

**UNIVERSIDADE DA REGIÃO DE JOINVILLE - UNIVILLE**

Bacharelado em Engenharia de Software (BES)

## **Estatística para computação**

**Professora Priscila Ferraz Franczak**

Engenheira Ambiental - UNIVILLE

Mestre em Ciência e Engenharia de Materiais - UDESC

Doutoranda em Ciência e Engenharia de Materiais - UDESC

[priscila.franczak@gmail.com](mailto:priscila.franczak@gmail.com)


# Plano de Aula

1. Gráficos em barra
2. *Boxplot*
3. Gráficos em setores
4. Adicionando algo em um gráfico existente
5. Interagindo com a janela gráfica
6. Pacotes gráficos adicionais
7. Exercícios



# 1. Gráficos em barra

- Composto por duas linhas ou eixos, um vertical e outro horizontal.
- No eixo vertical são construídas as barras que representam a variação de um fenômeno ou de um processo de acordo com sua intensidade.

- 
- Essa intensidade é indicada pela altura da barra.
  - No eixo horizontal especifica-se as categorias da variável.

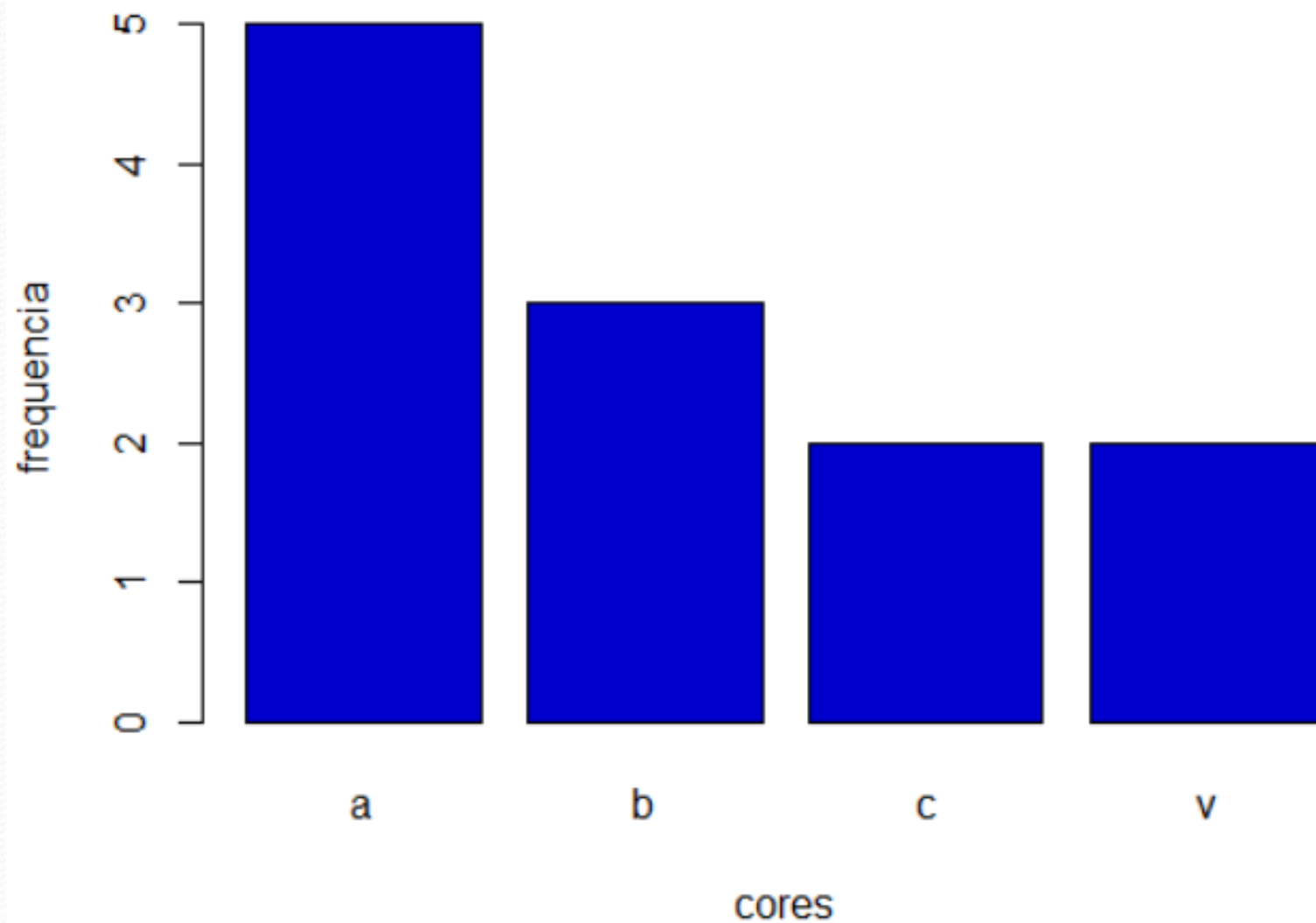
## Exemplo:

Dados se referem as categorias de cor: azul, branco, cinza e verde.

```
barplot(table(c("a", "a", "a", "a", "a",  
                "b", "b", "b", "c", "c", "v", "v")),  
        main = "Gráfico de cores",  
        xlab = "cores",  
        ylab = "frequencia",  
        bty="l",  
        col = "blue3"  
        )
```



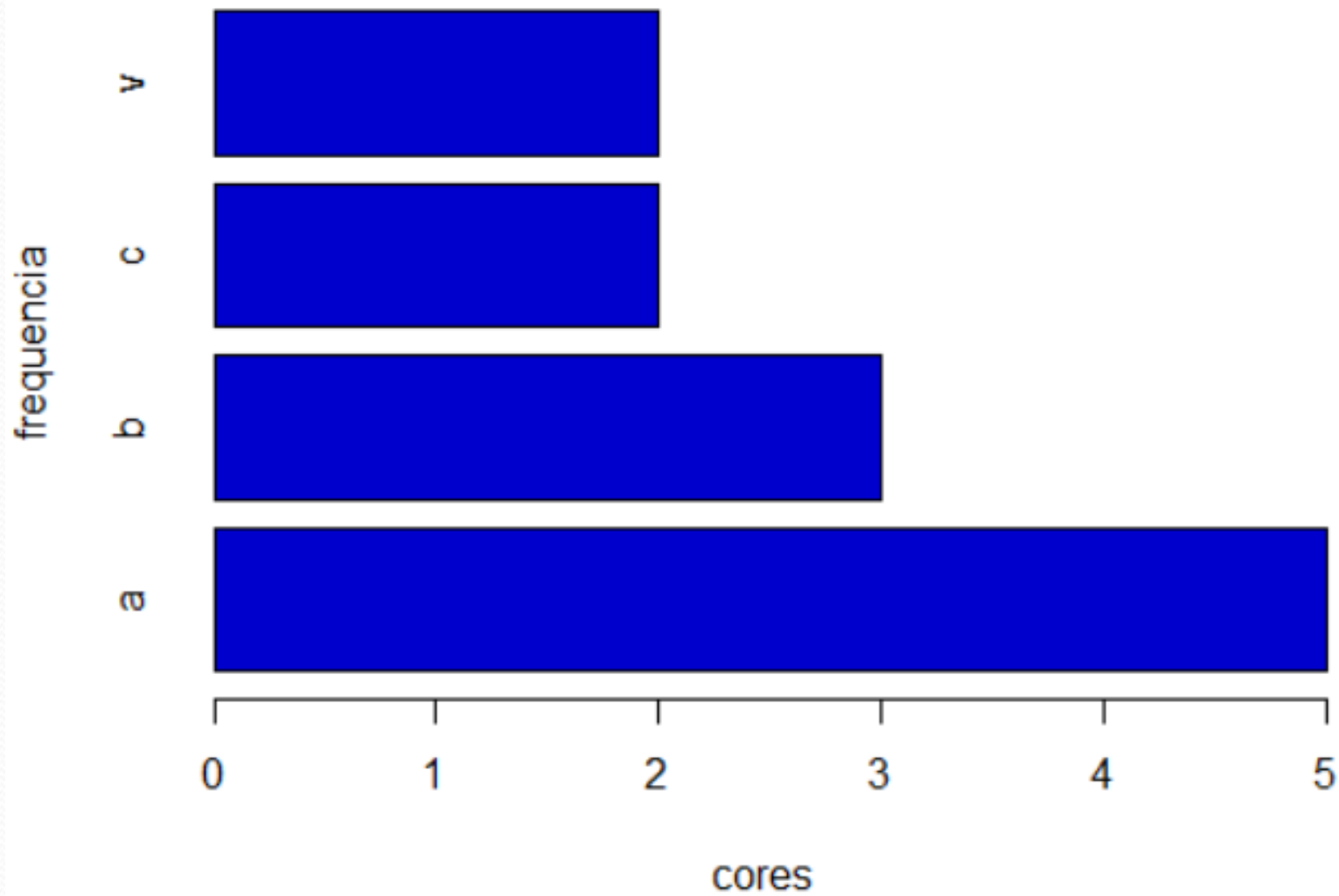
**Gráfico de cores**




- A orientação pode ser trocada:

```
barplot(table(c("a", "a", "a", "a", "a",  
                "b", "b", "b", "c", "c", "v", "v")),  
        main = "Gráfico de cores",  
        xlab = "cores",  
        ylab = "frequencia",  
        bty="l",  
        col = "blue3",  
horiz = T #disposição horizontal das categorias  
        )
```

