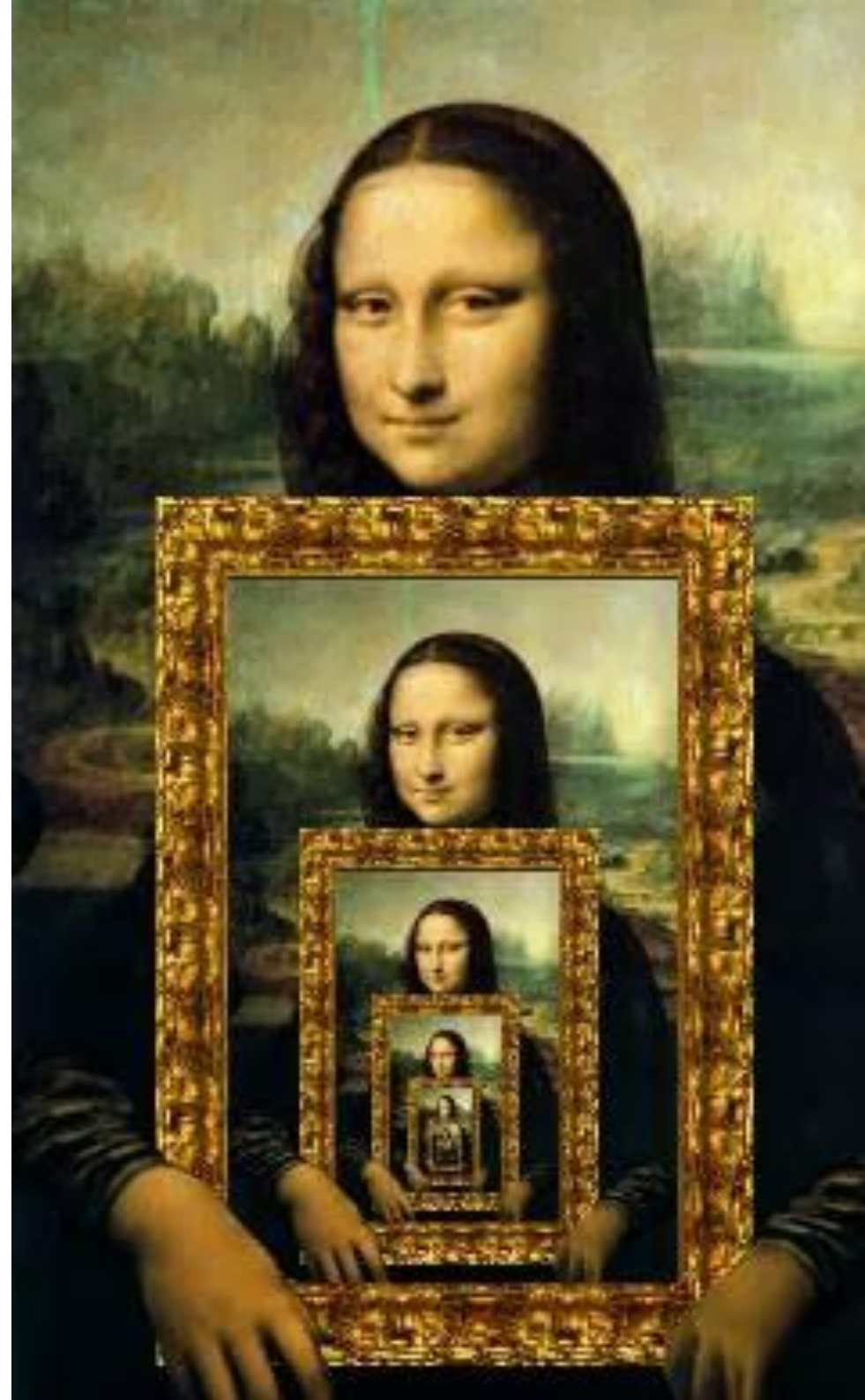


# RECURSIVIDADE

---

Gabriel Barbosa da Fonseca



# INTRODUÇÃO

- Definição: uma função é dita recursiva quando ela **chama a si própria**
- Uma função recursiva normalmente apresenta duas características básicas:
  - **Chamada recursiva**
  - **Condição de parada**
- Condições de parada podem ser similares/análogas às utilizadas em laços de repetição.

- Identifique as chamadas recursivas e condições de parada

```
int fat (int n){  
    int resp;  
    if (n == 1){  
        resp = 1;  
    } else {  
        resp = n * fat (n - 1);  
    }  
    return resp;  
}
```

```
int fib (int n){  
    int resp;  
    if (n == 0 || n == 1){  
        resp = 1;  
    } else {  
        resp = fib (n - 1) + fib(n - 2);  
    }  
    return resp;  
}
```

- Identifique as chamadas recursivas e condições de parada

```
int fat (int n){  
    int resp;  
    if (n == 1){  
        resp = 1;  
    } else {  
        resp = n * fat (n - 1);  
    }  
    return resp;  
}
```

Chamadas recursivas

```
int fib (int n){  
    int resp;  
    if (n == 0 || n == 1){  
        resp = 1;  
    } else {  
        resp = fib (n - 1) + fib(n - 2);  
    }  
    return resp;  
}
```

- Identifique as chamadas recursivas e condições de parada

```
int fat (int n){  
    int resp;  
    if (n == 1){  
        resp = 1;  
    } else {  
        resp = n * fat (n - 1);  
    }  
    return resp;  
}
```

Condições de parada

A cada chamada recursiva,  
o n se aproxima do último valor

```
int fib (int n){  
    int resp;  
    if (n == 0 | n == 1){  
        resp = 1;  
    } else {  
        resp = fib (n - 1) + fib(n - 2);  
    }  
    return resp;  
}
```

# Exemplo de Procedimento Iterativo vs Recursivo


- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar () {  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d", i);  
    }  
}
```


```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i) {  
    if (i < 4) {  
        printf("%d", i);  
        mostrar (i + 1);  
    }  
}
```

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3



```
void mostrar () {  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d", i);  
    }  
}
```



```
void auxiliar () {  
    mostrar (0);  
}  
void mostrar (int i) {  
    if (i < 4) {  
        printf("%d", i);  
        mostrar (i + 1);  
    }  
}
```

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar () {  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d", i);  
    }  
}
```

Tela

```
void auxiliar () {  
    mostrar (0);  
}  
void mostrar (int i) {  
    if (i < 4) {  
        printf("%d", i);  
        mostrar (i + 1);  
    }  
}
```



# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i	0
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

true

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i	0
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

0

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

0

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

0

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

1

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

0

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

1

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

0

true

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

1

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar () {  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d", i);  
    }  
}
```

Tela

0

1

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i) {  
    if (i < 4) {  
        printf("%d", i);  
        mostrar (i + 1);  
    }  
}
```

i

1

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

0

1

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

2



# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela
0
1

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i	2
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela
0
1

true

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i	2
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar () {  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d", i);  
    }  
}
```

Tela

0  
1  
2

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i) {  
    if (i < 4) {  
        printf("%d", i);  
        mostrar (i + 1);  
    }  
}
```

i	2
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1){  
        printf("%d",i);  
    }  
}
```

Tela
0
1
2

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i	3
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

Tela

0

1

2

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

3

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){\n    for (int i = 0; i < 4; i = i + 1) {\n        printf("%d",i);\n    }\n}
```

```
void auxiliar() {\n    mostrar (0);\n}\nvoid mostrar (int i){\n    if (i < 4) {\n        printf("%d",i);\n        mostrar (i + 1);\n    }\n}
```

Tela
0
1
2

true

i	3
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

Tela
0
1
2
3

i	3
---	---

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1){  
        printf("%d",i);  
    }  
}
```

Tela

0

1

2

3

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

4



# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1){  
        printf("%d",i);  
    }  
}
```

Tela

0

1

2

3

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

i

4

# Exemplo de Procedimento Iterativo vs Recursivo

- Procedimentos ITERATIVO e RECURSIVO para mostrar os números 0 à 3

```
void mostrar (){  
    for (int i = 0; i < 4; i = i + 1) {  
        printf("%d",i);  
    }  
}
```

```
void auxiliar() {  
    mostrar (0);  
}  
void mostrar (int i){  
    if (i < 4) {  
        printf("%d",i);  
        mostrar (i + 1);  
    }  
}
```

