RUDN university

5100

Программирование в среде R
Лекция 1

Шевцов Василий Викторович,
директор ДИТ РУДН, shevtsov_vv@rudn.university

Примерный план лекций

- **Вводная лекция**
  - Что такое данные и зачем их обрабатывать
  - Как обрабатывать данные
  - Общие сведения об R
- **Наборы данных**
  - Типы данных
  - Одномерные данные
  - Двумерные данные, связи
  - Анализ структуры
- **Работа с диаграммами. Базовые диаграммы**
  - Столбчатые диаграммы
  - Круговые диаграммы
  - Гистограммы
  - Диаграммы размахов
  - Точечные диаграммы

Примерный план лекций (продолжение)

- **Управление данными**
  - Создание новых переменных
  - Перекодировка переменных
  - Переименование переменных
  - Пропущенные значения
  - Календарные даты как данные
  - Преобразования данных из одного типа в другой
  - Сортировка данных
  - Объединение наборов данных
  - Разделение наборов данных на составляющие
  - Использование команд SQL для преобразования таблиц данных
- **Анализ временных рядов**
  - Что такое временной ряд
  - Тренд и период колебания
  - Построение временного ряда

Источники данных

- **Наблюдение**
  - Воздействие на объект исследования минимально
- **Эксперимент**
  - На объект исследования оказывается заранее рассчитанное воздействие
- **Генеральная совокупность и выборка**
  - Простой отбор – случайное извлечение объектов из генеральной совокупности с возвратом или без возврата.
  - Типический отбор, когда объекты отбираются не из всей генеральной совокупности, а из ее «типической» части.
  - Серийный отбор – объекты отбираются из генеральной совокупности не по одному, а сериями.
  - Механический отбор - генеральная совокупность «механически» делится на столько частей, сколько объектов должно войти в выборку и из каждой части выбирается один объект.
- **Исследования**
  - Сплошные
  - Выборочные

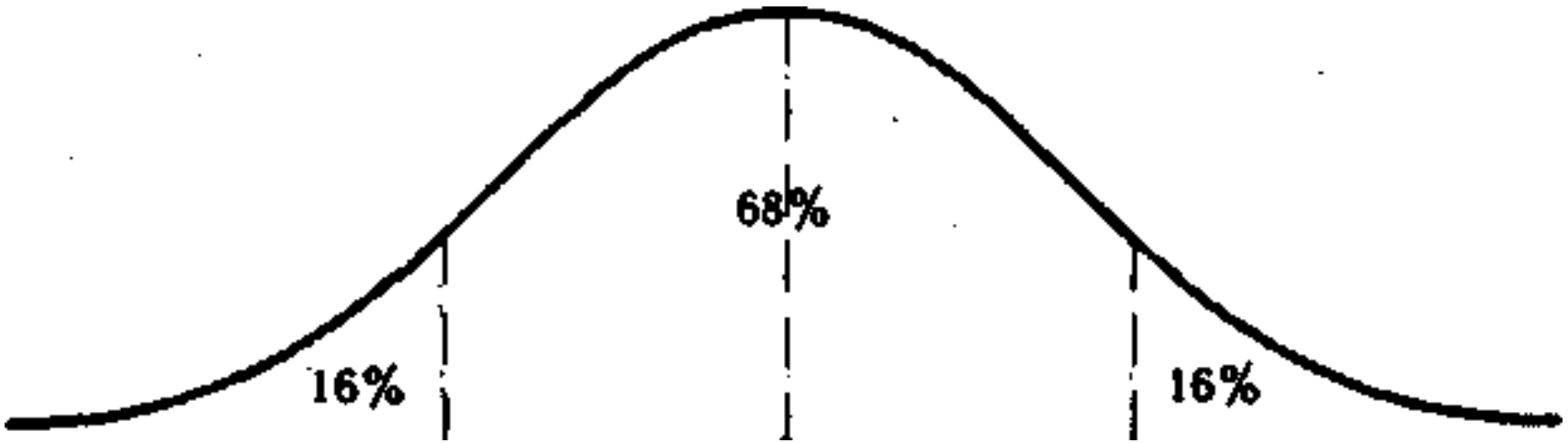
Пример сплошной выборки – перепись 1897г