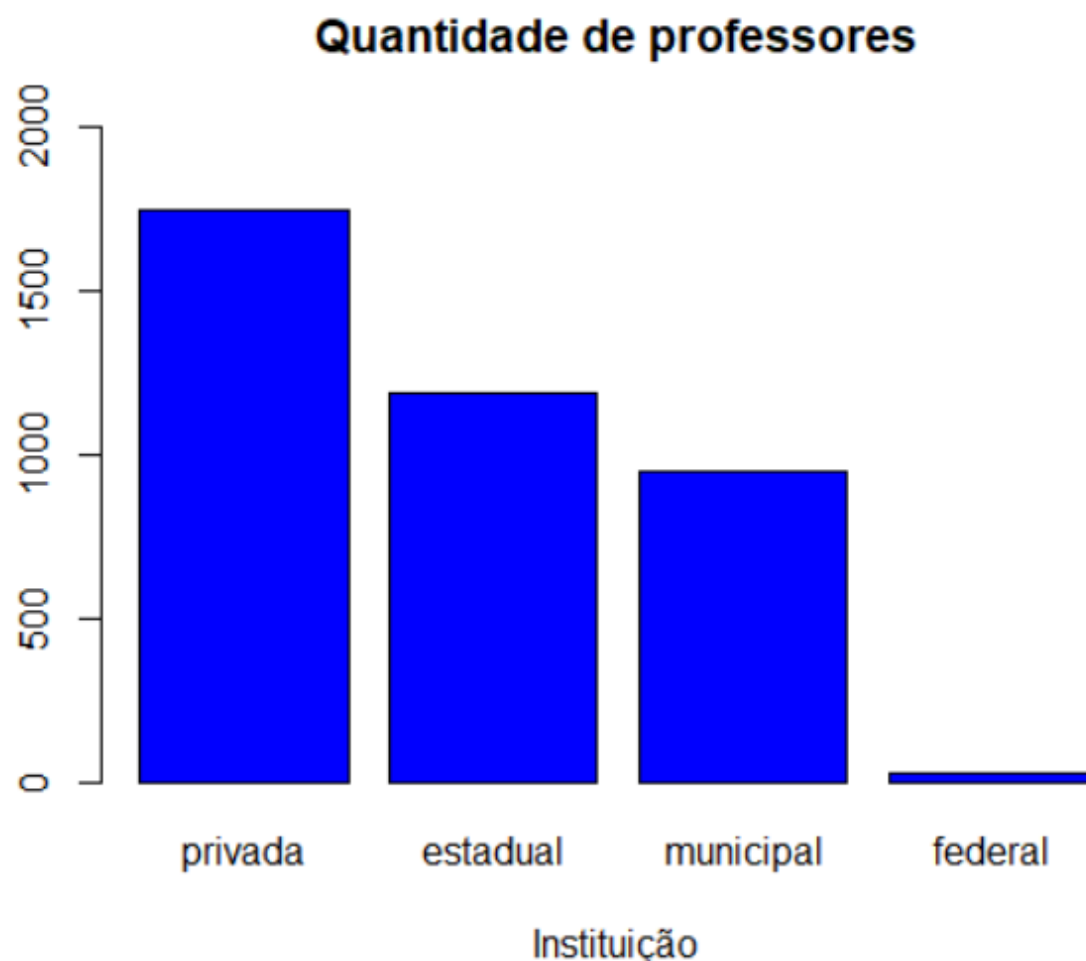
## Gráfico de cores





- 
- O primeiro passo na construção do gráfico é ter os dados armazenados em objeto apropriado.
  - No gráfico em barras é necessário que os dados estejam armazenados em um vetor ou matriz.

```
prof<-c(1751,1186,947,29)
escola<-c("privada","estadual","municipal","federal")
barplot(prof,names.arg = escola,
        main = "Quantidade de professores",
        xlab = "Instituição",
        ylim = c(0,2000),
        col = "blue")
```

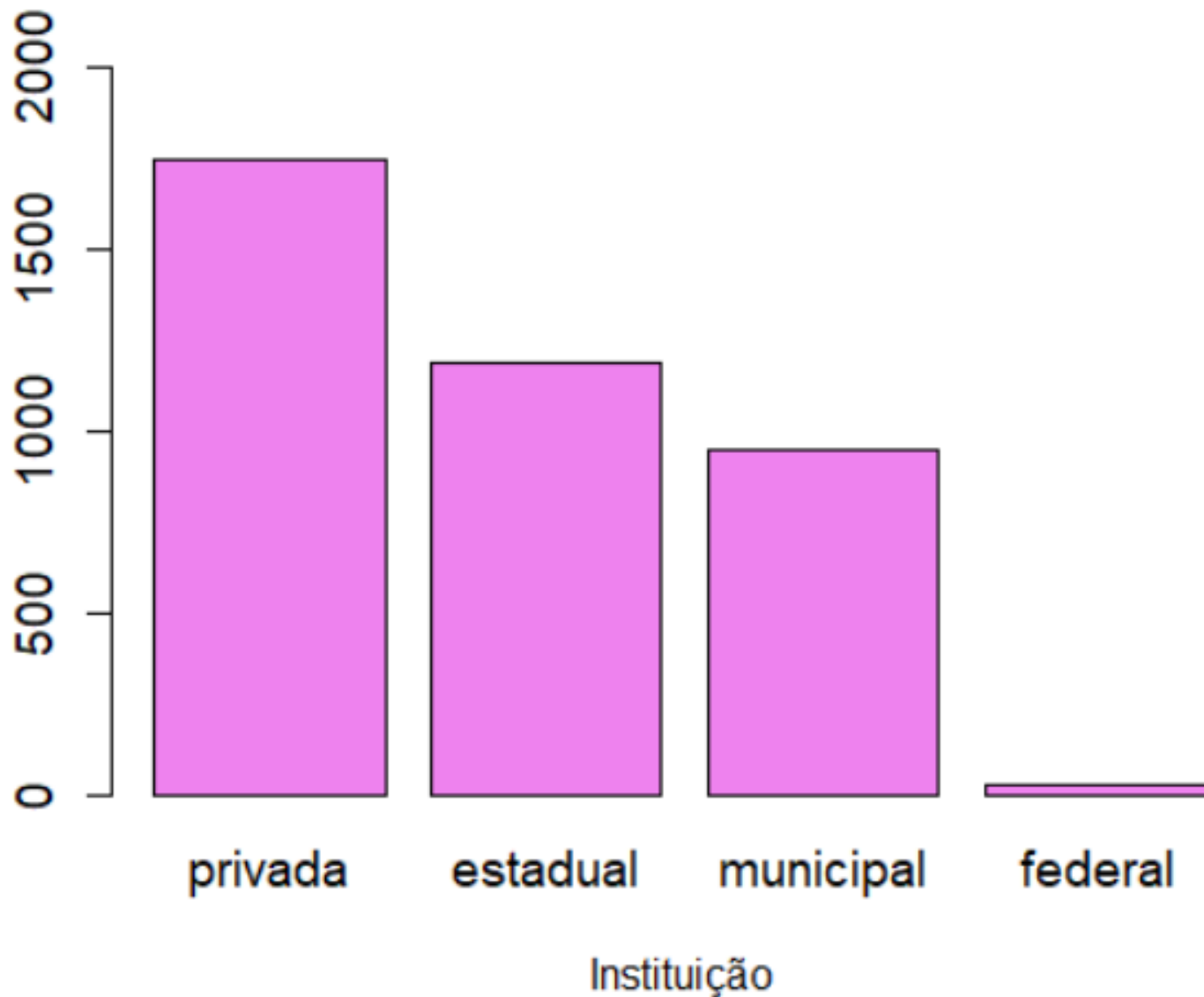


- É possível também nomear as posições através do comando `names( )` e aumentar a fonte do título e eixos:

```
prof<-c(1751,1186,947,29)
names(prof)<-c("privada","estadual","municipal","federal")
prof
privada  estadual  municipal  federal
   1751     1186     947        29
barplot(prof,
        main = "Quantidade de professores",
        cex.axis = 1.2,
        cex.names = 1.2,
        xlab = "Instituição",
        ylim = c(0,2000),
        col = "violet")
```



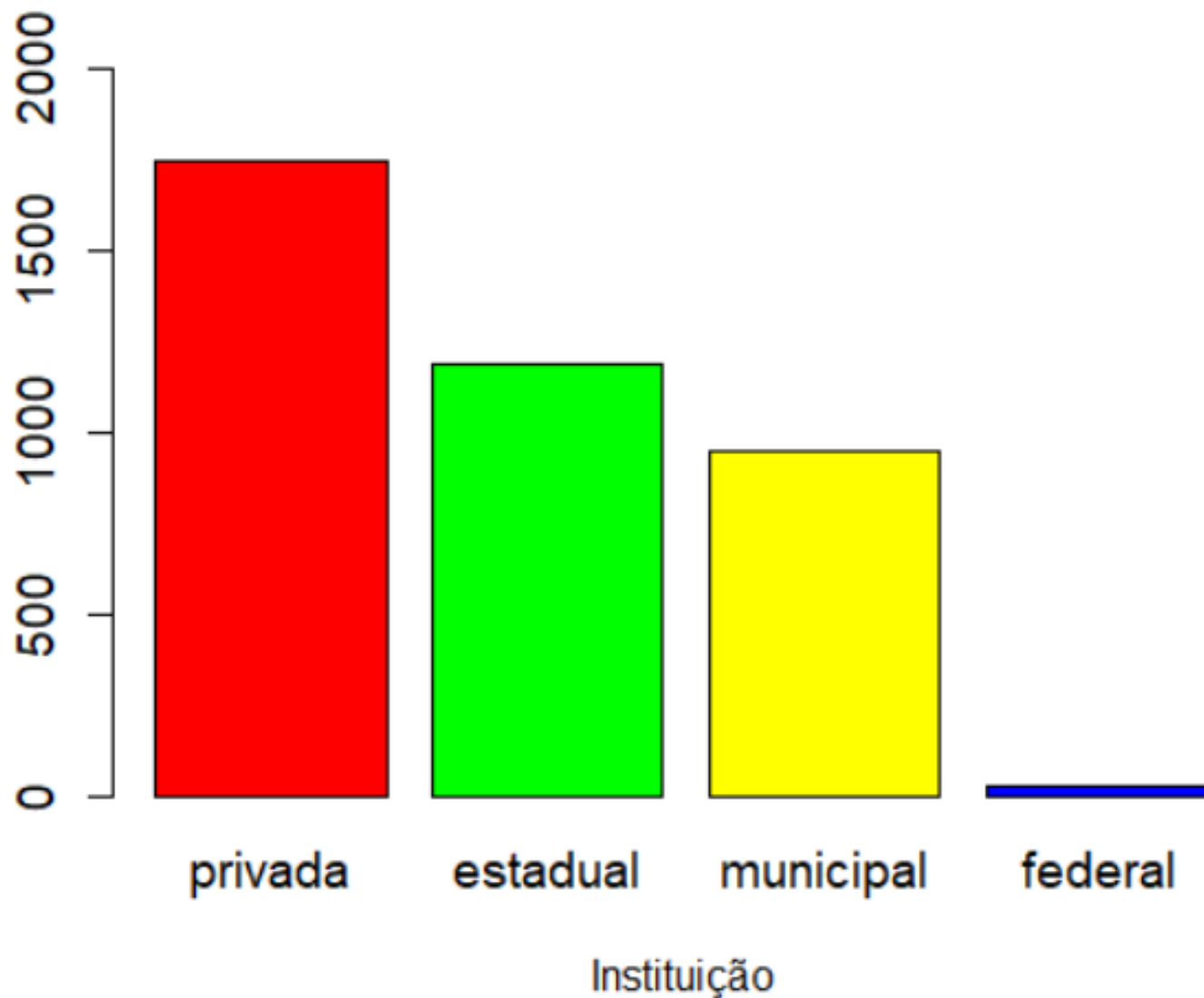
## Quantidade de professores



- Podemos também preencher cada barra com cor diferente:

```
prof<-c(1751,1186,947,29)
names(prof)<-c("privada","estadual","municipal","federal")
prof
barplot(prof,
        main = "Quantidade de professores",
        cex.axis = 1.2,
        cex.names = 1.2,
        xlab = "Instituição",
        ylim = c(0,2000),
        col = c("red","green","yellow","blue")
        )
```

## Quantidade de professores





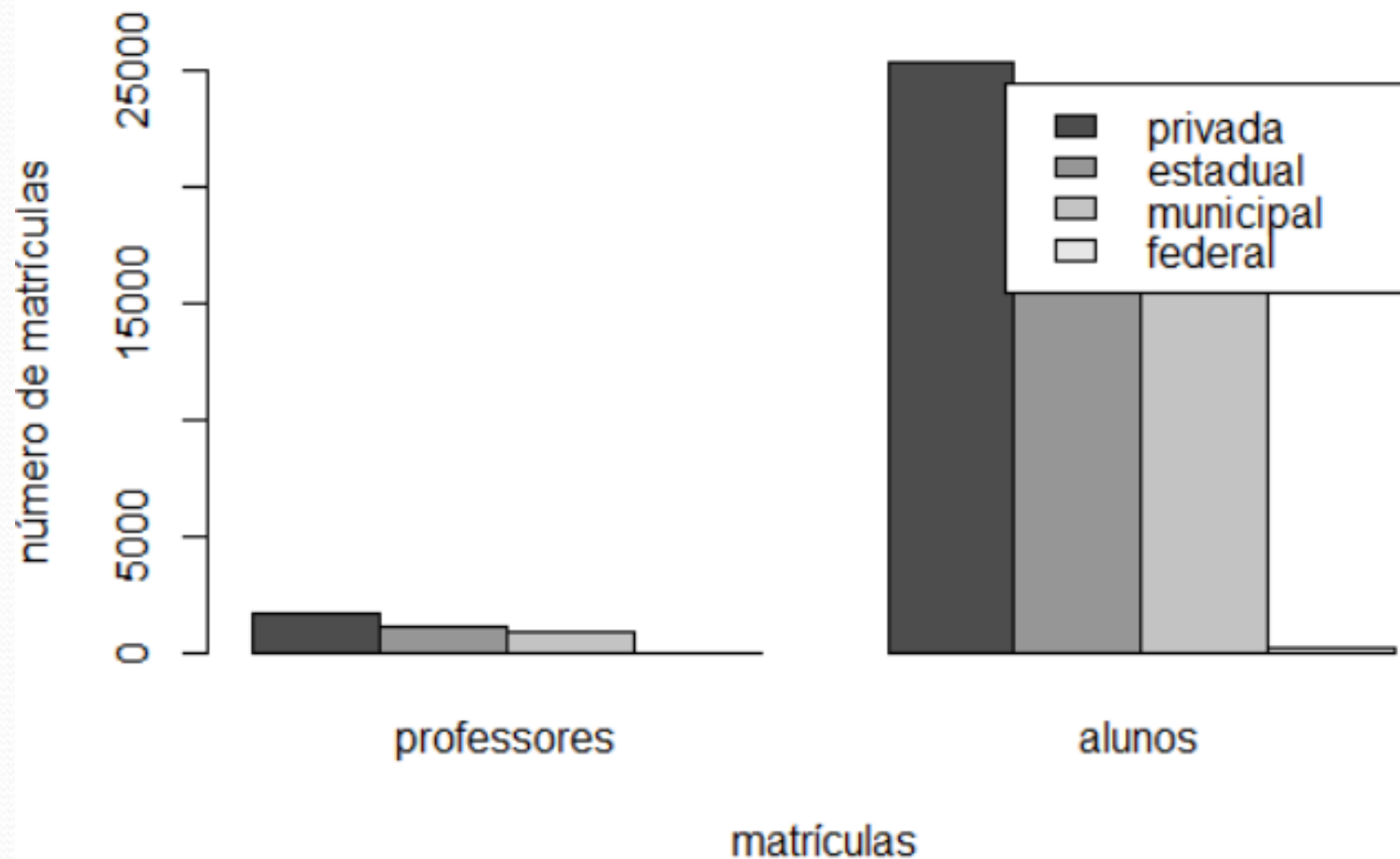
- Podemos criar o gráfico de barras de duas variáveis, um ao lado do outro, na mesma janela gráfica.
- Precisamos armazenar os dados em um objeto do tipo matriz:

```
> alunosprof<-matrix(c(1751,1186,947,29,25280,  
+                       21328,18432,280),nrow = 4,ncol = 2)  
> alunosprof  
      [,1] [,2]  
[1,] 1751 25280  
[2,] 1186 21328  
[3,]  947 18432  
[4,]   29   280
```

- Para colocarmos nome nas linhas e colunas, usamos o comando `dimnames( )`:

```
> dimnames(alunosprof)<-list(c("privada","estadual",  
+                             "municipal","federal"),  
+                             c("professores","alunos"))  
> barplot(alunosprof,  
+         beside=TRUE,  
+         main = "Distribuição de matrícula de alunos e professores",  
+         ylab="número de matrículas",  
+         xlab="matrículas",  
+         legend.text = rownames(alunosprof))
```

## Distribuição de matrícula de alunos e professores

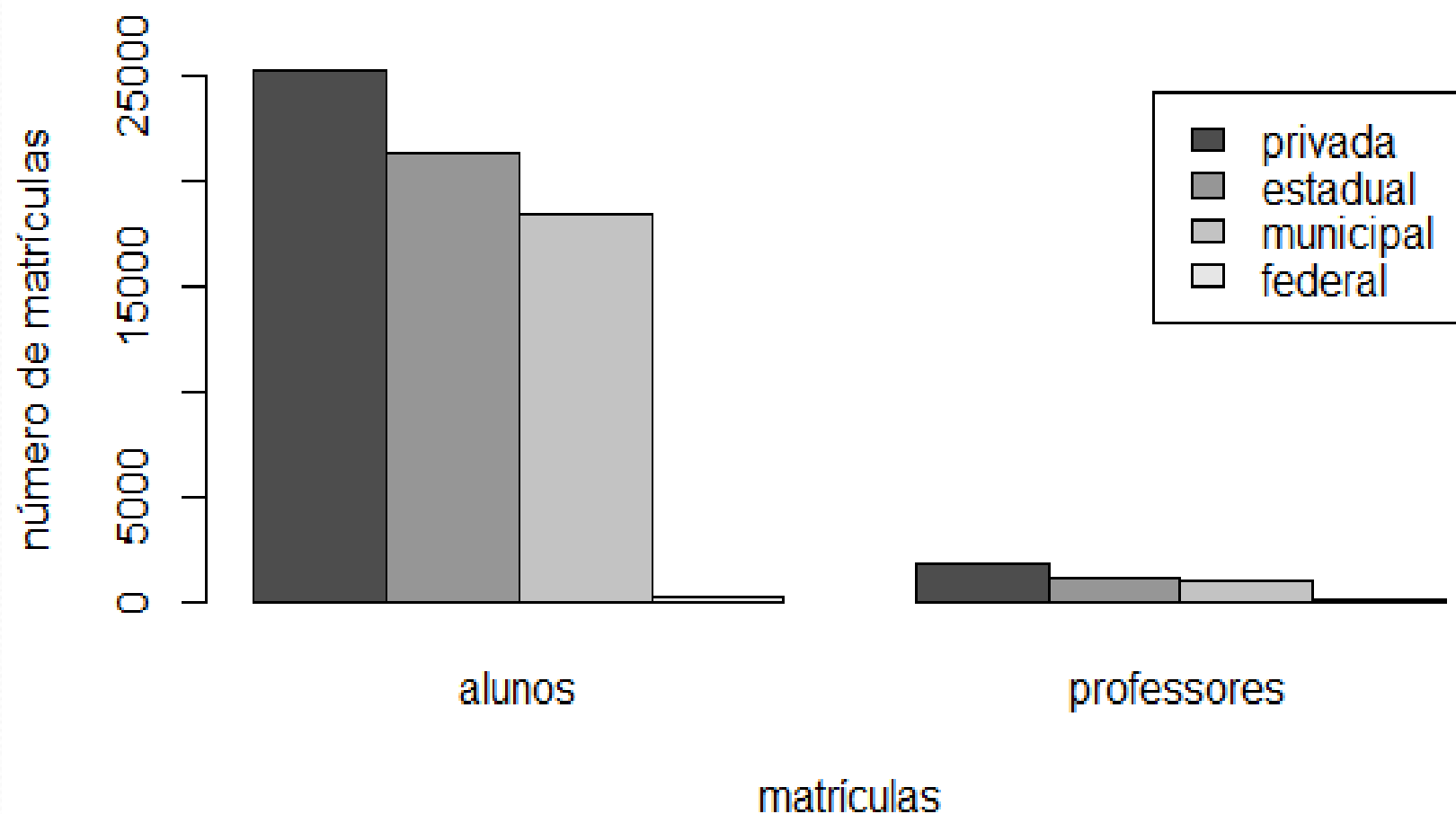




- Podemos trocar as colunas de lugar:

```
barplot(alunosprof[,2:1],  
        beside=TRUE,  
        main = "Distribuição de matrícula de alunos e professores",  
        ylab="número de matrículas",  
        xlab="matrículas",  
        legend.text = rownames(alunosprof))
```

## Distribuição de matrícula de alunos e professores



- Podemos trocar as colunas de lugar:

```
barplot(alunosprof[,2:1],  
        beside=TRUE,  
        main = "Distribuição de matrícula de alunos e professores",  
        ylab="número de matrículas",  
        xlab="matrículas",  
        legend.text = rownames(alunosprof))
```



## 2. *Boxplot*

- Este gráfico, também conhecido como **diagrama em caixa**, informa sobre a distribuição dos dados.
- **Somente se aplica a variáveis quantitativas**, informando o menor valor (pequena linha horizontal inferior) e valor máximo (pequena linha horizontal superior).

Valor da variável quantitativa

50% dos valores  
ocorrem dentro  
desta faixa

25% dos dados  
ocorrem, no máximo,  
até o valor equivalente  
ao percentil 25 (p25)

← Valor máximo

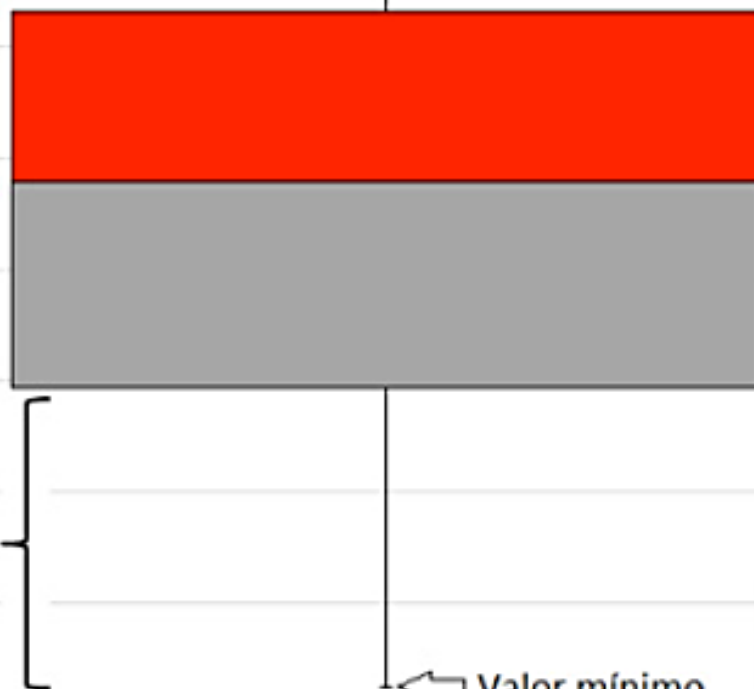
← Percentil 75 (p75)


← Mediana

← Percentil 25 (p25)

← Valor mínimo

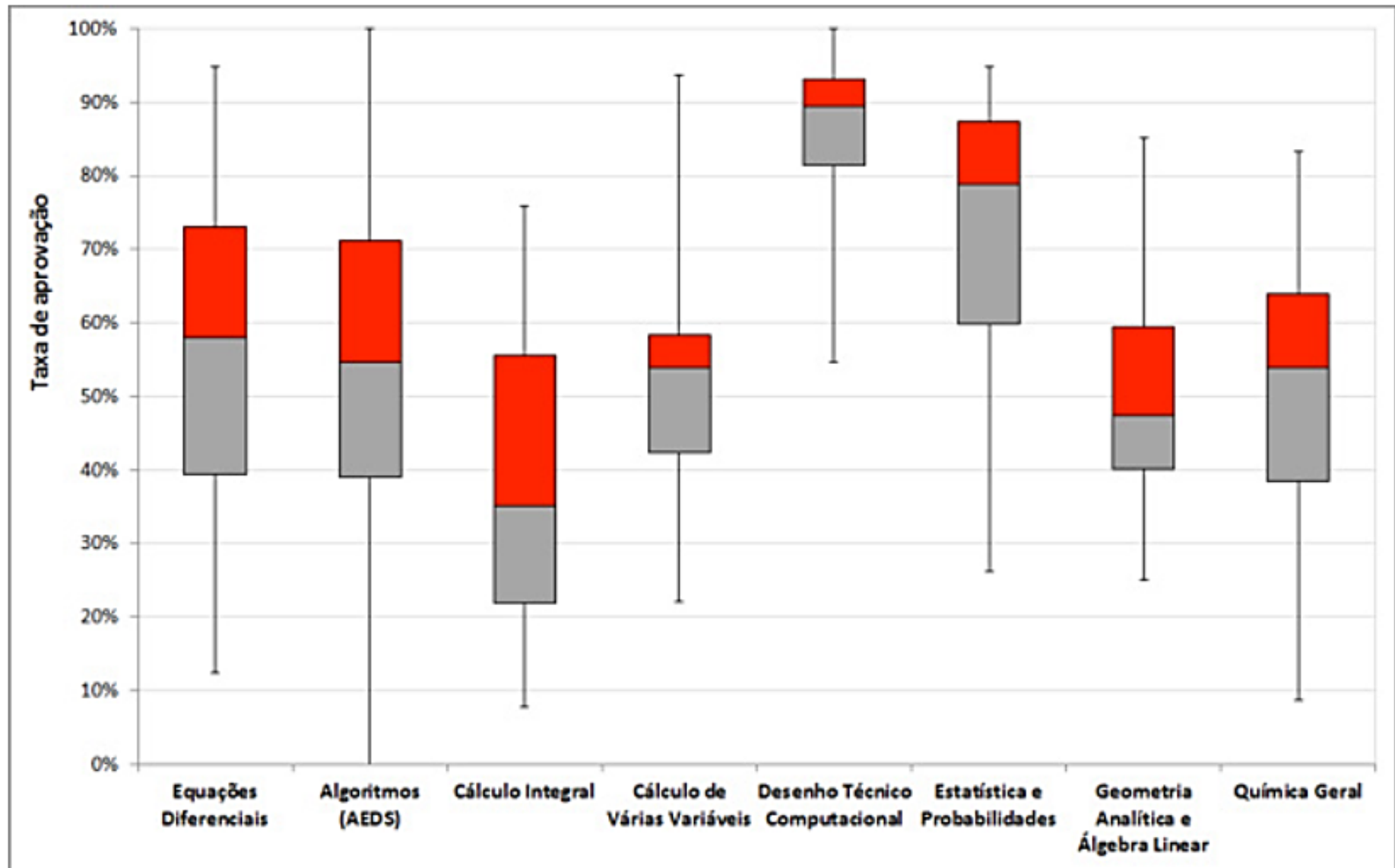
75% dos dados  
ocorrem, no máximo,  
até o valor equivalente  
ao percentil 75 (p75)



- 
- Assim como os histogramas, o *boxplot* nos informa sobre a maneira de distribuição dos dados, tendo a vantagem de permitir a visualização de grupos de dados.



# Resumo comparativo da taxa de aprovação de oito disciplinas de ciclo básico de cursos de Engenharia.





- Exemplo: conjunto de dados *InsectSprays* contido nos pacotes básicos de instalação do R.
- Contém dados de um experimento onde foram contabilizados o número de insetos sobreviventes quando da aplicação de seis diferentes inseticidas, nomeados de A a F.

