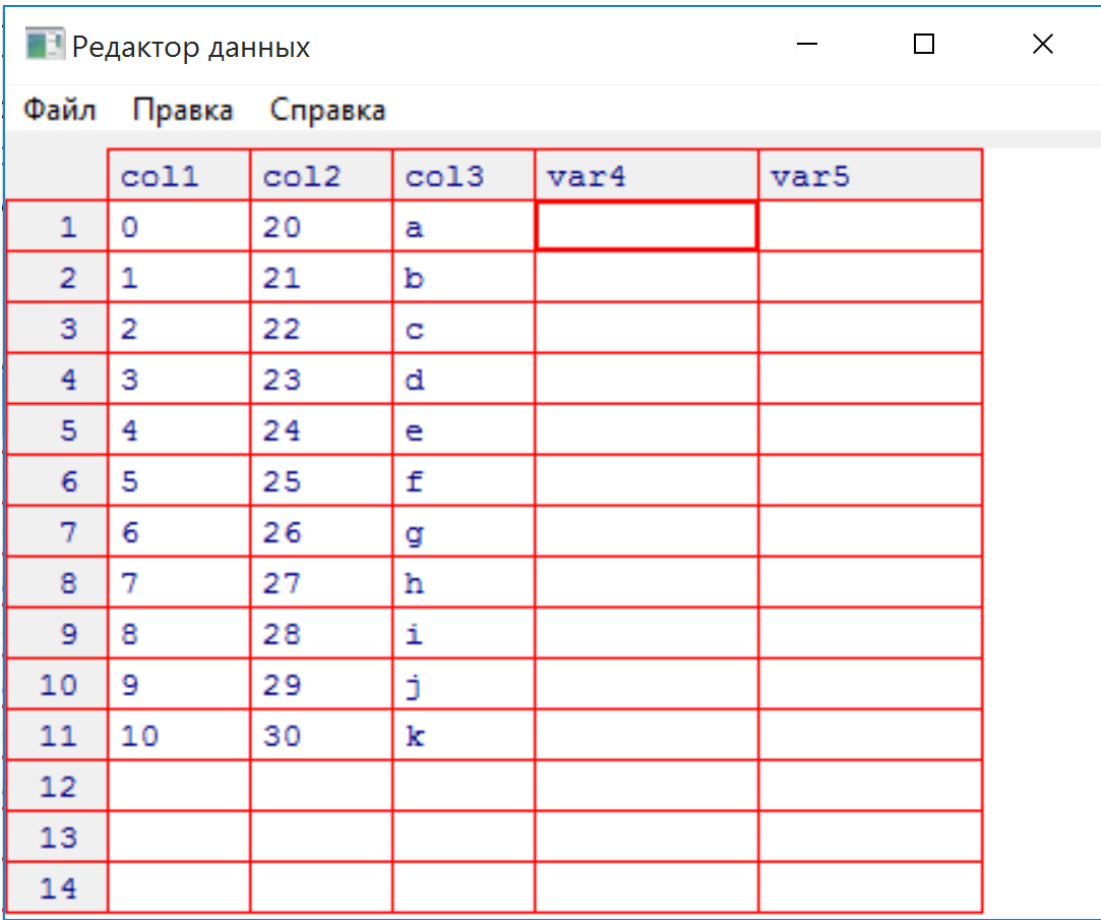
> with(mtcars, {mpg})
[1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.3 15.2 10.4 10.4 14.7
[18] 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.4 15.8 19.7 15.0 21.4
```



# Редактор данных

- `fix(data.frame)`

```
> fix(d)
```



	col1	col2	col3	var4	var5
1	0	20	a		
2	1	21	b		
3	2	22	c		
4	3	23	d		
5	4	24	e		
6	5	25	f		
7	6	26	g		
8	7	27	h		
9	8	28	i		
10	9	29	j		
11	10	30	k		
12					
13					
14					

# Редактор данных

	col1	col2	col3	var4	var5
1	0	20	a	ret	TRUE
2	1	21	b	ert	FALSE
3	2	22	c	ert	TRUE
4	3	23	d	tyju	TRUE
5	4	24	e	fg	TRUE
6	5	25	f	df	FALSE
7	6	26	g	sdf	FALSE
8	7	27	h	fg	TRUE
9	8	28	i	hjk	TRUE
10	9	29	j	fg	TRUE
11	10	30	k	df	TRUE