Возраст	Мужчины	Женщины	Оба пола		Возраст	Мужчины	Женщины	Оба пола
20	1 203 695	1 601 899	2 805 594		40	1 256 887	1 589 111	2 845 998
21	1 027 337	771 514	1 798 851		41	378 000	305 971	683 971
22	1 158 645	1 099 310	2 257 955		42	621 999	542 531	1 164 530
23	1 053 373	998 226	2 051 599		43	512 332	451 597	963 929
24	929 468	885 048	1 814 516		44	424 793	393 336	818 129
25	1 188 690	1 480 517	2 669 207		45	924 863	979 753	1 904 616
26	958 475	907 086	1 865 561		46	474 362	409 166	883 528
27	985 563	922 210	1 907 773		47	438 236	392 561	830 797
28	1 014 745	1 019 581	2 034 326		48	542 175	503 505	1 045 680
29	625 075	530 513	1 155 588		49	299 949	265 337	565 286
30	1 494 510	1 889 008	3 383 518		50	1 077 316	1 386 835	2 464 151
31	508 760	421 847	930 607		51	230 845	205 362	436 207
32	780 758	730 968	1 511 726		52	394 749	357 427	752 176
33	721 160	643 810	1 364 970		53	322 410	301 518	623 928
34	549 678	536 352	1 086 030		54	268 326	269 386	537 712
35	1 109 835	1 223 840	2 333 675		55	680 823	700 127	1 380 950
36	724 769	675 392	1 400 161		56	370 174	318 765	688 939
37	724 478	649 983	1 374 461		57	285 355	245 565	530 920
38	796 176	739 313	1 535 489		58	310 510	278 453	588 963
39	483 817	401 594	885 411		59	170 292	147 219	317 511
					60	899 491	1 142 709	2 042 200

Примеры данных с неопределенной достоверностью

- Оценки в школьном аттестате
- Результаты ЕГЭ
- Результаты работы в группе (коллективе)

Оценка знаний путем тестирования



Хороший нормативно-ориентированный тест обеспечивает нормальное распределение индивидуальных баллов репрезентативной выборки учеников, когда среднее значение баллов находится в центре распределения, а остальные значения концентрируются вокруг среднего по нормальному закону, т.е. примерно 70% значений в центре, а остальные сходят на нет к краям распределения

Как получать данные
Два принципа составления выборки

- **Принцип повторностей**
  - предполагает, что один и тот же эффект будет исследован несколько раз
  - повторности должны быть независимы друг от друга

  Все объекты отличаются.

  Если брать объекты поодиночке, то можно учитывать незначительные характеристики, не влияющие на сущность исследования

  Исследование нескольких объектов позволят выявить характеристики, влияние которых на объект незначительное

- **Рандомизация**
  - каждый объект должен иметь разные шансы попасть в выборку

Что дает анализ данных

- **Общие характеристики для больших выборок**
  - центральная тенденция
  - разброс, насколько сильно разбросаны данные
- **Сравнение между разными выборками**
  - насколько велика вероятность, что различия вызваны случайными причинами
- **Сведения о взаимосвязях**
  - соответствия
  - корреляции
  - зависимости
  - предсказания
- **Структура**
  - классификация объектов

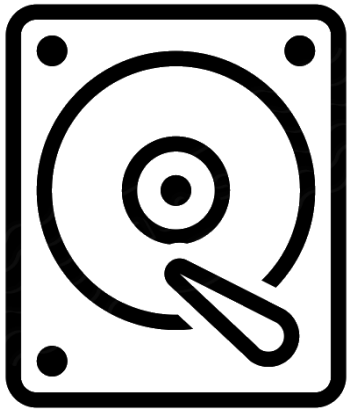
BigData

- Большие данные - обозначение структурированных и неструктурированных данных огромных объёмов и значительного многообразия, эффективно обрабатываемых горизонтально масштабируемыми программными инструментами
- В широком смысле о «больших данных» говорят как о социально-экономическом феномене, связанном с появлением технологических возможностей анализировать огромные массивы данных, в некоторых проблемных областях — весь мировой объём данных, и вытекающих из этого трансформационных последствий.
- С точки зрения информационных технологий в совокупность подходов и инструментов изначально включались средства массово-параллельной обработки неопределённо структурированных данных, прежде всего, системами управления базами данных категории NoSQL
- NoSQL → no relational → No SQL → Not Only SQL

BigData

- **Совокупность данных, методов их обработки, технологий хранения и доступа, применения**
  - большие объемы метаданных
    - количество записей остается, меняется объем самой записи
    - географически распределенные данные
    - системно распределенные данные
  - специализированные аппаратные комплексы
    - дисковые подсистемы выделяются в отдельный класс технологических решений
    - узким местом становятся системы коммутации и передачи данных
    - вычислительные узлы строятся на новых принципах
  - развитие и распространение языков программирования
    - S, R, Python
- **IP4 → IP6**
- **3G → 4G → 5G**