- Esse conjunto possui 72 observações estruturadas em um `data.frame`, onde **count** contém o número de insetos sobreviventes após a aplicação do inseticida, que está rotulado na segunda coluna como **spray**.

```
> InsectSprays
```

	count	spray
1	10	A
2	7	A
3	20	A
4	14	A
5	14	A
6	12	A
7	10	A
8	23	A
9	17	A
10	20	A
11	14	A

⋮



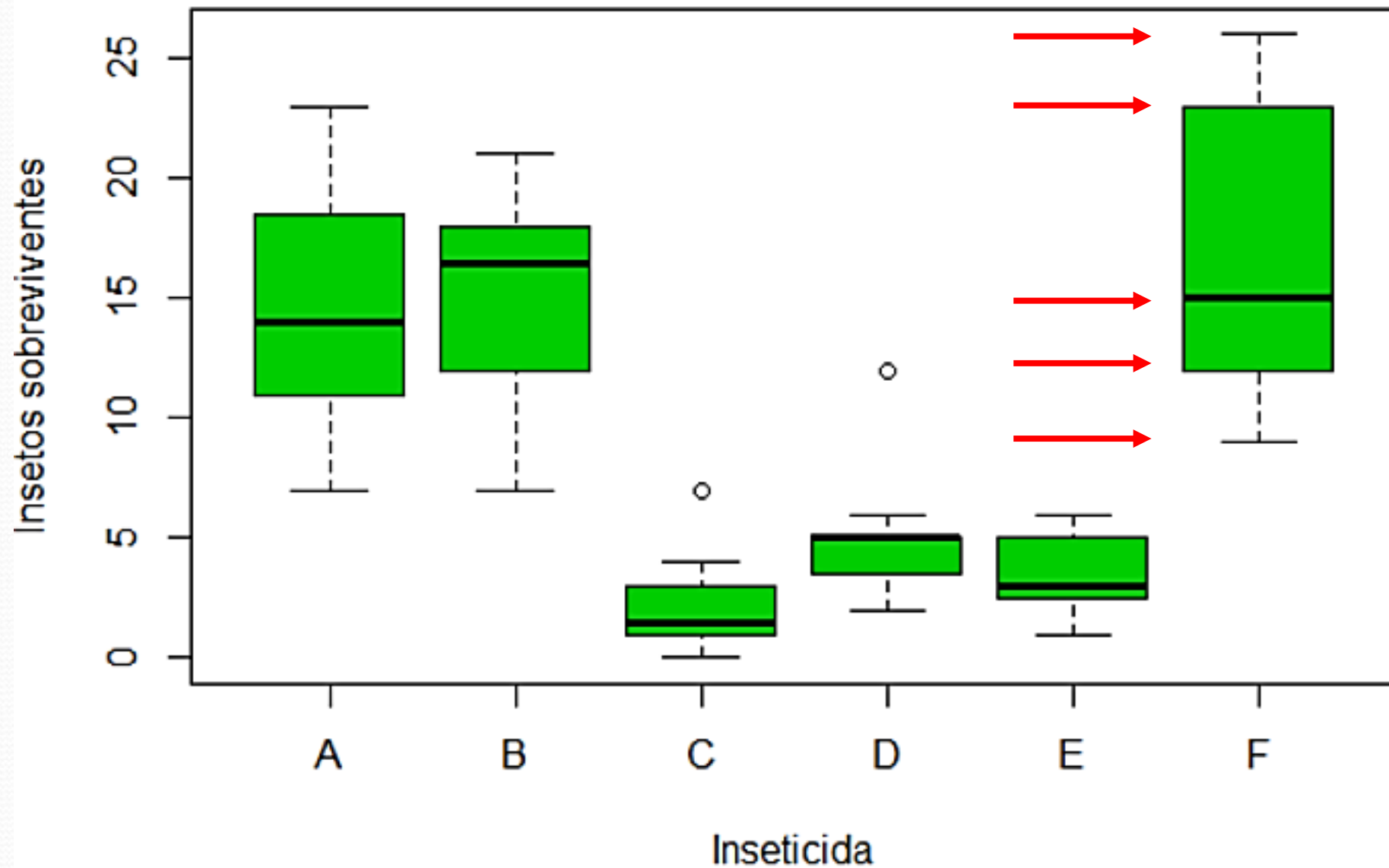
- A hipótese é que o número de insetos sobreviventes é função do tipo de inseticida aplicado.
- Ou seja, quanto mais eficiente o inseticida, menos insetos sobreviveriam.
- No R podemos expressar “count é uma função de spray” com a fórmula:  
count~spray

InsectSprays

```
boxplot(count~spray,  
        data=InsectSprays,  
        main="Boxplot",  
        xlab="Inseticida",  
        ylab="Insetos sobreviventes",  
        col=3)
```



## Boxplot



### 3. Gráficos em setores

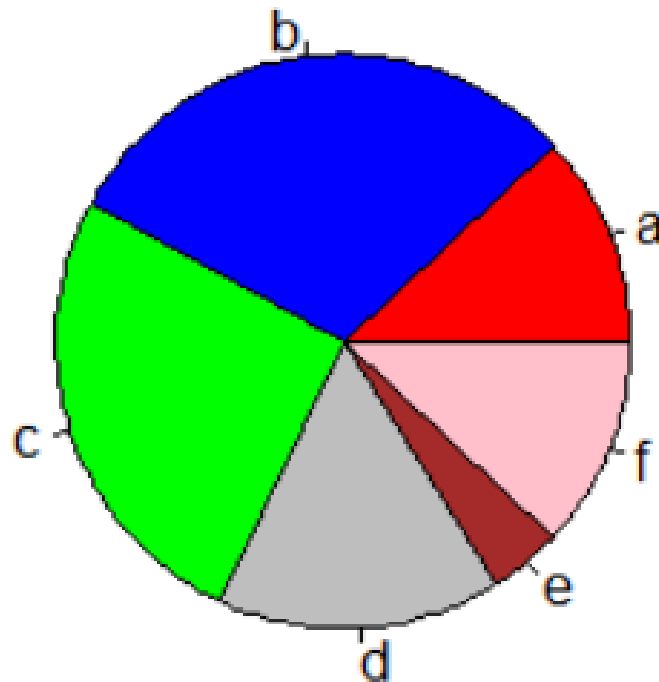
- Também chamado de gráfico de pizza.
- Exibem dados como proporção de um todo, permitindo fazer comparações entre grupos.
- Sintaxe:

`pie(dados,opções)`

- Exemplo:

```
a<-c(0.12,0.3,0.26,0.16,0.04,0.12)
names(a)<-c("a","b","c","d","e","f")
pie(a, col=c("red","blue","green","gray","brown","pink"),
    main = "Gráfico de setores")
```

**Gráfico de setores**





## 4. Adicionando algo em um gráfico existente

- Os comandos gráficos de baixo nível podem ser usados para adicionar informações a um gráfico existente.
- Os mais usados são o `points( )`, que adiciona pontos a um gráfico já elaborado e o `lines( )`, que adiciona linhas.



```
x<-c(0:20)
```

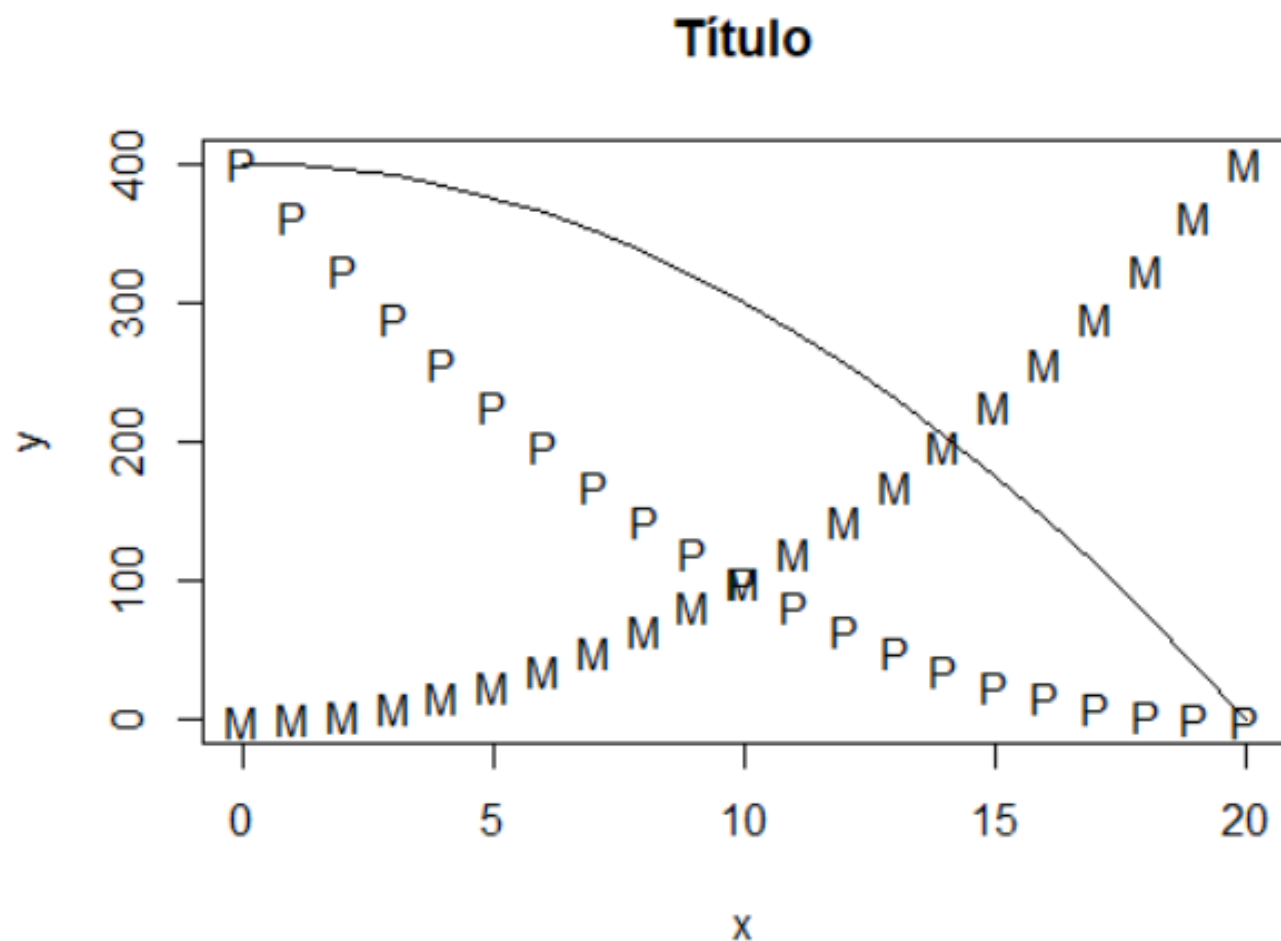
```
y=x^2
```

```
plot(x,y,pch="M") #plotando x e y
```

```
points(rev(x),y, pch="P") #adicionando pontos
```

