W4C1\_object\_variable

2025-03-19

# 1 数据类型(variable)设置，检查，修改与判断

## 数据类型：用于存储数值或信息，基本的数据类型包括：

## numeric(数值), integer(整数), double(小数),, complex(复数), logical(逻辑), character(字符),factor(因子)

## 检查数据类型

x <- 5.23  
class(x)

## [1] "numeric"

is.numeric(x)

## [1] TRUE

is.integer(x)

## [1] FALSE

is.double(x)

## [1] TRUE

class(x)

## [1] "numeric"

typeof(x)

## [1] "double"

mode(x)

## [1] "numeric"

## 改变数据类型

x1 <- as.integer(x)  
x1

## [1] 5

#字符型  
x <- 'elevated'  
typeof(x)

## [1] "character"

is.character(x)

## [1] TRUE

# 2 R对象(object)：数据对象和函数对象

## object-oriented program R语言将变量分配到对象，内存中保存的是数据结构而不是数据类型(如C#)

## R数据对象：向量 vector，矩阵 matrix，数组 array，数据框 dataframe，列表 list

### 向量：c(), seq(), rep()

num\_variable <- c(1,4,7)  
(num\_variable <- c(a=1, b=4, c=7)) # 加括号，赋值并打印

## a b c   
## 1 4 7

# 查看对象的基本信息  
  
typeof(num\_variable) # 对象其中的数据类型

## [1] "double"

str(num\_variable) # object的结构

## Named num [1:3] 1 4 7  
## - attr(\*, "names")= chr [1:3] "a" "b" "c"

is.vector(num\_variable)

## [1] TRUE

length(num\_variable) # 长度

## [1] 3

names(num\_variable) # 对象中变量的名称

## [1] "a" "b" "c"

attributes(num\_variable) # 查看变量的属性

## $names  
## [1] "a" "b" "c"

### Rectangular: matrices, dataframe等的统称（类似于方框，都拥有由向量组成的列）

* matrix：矩阵,通过matrix()创建，或通过rbind/cbind合并向量来创建

(m0 <- matrix(data = 1:9, nrow=3, byrow=TRUE) )

## [,1] [,2] [,3]  
## [1,] 1 2 3  
## [2,] 4 5 6  
## [3,] 7 8 9

# 创建一个3行的矩阵，数据按照行排列  
  
(m1 <- matrix(data = 1:9, nrow=3, byrow=FALSE) )

## [,1] [,2] [,3]  
## [1,] 1 4 7  
## [2,] 2 5 8  
## [3,] 3 6 9

x <- 1:3  
y <- 4:6  
z <- 7:9  
(m2 <- rbind(x,y,z))

## [,1] [,2] [,3]  
## x 1 2 3  
## y 4 5 6  
## z 7 8 9

(m3 <- cbind(x,y,z))

## x y z  
## [1,] 1 4 7  
## [2,] 2 5 8  
## [3,] 3 6 9

str(m3)

## int [1:3, 1:3] 1 2 3 4 5 6 7 8 9  
## - attr(\*, "dimnames")=List of 2  
## ..$ : NULL  
## ..$ : chr [1:3] "x" "y" "z"

dim(m3) # 查看维度

## [1] 3 3

length(m3) # 查看长度

## [1] 9

typeof(m3)

## [1] "integer"

attributes(m3)

## $dim  
## [1] 3 3  
##   
## $dimnames  
## $dimnames[[1]]  
## NULL  
##   
## $dimnames[[2]]  
## [1] "x" "y" "z"

# matrix中的取值  
m3[1,3] # 通过行、列号位置索引

## z   
## 7

# 运算  
mean(m3) # 全部取均值

## [1] 5

colMeans(m3) # 按列取均值

## x y z   
## 2 5 8

rowMeans(m3) # 按行取均值

## [1] 4 5 6

sum(m3) # 全部求和

## [1] 45

colSums(m3) # 按列求和

## x y z   
## 6 15 24

rowSums(m3) # 按行求和

## [1] 12 15 18

max(m3)

## [1] 9

max.col(m3) #找到每列中最大值所在的位置索引

## [1] 3 3 3

# max.row(m3)# 没有这个函数！

* dataframe：有行rows和列columns

(df1 <- as.data.frame(m3))

## x y z  
## 1 1 4 7  
## 2 2 5 8  
## 3 3 6 9

(df <- data.frame(a=c(1,2,3),b=c('1','2','3'),  
 stringsAsFactors = FALSE))

## a b  
## 1 1 1  
## 2 2 2  
## 3 3 3

# stringAsFactor：将字符作为一种因子，将string类型转换为factor  
# 作为factor时，按字符的类型进行分类，如male female当做两个值处理；  
# 作为character时，每个字符串都当做独立值处理，不做分类  
  
str(df)

## 'data.frame': 3 obs. of 2 variables:  
## $ a: num 1 2 3  
## $ b: chr "1" "2" "3"

dim(df)

## [1] 3 2

length(df[,2])

## [1] 3

typeof(df)

## [1] "list"

attributes(df)

## $names  
## [1] "a" "b"  
##   
## $class  
## [1] "data.frame"  
##   
## $row.names  
## [1] 1 2 3

* tibble:最新的许多函数均以tibble为操作对象，因此有时要把dataframe转化为tibble对象 []或subset索引取的是子集，仍然是dataframe或tibble 而df$a[条件]这样取的是对应的元素，不再是子集，数据类型会改变

(tb <- tibble::as\_tibble(df))

## # A tibble: 3 × 2  
## a b   
## <dbl> <chr>  
## 1 1 1   
## 2 2 2   
## 3 3 3

# tibble轻量级，占用资源少，直接给出行、列和每列的数据类型  
dim(tb)

## [1] 3 2

tb[2] # 直接取列

## # A tibble: 3 × 1  
## b   
## <chr>  
## 1 1   
## 2 2   
## 3 3

tb[1,] # 直接取行

## # A tibble: 1 × 2  
## a b   
## <dbl> <chr>  
## 1 1 1

tb$a[1] # 取指定列中的元素

## [1] 1

names(tb)[2] # 获取数据框列名

## [1] "b"

age <- rep(c(21:29),2)  
gender <- rep(c('male','female'),9)  
name <- rep(c('a','b','c'),6)  
  
(df\_a <- data.frame(name, age, gender, stringsAsFactors = TRUE))

## name age gender  
## 1 a 21 male  
## 2 b 22 female  
## 3 c 23 male  
## 4 a 24 female  
## 5 b 25 male  
## 6 c 26 female  
## 7 a 27 male  
## 8 b 28 female  
## 9 c 29 male  
## 10 a 21 female  
## 11 b 22 male  
## 12 c 23 female  
## 13 a 24 male  
## 14 b 25 female  
## 15 c 26 male  
## 16 a 27 female  
## 17 b 28 male  
## 18 c 29 female

# 进行条件索引的方法  
subset(df\_a, df\_a$name=='c') #获得符合条件的所有行

## name age gender  
## 3 c 23 male  
## 6 c 26 female  
## 9 c 29 male  
## 12 c 23 female  
## 15 c 26 male  
## 18 c 29 female

df\_a$name[df\_a$age > 21] #获得另一行符合条件的所有的元素,这里取的是元素

## [1] b c a b c a b c b c a b c a b c  
## Levels: a b c

subset(df\_a, age >22 & gender=='female') #与筛选

## name age gender  
## 4 a 24 female  
## 6 c 26 female  
## 8 b 28 female  
## 12 c 23 female  
## 14 b 25 female  
## 16 a 27 female  
## 18 c 29 female

subset(df\_a, age >22 | gender=='female') #或筛选

## name age gender  
## 2 b 22 female  
## 3 c 23 male  
## 4 a 24 female  
## 5 b 25 male  
## 6 c 26 female  
## 7 a 27 male  
## 8 b 28 female  
## 9 c 29 male  
## 10 a 21 female  
## 12 c 23 female  
## 13 a 24 male  
## 14 b 25 female  
## 15 c 26 male  
## 16 a 27 female  
## 17 b 28 male  
## 18 c 29 female

class(subset(df\_a, age >22 | gender=='female')) #获取的是子集，class仍是dataframe

## [1] "data.frame"

## 在R对象中获取特定的成分：访问，测试或替换对象中的元素(element)

# 方式：索引，子集或过滤  
vect <- c(a=1, b=4, c=7)  
  
# 索引：数据索引[], 逻辑索引which()  
  
vect[2]

## b   
## 4

vect[-2] #排除第2个元素

## a c   
## 1 7

which(vect==4) #返回为TRUE的元素的索引

## b   
## 2

# 取子集：不排除NA [条件]，排除NA用subset(v, 条件)  
v <- c(1:2, NA, 4:6, NA, 8:10)  
v[v>5] # 返回为TRUE的子集，保留NA

## [1] NA 6 NA 8 9 10

subset(v, v>5) # 返回为TREU的子集，排除缺失值NA

## [1] 6 8 9 10

df[3,1] # 按行列号索引

## [1] 3

df[,1] # 取第1列所有元素

## [1] 1 2 3