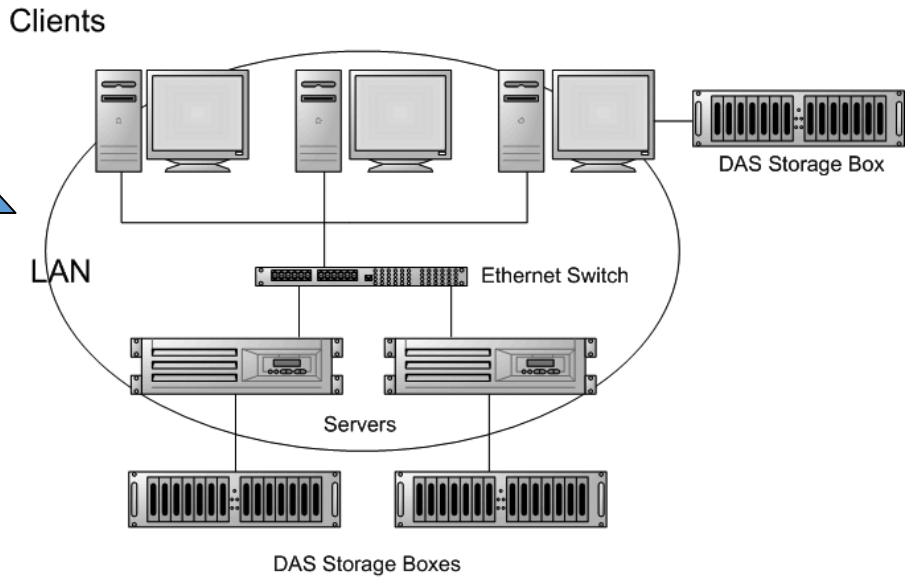
Коллизия развития дисковых подсистем для обработки данных



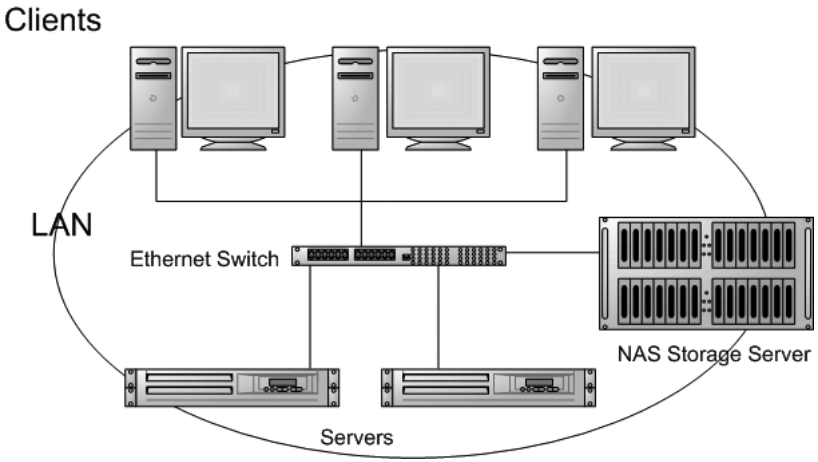
HDD
RAID

NAS, SAN

Direct Attached Storage



Network Attached Storage



DAS
система хранения данных с прямым подключением

RUDN university

5100

Техники и методы анализа, применимые к BigData

- Data Mining;
- Краудсорсинг;
- Смешение и интеграция данных;
- Машинное обучение;
- Искусственные нейронные сети;
- Распознавание образов;
- Прогнозная аналитика;
- Имитационное моделирование;
- Пространственный анализ;
- Статистический анализ;
- Визуализация аналитических данных.

Горизонтальная масштабируемость, которая обеспечивает обработку данных - базовый принцип обработки больших данных.

Данные распределены на вычислительные узлы, а обработка происходит без деградации производительности.

Применимость реляционных систем управления и Business Intelligence.

Программные средства обработки данных

- **Калькулятор**
- **Электронные таблицы**
- **СУБД**
  - MS Access
  - MS SQL Server
- **Специализированные статистические программы**
  - оконно-кнопочные системы
    - STATISTICA
    - STADIA
    - IBM SPSS Statistics Base
  - статистические среды
    - SAS
    - R

Почему R?

- **MATLAB**
  - дороговизна лицензий
  - неполная поддержка статистических функций
  - основная ориентировка на инженерные расчеты
- **MS Excel**
  - ограниченность в функционале
  - ограничение на количество записей
- **SAS**
  - дороговизна лицензий
- **SPSS**
  - дороговизна лицензий

Что такое R?

- R - это среда вычислений, разработанная учеными для обработки данных, математического моделирования и работы с графикой.
  - R можно использовать как простой калькулятор, можно редактировать в нем таблицы с данными, можно проводить простые статистические анализы (например, t-тест, ANOVA или регрессионный анализ) и более сложные длительные вычисления, проверять гипотезы, строить векторные графики и карты. Это далеко не полный перечень того, что можно делать в этой среде. Стоит отметить, что она распространяется бесплатно и может быть установлена как на Windows, так и на операционные системы класса UNIX (Linux и MacOS X). Другими словами, R - это свободный и кроссплатформенный продукт.

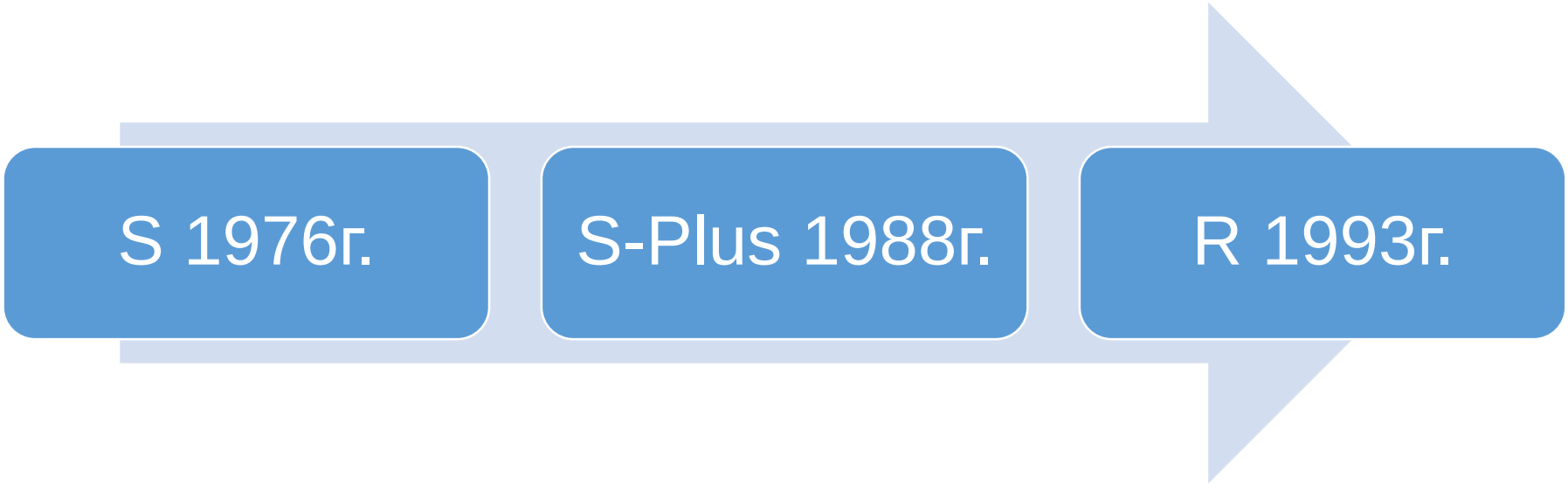
Что такое R?

- R - это язык программирования, благодаря чему можно писать собственные программы (скрипты), а также использовать и создавать специализированные расширения (пакеты).
  - Пакет - это набор функций, файлов со справочной информацией и примерами, собранных вместе в одном архиве. Пакеты играют важную роль, так как они используются как дополнительные расширения на базе R. Каждый пакет, как правило, посвящен конкретной теме, например: пакет 'ggplot2' используется для построения красивых векторных графиков определенного дизайна, а пакет 'qtl' идеально подходит для генетического картирования. Таких пакетов в библиотеке R насчитывается на данный момент более 10000! Все они проверены на предмет ошибок и находятся в открытом доступе.

Что такое R?

- R - это сообщество/движение
  - Так как R - это бесплатный продукт с открытым кодом, то его разработкой, тестированием и отладкой занимается не отдельная компания с нанятым персоналом, а сами пользователи. За два десятилетия из ядра разработчиков и энтузиастов сформировалось огромное сообщество. По последним данным, более 2 млн человек так или иначе помогали развивать и продвигать R на добровольной основе, начиная от переводов документации, создания обучающих курсов и заканчивая разработкой новых приложений для науки и промышленности. В интернете существует огромное количество форумов, на которых можно найти ответы на большинство вопросов, связанных с R.