```
> d
  col1 col2 col3 var4 var5
1    0   20    a  ret TRUE
2    1   21    b ert FALSE
3    2   22    c ert  TRUE
4    3   23    d tyju  TRUE
5    4   24    e  fg  TRUE
6    5   25    f  df FALSE
7    6   26    g  sdf FALSE
8    7   27    h  fg  TRUE
9    8   28    i  hjk  TRUE
10   9   29    j  fg  TRUE
11  10   30    k  df  TRUE

> str(d)
'data.frame':  11 obs. of  5 variables:
 $ col1: num  0 1 2 3 4 5 6 7 8 9 ...
 $ col2: num  20 21 22 23 24 25 26 27 28 29 ...
 $ col3: Factor w/ 11 levels "a","b","c","d",...: 1 2 3 4 5 6 7 8 9 10 ...
 $ var4: chr  "ret" "ert" "ert" "tyju" ...
 $ var5: chr  "TRUE" "FALSE" "TRUE" "TRUE" ...
```

	col1	col2	col3	var4	var5
1	0	20	a	ret	TRUE
2	1				
3	2				
4	3				
5	4				
6	5				
7	6				
8	7	27	h	fg	TRUE
9	8	28	i	hjk	TRUE
10	9	29	j	fg	TRUE
11	10	30	k	df	TRUE
12					
13					

Редактор переменной

имя

тип ☐ numeric ☒ character

# Факторы

# factors

- Факторы представляют собой структуру данных для хранения векторов категориальных данных (классов), т.е. величин, которые могут принимать значения из конечно, в общем случае, неупорядоченного множества

> mtcars

	mpg	cyl	dis	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4
Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	0	3	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	1	4	1
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	1	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	1	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	1	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	1	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	1	4	2

> mtcars\$cyl

[1] 6 6 4 6 8 6 8 4 4 6 6 8 8 8 8 8 8 4 4 4 4 8 8 8 8 4 4 4 8 6 8 4

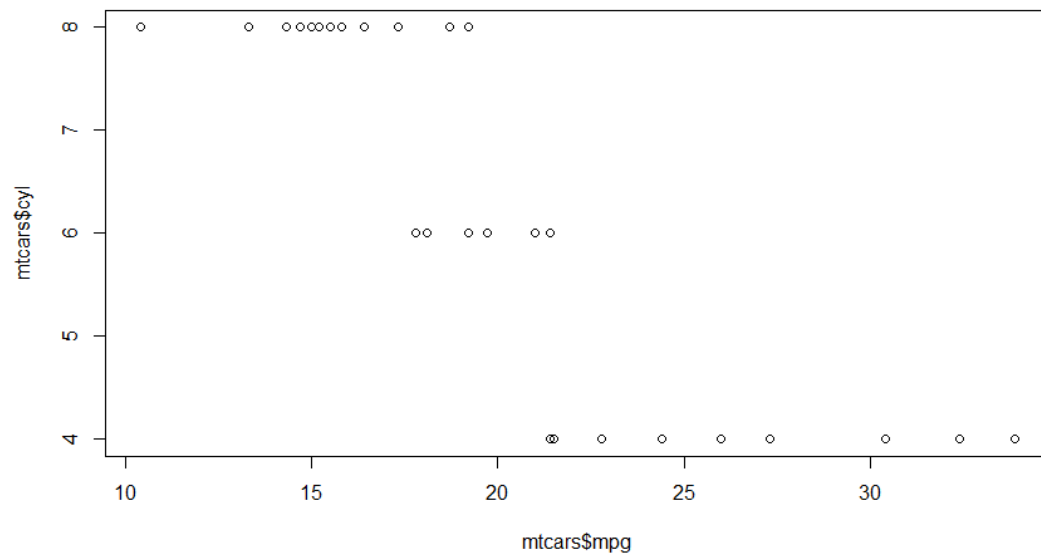
```

> is.factor((mtcars$am))
[1] FALSE
> t1 <- mtcars
> t1$am <- factor(t1$am)
> is.factor(t1$am)
[1] TRUE
> t1$am
[1] 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 1 1 1 1 1
Levels: 0 1
> t1$am <- factor(t1$am, levels=c(0,1), labels <- c("да", "нет"))
> t1