Tela
0
1
2
3

false

i	4
---	---

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1o – inicio");  
    segundo();  
    printf("1o – fim");  
}  
  
void segundo(){  
    printf("2o – inicio e fim");  
}  
  
void main (){  
    printf("main – inicio");  
    primeiro();  
    printf("main – fim");  
}
```

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1o – início");  
    segundo();  
    printf("1o – fim");  
}  
  
void segundo(){  
    printf("2o – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1o – início");  
    segundo();  
    printf("1o – fim");  
}  
  
void segundo(){  
    printf("2o – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

main – início

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1o – início");  
    segundo();  
    printf("1o – fim");  
}  
  
void segundo(){  
    printf("2o – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

main – início

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1o – início");  
    segundo();  
    printf("1o – fim");  
}  
  
void segundo(){  
    printf("2o – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

main – início

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

main – início

1º – início



# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

main – início

1º – início

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

main – início

1º – início

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

## TELA

main – início

1º – início

2º – início e fim

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

TELA

main – início

1º – início

2º – início e fim

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

## TELA

main – início

1º – início

2º – início e fim

1º – fim

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

## TELA

main – início

1º – início

2º – início e fim

1º – fim

# Exemplo de Procedimento Iterativo

- O que o programa iterativo abaixo mostra na tela?

```
void primeiro(){  
    printf("1º – início");  
    segundo();  
    printf("1º – fim");  
}  
  
void segundo(){  
    printf("2º – início e fim");  
}  
  
void main (){  
    printf("main – início");  
    primeiro();  
    printf("main – fim");  
}
```

## TELA

main – início

1º – início

2º – início e fim

1º – fim

main – fim

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){
    printRecursoivo(2);
}

void printRecursoivo(int i){
    printf("%d",i);
    if (i > 0){
        printRecursoivo(i - 1);
    }
    printf("%d",i);
}
```



# Exercício

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}  
  
void printRecursoivo(int i){  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

Temos como se cada chamada recursiva fosse uma função diferente!!!

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
(1) void printRecursoivo(){  
(2)     printRecursoivo(2);  
}
```

```
(3) void printRecursoivo(int i){ // i (2)  
(4)     printf("%d",i);  
(5)     if (i > 0){  
(6)         printRecursoivo(i - 1);  
        }  
        printf("%d",i);  
}
```

```
(7) void printRecursoivo(int i){ // i (1)  
(8)     printf("%d",i);  
(9)     if (i > 0){  
(10)        printRecursoivo(i - 1);  
        }  
        printf("%d",i);  
}
```

```
(11) void printRecursoivo(int i){ // i (0)  
(12) printf("%d",i);  
(13) if (i > 0){  
        printRecursoivo(i - 1);  
    }  
(14) printf("%d",i);  
}
```

# Exercício - Reavaliando

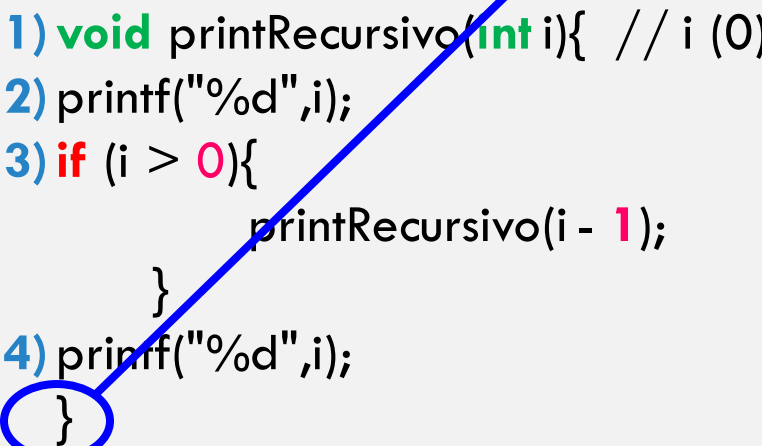
● Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
(1) void printRecursoivo(){  
(2)     printRecursoivo(2);  
}
```

```
(3) void printRecursoivo(int i){ // i (2)  
(4)     printf("%d",i);  
(5)     if (i > 0){  
(6)         printRecursoivo(i - 1);  
        }  
        printf("%d",i);  
}
```

```
(7) void printRecursoivo(int i){ // i (1)  
(8)     printf("%d",i);  
(9)     if (i > 0){  
(10)        printRecursoivo(i - 1); (15)  
        } (16)  
        printf("%d",i); (17)  
}
```

```
(11) void printRecursoivo(int i){ // i (0)  
(12) printf("%d",i);  
(13) if (i > 0){  
        printRecursoivo(i - 1);  
    }  
(14) printf("%d",i);  
    }
```



# Exercício - Reavaliando

● Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
(1) void printRecursoivo(){  
(2)     printRecursoivo(2);  
}
```

```
(3) void printRecursoivo(int i){ // i (2)  
(4)     printf("%d",i);  
(5)     if (i > 0){  
(6)         printRecursoivo(i - 1); (18)  
} (19)  
    printf("%d",i); (20)  
}
```

```
(7) void printRecursoivo(int i){ // i (1)  
(8)     printf("%d",i);  
(9)     if (i > 0){  
(10)        printRecursoivo(i - 1); (15)  
} (16)  
    printf("%d",i); (17)  
}
```

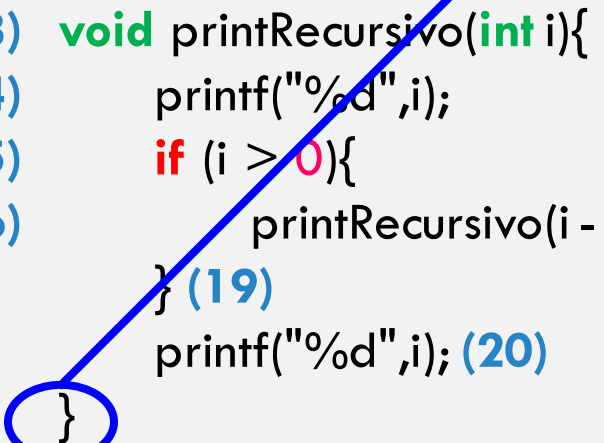
```
(11) void printRecursoivo(int i){ // i (0)  
(12) printf("%d",i);  
(13) if (i > 0){  
        printRecursoivo(i - 1);  
}  
(14) printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
(1) void printRecursoivo(){  
(2)     printRecursoivo(2); (21)  
}
```

```
(3) void printRecursoivo(int i){ // i (2)  
(4)     printf("%d",i);  
(5)     if (i > 0){  
(6)         printRecursoivo(i - 1); (18)  
            } (19)  
            printf("%d",i); (20)  
        }
```



```
(7) void printRecursoivo(int i){ // i (1)  
(8)     printf("%d",i);  
(9)     if (i > 0){  
(10)        printRecursoivo(i - 1); (15)  
            } (16)  
            printf("%d",i); (17)  
        }
```

```
(11) void printRecursoivo(int i){ // i (0)  
(12) printf("%d",i);  
(13) if (i > 0){  
            printRecursoivo(i - 1);  
        }  
(14) printf("%d",i);  
        }
```

# Exercício - Reavaliando

● Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
(1) void printRecursoivo(){  
(2)     printRecursoivo(2); (21)  
    }
```

```
(3) void printRecursoivo(int i){ // i (2)  
(4)     printf("%d",i);  
(5)     if (i > 0){  
(6)         printRecursoivo(i - 1); (18)  
    } (19)  
    printf("%d",i); (20)  
}
```

```
(7) void printRecursoivo(int i){ // i (1)  
(8)     printf("%d",i);  
(9)     if (i > 0){  
(10)        printRecursoivo(i - 1); (15)  
    } (16)  
        printf("%d",i); (17)  
}
```

```
(11) void printRecursoivo(int i){ // i (0)  
(12) printf("%d",i);  
(13) if (i > 0){  
        printRecursoivo(i - 1);  
    }  
(14) printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```



# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```


```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```


```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```


```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

**true**


```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

Vamos para o print do um,  
contudo, depois, voltaremos para (a)

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime 2, 1, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime **2**, **1**, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```



# Exercício - Reavaliando

- Por que o código abaixo imprime **2**, **1**, 0, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

Vamos para o print do zero,  
contudo, depois, voltaremos para (b)