R – среда для статистических расчетов



разработка фирмы AT&T Bell Labs, предназначен для обработки данных

коммерческая версия

бесплатный аналог S-Plus

Linux, R, 1C, PHP, JavaScript, Intel

R - преимущества и недостатки

- Гибкость
- Свободное распространение
- Свободный код
- Кроссплатформенность
- Богатый арсенал стат. методов
- Качественная векторная графика
- Более 10000 проверенных пакетов
- Взаимодействует с другими языками, такими: C, Java и Python
- Может работать с форматами данных для SAS, SPSS и STATA
- активное сообщество пользователей и разработчиков
- регулярные обновления, хорошая документация и тех. поддержка.

- Сложность обучения. Необходимы навыки программирования
- Относительная медлительность
- Проблемы с источниками на русском языке
- Встречается неподдерживаемость старых кодов

Где находится R?

- R и дополнительные пакеты распространяются через CRAN (Comprehensive R Archive Network).
- В настоящее время в мире доступны более 60 зеркал CRAN.
- Головной узел — (http://cran.r-project.org/) расположен в Вене (Австрия).

Как выглядит среда R?

- Rgui - это стандартный графический интерфейс, встроенный в R по умолчанию. Эта оболочка имеет вид командной строки в окне, называемым консолью. Командная строка работает по принципу "вопрос-ответ".
  - > 2 + 2 * 2 # наш вопрос/запрос
  - [1] 6 # ответ компьютера
- Для записи сложного алгоритма команд в Rgui существует дополнительное скриптовое окно, где пишется программа (скрипт). Команды скрипта выполняются пакетно.
- Третьим элементом оболочки является графический модуль, который появляется при необходимости отображения графиков.

RStudio

- **RStudio - интегрированная среда разработки**
  - У данной оболочки есть заранее разделенные области и дополнительные модули (например, история команд, рабочая область)
- **Open Source Edition**
  - Доступ к RStudio локально
  - Подсветка синтаксиса, завершение кода и интеллектуальный отступ
  - Выполнение кода R непосредственно из исходного редактора
  - Быстрый переход к определениям функций
  - Легко управлять несколькими рабочими каталогами с помощью проектов
  - Интегрированная справка и документация R
  - Интерактивный отладчик для быстрой диагностики и исправления ошибок
  - Обширные инструменты разработки пакетов
- **Commercial License**
  - Коммерческая лицензия для организаций, не способных использовать программное обеспечение AGPL (свободная лицензия, предоставляет разрешения на осуществление прав: создание, использование, воспроизведение, распространение, модификация)
  - Доступ к приоритетной поддержке

Именование

- Символ # означает начало комментария. Всё, что находится после этого знака в рамках одной строки, игнорируется программой.
- В R можно создавать имена для различных объектов (переменных) как на латинице, так и на кириллице
- R чувствителен и к регистру, т.е. строчные и заглавные буквы в нём различаются
- Имена переменных (идентификаторов) в R состоят из букв, цифр и знаков точки (.) и подчёркивания (_).
- Имя объекта не может начинаться с цифры, и если первый символ — это точка, то цифра не может быть вторым символом . При помощи ?имя можно проверить, есть ли такая переменная или функция с тем же именем

Операторы присваивания

```
> aa<-5
> aa
[1] 5
> 6->bb;bb
[1] 6
> c=7;c
[1] 7
> cc<-dd<-8;cc
[1] 8
> dd
[1] 8
>
```

Операторами присваивания в R выступают:

=

<- присваивание справа,

-> присваивание слева,

как поодиночке, так и в последовательности.

выводимые результаты начинаются с [1]. Это объясняется тем, что R рассматривает любые вводимые данные (если не указано иное) как массив. Выводимое число — это вектор длины 1, первый и единственный элемент которого и обозначается [1].

Операторы присваивания

```
> bb->cc<-7
Error in (cc <- bb) <- 7 : could not find function "<-<-"
> cc->7
Error in 7 <- cc : invalid (do_set) left-hand side to assignment
> cc<-7
> cc<-5;cc
[1] 5
>
```

```
> z<-6
> z
[1] 6
> (z<-7)
[1] 7
>
```

Чтобы для вывода результатов повторно не обращаться к переменной, достаточно взять операцию в круглые скобки

