

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

## ENTRADA DE DADOS EM R

Funções de entrada de dados:

- `c()` comando de combinação

Exemplo:

```
> a=12
> b=2
> c=3
> z=c(a,b,c)
> z
[1] 12 2 3
```

- `rep()` comando de repetição de x, n vezes

Exemplo:

```
> rep(2)
[1] 2
> rep(200)
[1] 200
> rep(100,10)
[1] 100 100 100 100 100 100 100 100 100 100
```

- `seq()` comando sequência números de x(máximo) a y (mínimo)

```
> seq(100,10)
[1] 100 99 98 97 96 95 94 93 92 91
[11] 90 89 88 87 86 85 84 83 82 81
[21] 80 79 78 77 76 75 74 73 72 71
[31] 70 69 68 67 66 65 64 63 62 61
[41] 60 59 58 57 56 55 54 53 52 51
[51] 50 49 48 47 46 45 44 43 42 41
[61] 40 39 38 37 36 35 34 33 32 31
[71] 30 29 28 27 26 25 24 23 22 21
[81] 20 19 18 17 16 15 14 13 12 11
[91] 10
> seq(20,3)
[1] 20 19 18 17 16 15 14 13 12 11 10 9 8
[14] 7 6 5 4 3
```

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

Objetos de classes mais específicas como entrada de dados:

- `matrix()`

Para entendermos classes de matrizes, vamos estudar o comportamento de um vetor: vetores podem ser criados usando o comando `c` (*combine*), veja a seguir 2 exemplos, restrições e combinação de 2 vetores.

<pre>&gt; xvector&lt;- c(1,2,3,4,5) &gt; xvector [1] 1 2 3 4 5</pre>	<pre>&gt; yvector &lt;- c('cpu,memoria,disco') &gt; yvector [1] "cpu,memoria,disco"</pre>
<p><b>RESTRIÇÕES</b></p>	<pre>&gt; restrictvector&lt;- c(1,2, 'cpu', 'memoria') &gt; restrictvector [1] "1"    "2"    "cpu" [4] "memoria"</pre>
<p><b>Combinando dois vetores</b></p>	<pre>&gt; xvector&lt;- c(1,2,3,4,5) &gt; yvector&lt;-c(10,20,30,40,50) &gt; print(c(xvector,yvector)) [1] 1 2 3 4 5 10 20 30 40 50</pre>

Não me lembro mais dos objetos que criei, valem os dar o comando `ls()` para listar os objetos

```
> ls()
[1] "a"          "altura"
[3] "b"          "c"
[5] "dataset2"   "IMC"
[7] "info"       "name"
[9] "nome"       "peso"
[11] "restrictvector" "xvector"
[13] "yvector"    "z"
```

Temos vetores como o `xvector` e `yvector`, como tirar a média de um objeto `xvector` por exemplo.

```
> mean(xvector)
```

```
[1] 3
```

```
> summary(xvector)
```

```
Min. 1st Qu.  Median Mean 3rd Qu.  Max.
```

```
1      2      3      3      4      5
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

Vamos agora fazer uma simulação se houver dados NA nesse vetor.

NA é o retorno quando o argumento não é numérico e nem lógico.

```
> xvector<- c(1,2,3,4,5, NA)
```

```
> mean(xvector)
```

```
[1] NA
```

Veja que o comando mean () que calcula a média retorna o NA.

Vamos remover o NA

```
> xvector<- c(1,2,3,4,5, NA)
```

```
> xvector
```

```
[1] 1 2 3 4 5 NA
```

```
> mean(xvector, NA.rm=TRUE)
```

```
[1] NA
```

```
> mean(xvector, na.rm=TRUE)
```

```
[1] 3
```

Concatenação com vetor

```
> x<-(1:5)
```

```
> x
```

```
[1] 1 2 3 4 5
```

```
> cat("numeros de 1 a 5 ", x)
```

```
numeros de 1 a 5 1 2 3 4 5
```

Vamos agora analisar os objetos vetoriais criados a seguir.

a) Tamanho do vetor

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
> x<- c(1,1,2,2,3,3,4,4,5,5)
```

```
> vectort<-length(x)
```

```
> cat(paste('O tamanho do vetor é',vectort))
```

O tamanho do vetor é 10

### b) Matrix

```
> m<-matrix(2:30, nrow=5,ncol=6)
```

Warning message:

In matrix(2:30, nrow = 5, ncol = 6) :

comprimento dos dados [29] não é um submúltiplo ou múltiplo do número de linhas [5]

```
> m<-matrix(1:30, nrow=5,ncol=6)
```

```
> m
```

```
 [,1] [,2] [,3] [,4] [,5] [,6]
```

```
[1,]  1   6  11  16  21  26
```

```
[2,]  2   7  12  17  22  27
```

```
[3,]  3   8  13  18  23  28
```

```
[4,]  4   9  14  19  24  29
```

```
[5,]  5  10  15  20  25  30
```

```
> numberlines<-nrow(m)
```

```
> numbercol<-ncol(m)
```

```
> matrizt<-length(m)
```

```
> cat(paste('\n',numberlines,'\n',numbercol,'\n',matrizt))
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

5

6

30

Mas falta saber a dimensão da matriz

```
> print(dim(m))
```

```
[1] 5 6
```

Pronto, a matriz de duas dimensões já sabemos utilizar, agora vamos fazer matrizes com mais de duas dimensões.

```
> array1<-array(1:20, dim = c(4,4,4))
```

```
> print(array1)
```

```
,, 1
```

```
 [1] [2] [3] [4]
```

```
[1,] 1 5 9 13
```

```
[2,] 2 6 10 14
```

```
[3,] 3 7 11 15
```

```
[4,] 4 8 12 16
```

```
,, 2
```

```
 [1] [2] [3] [4]
```

```
[1,] 17 1 5 9
```

```
[2,] 18 2 6 10
```

```
[3,] 19 3 7 11
```

```
[4,] 20 4 8 12
```

```
,, 3
```

```
 [1] [2] [3] [4]
```

```
[1,] 13 17 1 5
```

```
[2,] 14 18 2 6
```

```
[3,] 15 19 3 7
```

```
[4,] 16 20 4 8
```

```
,, 4
```

```
 [1] [2] [3] [4]
```

```
 [1,] 9 13 17 1
```

```
[2,] 10 14 18 2
```

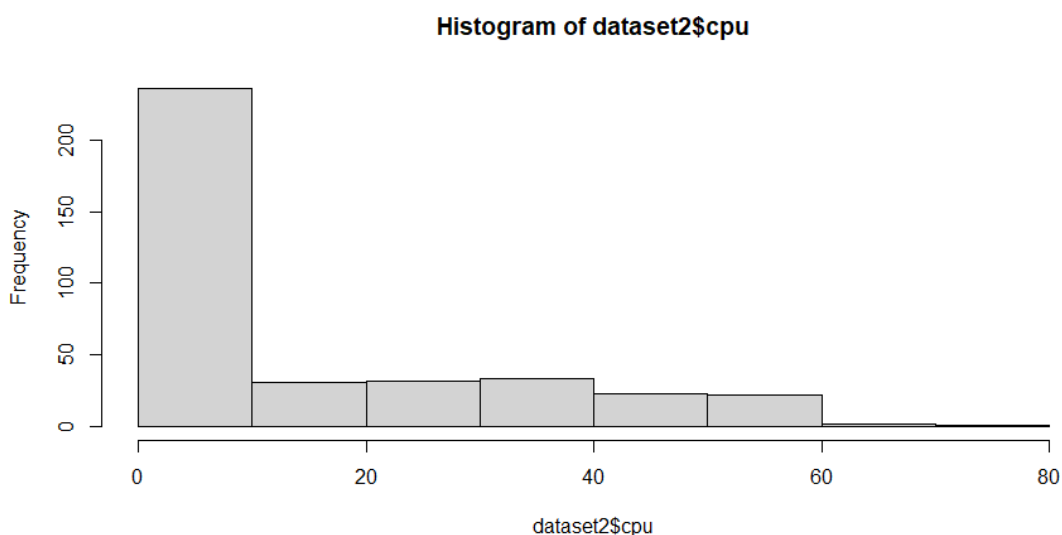
```
[3,] 11 15 19 3
```

```
[4,] 12 16 20 4
```

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

Vamos aplicar este estudo em nossa tabela, matriz de duas dimensões

```
> mean(dataset2$cpu)
[1] 15.70737
> cpu_media<-mean(dataset2$cpu)
> round(cpu_media,2)
[1] 15.71
> summary(dataset2$cpu)
  Min. 1st Qu. Median Mean 3rd Qu. Max.
  0.80  3.80  7.20 15.71 26.85 75.20
> hist(dataset2$cpu)
> hist(dataset2$cpu,xlab='cpu_uso',ylab = 'frequencia')
> hist(dataset2$cpu,xlab='cpu_uso',ylab = 'frequencia', main = 'Frequência uso cpu')
```



Observe esta distribuição, em que faixa ocorre o maior uso de cpu?

O que será que ocorre com a memória, memória virtual e disco se aplicar a mesma técnica.?

- `list()`

Este comando lista os dados do vetor ou array.

Faça o teste, veja o comando a seguir:

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
> list(dataset2$cpu)
[[1]]
[1] 8.2 4.0 14.8 5.7 6.5 3.4 28.4 10.0 9.5 7.8 7.0 17.9
[13] 3.2 3.4 5.6 4.8 10.7 8.6 4.8 5.9 2.2 2.3 6.7 2.0
[25] 2.7 4.9 5.4 5.2 2.8 4.7 5.5 4.8 7.2 2.4 2.5 5.2
[37] 6.7 3.7 9.0 5.2 3.7 4.4 11.9 5.2 2.3 7.0 4.7 7.8
[49] 5.9 5.5 5.5 5.8 4.7 6.7 7.9 5.5 2.5 4.1 3.8 28.2
.....
```

Aqui você começa a entender a importância da representação gráfica dos dados.

