Linguagem de Programação

Cristiano de Carvalho Santos

Departamento de Estatística,
Universidade Federal de Minas Gerais (UFMG)

Agrupando comandos

- É possível atribuir os mesmos valores a vários objetos de uma só vez utilizando atribuições múltiplas de valores
  a <- b <- 10;
- Um grupo de comandos podem ser agrupado com "{ }" e separados por ";" para digitação em uma mesma linha. Em certas situações o uso das chaves é opcional

Execução condicional

Execuções condicionais são controladas por funções especiais que verificam se uma condição é satisfeita para permitir a execução de um comando. As seguintes funções e operadores podem ser usadas para controlar execução condicional.

- if() (opcionalmente acompanhado de else)
- ifelse()
- switch()
- &, |, && e ||

if

- A estrutura if() else é comumente usada, em especial dentro de funções;
- Quando a expressão que segue o if() e/ou else tem uma única linha ela pode ser escrita diretamente, entretanto, caso sigam-se mais de duas linhas deve-se usar chaves;

Exemplo 1:

```
if(condição){
    Ações a serem realizadas se
    condição for verdadeira
}
```

Exemplo 2:

```
if(condição){
    Ações a serem realizadas se
    a condição for verdadeira
} else{
    Ações a serem realizadas se
    a condição for falsa
}
```

else if é usado para especificar um segunda condição para o caso em que a primeira não foi atendida.

Exemplo 3:

```
if(condição 1){
   Ações a serem realizadas se
   a condição 1 for verdadeira
} else if(condição 2){
   Ações a serem realizadas se
   a condição 2 for verdadeira
}
```

ifelse

Uso:

`ifelse(teste, yes, no)`

- "teste" é a condição a ser testada;
- "yes" é o valor retornado se a condição for verdadeira;
- "no" é o valor retornado se a condição for falsa.

switch

- Permite várias comparações sem a necessidade de vários comandos de if;
- É mais rápido que o if.

Exemplo:

```
switch(tipo,
mean = 1,
median = 2,
trimmed = 3)
```

- "tipo" neste caso é uma variável (com números ou caracteres) que será testada;
- mean, median e trimmed são possíveis respostas para tipo. Se tipo for igual a mean a função retornará 1 e assim por diante.

comandos lógicos

- & e && são usados para fazer interseção de condições. No caso vetorial eles se diferem, sendo que & faz a interseção para cada entrada do vetor e && faz usando apenas a primeira entrada do vetor;
- | e || são usados para fazer união de condições e diferem da mesma forma que o comando de interseção;
- ! pode ser usado para negar uma condição.

Loops no R

- O controle de fluxo no R é implementado pelas funções for(), while() e repeat().
- A escolha de qual usar vai depender do contexto e objetivo do código e em geral não existe solução única, sendo que uma mesma tarefa pode ser feita por uma ou outra.

Estruturas

for

```
for(i in conjunto de indices){
    Ações a serem executadas para i
    igual a todos os indices
}
```

while

```
i <- 1
while(condição){
    Ações a serem executadas enquanto
    a condição for verdadeira
    i <- i + 1
}
```

repeat

```
i <- 1
repeat{
    Ações a serem executadas
    if(condição){
        break
    }
    i <- i+1
}
```

Tempo computacional

- **proc.time** determina o tempo real e tempo de CPU (em segundos) que o atual processo do R está em execução. Para descobrir o tempo de execução de uma função podemos subtrair o tempo depois e antes de rodar a função;
- **system.time** retorna o tempo de CPU e outros que o comando avaliado usou;
- Função microbenchmark compara o tempo computacional de execução de duas funções.