Операторы присваивания

```
> {z<-5;
+ c=4;z
+ c}
[1] 4
>
```

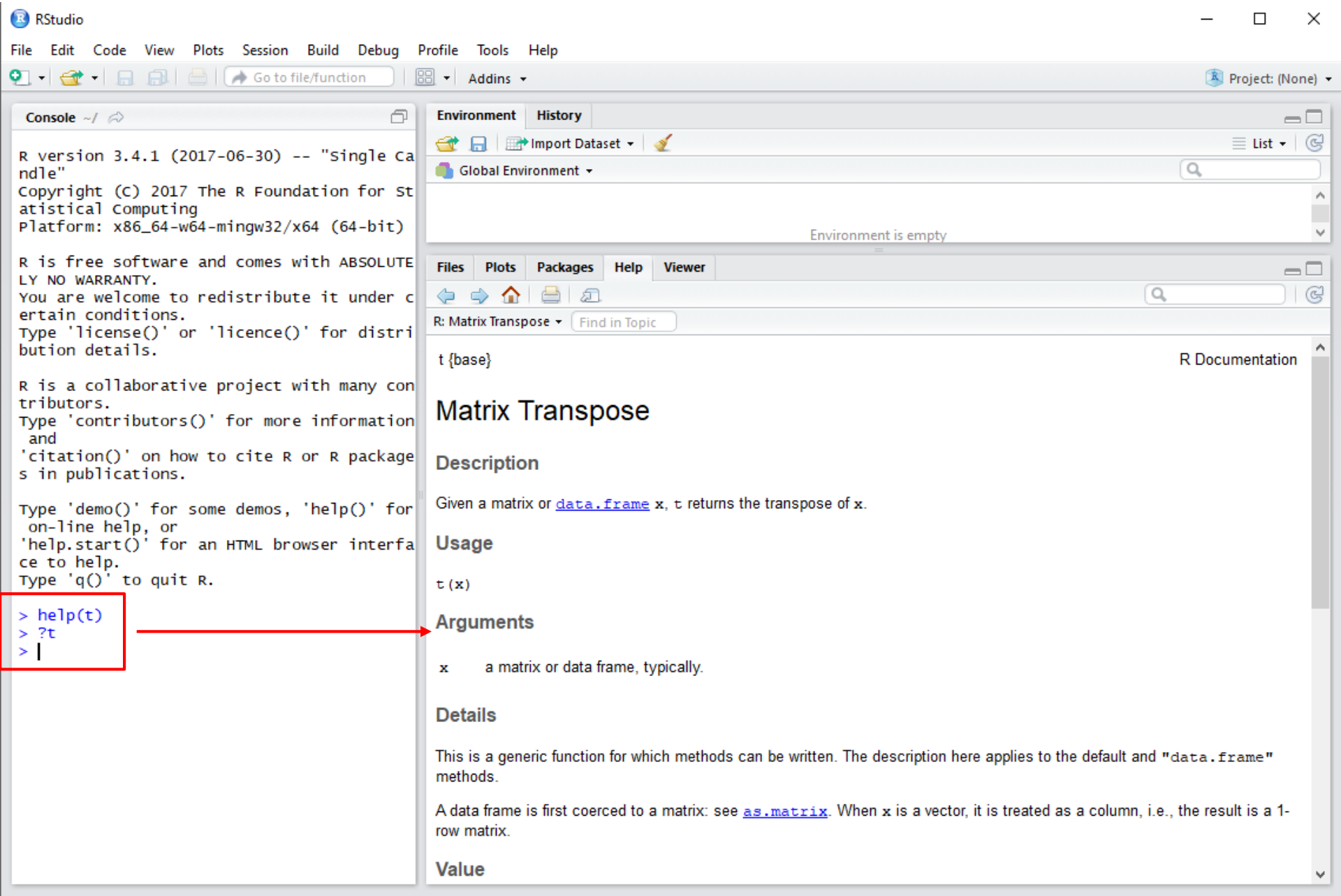
Если нужно, чтобы выводились результаты только последнего выражения, нужно весь блок команд взять в фигурные скобки. Выражения в {} воспринимаются как единое целое.

Проверка переменных

```
> t
function (x)
UseMethod("t")
<bytecode: 0x00000000167bdbe0>
<environment: namespace:base>
> c
function (...)  .Primitive("c")
> q
function (save = "default", status = 0, runLast = TRUE)
.Internal(quit(save, status, runLast))
<bytecode: 0x00000000167a7e10>
<environment: namespace:base>
> w
Error: object 'w' not found
>
```

В R много функций, чьё имя состоит только из одной буквы (например, c() или t()), поэтому создание переменной со схожим названием может привести к ряду проблем

Справка по функции



Типы данных

В отличие от других языков программирования, таких как C и Java в R, переменные не объявляются, как какой-то тип данных. Переменные присваиваются с R-объектами, и тип данных R-объекта становится тип данных переменной.

Есть много типов R-объектов. Часто используемые:

- Векторы
- Списки
- Матрицы
- Массивы
- факторы
- Кадры данных

Классы данные (переменных)

- **numeric** — название класса, а также типа объектов. К нему относятся действительные числа. Объекты данного класса делятся на
  - целочисленные (integer)
  - действительные (double или real).
- **complex** — объекты комплексного типа.
- **logical** — логические объекты, принимают только два значения:
  - FALSE (F)
  - TRUE (T)
- **character** — символьные объекты. символьные переменные задаются либо в двойных кавычках, либо в одинарных.
- **raw** — объекты потокового типа

Иерархия типов: raw < logical < integer < real < complex < character.

Numeric

Объект класса **numeric** создаётся при помощи команды **numeric(n)**, где n — количество элементов данного типа.
Создаётся нулевой вектор длины n.
>x=numeric(5)
> x
> [1] 0 0 0 0 0
В результате создан нулевой вектор типа numeric длины 5.

Тип	Создается	Проверяется
numeric	numeric(n)	is.numeric(имя_объекта) → TRUE/FALSE
integer	integer(n)	is.integer(имя_объекта) → TRUE/FALSE
double	double(n)	is.double(имя_объекта) → TRUE/FALSE

Десятичным разделителем для чисел является точка

Numeric

```
> x<-double(1)
> x<-5
> y<-integer(1)
> y<-7
> is.integer(x)
[1] FALSE
> is.double(x)
[1] TRUE
> is.integer(y)
[1] FALSE
> is.double(y)
[1] TRUE
> is.numeric(x)
[1] TRUE
> is.double(y)
[1] TRUE
> y<-integer(1)
> is.integer(y)
[1] TRUE
>
```

1. Создали переменную
2. Присвоили значение
3. Проверили принадлежность

По умолчанию, все числа в R являются вещественными. Чтобы сделать их целочисленными, надо воспользоваться командой **as.integer(имя_объекта)**

Numeric

```
>
> x<-double(1)
> x<-5
> is.double(x)
[1] TRUE
> is.integer(x)
[1] FALSE
> x<-as.integer(x)
> is.double(x)
[1] FALSE
> is.integer(x)
[1] TRUE
>
```

1. Создали переменную
2. Присвоили значение
3. Проверили принадлежность
4. Назначили тип данных
5. Проверили принадлежность

Logical

- Объекты этого класса принимают два возможных значения: TRUE (истина) и FALSE (ложь). Сокращенные наименования: T, F
- создаются при помощи команды **logical(n)**, где n — это длина создаваемого вектора.

```
> x<-logical(4)
> x
[1] FALSE FALSE FALSE FALSE
> x<-1;y<-F
> is.logical(x)
[1] FALSE
> is.logical(y)
[1] TRUE
>
```

Проверка типа данных – **is.logical(x)**

Logical

- преобразование данных

```
> x<-1
> is.double(x)
[1] TRUE
> z<-as.logical(x)
> is.logical(z)
[1] TRUE
> z
[1] TRUE
> x<-0
> z<-as.logical(x)
> z
[1] FALSE
>
```

- присвоение числового значения (double)
- преобразование в logical и присвоение значения
- проверка типа данных

Character

- **character** - символьные объекты
- Создаются при помощи команды **character(n)**, результат — пустой символьный вектор размерности n
- Символьные объекты обязательно задаются в кавычках (одинарных или двойных)
- Символьным объектом может быть как просто символ, так и строка.

```
> x<-'q'
> x
[1] "q"
> x<-"w"
> x
[1] "w"
> x<-"'r'"
> x
[1] "'r'"
> x<-"язык программирования R"
> x
[1] "язык программирования R"
>
```

```
> x<-"qqq";x
[1] "qqq"
> x<-"'qqq'";x
[1] "'qqq'"
> x<-""qqq"";x
Error: unexpected symbol in "x<-""qqq"
> x<-'"qqq"';x
[1] "\"qqq\""
> x<-"\"qqq\"";x
[1] "\"qqq\""
>
```

Преобразование в character

- Объекты любого типа можно перевести в символьные. Для этого нужно воспользоваться командой **as.character(имя_объекта)**

```
> x<-F;y<-1.23
> x;y
[1] FALSE
[1] 1.23
> z<-as.character(x);x
[1] FALSE
> z
[1] "FALSE"
> z<-as.character(y);y
[1] 1.23
> z
[1] "1.23"
>
```

```
> x<-0123;x
[1] 123
> y<-as.character(x);y
[1] "123"
> y<-as.character("0123");y
[1] "0123"
>
```