- `data.frame()`

Um Data Frame é essencialmente uma lista de vetores e fatores. Cada vetor ou fator representa uma coluna em um data frame (AWS Application about Chang,2012).

Diferentemente de vetores, que não podem agregar elementos de classes diferentes, um data frame pode conter colunas com **vetores numéricos e variáveis categóricas**.

Variáveis quantitativas podem ser classificadas como discretas ou contínuas.

**variáveis categóricas: representam categorias ou classes: escolaridade, cidade, sexo, etc.**

É por esse motivo que a maior parte dos dados que utilizamos em análises estatísticas são formatados como data frames. Em um data frame, as linhas representam as observações e as colunas representam as variáveis.

Como criar data frames a partir de vetores?

Imagine que temos dados de crescimento de plântulas (altura em cm) que foram submetidas a dois níveis de luminosidade: (i) com muita luz e (ii) com pouca luz. Podemos utilizar a função `data.frame` para juntar esses dois vetores em um objeto do tipo `data.frame`:

```
> alt <- c(3,4,3,2,5,2,3,1,3,2,6,5,4,8,6,3,5,3,7,8)
> luminosidade <- rep(c("muita", "pouca"),each=10)
> dados <- data.frame(alt,luminosidade)
> dados
  alt luminosidade
1  3      muita
2  4      muita
3  3      muita
4  2      muita
5  5      muita
6  2      muita
7  3      muita
8  1      muita
9  3      muita
10 2      muita
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
11 6 pouca
12 5 pouca
13 4 pouca
14 8 pouca
15 6 pouca
16 3 pouca
17 5 pouca
18 3 pouca
19 7 pouca
20 8 pouca
```

```
> str(dados)
'data.frame': 20 obs. of 2 variables:
 $ alt : num 3 4 3 2 5 2 3 1 3 2 ...
 $ luminosidade: chr "muita" "muita" "muita" "muita" ...
```

Verificar as variáveis de forma individualizada use o class()

```
> class(alt)
[1] "numeric"
> class(luminosidade)
[1] "character"
> class(data.frame)
[1] "function"
```

Criar outro objeto no data frame

```
> fol<-c(19,21,18,18,16,21,23,21,25,22,9,7,6,7,12,9,12,2,9,4)
> dados$fol<-fol
> dados
  alt luminosidade fol
1  3      muita  19
2  4      muita  21
3  3      muita  18
4  2      muita  18
5  5      muita  16
6  2      muita  21
7  3      muita  23
8  1      muita  21
9  3      muita  25
10 2      muita  22
11 6      pouca   9
12 5      pouca   7
13 4      pouca   6
14 8      pouca   7
15 6      pouca  12
16 3      pouca   9
17 5      pouca  12
18 3      pouca   2
```



# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
19 7 pouca 9
20 8 pouca 4
```

```
> names(dados)
```

```
[1] "alt" "luminosidade" "fol"
```

```
> names(dados)<-c("altura","luz","folhas")
```

```
> dados
```

```
altura luz folhas
1 3 muita 19
2 4 muita 21
3 3 muita 18
4 2 muita 18
5 5 muita 16
6 2 muita 21
7 3 muita 23
8 1 muita 21
9 3 muita 25
10 2 muita 22
11 6 pouca 9
12 5 pouca 7
13 4 pouca 6
14 8 pouca 7
15 6 pouca 12
16 3 pouca 9
17 5 pouca 12
18 3 pouca 2
19 7 pouca 9
20 8 pouca 4
```

```
> dados$folhas
```

```
[1] 19 21 18 18 16 21 23 21 25 22          9 7 6 7 12 9 12
[18] 2 9 4
```

```
> head(dados)
```

```
altura luz folhas
1 3 muita 19
2 4 muita 21
3 3 muita 18
4 2 muita 18
5 5 muita 16
6 2 muita 21
```

```
> tail(dados)
```

```
altura luz folhas
15 6 pouca 12
16 3 pouca 9
17 5 pouca 12
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

18	3 pouca	2
19	7 pouca	9
20	8 pouca	4

> dados[1,]

	altura	luz	folhas
1	3	muita	19

> dados[1:3,]

	altura	luz	folhas
1	3	muita	19
2	4	muita	21
3	3	muita	18

> dados[c(1,2,5),]

	altura	luz	folhas
1	3	muita	19
2	4	muita	21
5	5	muita	16

> dados[,2]

[1]	"muita"	"muita"	"muita"	"muita"	"muita"	"muita"
[7]	"muita"	"muita"	"muita"	"muita"	"pouca"	"pouca"
[13]	"pouca"	"pouca"	"pouca"	"pouca"	"pouca"	"pouca"
[19]	"pouca"	"pouca"				

> dados[8,3] [1]

21

> dados[3,1:2]

	altura	luz
3	3	muita

> dados

	altura	luz	folhas	altura1
1	3	muita	19	3
2	4	muita	21	4
3	3	muita	18	3
4	2	muita	18	2
5	5	muita	16	5
6	2	muita	21	2
7	3	muita	23	3
8	1	muita	21	1
9	3	muita	25	3
10	2	muita	22	2
11	6	pouca	9	6
12	5	pouca	7	5
13	4	pouca	6	4
14	8	pouca	7	8

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

15	6 pouca	12	6
16	3 pouca	9	3
17	5 pouca	12	5
18	3 pouca	2	3
19	7 pouca	9	7
20	8 pouca	4	8

```
> dados$altura1<-NULL
```

```
> dados
```

	altura	luz folhas
1	3 muita	19
2	4 muita	21
3	3 muita	18
4	2 muita	18
5	5 muita	16
6	2 muita	21
7	3 muita	23
8	1 muita	21
9	3 muita	25
10	2 muita	22
11	6 pouca	9
12	5 pouca	7
13	4 pouca	6
14	8 pouca	7
15	6 pouca	12
16	3 pouca	9
17	5 pouca	12
18	3 pouca	2
19	7 pouca	9
20	8 pouca	4

```
>dados
```

	altura	luz folhas
1	3 muita	19
2	4 muita	21
3	3 muita	18
4	2 muita	18
5	5 muita	16
6	2 muita	21
7	3 muita	23
8	1 muita	21
9	3 muita	25
10	2 muita	22
11	6 pouca	9
12	5 pouca	7
13	4 pouca	6
14	8 pouca	7
15	6 pouca	12
16	3 pouca	9
17	5 pouca	12
18	3 pouca	2
19	7 pouca	9
20	8 pouca	4

Podemos criar subconjuntos de data frames

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
> subset(dados, luz == 'muita')
```

```
altura luz folhas
1    3 muita   19
2    4 muita   21
3    3 muita   18
4    2 muita   18
5    5 muita   16
6    2 muita   21
7    3 muita   23
8    1 muita   21
9    3 muita   25
10   2 muita   22
```

```
> dados[dados$altura > 3 & dados$luz == "muita", ]
```

```
altura luz folhas
2    4 muita   21
5    5 muita   16
```

```
> subset(dados, altura > 3)
```

```
altura luz folhas
2    4 muita   21
5    5 muita   16
11   6 pouca    9
12   5 pouca    7
13   4 pouca    6
14   8 pouca    7
15   6 pouca   12
17   5 pouca   12
19   7 pouca    9
20   8 pouca    4
```