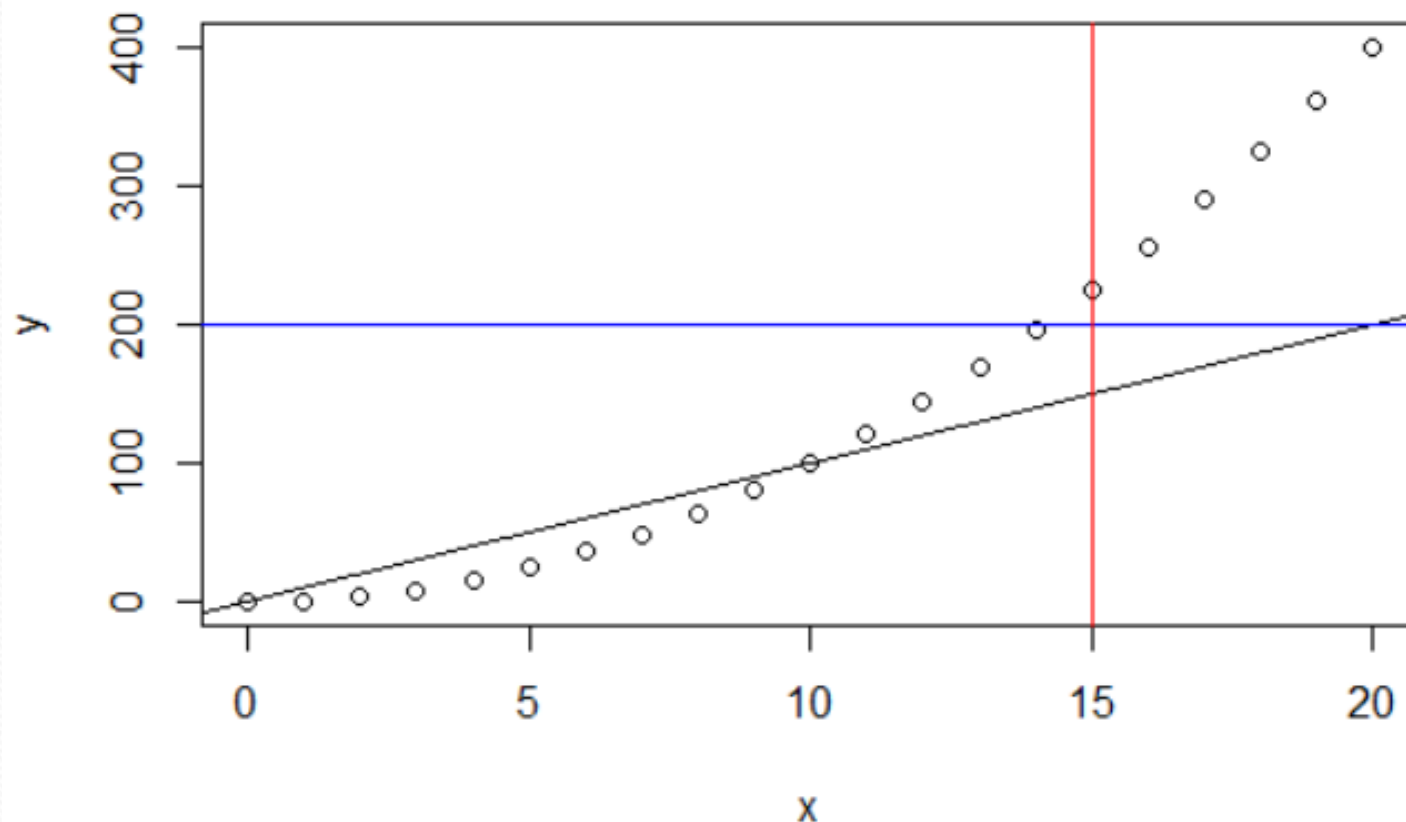
```
lines(x,400-y) #adicionando linhas
```

```
title("Título") #adicionando um título
```



- Outro comando útil para desenhar retas em gráficos já existentes é o comando `abline( )`.
- Ele desenha retas tanto com base em um intercepto e um coeficiente de inclinação quanto com base em valores verticais e horizontais fixos.

```
x<-c(0:20)
y=x^2
plot(x,y)
abline(0,10) #reta passando pelo 0 e inclinação 10
abline(h=200, col=4) #reta horizontal em y=200 azul
abline(v=15, col=2) #reta vertical em x=15 vermelha
```

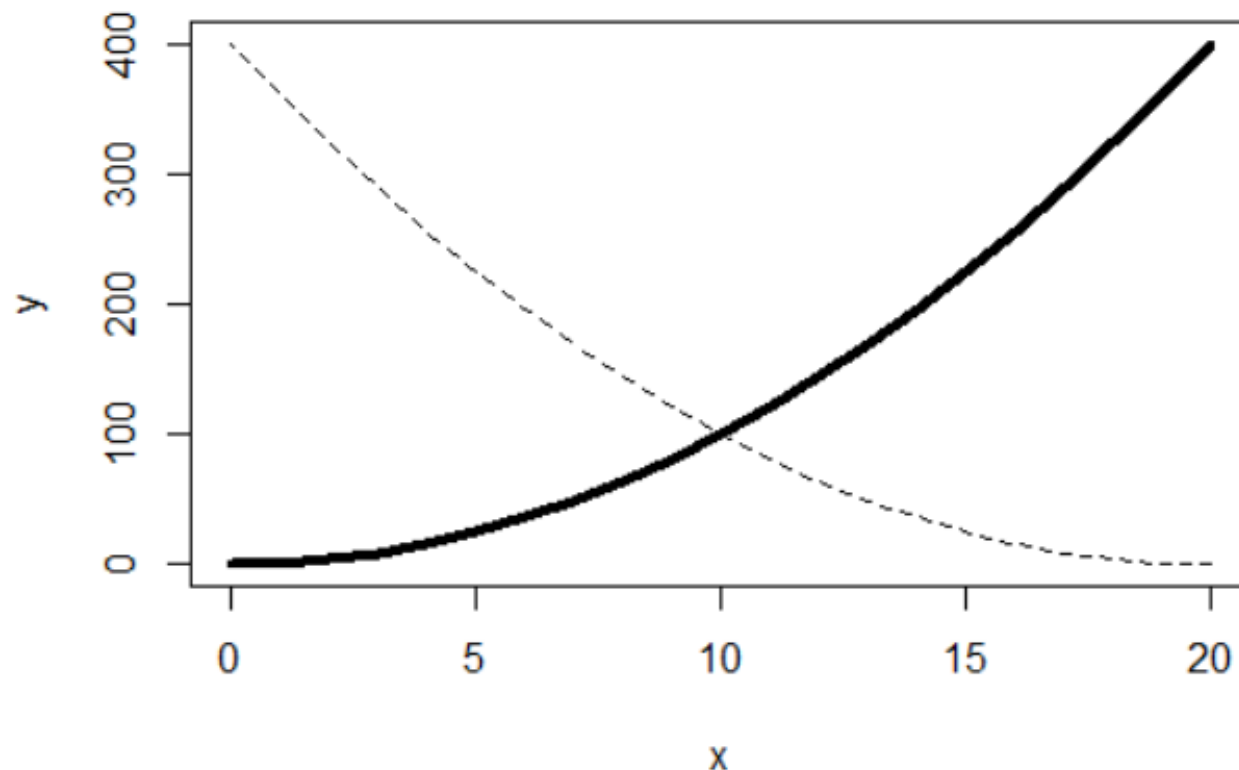


# Mudando linhas

- A largura das linhas pode ser mudada com o argumento **lwd** (quanto maior o número, mais grossa a linha).
- O estilo da linha pode ser modificado com o argumento **lty**.



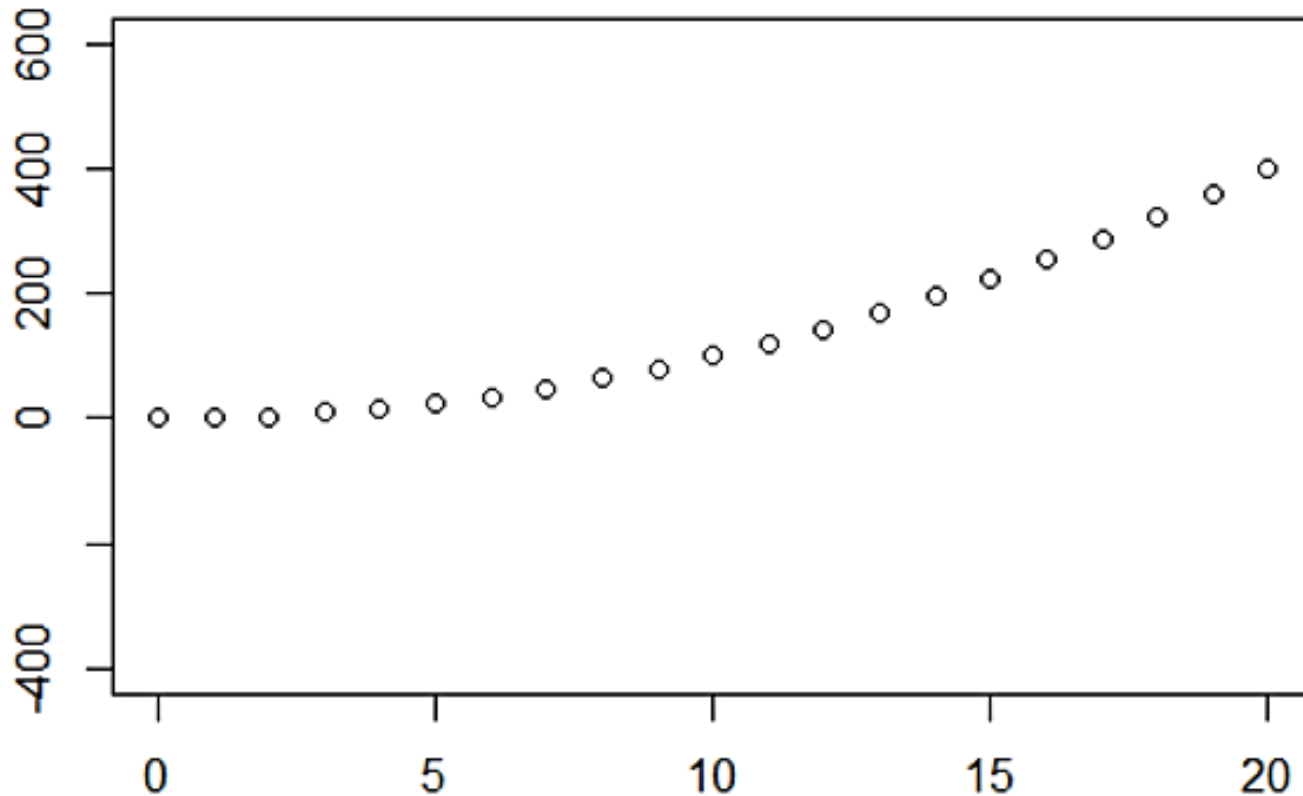
```
x<-c(0:20)
y=x^2
plot(x,y,type="n")
lines(x,y,lwd=4) #linha grossa
lines(rev(x),y,lty=2) #linha tracejada
```



## Definindo manualmente o intervalo dos eixos

- O R define os intervalos dos eixos com base nos valores a serem plotados.
- Porém é possível definir manualmente esses intervalos usando os comandos `xlim` e `ylim`.

```
plot(x,y,  
      ylim = c(-400,600))
```

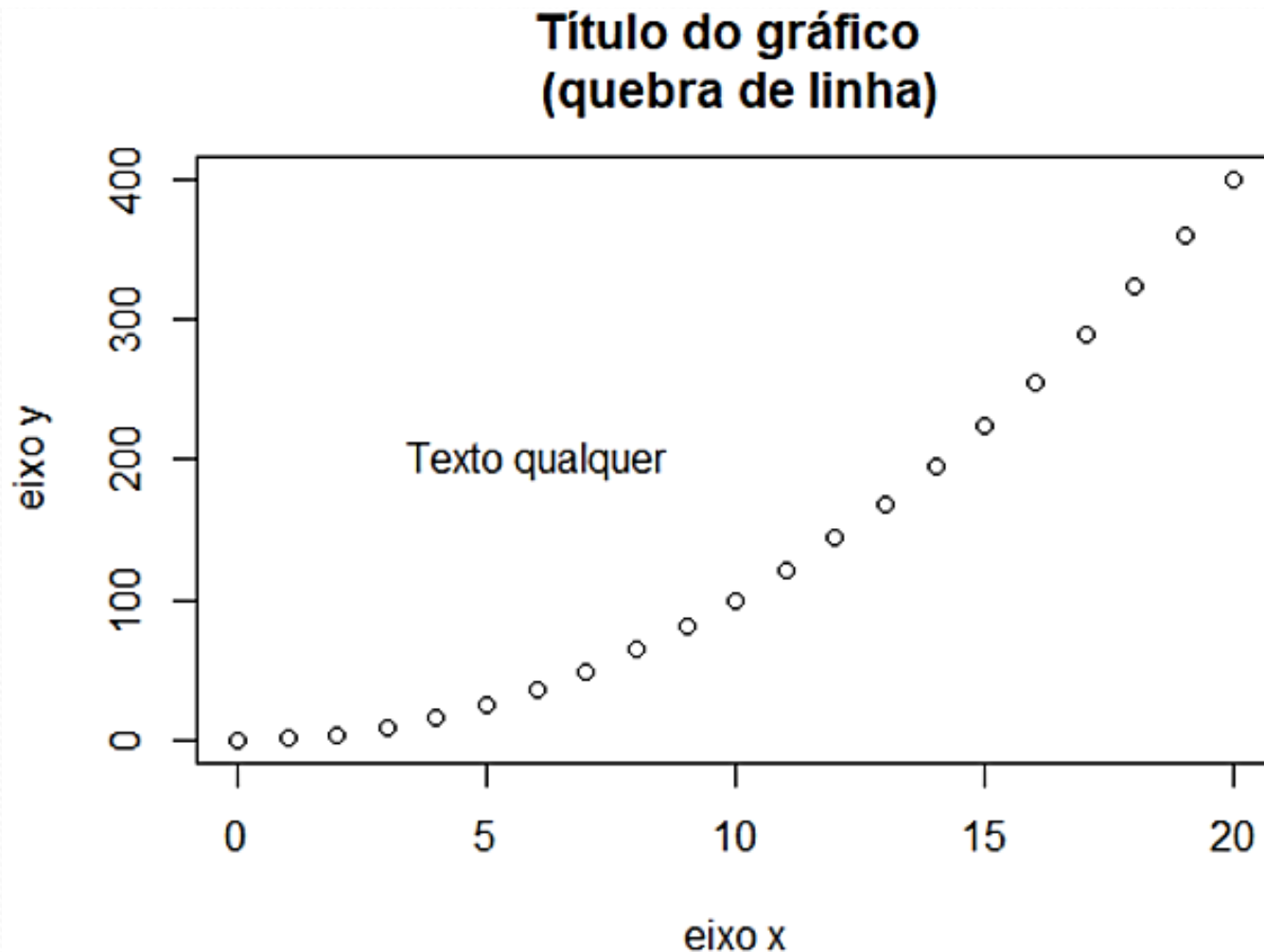




# Adicionando textos no gráfico

- Podemos adicionar textos em qualquer lugar do gráfico com o comando `text( )`, dando as coordenadas para inserção do texto.

```
plot(x,y,xlab="eixo x", ylab="eixo y")  
title("Título do gráfico \n (quebra de linha)")  
text(6,200,"Texto qualquer")
```



# 5. Interagindo com a janela gráfica

## Identificadores nos gráficos

- Podemos identificar um ponto ou conjunto de pontos em um gráfico gerado.
- Essa identificação pode ocorrer de maneira bem interativa quando usamos o comando `identify( )`.
- Esse comando se enquadra na terceira categoria de comandos gráficos: **os comandos interativos**.



## Exemplo:

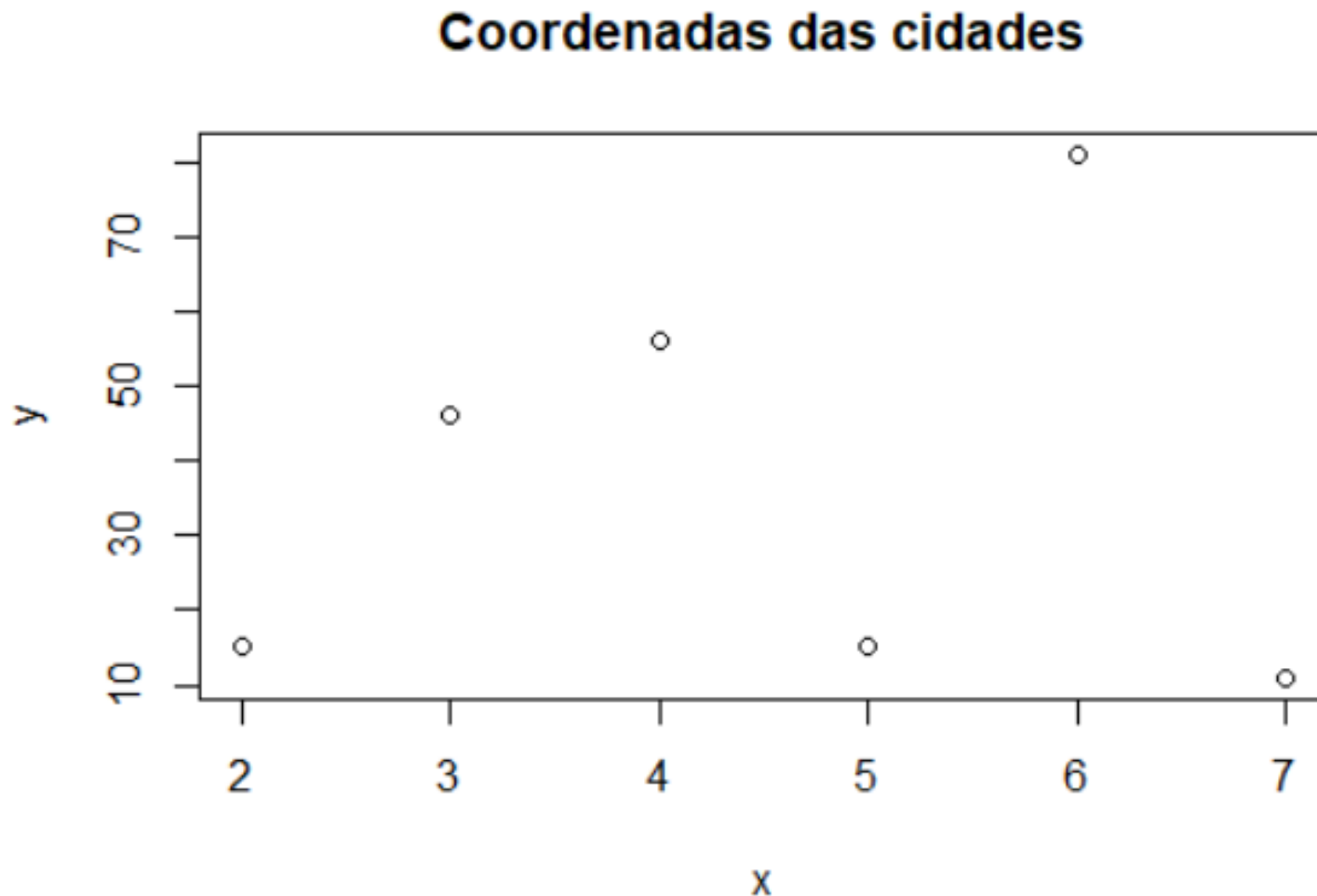
- Suponhamos que haja um conjunto de cidades (de A a F) e suas coordenadas planas (x e y):

```
> x<-c(2,3,4,5,6,7)           #coordenadas x
> y<-c(15,46,56,15,81,11)      #coordenadas y
> nomes<-LETTERS[1:6]          #nomes das cidades
> cidades<-data.frame(x,y,row.names = nomes)
> cidades
```

	x	y
A	2	15
B	3	46
C	4	56
D	5	15
E	6	81
F	7	11

- Podemos plotar as coordenadas das cidades:

```
plot(cidades,  
     main="Coordenadas das cidades")
```