Преобразование character

- Символьный объект можно перевести в числовой, если он представляет из себя число, окружённое кавычками.
- Если же в кавычках стоял непосредственно символ (или набор символов), то такой перевод приведёт к появлению NA (Not Available)

```
> x<-"123"
> y<-as.numeric(x)
> y
[1] 123
> x<-"0123"
> y<-as.numeric(x)
> y
[1] 123
> x<-"q123"
> y<-as.numeric(x)
Warning message:
NAs introduced by coercion
> 
```

Преобразования

- Тип любого объекта можно проверить (и изменить) при помощи функции **mode(имя_объекта)**

```
> x<-as.character("T")
> y<-as.logical(x)
> y
[1] TRUE
> z<-as.integer(y)
> z
[1] 1
> mode(x)
[1] "character"
> mode(y)
[1] "logical"
> mode(z)
[1] "numeric"
>
```

```
> x<-as.character("T")
> x
[1] "T"
> mode(x)<-'logical'
> x
[1] TRUE
> x<-as.character("Q")
> x
[1] "Q"
> mode(x)<-'logical'
> x
[1] NA
>
```

Специальные переменные в R

- **Inf**
  - бесконечность: положительная (+∞ — Inf) и отрицательная (−∞ -Inf);
- **NA**
  - отсутствующее значение (Not Available);
- **NaN**
  - не число (Not a Number);
- **NULL**
  - ничто

Специальные переменные в R

- **Inf** появляется при переполнении и в результате операций вида a/0, где a <>0
- Проверить объект на конечность / бесконечность можно при помощи команд **is.finite()** / **is.infinite()**

```
> x<-0;y<-1
> z<-y/x
> z
[1] Inf
> z<-log(x)
> z
[1] -Inf
>
```

```
> x<-0;y<-1
> z<-y/x
> is.finite(z)
[1] FALSE
> is.infinite(z)
[1] TRUE
>
```

Специальные переменные в R

- Объект **NaN** — «не число», появляется при операциях над числами, результат которых не определён (не является числом)
- При помощи **is.nan(имя_объекта)** можно проверить, является ли объект NaN:

```
> x<-0;y<-0
> z<-x/y
> z
[1] NaN
> z<-Inf-Inf
> z
[1] NaN
> x<-Inf;y<-Inf
> z<-x-y
> z
[1] NaN
> x<--2
> z<-log(x);z
Warning message:
In log(x) : NaNs produced
[1] NaN
>
```

```
> x<-Inf;y<-Inf
> z<-x-y
> is.nan(z)
[1] TRUE
>
```

Специальные переменные в R

- Отсутствующее значение **NA** возникает, если значение некоторого объекта не доступно (не задано). Включает в себя и **NaN**.
- Проверка,относится ли объект к **NA**, делается при помощи **is.na(имя_объекта)**

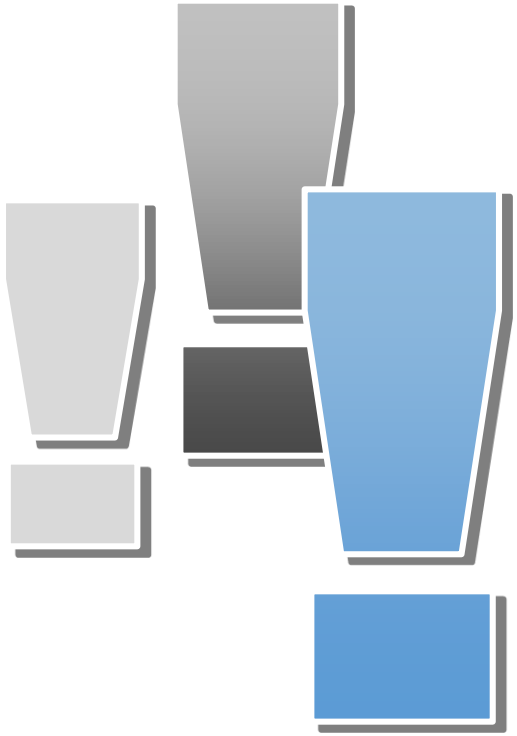
```
> x<-NaN
> y<-NA
> is.na(x)
[1] TRUE
> is.na(y)
[1] TRUE
> 
```

Специальные переменные в R

- Ничто **NULL** нулевой (пустой) объект. Возникает как результат выражений (функций), чьи значения не определены. Обнулить объект можно при помощи команды **as.null(имя_объекта)**
- Проверить объект на принадлежность к **NULL** можно при помощи функции **is.null(имя_объекта)**

```
> x<-1
> y<-as.null(x)
> is.null(x)
[1] FALSE
> is.null(y)
[1] TRUE
>
```

Спасибо за внимание!



Шевцов Василий Викторович

shevtsov_vv@rudn.university
+7(903)144-53-57