```
scaread
```

```
> subset(dados, altura > 3 & luz == "muita")
```

```
altura luz folhas
2    4 muita   21
5    5 muita   16
```

```
> subset(dados, altura > 3 & luz == "muita" & folhas > 20)
```

```
altura luz folhas
2    4 muita   21
```

```
> dados[1,1] <- 4
```

```
> dados
```

```
altura luz folhas
1    4 muita   19
2    4 muita   21
3    3 muita   18
4    2 muita   18
5    5 muita   16
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
6  2 muita  21
7  3 muita  23
8  1 muita  21
9  3 muita  25
10 2 muita  22
11 6 pouca  9
12 5 pouca  7
13 4 pouca  6
14 8 pouca  7
15 6 pouca  12
16 3 pouca  9
17 5 pouca  12
18 3 pouca  2
19 7 pouca  9
20 8 pouca  4
```

```
> dados[2,1] <- 5
> dados
```

```
altura luz folhas
1  4 muita  19
2  5 muita  21
3  3 muita  18
4  2 muita  18
5  5 muita  16
6  2 muita  21
7  3 muita  23
8  1 muita  21
9  3 muita  25
10 2 muita  22
11 6 pouca  9
12 5 pouca  7
13 4 pouca  6
14 8 pouca  7
15 6 pouca  12
16 3 pouca  9
17 5 pouca  12
18 3 pouca  2
19 7 pouca  9
20 8 pouca  4
```

```
> dados[dados$altura > 3,1]
[1] 4 5 5 6 5 4 8 6 5 7 8
```

```
> dados[dados$altura > 3,1]<-10
> dados
```

```
altura luz folhas
1  10 muita  19
2  10 muita  21
3  3 muita  18
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

4	2 muita	18
5	10 muita	16
6	2 muita	21
7	3 muita	23
8	1 muita	21
9	3 muita	25
10	2 muita	22
11	10 pouca	9
12	10 pouca	7
13	10 pouca	6
14	10 pouca	7
15	10 pouca	12
16	3 pouca	9
17	10 pouca	12
18	3 pouca	2
19	10 pouca	9
20	10 pouca	4

Entrada de dados via teclado, por digitação do usuário:

- `scan()`

```
foi<-scan()
```

```
1: 1
```

```
2: 2
```

```
3: 3
```

```
4: 4
```

```
5: 5
```

```
6: 6
```

```
7: 7
```

```
8: 8
```

```
9: 9
```

```
10: 000
```

```
Read 10 items
```

Entrada de dados no formato texto:

- `readlines()`

```
> readline(prompt = "")
```

```
olá tudo bem
```

```
[1] "olá tudo bem"
```

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

**EX1)** Tomando o exercício de IMC da aula anterior, realizado por meio de comandos em R, vamos agora fazer um pequeno **script** de automação, interagindo com o usuário.

$$\text{IMC} = \text{Peso} / (\text{altura} * \text{altura})$$

Peso = 75

Altura = 1.80      # atenção use ponto em não vírgula para decimais

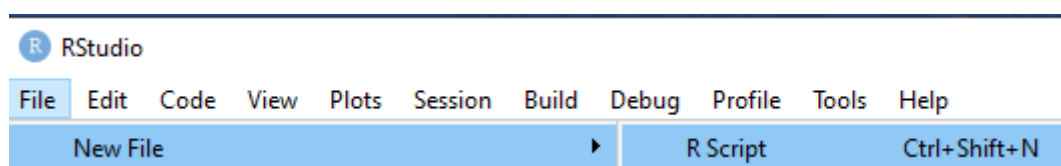
Os comandos em R ficaram assim:

```
> peso=75  
  
> altura = 1.80  
  
> IMC = 75/(1.80 *1.8)  
  
> IMC  
  
[1] 23.14815  
  
> round(IMC, 1)  
  
[1] 23.1
```

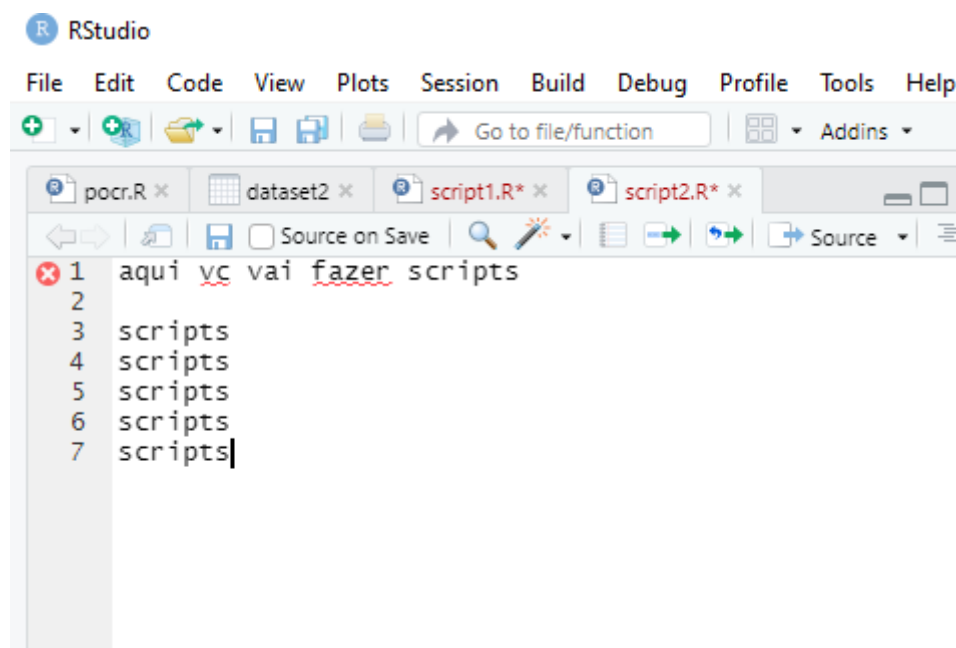
Vamos automatizar como um script os comandos anteriores:

Vamos abrir o RScript:

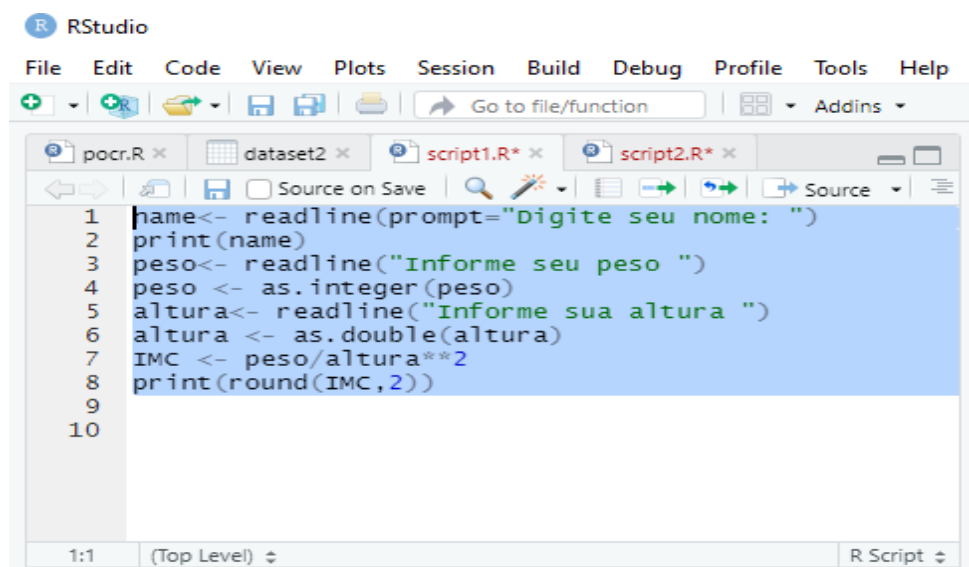
Vá em File, New File e R Script



# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC



Digite o pequeno script na tela a seguir

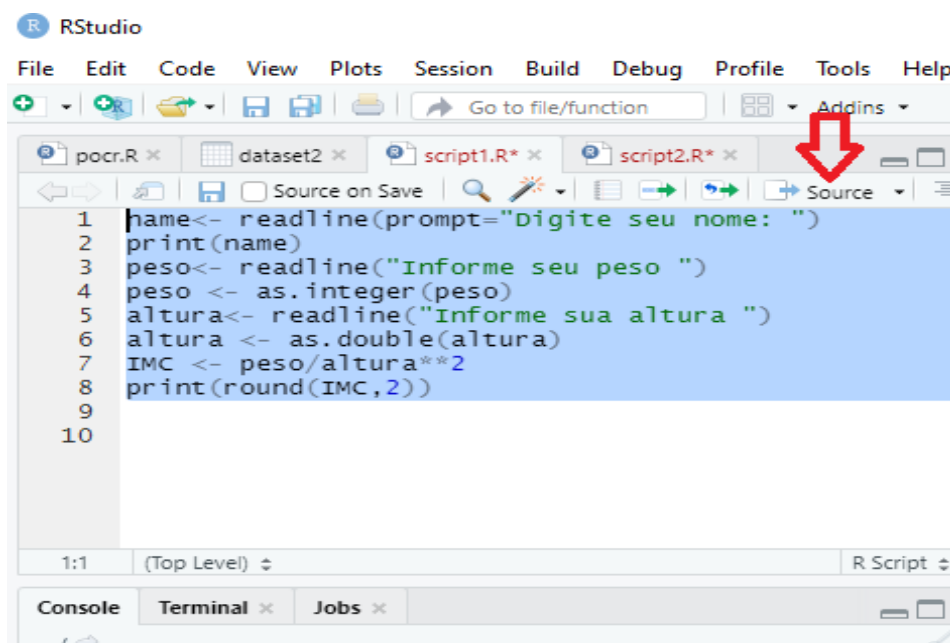


Vá em file, e salve o script por exemplo em sua área de trabalho em uma pasta scriptsR

Clic em source para rodar o script na console de interação



# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC



```

1 hame<- readline(prompt="Digite seu nome: ")
2 print(name)
3 peso<- readline("Informe seu peso ")
4 peso <- as.integer(peso)
5 altura<- readline("Informe sua altura ")
6 altura <- as.double(altura)
7 IMC <- peso/altura**2
8 print(round(IMC,2))
9
10

```

Ou CTRL + shift + s para rodar o script salvo

> source('C:/Users/Marise/Desktop/dados para R/script1.R')

Digite seu nome: marise

[1] "marise"

Informe seu peso 35

Informe sua altura 3

[1] 3.89

Agora que você já sabe fazer os scripts, vamos ver os tipos de dados que utilizamos, use o comando typeof(), que verificar o tipo de dado da variável.

> typeof(name)

[1] "character"

> typeof(IMC)

[1] "double"

> typeof(peso)

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
[1] "integer"  
> typeof(altura)  
[1] "double"
```

O mesmo pode ser obtido com a função `class()`

```
> class(name)  
[1] "character"
```

Se dermos o comando `paste`, vai concatenar as variáveis numéricas e strings)

```
> paste(name,peso,altura,print(round(IMC,2)))  
[1] 21.22  
[1] "marise 65 1.75 21.22"
```

Se utilizarmos o comando `c` de **combine**, será necessário atribuir essa combinação a uma nova variável.

```
> info<-c(name,peso,altura)  
> info  
[1] "marise" "65" "1.75"
```

### EX2) CONTINUAÇÃO: Calculando o IMC como “Projeto a ser aplicado a conjunto de dados”, integração de funções

Vamos criar um dataset com pelo menos 5 colunas numéricas e 3 colunas categóricas, através do uso de **funções de criação de distribuições randômicas (aleatórias)**.

Tarefas:

1. Calcular somas e médias
2. Criar gráficos simples

A especificação detalhada de como deve ser o dataset não foi especificada, usaremos a simulação para definir variáveis que visem a demonstração do modelo.

#### Criando um dataset simulado

**Contexto simulado:** estudo sobre o índice de massa corpórea em adultos jovens na qual foram inserido dados simulados antropométricos, demográficos e de alguns fatores de risco.

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

As variáveis coletadas estão detalhadas na tabela abaixo (incluindo o tipo da variável, sua representação em R e outras informações importantes):

Variável	Observações	Tipo	Representação no R
idade	Em anos completos	Dimensional de razão, discreta	Numeric
altura	Em metros (m)	Dimensional de razão, contínua	Numeric
peso	Em quilo (Kg)	Dimensional de razão, contínua	Numeric
imc	peso/altura2	Índice dimensional de razão, contínuo	Numeric
sexo	1 = Masculino		
2 = Feminino	Nominal	Unordered Factor	
escolaridade	0 = Analfabeto		
1 = 1º grau completo			
2 = 2º grau completo			
3 = 3º grau completo			
4 = mestrado			
5 = doutorado			
6 = pós-doutorado	Ordinal	Ordered Factor	
profissao	1 = Humanas		
2 = Exatas			
3 = Biológicas	Nominal	Unordered Factor	
fumante	0 = Não		
1 = Sim	Binária	Ordered Factor	
salario	Em reais (R\$)	Dimensional de razão, contínua	Numeric
carros	Número de carros	Dimensional de razão, discreta	Numeric
filhos	Número de filhos	Dimensional de razão, discreta	Numeric

## Simulação das variáveis do dataset

Podemos propor, como teste, que as informações simuladas advém de 20.000 observações, retiradas aleatoriamente por algum processo de amostragem a partir de uma população de 200.000 indivíduos. Sendo  $n$ , número de observações e  $p$  o universo amostral.

$n <- 20000$

$p <- 200000$

Variáveis dimensionais:

A1) Idade

A variável idade (em anos completos) foi simulada a partir de uma distribuição normal, com o uso da função `rnorm` ajustada para uma média de idade de 37 anos com desvio padrão de 7 anos. Para evitar quaisquer números negativos foi utilizada a função `abs` e para manter a idade em anos completos o resultado foi arredondado para zero casas decimais com a função `round`. A função `set.seed` foi utilizada para tornar os resultados reprodutíveis.

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
> n <- 20000
```

Observe que já atribuímos o valor para n anteriormente

```
> set.seed(1234)
```

```
> idade <- abs(round(rnorm(n, 35, 7),0))
```

```
> summary(idade)
```

```
Min. 1st Qu. Median Mean 3rd Qu. Max.
```

```
6.00 30.00 35.00 34.99 40.00 61.00
```

Agora vamos representar essas idades do novo conjunto de dados

```
> idade
```

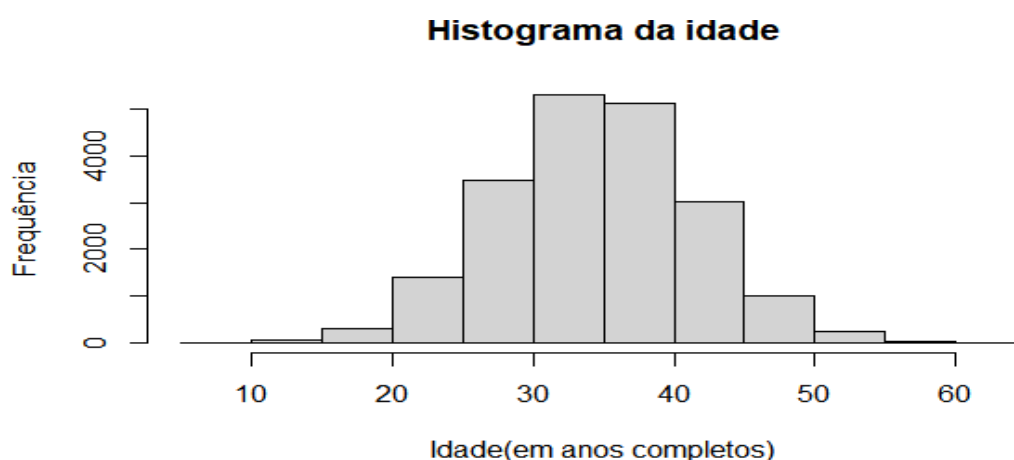
```
[1] 27 37 43 19 38 39 31 31 31 29 32 28 30 35 42 34 31 29 29 52 36 32 32 38 30 25 39 28 35 28 43 32 30
[34] 31 24 27 20 26 33 32 45 28 29 33 28 28 27 26 31 32 22 31 27 28 34 39 47 30 46 27 40 53 35 30 35 47
[67] 27 45 44 37 35 32 32 40 49 34 25 30 37 33 34 34 25 34 41 40 39 32 34 27 35 37 47 42 32 37 27 41 42
[100] 50 38 32 35 31 29 36 29 36 37 35 34 30 27 41 35 41 26 36 40 35 34 30 49 40 48 36 31 24 31 37 42 37
[133] 27 40 23 32 33 21 41 31 33 45 39 34 39 38 47 37 39 37 32 36 46 29 36 45 33 28 29 32 29 33 32 34 38
[166] 39 47 35 33 45 47 35 33 22 45 29 27 56 37 35 16 34 42 38 41 49 43 31 40 34 31 15 29 38 50 39 39 28
[199] 36 20 38 40 36 40 37 40 48 43 35 27 38 32 45 24 32 38 34 31 33 39 41 40 51 43 37 30 55 40 30 36 33
[232] 33 28 36 37 27 12 33 35 39 35 38 27 40 40 37 45 38 37 37 38 42 38 40 27 32 45 26 37 38 42 33 36 23
[265] 37 30 33 27 38 40 33 48 40 42 49 30 41 42 35 37 28 38 30 41 29 37 29 33 24 36 42 36 23 31 47 48 27
[298] 32 37 37 31 28 34 42 35 30 31 46 32 40 45 38 32 35 32 44 34 46 31 30 36 35 31 24 35 30 43 28 43 30
[331] 37 41 31 39 34 27 33 46 34 29 35 37 42 43 36 43 30 19 32 31 19 39 30 33 24 39 48 29 34 49 42 48 35
[364] 34 31 45 26 37 35 51 31 24 37 35 29 36 33 30 39 33 22 16 31 45 41 44 42 37 24 36 46 11 30 43 39 41
[397] 38 39 35 29 26 35 32 29 38 36 45 27 31 34 25 33 30 20 33 31 43 31 39 36 26 38 34 36 40 36 46 36 26
[430] 40 26 22 29 30 31 46 46 40 26 35 38 34 36 28 26 45 37 32 37 49 38 45 45 32 40 30 25 27 34 48 36 29
[463] 37 40 26 28 21 26 29 33 45 41 37 31 35 39 31 42 43 27 30 43 23 38 37 57 16 29 40 47 41 25 51 23 27
[496] 35 51 39 40 47 42 26 40 34 47 33 24 38 37 35 28 29 35 42 32 29 16 38 38 44 45 34 44 35 30 29 33 37
[529] 35 36 36 40 33 46 38 27 33 39 32 37 38 44 30 35 31 27 22 36 39 39 25 27 27 30 36 39 29 15 42 39 30
[562] 27 33 36 40 45 29 37 44 29 33 23 37 31 29 30 38 20 32 28 28 38 37 34 39 31 21 43 33 32 36 40 38 41
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
[595] 35 31 33 41 29 28 37 39 23 40 35 34 43 45 27 42 31 36 38 36 25 31 46 36 29 30 42 37 29 35 39 47 29
[628] 32 42 32 45 42 23 38 25 25 35 35 35 44 37 30 36 31 35 27 34 36 30 35 30 30 34 38 34 36 42 33 34 39
[661] 26 39 31 27 23 30 27 27 46 27 42 39 29 31 32 36 33 41 33 20 34 22 42 30 27 31 39 22 37 28 35 40 39
[694] 46 36 30 50 33 41 42 38 45 34 26 32 31 37 28 40 27 47 45 32 36 32 27 16 35 35 46 21 30 41 37 36 31
[727] 36 33 36 31 43 42 44 32 33 33 45 33 36 38 40 33 51 29 25 35 47 35 36 41 34 36 43 36 30 38 27 25 34
[760] 27 39 34 30 46 34 45 34 33 38 42 54 28 34 43 37 34 26 33 34 34 46 32 24 38 34 34 39 15 31 34 39 33
[793] 28 38 35 36 43 31 26 26 28 25 35 48 34 40 27 33 39 33 40 40 31 30 52 34 35 35 31 29 27 41 28 40 45
[826] 43 34 31 27 35 33 36 31 39 40 43 51 36 31 34 24 36 32 38 33 25 35 26 41 22 34 38 24 45 29 31 36 33
[859] 28 38 31 39 31 28 34 40 37 35 21 45 35 29 34 37 24 40 37 43 42 31 32 28 32 42 48 41 22 41 42 39 29
[892] 42 40 40 32 41 36 35 45 40 15 32 25 39 32 41 38 35 34 32 36 27 40 32 30 33 48 23 33 36 31 32 38 37
[925] 30 39 36 40 26 30 34 42 31 28 40 38 26 49 45 25 50 23 36 27 32 33 37 27 16 34 30 36 29 41 35 39 28
[958] 34 37 34 47 21 40 35 42 34 16 38 33 28 30 35 52 36 28 48 32 29 26 30 49 41 40 36 37 39 37 36 42 34
[991] 37 29 46 45 22 33 39 38 40 32

[ reached getOption("max.print") -- omitted 19000 entries ]
```

```
> hist. (idade,
  main = "Histograma da idade",
  ylab = "Frequência",
  xlab= "Idade (em anos completos)")
```



Observe que pelo gráfico conseguimos compreender o conjunto de dados.

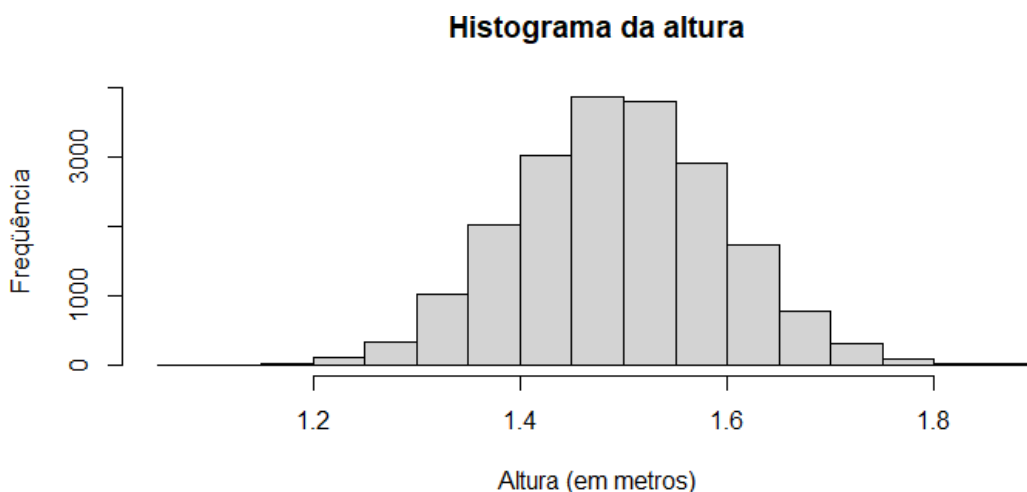
# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 -

## Aplicação em Case de IMC

### A2) Altura

A altura (em metros) seguiu a mesma lógica da simulação da idade, utilizando-se uma distribuição normal com média 1,50 m e desvio padrão de 0,2 m. Entretanto, como a altura é uma variável dimensional de razão e contínua, utilizei duas casas decimais na simulação:

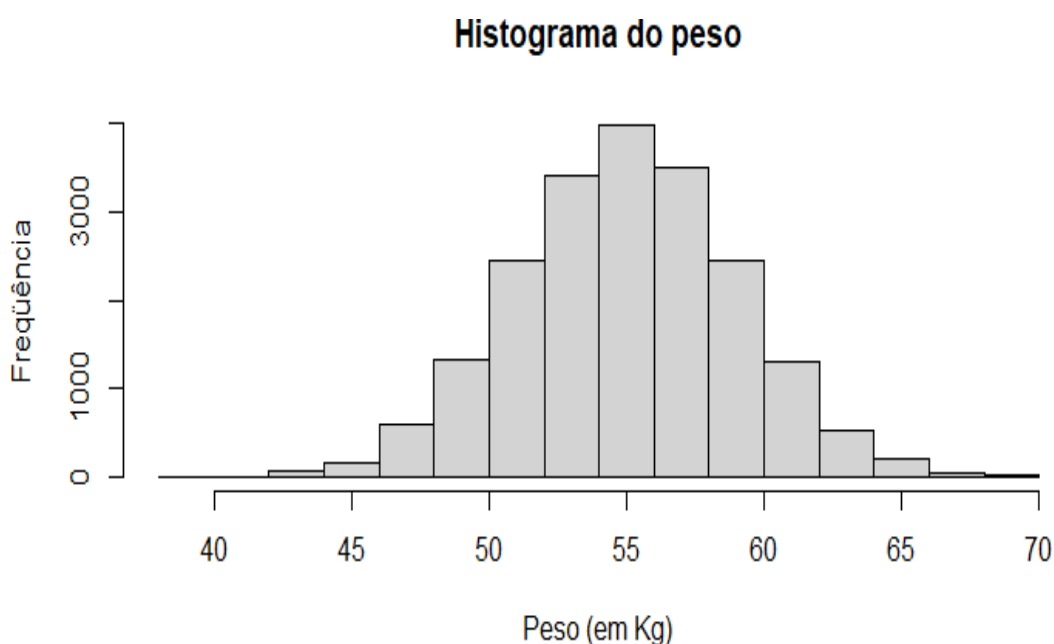
```
> set.seed(1234)
> altura <- abs(round(rnorm(n, 1.50, 0.1), 2))
> summary(altura)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 1.09  1.43   1.50   1.50  1.57  1.87
> hist(altura,
+   main = "Histograma da altura",
+   ylab = "Frequência",
+   xlab = "Altura (em metros)")
```