```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (b)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```


# Exercício - Reavaliando

● Por que o código abaixo imprime **2**, **1**, 0, 0, 1 e 2?


```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```



```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (b)  
    }  
    printf("%d",i);  
}
```



```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime **2**, **1**, **0**, 0, 1 e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (b)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime **2, 1, 0, 0, 1** e **2**?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (b)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

**false**

# Exercício - Reavaliando

● Por que o código abaixo imprime **2, 1, 0, 0, 1** e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (b)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime **2, 1, 0, 0, 1** e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

Voltando para (b)

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (b)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (0)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

- Por que o código abaixo imprime **2, 1, 0, 0, 1** e **2**?

```
void printRecursivo(){  
    printRecursivo(2);  
}
```

```
void printRecursivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

Voltando para (b)

```
void printRecursivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursivo(i - 1); (b)  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

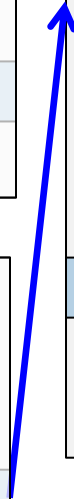
- Por que o código abaixo imprime **2, 1, 0, 0, 1** e **2**?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```





# Exercício - Reavaliando

● Por que o código abaixo imprime **2, 1, 0, 0, 1** e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1); (a)  
    }  
    printf("%d",i);  
}
```

```
void printRecursoivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime **2, 1, 0, 0, 1** e 2?

```
void printRecursivo(){  
    printRecursivo(2);  
}
```

```
void printRecursivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursivo(i - 1);  
    }  
    printf("%d",i);  
}
```

```
void printRecursivo(int i){ // i (1)  
    printf("%d",i);  
    if (i > 0){  
        printRecursivo(i - 1);  
    }  
    printf("%d",i);  
}
```

Voltando para (a)

# Exercício - Reavaliando

● Por que o código abaixo imprime **2, 1, 0, 0, 1** e 2?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```

```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```


(a)

Voltando para (a)

## Exercício - Reavaliando

● Por que o código abaixo imprime **2**, **1**, **0**, **0**, **1** e **2**?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

## Exercício - Reavaliando

● Por que o código abaixo imprime **2**, **1**, **0**, **0**, **1** e **2**?

```
void printRecursoivo(){  
    printRecursoivo(2);  
}
```



```
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

# Exercício - Reavaliando

● Por que o código abaixo imprime **2**, **1**, **0**, **0**, **1** e **2**?

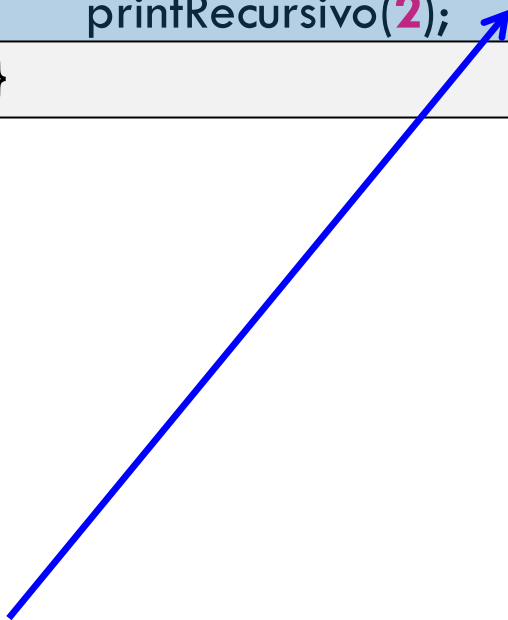
```
void printRecursoivo(){  
    printRecursoivo(2);  
}  
  
void printRecursoivo(int i){ // i (2)  
    printf("%d",i);  
    if (i > 0){  
        printRecursoivo(i - 1);  
    }  
    printf("%d",i);  
}
```

Voltando para (**primeiro**)

## Exercício - Reavaliando

- Por que o código abaixo imprime **2**, **1**, **0**, **0**, **1** e **2**?

```
void printRecursivo(){  
    printRecursivo(2);  
}
```



## Exercício - Reavaliando

● Por que o código abaixo imprime **2**, **1**, **0**, **0**, **1** e **2**?

```
void printRecursivo(){  
    printRecursivo(2);  
}
```



# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?

# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?

$$\text{Fat}(5) = 5 * \text{Fat}(4)$$


# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

$$\text{Fat}(4) = 4 * \text{Fat}(3)$$

- Qual é o valor do fatorial de 5?

$$\text{Fat}(5) = 5 * \text{Fat}(4)$$


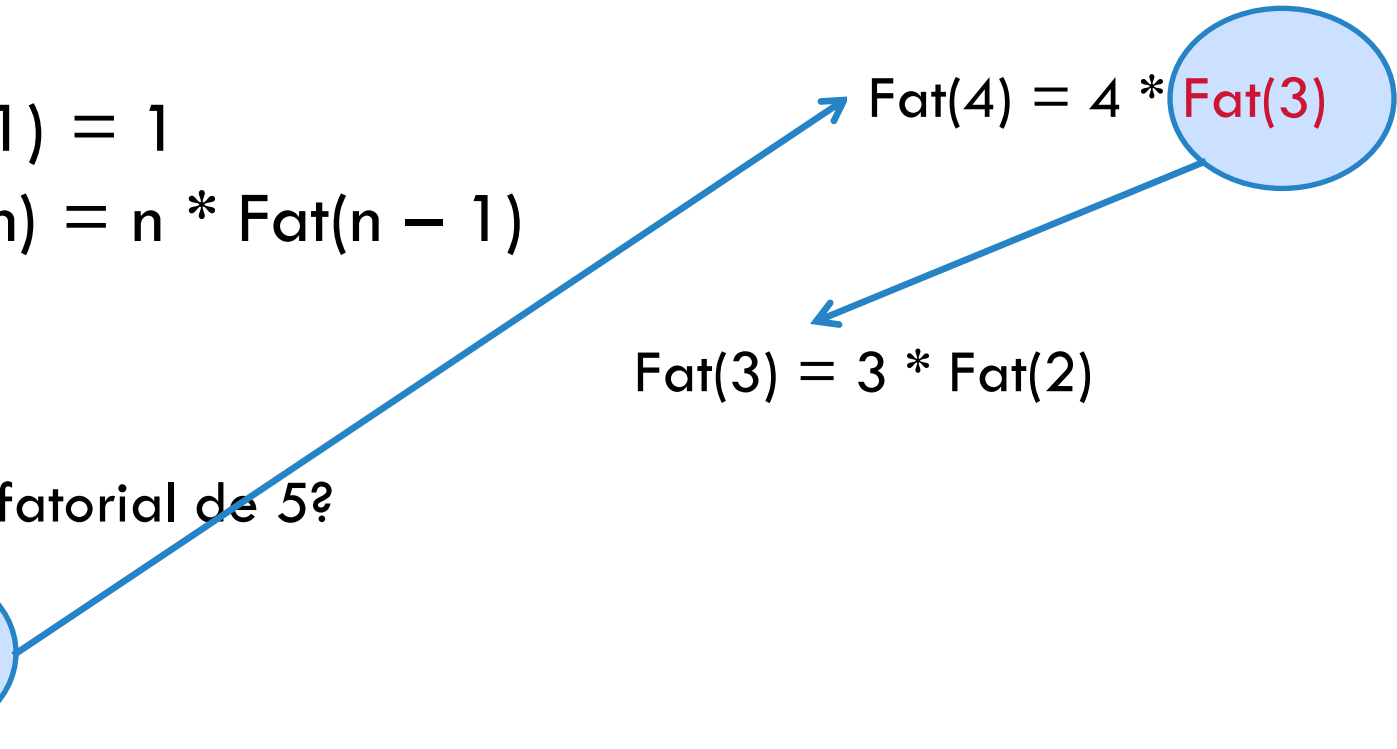
# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?

$$\text{Fat}(5) = 5 * \text{Fat}(4)$$

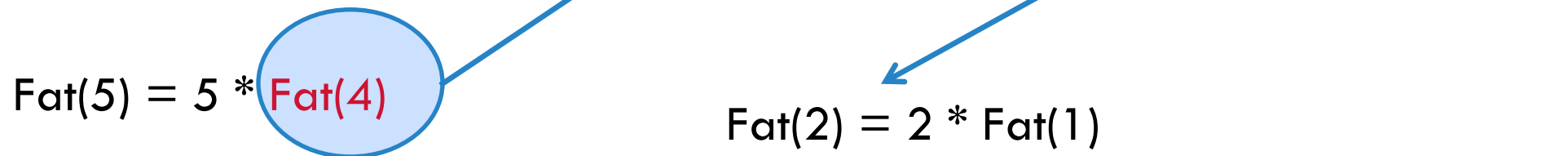
$$\text{Fat}(4) = 4 * \text{Fat}(3)$$
$$\text{Fat}(3) = 3 * \text{Fat}(2)$$


# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?

