|  |  |
| --- | --- |
| **Alt+Shift+K** | горячие клавиши  Ctrl+Enter - запустить строку кода  Ctrl+Alt+R - запустить весь код |
| **help(sqrt)**  **?sqrt** | справка |
| **search()** | # список установленных пакетов |
| **install.packages("cluster")**  **library(cluster)**  **animals** | инсталлирование пакета  загрузка пакета в программу  обращение к содержимому пакета |
| **getwd()** | вывод рабочей директории  измерение рабочей директории (session/set working directory/choose directory) |
| **a <- 10**  **50 -> b**  **c = 20** | присвоить значение переменной |
| **|** | или  1<2 | 2<1 # опертор или |
| **&** | и  1<2 & 2<1 # опертор и |
| **class(b)** | тип переменной |
| **rm(b)** | удалить переменную |
| **4^3** | возведение в степень |
| **sqrt(25)** | корень квадратный |
| **factorial()** | факториал |
| **log2(8)**  **log(27, base = 3)** | логарифм по основанию 2  log(27, base = 3) |
| **exp(1)** | экспонента |
| **10:1** | создание множества  x <- 10:1 # создание множества от 10 до 1 |
| **seq(2,12,by=2)**  **seq(5,2,length.out = 9)** | создание последовательности  seq(2,12,by=2) # последоват от 2 до 12 с шагом 2  seq(5,2,length.out = 9) # от 5 до 2 длиной 9 элементов |
| **c()** | создание вектора  a <- c(0,1,2,3) # перечисл значений вектора  b <- c(rep(7, time=5)) # повторить 7, кол-во 5 раз  b <- c(rep(7, 5)) # повторить 7, кол-во 5 раз  a[1] # значение вектора с индексом 1  a[1:3] # значение вектора с 1 по 3 индексы  a[c(1,3)] # значение вектора 1 и 3 индексы  a[-c(1,3)] # значение вектора кроме 1 и 3 индексаa+b # сложение векторов  d <- c(seq(1,3), rep(c(0,2),3)) # 1 2 3 0 2 0 2 0 2  f <- rep(c(0,2),each=3) # 0 0 0 2 2 2 |
| **sample(d)** | перемешивание значений вектора  sample(letters, 100, replace=TRUE) # перемешивание значений вектора (вектор, кол-во, дополнить из вектора до кол-во) |
| **length(a)** | длина вектора, переменной |
| **dim(animals)** | размер датафрейма, матрицы |
| **str(animals)** | структура датафрейма (строки, столбцы, типы данных) |
| **min(x)** | минимум |
| **max(x)** | максимум |
| **sum(x)** | сумма |
| **median(d)** | медиана |
| **mean(x)** | среднее арифметическое |
| **table(d)** | сколько раз встречаются элементы (мода - самое частое значение)  d <- c(seq(1,3), rep(0,3)) # 1 2 3 0 0 0  table(d)  0 1 2 3  3 1 1 1 |
| **round(y, 1)** | банковское округление до такого-то знака |
| **sort(y)** | сортировка  sort(y) # сортировка по возрастанию  sort(y, decreasing = TRUE) # сортировка по убыванию |
| **set.seed(1)** | от какого числа высчитывается random |
| **rnorm(n=50, mean=0, sd=1)** | создание нормального распределения (кол-во, ср арифмет, стандартное отклонение) |
| **rpois(100,10)** | создание распределения Пуассона (кол-во, неблагоприятн события) |
| **rbinom(100, 10, 0.5)** | создание биномиального распределения (100 экспериментов, по 10 испытаний, вероятность благоприят события) |
| **letters** | текстовой вектор из 26 строчных англ букв |
| **LETTERS** | текстовой вектор из 26 заглавных англ букв |
| **paste()** | paste(letters, seq(1,26)) # поэлементное объединение нескольких последовательностей через пробел [a 1]  paste(letters, set="-", seq(1,26)) # с разделителем через пробел [a - 1] |
| **paste0()** | paste() но без пробела |
| **ISOdate(2020, 8, 1:30)** | создание последовательности из даты |
| **format(ISOdate(2020, 8, 1:30), "%d")** | создание последовательности из даты в определенном формате (b - назв месяцев, a - дни нед) |
| **df\_1 <- data.frame(s\_1, s\_2)** | создание датафрейма  s\_1 <- sample(letters, 100, replace=TRUE) # столб 1 df  s\_2 <- round((sample(seq(5,2,length.out = 100), 100)),2)  df\_1 <- data.frame(s\_1, s\_2) # создание df из векторов  s\_3 <- c(rep('a', 50), rep('b', 50))  df\_2 <- data.frame(s\_1, s\_2, s\_3, stringsAsFactors = TRUE) # создание df символьный вектор сделать Фактором  animals[animals$fly == 2,] # только те строки в которых fly имеет значения 2 |
| **head(df\_1, 5)** | вывести на экран первые 5 строк (по умолч 6) |
| **tail(df\_1, 5)** | вывести на экран последние 5 строк (по умолч 6) |
| **order(df\_1$s\_2)** | отсортировать индексы по возр значений столбца s\_2  (sort выведет сами значения, а не индексы) |
| **df\_1$s\_2** | вывести из df\_1 столбец s\_2 |
| **df\_1[3:7, 1:2]** | вывести из df\_1 с 3 по 7 строку 1 и 2 столбцов (включ)  df\_1[,2] # вывести из df\_1 столбец s\_2 |
| **df\_1[ind,]** | вывести df - вывести df - строки отсортировать по переменной ind |
| **df\_1[,st\_1]** | вывести df - столбцы отсортировать по переменной st\_1 |
| **data()** | список датасетов  data(package = .packages(all.available = TRUE)) # список датасетов по пакетам |
| **is.na(animals)** | пропущенные значения (TRUE)  sum(is.na(animals)) # кол-во пропущенных значений  animals$fly[is.na(animals$end)] # значения столбца fly в строках, где столбец end не имеет значения  animals$end[!is.na(animals$end)] # значения столбца end, где есть значения |
| **unique(iris$Species)** | уникальные значения (тип Фактор + levels) |
| **levels(iris$Species)** | уровни (только для тип Фактор) |
| **matrix(n,10,3)** | создание матрицы  m <- 1:30 # создание вектора  dim(m)<-c(10,3) # 10:3 (присвоить вектору объем)  n <- 1:30 # создание вектора  n <- matrix(n,10,3) # матрица 10:3 |
| **rownames(m)** | имена строк df и матриц  name\_row <- LETTERS[1:10]  rownames(m, do.NULL = TRUE, prefix = "row")  row.names(m) <- name\_row |
| **colnames(df\_1)** | имена столбцов df и матриц  name\_col <- LETTERS[1:3]  colnames(m, do.NULL = TRUE, prefix = "col")  colnames(m) <- paste('day',1:3) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |