Vectors

Marcella Pedro

05/10/2021

Vetores

Vetor é uma estrutura de dados que permite armazenar um conjunto de dados do mesmo tipo.

Criar um vetor

Há várias formas de criar um vetor:

Através de um intervalo

Aqui cria um vetor que vai de 1 a 10

```
vetor_range <- 1:10
vetor_range
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

Através da função rep

A função cria uma repetição do número 1 por 10 vezes

```
vetor_rep <- rep(1, times=10)
vetor_rep
```

```
##  [1] 1 1 1 1 1 1 1 1 1 1
```

Através da função seq

Funciona similar ao intervalo já citado, mas aqui com mais flexibilidade.

Aqui foi possível criar uma sequência de 1 a 5 pulando a cada 0,5.

```
vetor_seq <- seq(1.0, 5.0, 0.5)
vetor_seq
```

```
## [1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
```

Através da função c

Função c é comumente utilizada par concatenar valores já estabelecidos.

```r
vetor_c <- c("A","B","C","D")
vetor_c
```

```
## [1] "A" "B" "C" "D"
```

Através de outros vetores

Podemos juntar vetores existentes para criar um novo

```r
vetor_final <- c(vetor_range, vetor_seq)
vetor_final
```

```
##  [1]  1.0  2.0  3.0  4.0  5.0  6.0  7.0  8.0  9.0 10.0  1.0  1.5  2.0  2.5  3.0
## [16]  3.5  4.0  4.5  5.0
```

Extrair informação do vetor

Acessar uma posição específica do vetor

Diferente da maioria das linguagens de programação, em R, a primeira posição do vetor é de fato 1, e não 0.

```r
vetor_c[1] #Traz primeira posição do vetor
```

```
## [1] "A"
```

Acessar várias posições em sequência do vetor

Traz os valores das posições 1, 2 e 3.

```r
vetor_c[1:3]
```

```
## [1] "A" "B" "C"
```

Acessar várias posições diferentes do vetor

Quando queremos várias posições específicas que não estão em sequência é necessário utilizar a função c.

```r
vetor_c[c(1,3,4)]
```

```
## [1] "A" "C" "D"
```

Filtrar posições a extrair

É possível filtrar as posições através do uso de outro vetor de booleanos. Quando o valor for `TRUE` apresenta o valor e quando `FALSE` não.

```r
filter <- c(TRUE, TRUE, FALSE, TRUE)
vetor_c[filter]
```

```
## [1] "A" "B" "D"
```

Modificar valores de um vetor

Alterar uma posição específica do vetor

Altera a primeira posição do vetor.

```
vetor_c[1] <- "Z"
vetor_c
```

```
## [1] "Z" "B" "C" "D"
```

Modificar várias posições em sequência do vetor

Altera os valores das posições 1, 2 e 3.

```
vetor_c[1:3] <- "Y"
vetor_c
```

```
## [1] "Y" "Y" "Y" "D"
```

Alterar várias posições diferentes do vetor

Altera em várias posições não-sequenciais.

```
vetor_c[c(1,3,4)] <- "X"
vetor_c
```

```
## [1] "X" "Y" "X" "X"
```

Filtrar posições a alterar

Altera as posições onde o filtro for `TRUE`.

```
filter <- c(TRUE, TRUE, FALSE, TRUE)
vetor_c[filter] <- "W"
vetor_c
```

```
## [1] "W" "W" "X" "W"
```

Eliminar valores de um vetor

Eliminar uma posição específica do vetor

Elimina a primeira posição do vetor.

```
vetor_final[-1]
```

```
##  [1]  2.0  3.0  4.0  5.0  6.0  7.0  8.0  9.0 10.0  1.0  1.5  2.0  2.5  3.0  3.5
## [16]  4.0  4.5  5.0
```

Eliminar várias posições em sequência do vetor

Elimina os valores das posições 1, 2 e 3.

```
vetor_final[-1:-3]
```

```
##  [1]  4.0  5.0  6.0  7.0  8.0  9.0 10.0  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5
## [16]  5.0
```

Eliminar várias posições diferentes do vetor

Elimina em várias posições não-sequenciais.

```
vetor_final[c(-1,-3,-4)]
```

```
##  [1]  2.0  5.0  6.0  7.0  8.0  9.0 10.0  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5
## [16]  5.0
```

Uma segunda alternativa

```
vetor_final[-c(1,3,4)]
```

```
##  [1]  2.0  5.0  6.0  7.0  8.0  9.0 10.0  1.0  1.5  2.0  2.5  3.0  3.5  4.0  4.5
## [16]  5.0
```

Realizar operações com vetores

Soma de vetores do mesmo tamanho

```
resultado<- vetor_rep + vetor_range
resultado
```

```
##  [1]  2  3  4  5  6  7  8  9 10 11
```

Soma de vetores de tamanhos distintos

Quando somamos vetores de tamanhos diferentes o R recicla os valores para completar o cálculo.

A partir do momento que um dos vetores chega ao fim o R retoma ao primeiro valor para continuar o cálculo enquanto o maior vetor não concluir.

```
resultado<- vetor_rep + vetor_seq
```

```
## Warning in vetor_rep + vetor_seq: comprimento do objeto maior não é múltiplo do
## comprimento do objeto menor
```

```
resultado
```

```
##  [1] 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 2.0
```

Utilizar nomes para indexar valores

Quando não atribuímos nomes, automaticamente os vetores são indexados por números iniciados em 1.

Mas podemos rotular cada valor:

```r
vetor_nomes<- c("A"=10, "B"=20, "C"=30)
vetor_nomes
```

```
##  A  B  C
## 10 20 30
```

Acessando valores pelo nome

```r
vetor_nomes["A"]
```

```
##  A
## 10
```

Acessando apenas nomes

```r
names(vetor_nomes)
```

```
## [1] "A" "B" "C"
```

Acessando apenas valores

```r
as.numeric(vetor_nomes)
```

```
## [1] 10 20 30
```

Por enquanto é só!