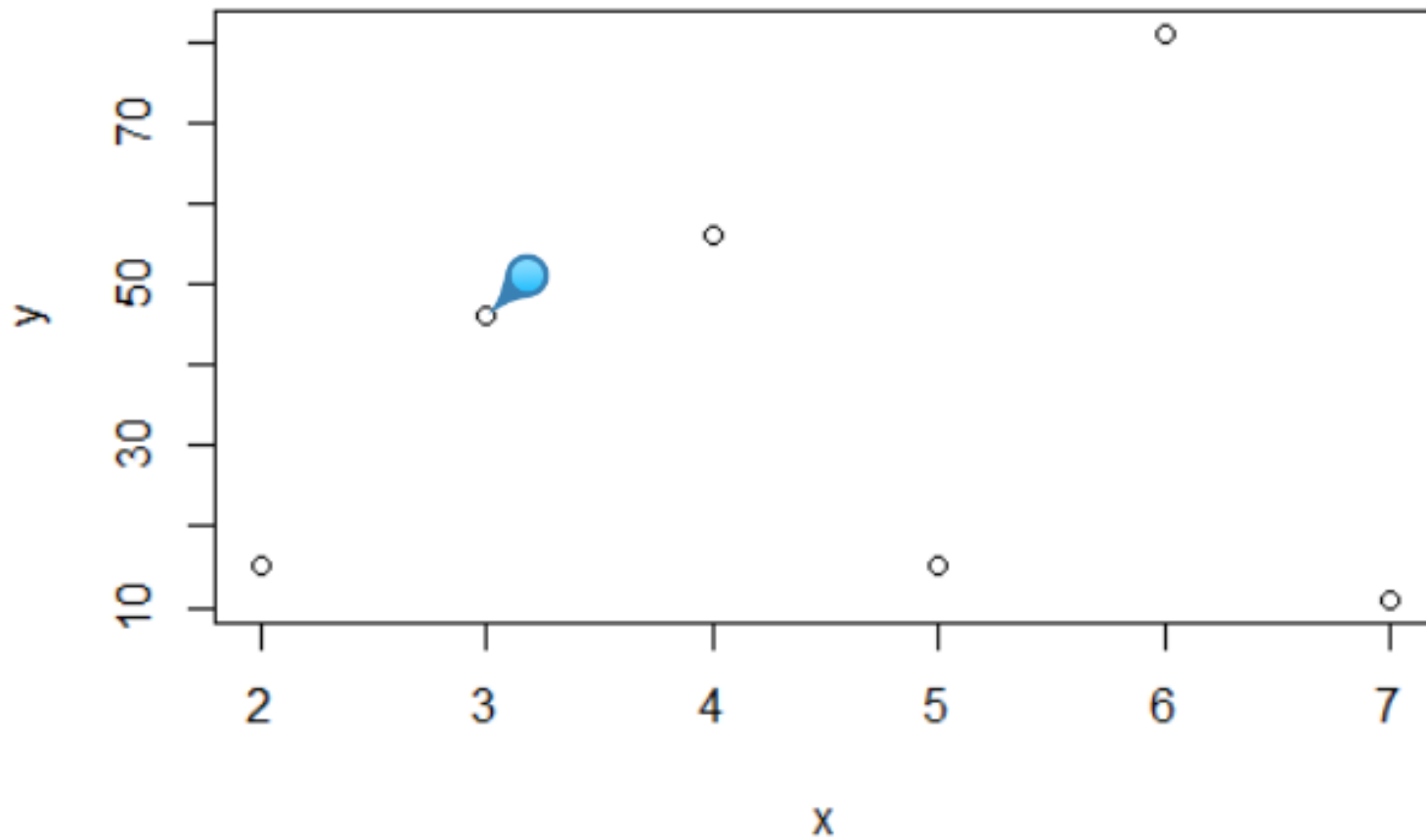
## Podemos identificar os nomes das cidades:

- Após dar o comando `identify( )`, quando passamos o mouse sobre o gráfico, **ele ganha forma de cruz**, e, ao clicar no ponto que se deseja identificar, sua descrição é exibida.
- Se quisermos identificar três cidades:

```
identify(x,y,nomes,n=3)
```



## Coordenadas das cidades



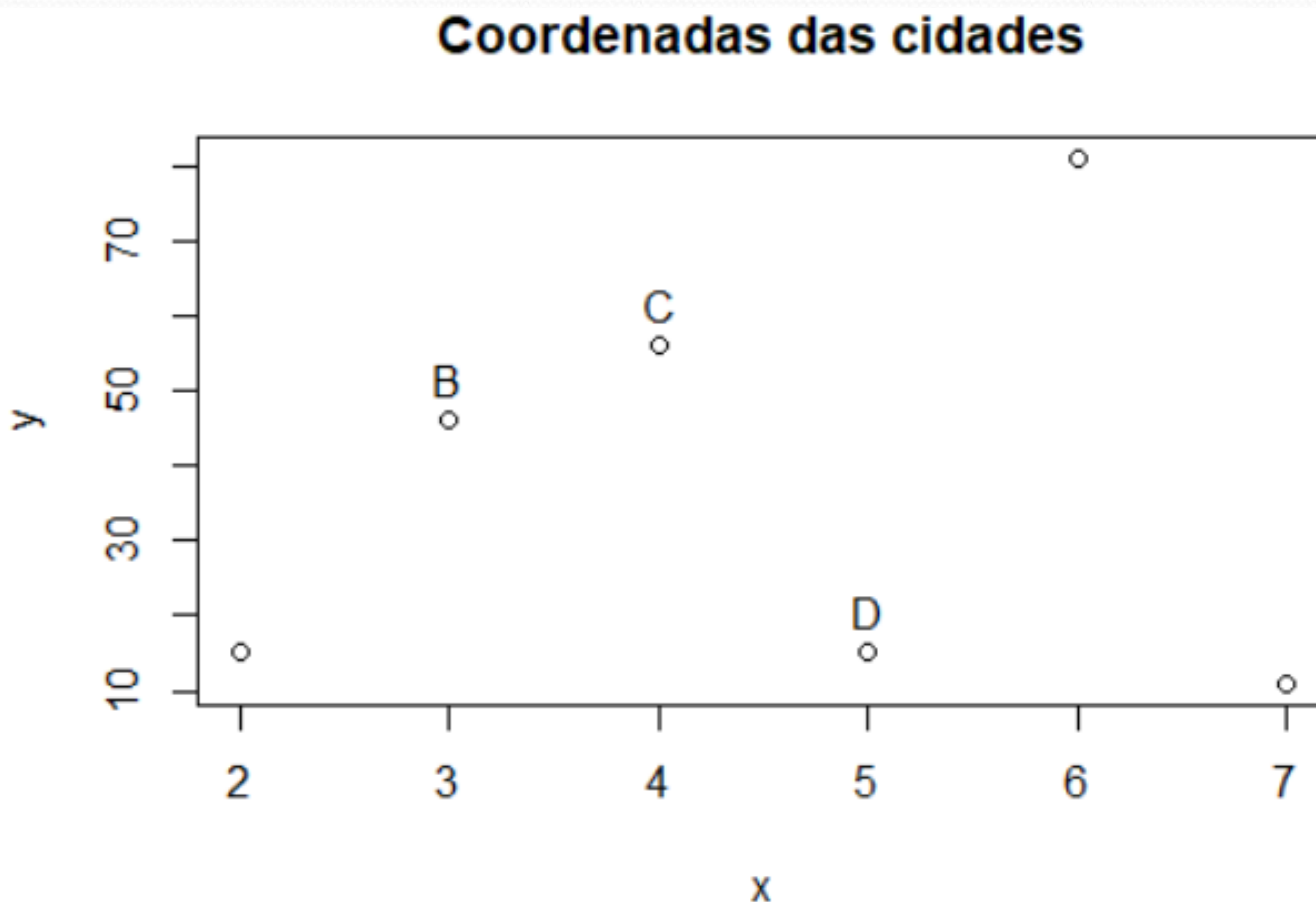
```
> identify(x,y,nomes,n=3)
```

aviso: ponto mais próximo já identificado

aviso: ponto mais próximo já identificado

aviso: ponto mais próximo já identificado

[1] 2 3 4 ←



## 6. Pacotes gráficos adicionais:

- lattice
- ggplot2





# 7. Exercícios