### A3) Peso

```
> set.seed(1234)
> peso <- abs(round(rnorm(n, 55, 4), 2))
> summary(peso)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 38.49 52.27 55.01 55.00 57.69 69.91
> hist(peso,
+   main = "Histograma do peso",
+   ylab = "Frequência",
+   xlab = "Peso (em Kg)")
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

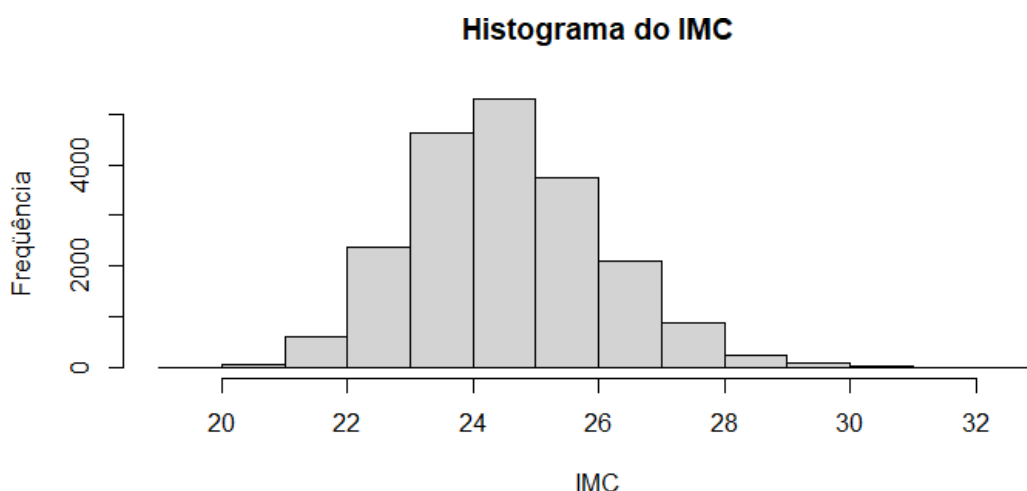


### A4) IMC

Utilize a fórmula padrão sem ajuste antropométrico, com arredondamento de 1 casa decimal.

```
> imc <- round(peso/altura^2, 2)
> summary(imc)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
19.99 23.49  24.44 24.54 25.49  32.40
> hist(imc,
+   main = "Histograma do IMC",
+   ylab = "Frequência",
+   xlab = "IMC")
```

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC



Sobre geração de números aleatórios.

**Set.seed ();** Foram gerados números aleatoriamente em um data frame. Para que a geração de números aleatórios mantenha certa constância, ou seja, as amostras devem ser idênticas, cada vez que são geradas. A geração dessa amostra depende de um *gerador de números aleatórios* que é controlado por uma *semente (seed em inglês)*. Cada vez que o comando `rnorm` é chamado diferentes elementos da amostra são produzidos, porque a *semente* do gerador é automaticamente modificada pela função. Em geral o usuário não precisa se preocupar com este mecanismo. Mas caso necessário a função `set.seed` pode ser usada para controlar o comportamento do gerador de números aleatórios.

**rnorm** é um comando de geração de um vetor numérico randômico com comportamento de distribuição normal

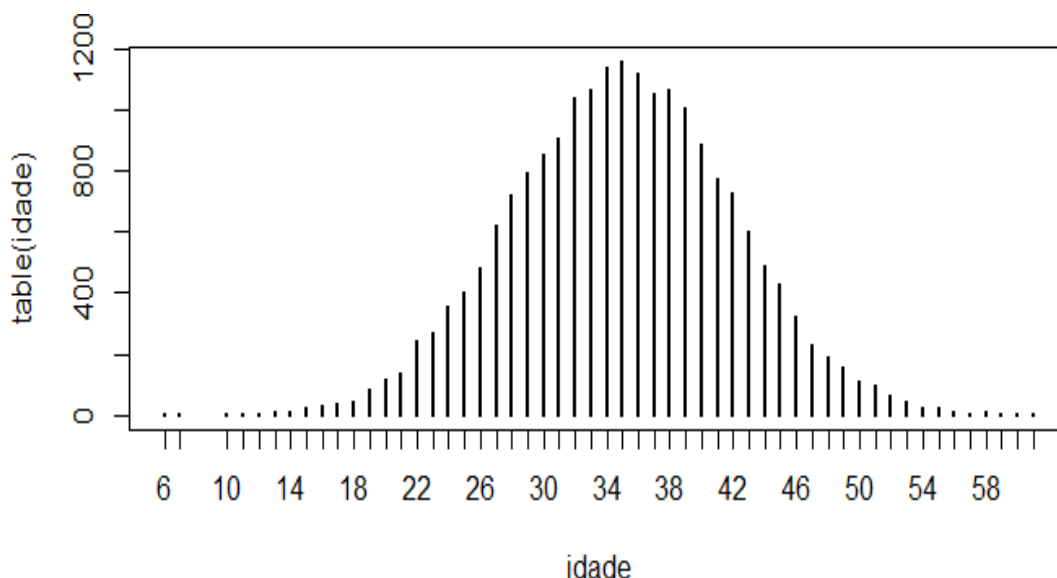
Vamos fazer um teste como é essa distribuição normal, por frequências.

```
> table(idade)
idade
 6 7 10 11 12 13 14 15 16 17 18 19
 1 1 1 1 3 11 12 21 30 34 45 81
20 21 22 23 24 25 26 27 28 29 30 31
116 140 240 272 353 406 485 620 719 794 857 905
32 33 34 35 36 37 38 39 40 41 42 43
1039 1068 1141 1159 1121 1057 1069 1007 886 778 726 600
44 45 46 47 48 49 50 51 52 53 54 55
491 428 323 227 188 154 111 97 64 41 23 23
56 57 58 59 60 61
12 4 8 1 5 1
```

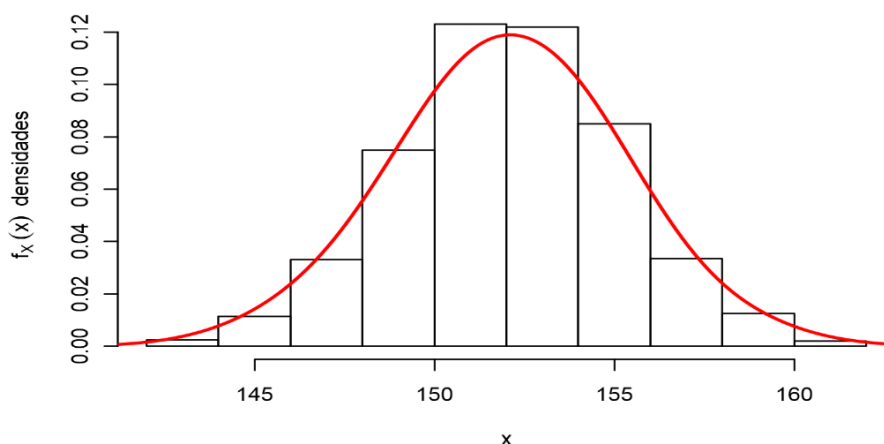
```
> plot(table(idade))
```



# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC



Vamos só recordar como são os conjuntos de dados em dois tipos de distribuições: Distribuição normal os números se distribuem equilibradamente, concentração no meio do gráfico.



## A5) Salário

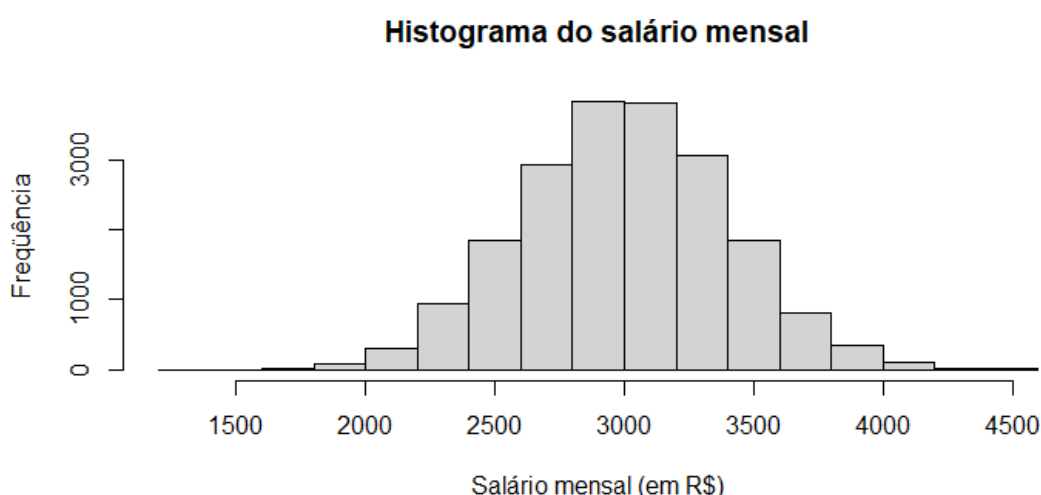
A variável salário (em reais) foi simulada através de uma distribuição normal:

```
> salario <- abs(round(rnorm(n, 3000, 400), 2))
> summary(salario)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1349	2727	3001	3000	3269	4491

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
> hist(salario,  
+      main = "Histograma do salário mensal",  
+      ylab = "Frequência",  
+      xlab = "Salário mensal (em R$)")
```



## A6) Carros

A variável número de carros foi simulada através de uma amostragem de valores de uma população de números (de 0 a 3). A população de valores foi criada com a função `rep` e a amostra foi retirada com a função `sample`.