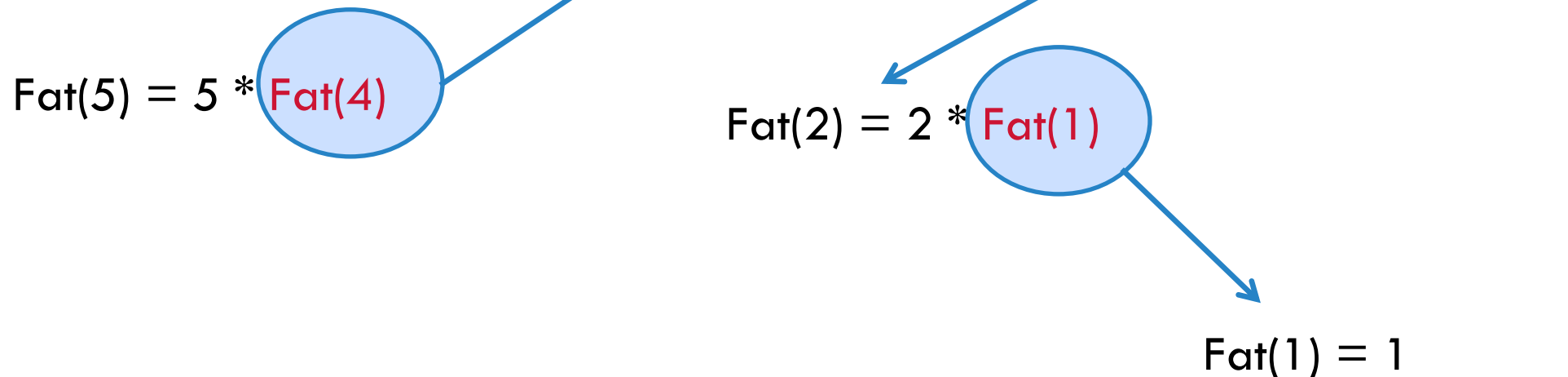


# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?

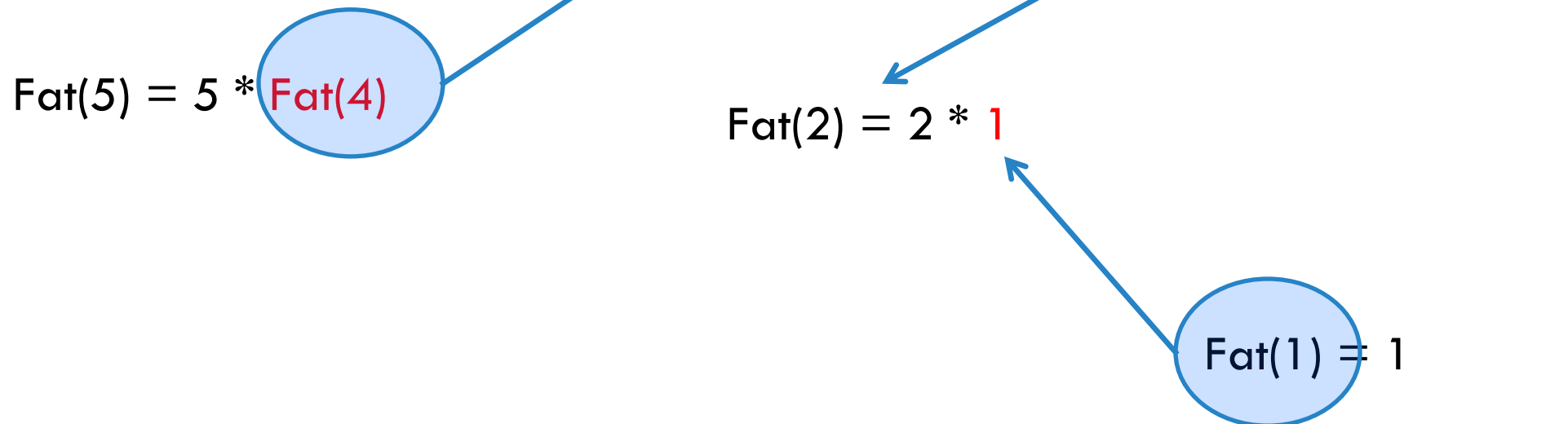


# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?



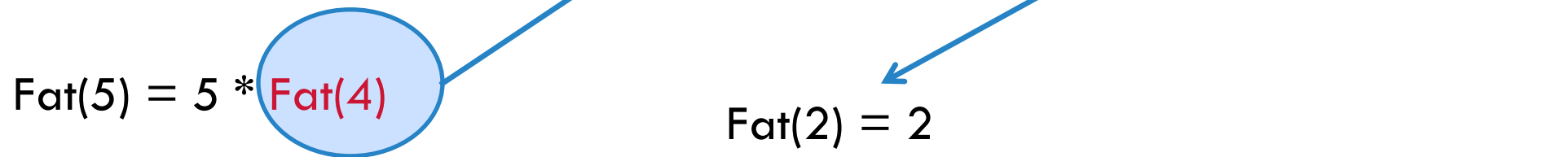


# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?



# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

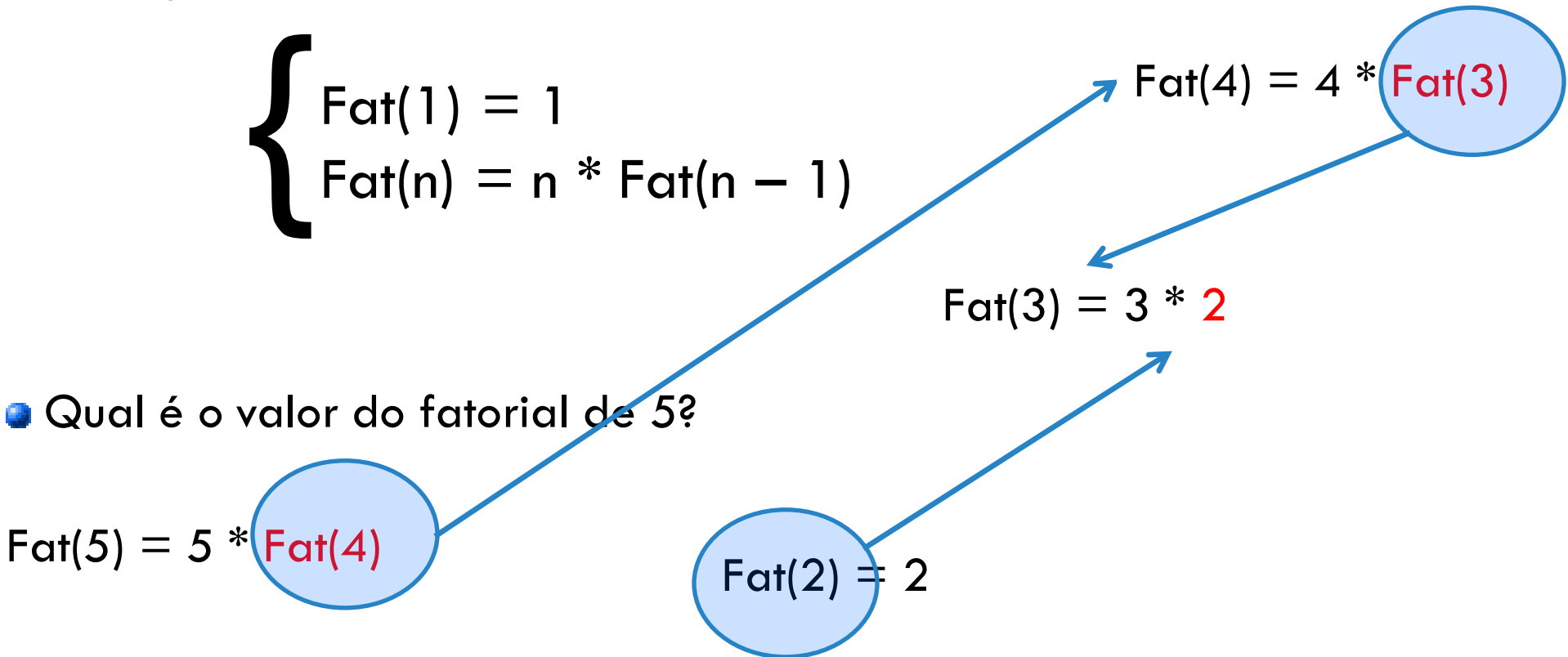
- Qual é o valor do fatorial de 5?