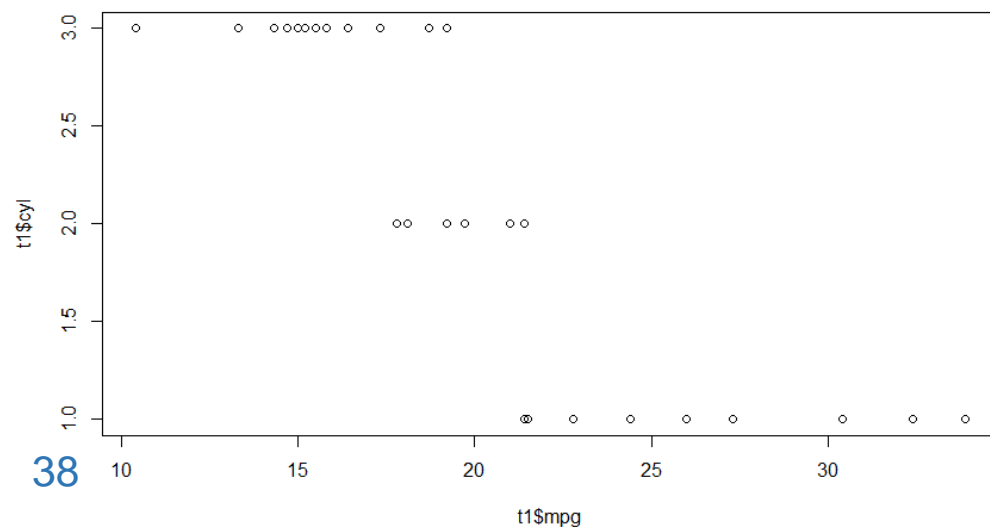
```

	mpg	cyl	dis	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	нет	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	нет	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	нет	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	да	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	да	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	да	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	да	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	да	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	да	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	да	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	да	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	да	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	да	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	да	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	да	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	да	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	да	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	нет	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	нет	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	нет	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	да	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	да	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	да	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	да	3	4
Pontiac Firebird	19.2	8	400.0	175	3.08	3.845	17.05	0	да	3	2
Fiat X1-9	27.3	4	79.0	66	4.08	1.935	18.90	1	нет	4	1
Porsche 914-2	26.0	4	120.3	91	4.43	2.140	16.70	0	нет	5	2
Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	нет	5	2
Ford Pantera L	15.8	8	351.0	264	4.22	3.170	14.50	0	нет	5	4
Ferrari Dino	19.7	6	145.0	175	3.62	2.770	15.50	0	нет	5	6
Maserati Bora	15.0	8	301.0	335	3.54	3.570	14.60	0	нет	5	8
Volvo 142E	21.4	4	121.0	109	4.11	2.780	18.60	1	нет	4	2

```
> t1$cyl <- factor(t1$cyl)
> plot(mtcars$mpg,mtcars$cyl)
```



```
> plot(t1$mpg,t1$cyl)
> t1$cyl
 [1] 6 6 4 6 8 6 8 4 4 6 6 8 8 8 8 8 8 8 4 4 4 4 8 8 8 8 4 4 4 8 6 8 4
Levels: 4 6 8
```



```
> summary(t1$cyl)
 4  6  8
11  7 14
> summary(mtcars$cyl)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 4.000  4.000   6.000   6.188   8.000   8.000
```

# Операции с факторами

```
> t1 <- mtcars
> t1$cyl <- factor(t1$cyl)
> t1$cyl
 [1] 6 6 4 6 8 6 8 4 4 6 6 8 8 8 8 8 8 4 4 4 4 8 8 8 8 4 4 4 8 6 8 4
Levels: 4 6 8
> t1$cyl[3]
[1] 4
Levels: 4 6 8
> t1$cyl[3] <- 1
warning message:
In `[<-.factor`(`*tmp*`, 3, value = c(2L, 2L, NA, 2L, 3L, 2L, 3L,  :
  invalid factor level, NA generated
> t1$cyl[3] <- 8
> t1$cyl[3]
[1] 8
Levels: 4 6 8
```

# Спасибо за внимание!



Шевцов Василий Викторович

shevtsov\_vv@rudn.university  
+7(903)144-53-57