```
> pop.carros <- rep(c(0,1,2,3), p)  
> pop.carros  
[1] 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3  
[33] 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3  
[65] 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3  
[513] 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3  
[545] 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3  
.....  
[929] 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3  
[961] 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3 0 1 2 3  
[993] 0 1 2 3 0 1 2 3
```

```
> rm(pop.carros)
```

```
> pop.carros
```

```
> summary(carros)
```

```
Min. 1st Qu. Median Mean 3rd Qu. Max.
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

0.000 0.000 1.000 1.494 2.000 3.000

> carros

```
[1] 3 1 3 1 2 1 2 1 3 3 3 3 1 2 3 2 0 1 3 1 2 0 3 1 0 3 0 1 2 3 1 1
[33] 1 0 2 1 3 2 3 0 1 3 3 0 2 2 1 0 1 1 0 1 1 3 3 2 3 1 1 2 0 1 0 0
[65] 3 1 1 1 2 1 2 2 2 3 3 1 3 0 2 2 2 3 2 2 2 0 0 2 3 3 0 2 1 1 1 2
[97] 2 3 0 2 3 2 1 3 3 1 0 3 0 2 3 2 0 0 1 3 0 1 0 0 0 1 1 2 3 0 1 0
[129] 3 3 3 0 3 1 0 3 2 0 1 0 0 2 1 0 2 3 0 2 2 1 2 3 3 2 2 1 1 0 1 3
[161] 0 2 1 0 1 3 2 0 1 3 1 1 1 1 0 1 0 2 1 3 3 2 3 2 3 3 1 2 2 2 2 0
[193] 2 1 1 1 3 1 2 1 3 0 3 3 2 3 3 3 0 1 1 1 2 0 3 3 2 0 3 1 0 1 1 3
[225] 1 0 2 0 0 2 1 0 0 0 3 0 0 1 1 1 0 1 2 0 3 3 3 1 2 3 2 3 0 2 1 1
```

> table(carros)

```
carros
 0   1   2   3
5043 5028 4934 4995
```

Repare que a soma das pessoas que tem nenhum carro, 1 carro, 2 carros e 3 carros são 20.000, igual ao valor atribuído a n, no início do projeto.

A7) Filhos

Usaremos a mesma estratégia da tabela carros

> pop.filhos<-rep(c(0,1,2),p)

> pop.filhos

```
[1] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
[28] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
[55] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
[82] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
[109] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
[136] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
[163] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
[190] 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2
```

.....

Vamos travar a inicialização do gerador de números aleatórios garantindo que serão os mesmos gerados.

> set.seed(1)

> filhos <- sample(pop.filhos, n)

> rm(pop.filhos)

> filhos

```
[1] 2 0 2 1 2 2 0 2 0 1 1 1 0 0 1 0 0 1 0 1 2 1 0 1 0 0 2
[28] 1 2 0 0 1 1 0 0 1 0 1 2 2 1 0 0 0 2 1 0 0 2 1 2 1 2 2
[55] 2 0 2 1 0 1 2 0 2 0 1 0 2 2 1 2 1 2 0 2 2 1 2 0 2 1 2
[82] 1 0 0 2 1 2 0 0 1 1 2 1 1 1 1 0 2 2 1 0 1 0 0 2 0 0 0
```

## Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

```
[109] 2 0 2 1 2 1 1 2 1 0 2 1 0 1 0 1 2 0 1 1 2 1 0 2 1 1 2
[136] 0 1 1 0 0 0 2 2 1 1 2 1 1 0 2 2 2 2 2 1 0 0 1 2 2 0 0
```

.....

```
> summary(filhos)
```

```
Min. 1st Qu. Median Mean 3rd Qu. Max.
```

```
0.000 0.000 1.000 1.017 2.000 2.000
```

```
> table(filhos)
```

```
filhos
 0 1 2
6617 6694 6689
```

### A8) Escolaridade

O objeto escolaridade tem uma escala ordinal, sem escolaridade ( analfabeto) até pós doutorado. Para ficar mais fácil atribuísse uma escala numérica para designar cada variável categórica. Usaremos o rep para popular a base de dados com números que variam de 0 a 6. Usaremos a função sample() para retirar a amostra dessa população criada. Finalmente usaremos a função factor() para associar cada nível uma labels.

```
> pop.escolaridade <- rep(c(0, 1, 2, 3, 4, 5, 6), p)
> set.seed(1234)
> escolaridade.temp <- sample(pop.escolaridade, n)
> escolaridade <- factor(escolaridade.temp,
+       levels = c(0, 1, 2, 3, 4, 5, 6),
+       labels = c("Analfabeto", "1º Grau", "2º Grau", "3º Grau",
+       "Mestrado", "Doutorado", "PósDoc"),
+       ordered = TRUE
+ )
```

```
> rm(pop.escolaridade, escolaridade.temp)
> str(escolaridade)
Ord.factor w/ 7 levels "Analfabeto"<"1º Grau"<...: 7 2 5 4 6 4 7 5 7 1 ...
> summary(escolaridade)
Analfabeto 1º Grau 2º Grau 3º Grau Mestrado
 2768    2869    2951    2851    2951
Doutorado  PósDoc
 2840    2770
> table(escolaridade)
```

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

escolaridade  
Analfabeto 1º Grau 2º Grau 3º Grau Mestrado  
2768 2869 2951 2851 2951  
Doutorado PósDoc  
2840 2770

## A8) Fumante

Quando temos apenas duas variáveis a serem distribuídas na amostra devemos utilizar outra técnica, pois há somente a variável binária fumante. No nosso caso vamos estabelecer um grupo de prováveis fumantes de 40% da amostra n. A distribuição binominal é dada pela função rbinom. Classificaremos fumante sim como 1, fumante não como 0

```
> set.seed(1234)
> fumante.n <- rbinom(n, 1, .40)
> fumante.f <- factor(fumante.n,
+                     levels = c(0, 1),
+                     labels = c("não", "sim"),
+                     ordered = TRUE)
> str(fumante.f)
Ord.factor w/ 2 levels "não"<"sim": 1 2 2 2 2 1 1 1 2 1 ...
> summary(fumante.f)
 não sim
11987 8013
> str(fumante.n)
int [1:20000] 0 1 1 1 1 0 0 0 1 0 ...
> mean(fumante.n)
[1] 0.40065
>
```

## A9) Sexo

Atribuímos a variável sexo F o valor 2 e sexo M valor 1.

```
> pop.sexo <- rep(c(1, 2), p)
> set.seed(1234)
> sexo.temp <- sample(pop.sexo, n)
> sexo <- factor(sexo.temp,
+               levels = c(1, 2),
+               labels = c("M", "F"),
+               ordered = FALSE)
> rm(pop.sexo, sexo.temp)
> str(sexo)
```

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

Factor w/ 2 levels "M","F": 2 2 2 2 1 2 1 2 2 2 ...

```
> summary(sexo)
```

M	F
---	---

9923	10077
------	-------

## A9) Profissão

Esta variável também está categorizada, associado uma variável numérica a cada uma delas.

```
> pop.profissao <- rep(0:2, p)
> set.seed(1234)
> profissao.temp <- sample(pop.profissao, n)
> profissao <- factor(profissao.temp,
+                     levels = c(0, 1, 2),
+                     labels = c("Humanas", "Exatas", "Biológicas"),
+                     ordered = FALSE
+ )
> rm(pop.profissao, profissao.temp)
> str(profissao)
Factor w/ 3 levels "Humanas","Exatas",...: 1 1 1 3 2 1 3 2 3 3 ...
```

```
> summary(profissao)
```

Humanas	Exatas	Biológicas
---------	--------	------------

6876	6583	6541
------	------	------

## A10) Criação do dataset

A partir de todas as variáveis criadas agora é possível criar o conjunto de dados usando o data.frame para combinar todas elas.

```
> df <- data.frame(id = 1:n,
+                  altura,
+                  peso,
+                  imc,
+                  sexo,
+                  escolaridade,
+                  profissao,
+                  fumante.f,
+                  fumante.n,
+                  salario,
+                  carros,
+                  filhos)
```

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

+ )

> str(df)

```
'data.frame': 20000 obs. of 12 variables:
 $ id      : int 1 2 3 4 5 6 7 8 9 10 ...
 $ altura  : num 1.38 1.53 1.61 1.27 1.54 1.55 1.44 1.45 1.44 1.41 ...
 $ peso    : num 50.2 56.1 59.3 45.6 56.7 ...
 $ imc     : num 26.3 24 22.9 28.3 23.9 ...
 $ sexo    : Factor w/ 2 levels "M","F": 2 2 2 2 1 2 1 2 2 2 ...
 $ escolaridade: Ord.factor w/ 7 levels "Analfabeto"<"1º Grau"<..: 7 2 5 4 6 4 7
 5 7 1 ...
 $ profissao : Factor w/ 3 levels "Humanas","Exatas",...: 1 1 1 3 2 1 3 2 3 3 ...
 $ fumante.f : Ord.factor w/ 2 levels "não"<"sim": 1 2 2 2 2 1 1 1 2 1 ...
 $ fumante.n : int 0 1 1 1 1 0 0 0 1 0 ...
 $ salario   : num 2517 3111 3434 2062 3172 ...
 $ carros    : num 3 1 3 1 2 1 2 1 3 3 ...
 $ filhos    : num 0 2 0 2 1 0 1 2 1 2 ...
```

Agora vamos salvar essa tabela.

```
> write.table(df, file = "projeto01.csv", sep = ",", col.names = TRUE, fileEncoding = "UTF-8")
```

Vai ficar em uma área do R por enquanto.