$$\text{Fat}(5) = 5 * \text{Fat}(4)$$

$$\text{Fat}(2) = 2$$

$$\text{Fat}(3) = 3 * 2$$

$$\text{Fat}(4) = 4 * \text{Fat}(3)$$



# Exemplo: Fatorial Recursivo

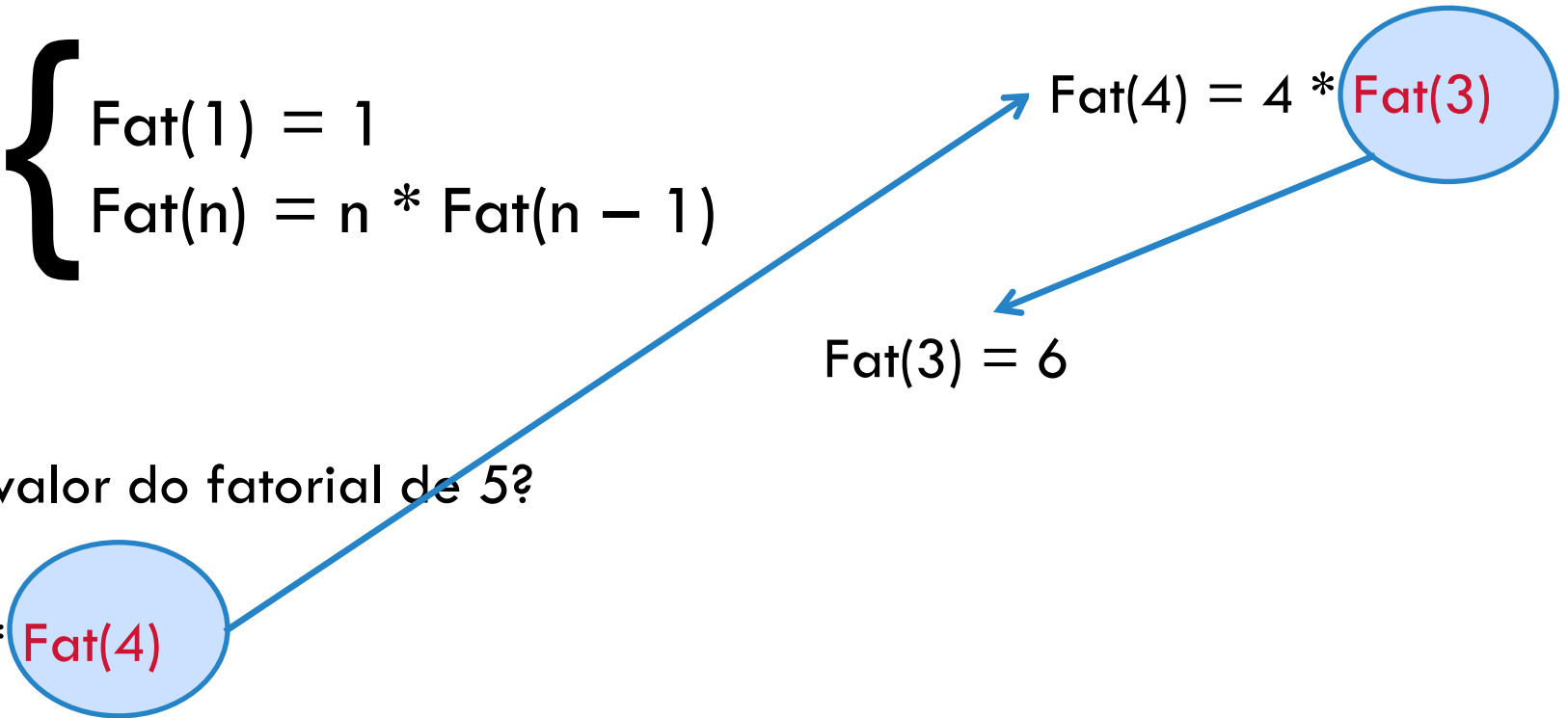
- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?

$$\text{Fat}(5) = 5 * \text{Fat}(4)$$

$$\text{Fat}(4) = 4 * \text{Fat}(3)$$
$$\text{Fat}(3) = 6$$



# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?



# Exemplo: Fatorial Recursivo

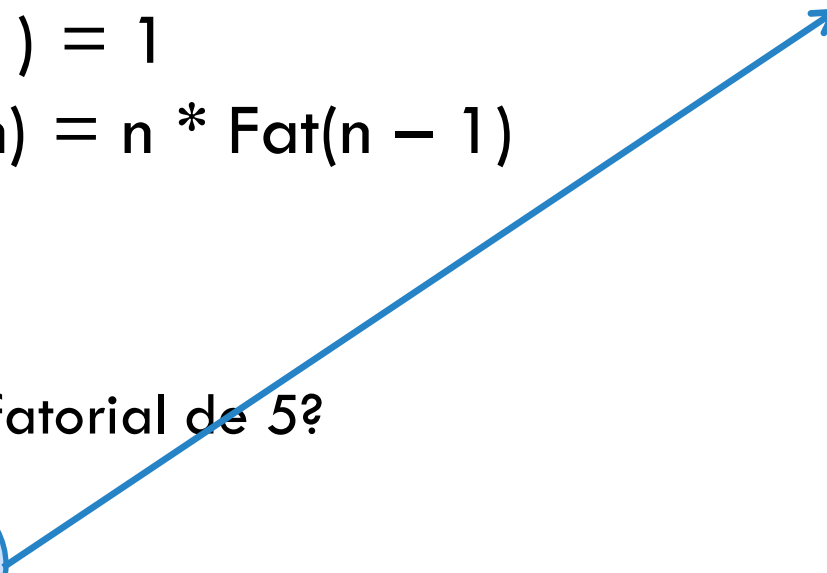
- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

$$\text{Fat}(4) = 24$$

- Qual é o valor do fatorial de 5?

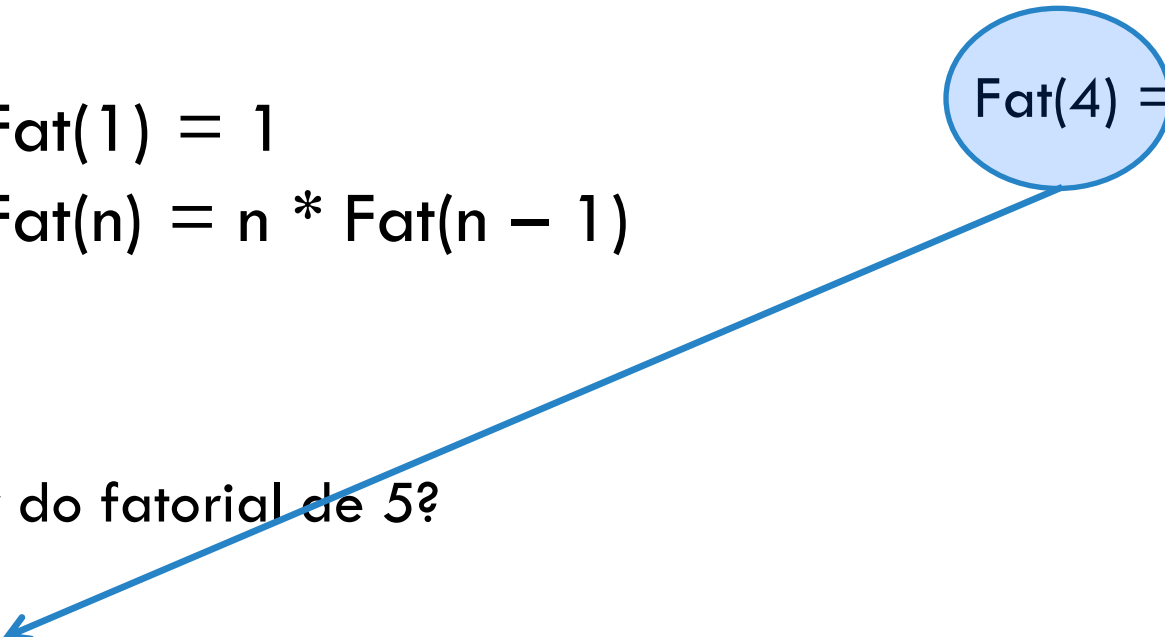
$$\text{Fat}(5) = 5 * \text{Fat}(4)$$



# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$


$$\text{Fat}(4) = 24$$

- Qual é o valor do fatorial de 5?

$$\text{Fat}(5) = 5 * 24$$

# Exemplo: Fatorial Recursivo

- Definição do fatorial é recursiva:

$$\begin{cases} \text{Fat}(1) = 1 \\ \text{Fat}(n) = n * \text{Fat}(n - 1) \end{cases}$$

- Qual é o valor do fatorial de 5?

$$\text{Fat}(5) = 120$$

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){  
    int resp;  
    if (n == 1){  
        resp = 1;  
    } else {  
        resp = n * fatorial(n - 1);  
    }  
    return resp;  
}  
void main(){  
    int valor = fatorial(5);  
    printf("%d",valor);  
}
```



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){  
    int resp;  
    if (n == 1){  
        resp = 1;  
    } else {  
        resp = n * fatorial(n - 1);  
    }  
    return resp;  
}
```

```
void main(){  
    int valor = fatorial(5);  
    printf("%d",valor);  
}
```

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){  
    int resp;  
    if (n == 1){  
        resp = 1;  
    } else {  
        resp = n * fatorial(n - 1);  
    }  
    return resp;  
}  
void main(){  
    int valor = fatorial(5);  
    printf("%d",valor);  
}
```

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){ // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}
void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}
void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {             false
        resp = n * fatorial(n - 1);
    }
    return resp;
}
void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$

$\text{fatorial}(3) = 3 * \text{fatorial}(2)$

$\text{fatorial}(2) = 2 * \text{fatorial}(1)$

$\text{fatorial}(1) = 1$

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){ // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$





# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}
void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {             false
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){ // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$

$\text{fatorial}(3) = 3 * \text{fatorial}(2)$

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}
void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {             false
        resp = n * fatorial(n - 1);
    }
    return resp;
}
void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$

$\text{fatorial}(3) = 3 * \text{fatorial}(2)$

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 4 * \text{fatorial}(3)$

$\text{fatorial}(3) = 3 * \text{fatorial}(2)$



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){ // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {             false
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){ // n (1)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

fatorial (1) = 1

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (1)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

fatorial (1) = 1



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (1)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

fatorial (1) = 1

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (1)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

fatorial (1) = 1

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (1)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* fatorial (1)

fatorial (1) = 1

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2 \* 1

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (2)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* fatorial (2)

fatorial (2) = 2

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 3 \* 2

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 6



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (3)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* fatorial (3)

fatorial (3) = 6

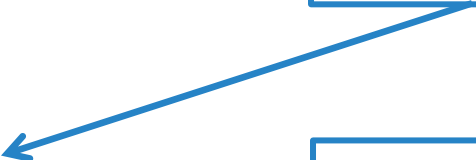
# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 5 \* fatorial (4)

fatorial (4) = 4 \* 6



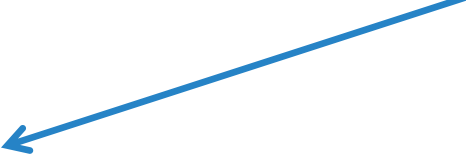
# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 24$




# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (4)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$\text{fatorial}(5) = 5 * \text{fatorial}(4)$

$\text{fatorial}(4) = 24$



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

$$\text{fatorial}(5) = 5 * \boxed{24}$$

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 120

# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 120



# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}

void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

fatorial (5) = 120





# Exemplo: Fatorial Recursivo