Obtenha a importação dessa tabela.

Paradinha conhecimento:

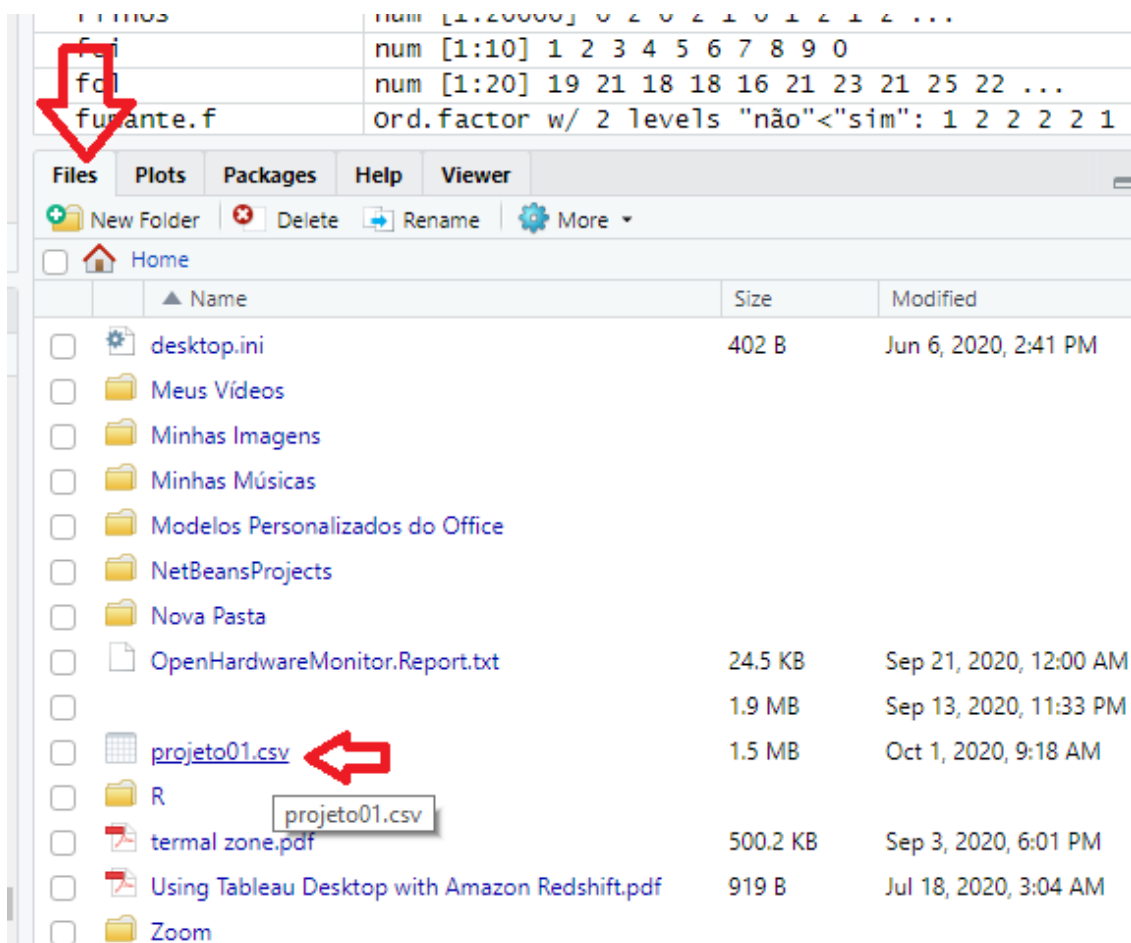
**Unicode** é um padrão que permite aos computadores representar e manipular, de forma consistente, texto de qualquer sistema de escrita existente.

**UTF-8** (8-bit Unicode Transformation Format- informática avançada) é um tipo de codificação binária (Unicode) de comprimento variável.

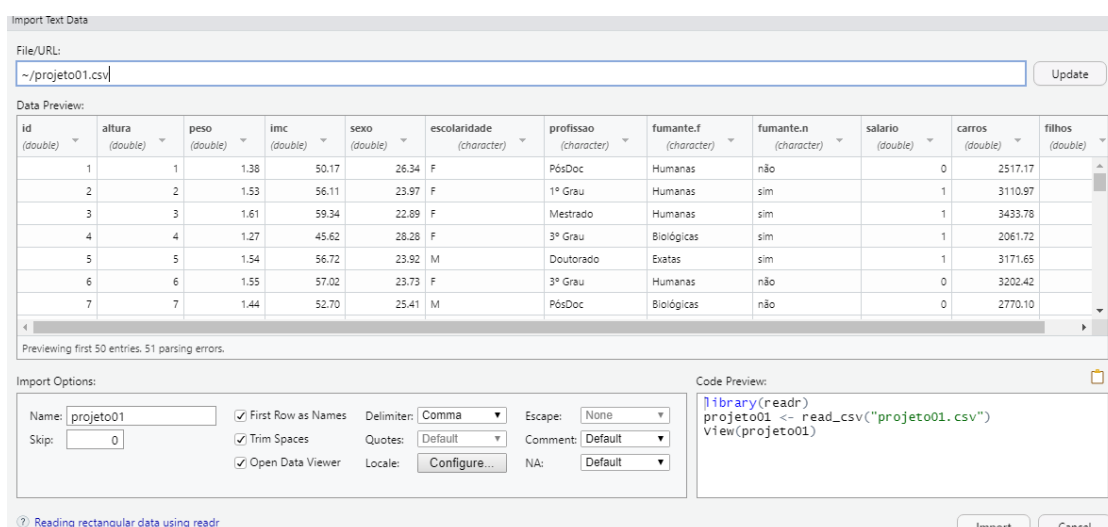
É necessário apenas um byte para codificar os 128 caracteres ASCII (Unicode U+0000 a U+007F). São necessários dois bytes para caracteres Latinos.

número Unicode	caractere	código UTF-8	hexadecimal
U+0021	!	00100001	0x21
U+0022	"	00100010	0x22
U+002D	-	00101101	0x2D
U+0039	9	00100111	0x39
U+0041	A	01000001	0x41
U+0042	B	01000010	0x42
U+0061	a	01100001	0x61
U+0062	b	01100010	0x62

# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC



Clique com o botão direito para importar este conjunto de dados.





# Aplicando modelagem de cálculo computacional com R aos data sets - Parte 2 - Aplicação em Case de IMC

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

procr.R dataset2 projeto01 script1.R\* script2.R\*

Filter

	id	altura	peso	imc	sexo	escolaridade	profissao	fumante.f	fumante.n	salario	carros
1	1	1	1.38	50.17	26.34	F	PósDoc	Humanas	não	0	25
2	2	2	1.53	56.11	23.97	F	1º Grau	Humanas	sim	1	31
3	3	3	1.61	59.34	22.89	F	Mestrado	Humanas	sim	1	34
4	4	4	1.27	45.62	28.28	F	3º Grau	Biológicas	sim	1	20
5	5	5	1.54	56.72	23.92	M	Doutorado	Exatas	sim	1	31
6	6	6	1.55	57.02	23.73	F	3º Grau	Humanas	não	0	32
7	7	7	1.44	52.70	25.41	M	PósDoc	Biológicas	não	0	27
8	8	8	1.45	52.81	25.12	F	Mestrado	Exatas	não	0	27

Showing 1 to 8 of 20,000 entries, 12 total columns

Pronto o seu data set está disponível para análises.

Atividade de fixação

Carregue a tabela gerada pela API Pythonics

Faça para os quatro objetos:

Contagem das frequências (será que vai funcionar?)

Gráfico de distribuição de frequências, analise o comportamento

Fazer o summary e analise o comportamento do resultado

Fazer o histograma e analise o comportamento dos dados.