```
int fatorial (int n){    // n (5)
    int resp;
    if (n == 1){
        resp = 1;
    } else {
        resp = n * fatorial(n - 1);
    }
    return resp;
}
void main(){
    int valor = fatorial(5);
    printf("%d",valor);
}
```

# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

Fibonacci(4)

# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

$$\begin{array}{c} \text{Fibonacci}(4) \\ \swarrow \quad \searrow \\ (\text{Fibonacci}(3) + \text{Fibonacci}(2)) \end{array}$$

$$(\text{Fibonacci}(2) + \text{Fibonacci}(1)) + \text{Fibonacci}(2)$$

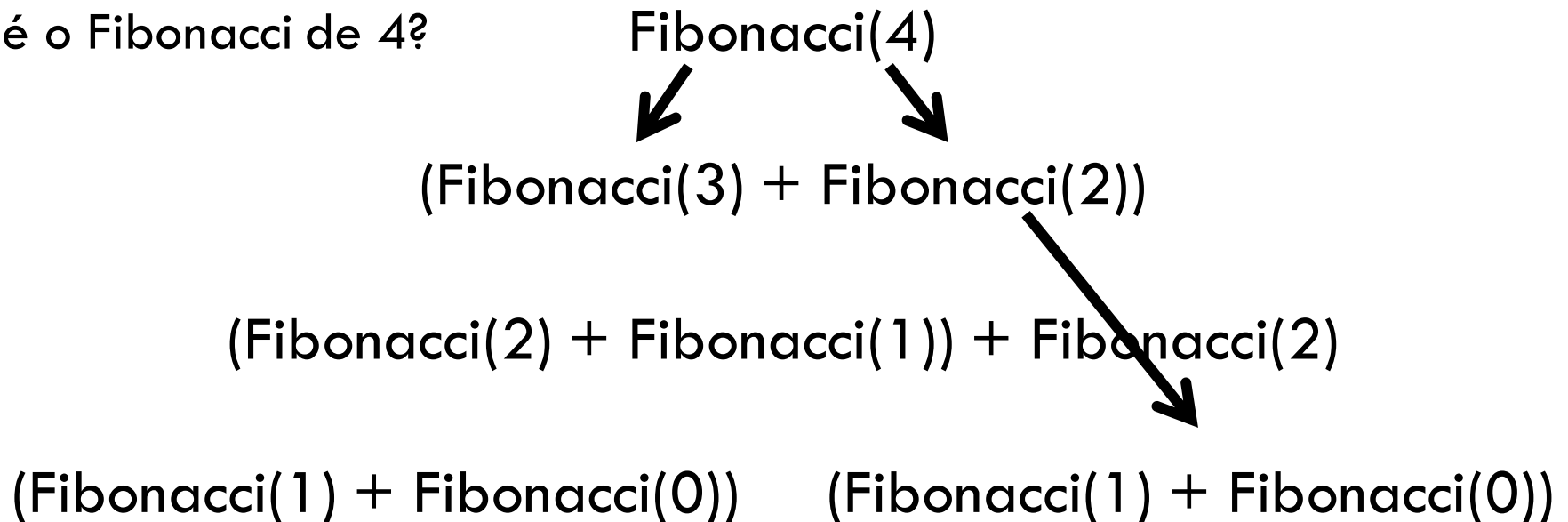
$$(\text{Fibonacci}(1) + \text{Fibonacci}(0)) \quad (\text{Fibonacci}(1) + \text{Fibonacci}(0))$$

# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?



# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

$$\begin{array}{c} \text{Fibonacci}(4) \\ \swarrow \quad \searrow \\ (\text{Fibonacci}(3) + \text{Fibonacci}(2)) \\ \searrow \\ (\text{Fibonacci}(2) + \text{Fibonacci}(1)) + \text{Fibonacci}(2) \\ \searrow \\ (1 + 1) \end{array}$$

# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

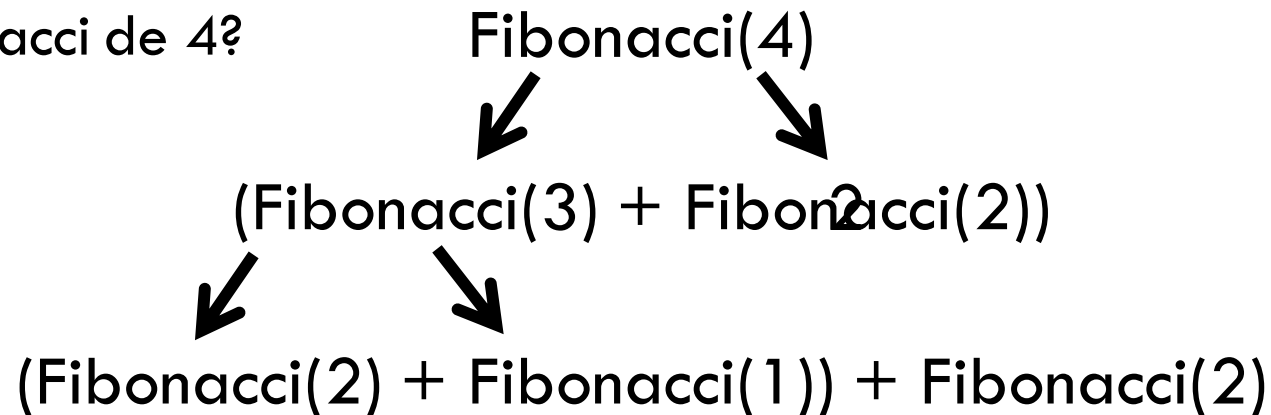
$$\begin{array}{c} \text{Fibonacci}(4) \\ \swarrow \quad \searrow \\ (\text{Fibonacci}(3) + \text{Fibonacci}(2)) \\ \\ (\text{Fibonacci}(2) + \text{Fibonacci}(1)) + \text{Fibonacci}(2) \end{array}$$

# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?



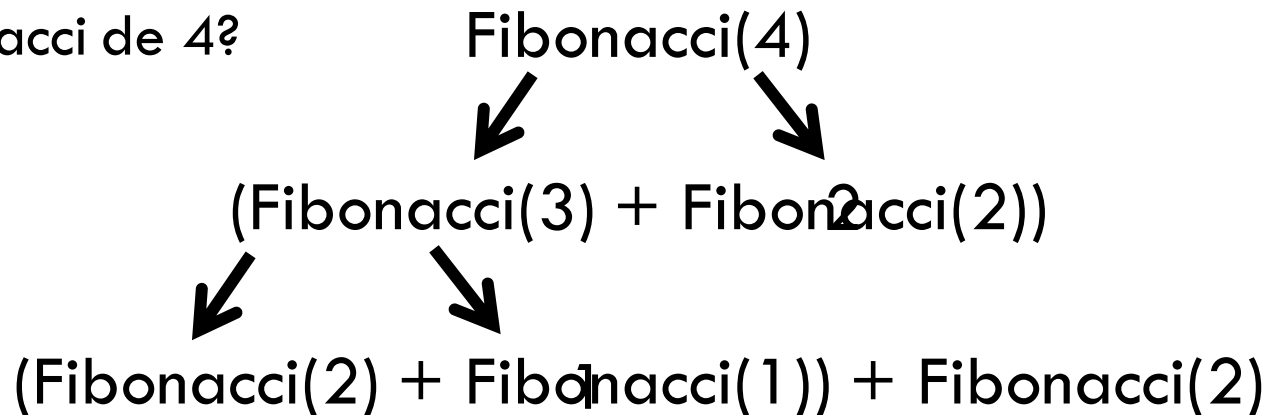


# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

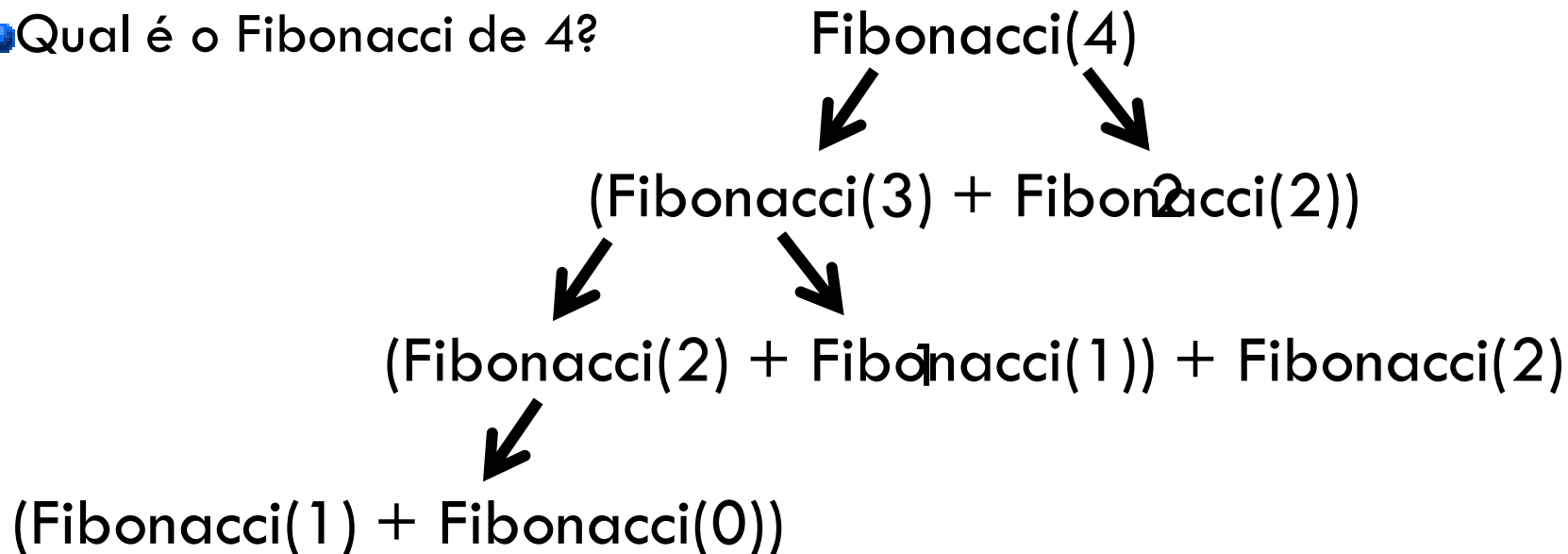


# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

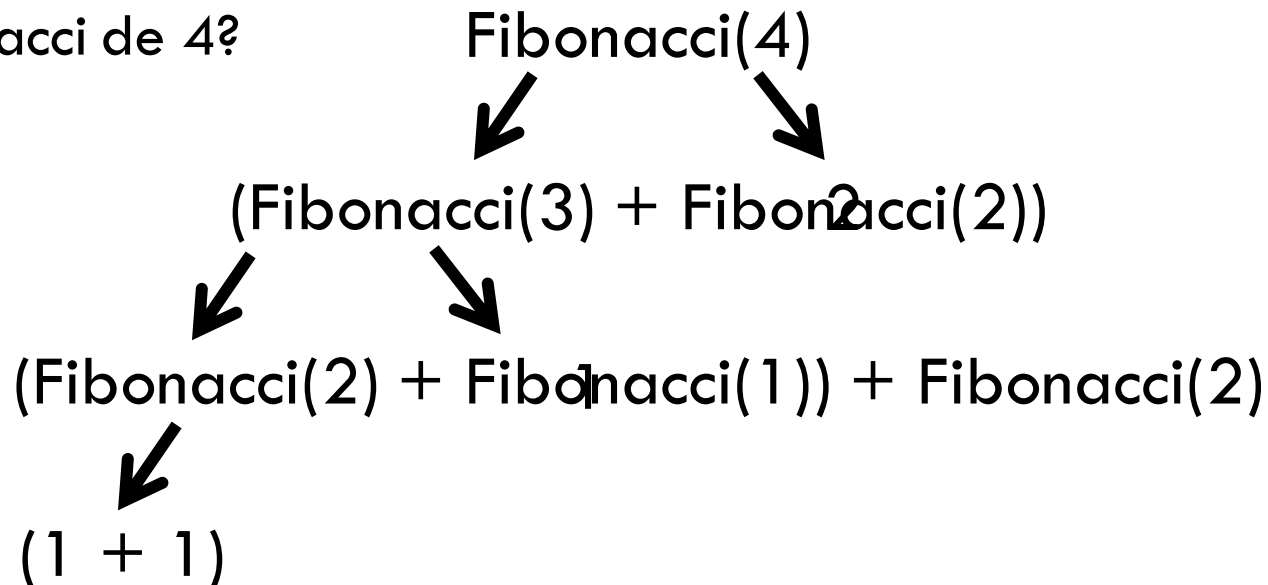


# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

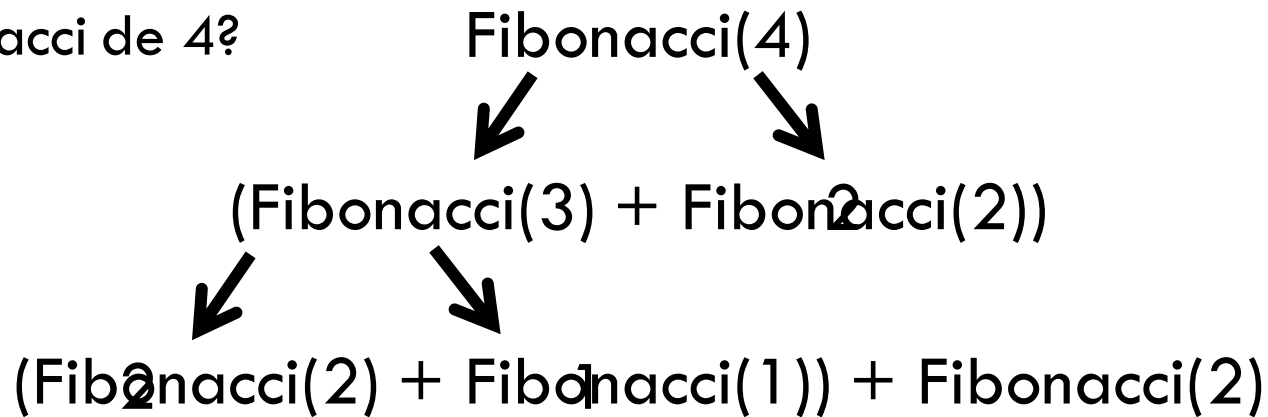


# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?



# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

$$\begin{array}{c} \text{Fibonacci}(4) \\ \swarrow \quad \searrow \\ (\text{Fibonacci}(3) + \text{Fibonacci}(2)) \end{array}$$

# Exemplo: Fibonacci Recursivo

- Definição do fatorial é recursiva:

$$\left\{ \begin{array}{l} \text{Fib}(0) = 1 \\ \text{Fib}(1) = 1 \\ \text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2) \end{array} \right.$$

- Qual é o Fibonacci de 4?

5

# Exemplo: Fibonacci Recursivo

```
int fibonacci (int n){  
    int resp;  
    if (n == 0 || n == 1){  
        resp = 1;  
    } else {  
        resp = fibonacci(n - 1) + fibonacci(n - 2);  
    }  
    return resp;  
}  
void main(){  
    int valor = fibonacci(4);  
    printf("%d",valor);  
}
```

- Faça uma função recursivo que receba dois números inteiros e retorne a multiplicação do primeiro pelo segundo fazendo somas

$$5 \times 13 = \underbrace{5 + 5 + \dots + 5}_{13 \text{ vezes}}$$



- Faça uma função recursivo que receba dois números inteiros e retorne a multiplicação do primeiro pelo segundo fazendo somas

```
int multiplicacao (int a, int b){  
    int resp = 0;  
  
    if (b > 0){  
        resp = a + multiplicacao(a, b - 1);  
    }  
  
    return resp;  
}  
  
void main (...){  
    multiplicacao(4, 3);  
}
```

# Exercício

- Faça uma função recursivo que receba dois números inteiros e retorne a multiplicação do primeiro pelo segundo fazendo somas (**outra resposta**)

```
int multiplicacao (int a, int b, int i){  
    int resp = 0;  
  
    if (i < b){  
        resp = a + multiplicacao(a, b, i + 1);  
    }  
  
    return resp;  
}  
  
int multiplicacao (int a, int b){  
    return multiplicacao(a, b, 0);  
}
```

```
int multiplicacao (int a, int b){  
    int resp = 0;  
  
    for (int i = 0; i < b; i = i + 1){  
        resp += a;  
    }  
  
    return resp;  
}
```

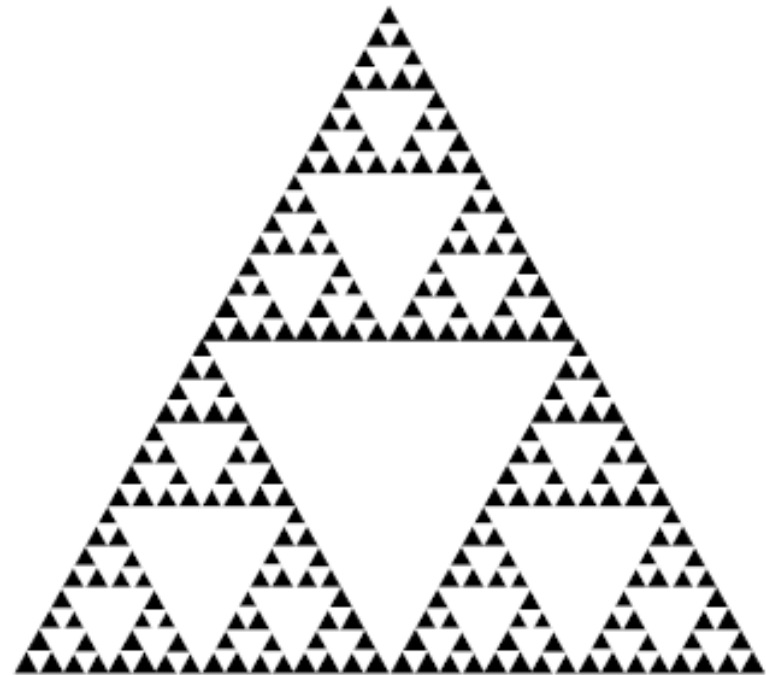
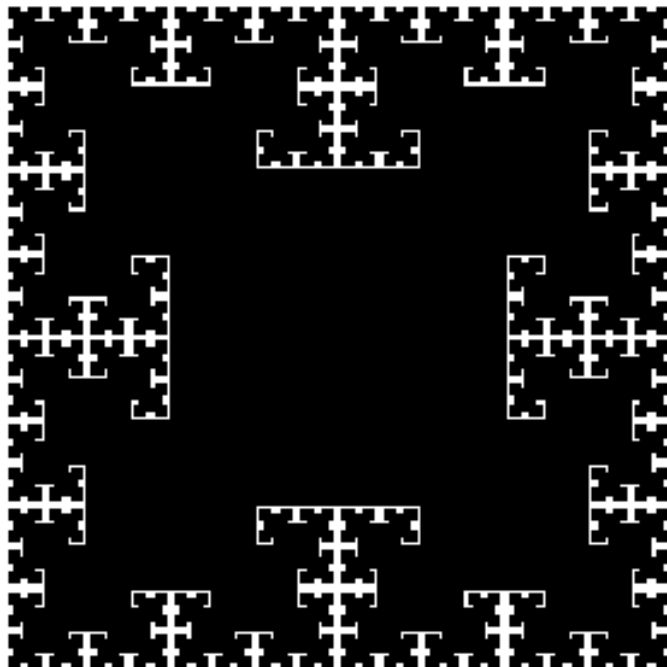
- Todo programa iterativo pode ser feito de forma recursiva e vice-versa
  - Algumas vezes é mais “fácil” fazer um programa de forma recursiva!!!
- O conceito de recursividade é fundamental na computação e na matemática (por exemplo, número naturais, fatorial e outros)
- A recursividade pode ser direta ou indireta (A chama B que chama A)

- O SO usa uma pilha para armazenar o estado corrente do programa antes de cada chamada não terminada e quando uma chamada termina, o SO recupera o estado armazenado na pilha

- As variáveis locais são recriadas para cada chamada recursiva

- Por que na prática é importante manter um nível “limitado” de chamadas recursivas?

- Outro exemplo de recursividade são os fractais, pequenos padrões geométricos que ao serem repetidos diversas vezes de forma recursiva criam desenhos mais sofisticados



- Outro exemplo de recursividade são os fractais, pequenos padrões geométricos que ao serem repetidos diversas vezes de forma recursiva criam desenhos mais sofisticadas

