

Engenharia de Computação

Fundamentos de Computação

Aula 13 – Recursividade

Prof. Fernando Barreto

- Aciona/Chama a própria função
 - A cada chamada, o estado atual da função (variáveis e valores) é suspenso e um novo estado é criado para a nova chamada.
- Deve-se ter um critério de parada, denominado **caso-base**
- Ideia de "dividir para conquistar"
 - Solução de problemas menores....
- Chamar a própria função com parâmetro diferente para que, em algum momento, o **caso-base** ocorra...
- Deve analisar se a recursividade pode ser aplicada ou vale a pena...

```
void recursao(int a) {  
    if (a>0) {  
        printf("%i ",a);  
        recursao(a-1);  
        printf("%i ", a);  
    }  
    else printf("\n");  
}
```

```
int main( ) {  
    recursao(4);  
    return 0;  
}
```

- O tipo **void** usado indica que não possui retorno, para facilitar o exemplo....
OBS: função recursiva pode ter valor de retorno !!!
- Veja análise no próximo slide

```
void recursao(int a) {  
    if (a>0) {  
        printf("%i ",a);  
        recursao(a-1);  
        printf("%i ", a);  
    }  
    else printf("\n");  
}
```

```
int main() {  
    recursao(4);  
    return 0;  
}
```

SAÍDA: 4 3 2 1
1 2 3 4

Estado: e1 if(a>0){
int a==4 printf("%i ",a);
 recursao(a-1);
 printf("%i ",a);
 }

Caminho Ida
Caminho Volta

```
void recursao(int a) {  
    if (a>0) {  
        printf("%i ",a);  
        recursao(a-1);  
        printf("%i ", a);  
    }  
    else printf("\n");  
}
```

```
int main() {  
    recursao(4);  
    return 0;  
}
```

SAÍDA: 4 3 2 1
1 2 3 4

Tela:

```
Estado: e1  if(a>0){  
int a==4      printf("%i ",a);  
              recursao(a-1);  
              printf("%i ",a);  
              }
```

Caminho Ida
Caminho Volta

```
void recursao(int a) {  
    if (a>0) {  
        printf("%i ",a);  
        recursao(a-1);  
        printf("%i ", a);  
    }  
    else printf("\n");  
}
```

```
int main() {  
    recursao(4);  
    return 0;  
}
```

SAÍDA: 4 3 2 1
1 2 3 4

Tela:
4

```
Estado: e1  if(a>0){  
int a==4      printf("%i ",a);  
              recursao(a-1);  
              printf("%i ",a);  
              }
```

Caminho Ida
Caminho Volta

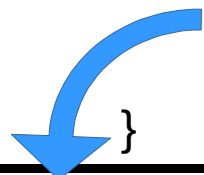
```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

```
int main() {
    recursao(4);
    return 0;
}
```

SAÍDA: 4 3 2 1
1 2 3 4

Tela:
4

Estado: e1 if(a>0){
int a==4 printf("%i ",a);
 recursao(a-1);
 printf("%i ",a);
 }



Tela:
4 3

Estado: e2 if(a>0){
int a==3 printf("%i ",a);
 recursao(a-1);
 printf("%i ",a);
 }

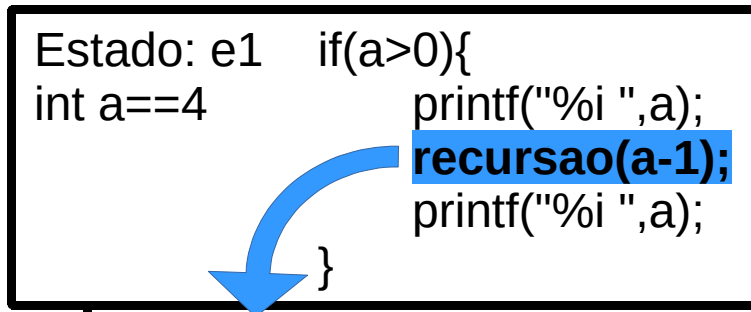
Caminho Ida
Caminho Volta

```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

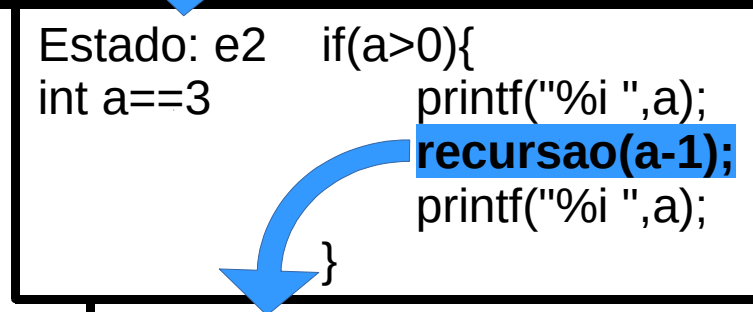
```
int main() {
    recursao(4);
    return 0;
}
```

SAÍDA: 4 3 2 1
1 2 3 4

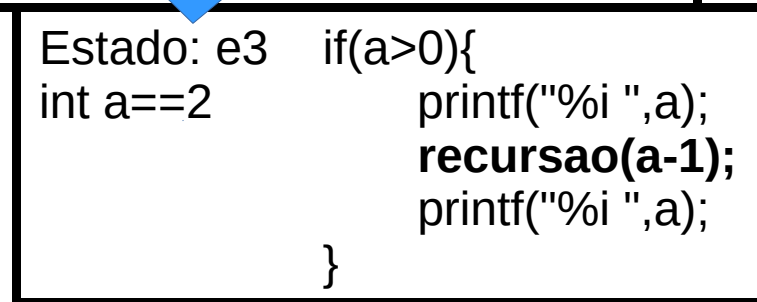
Tela:
4



Tela:
4 3



Tela:
4 3 2

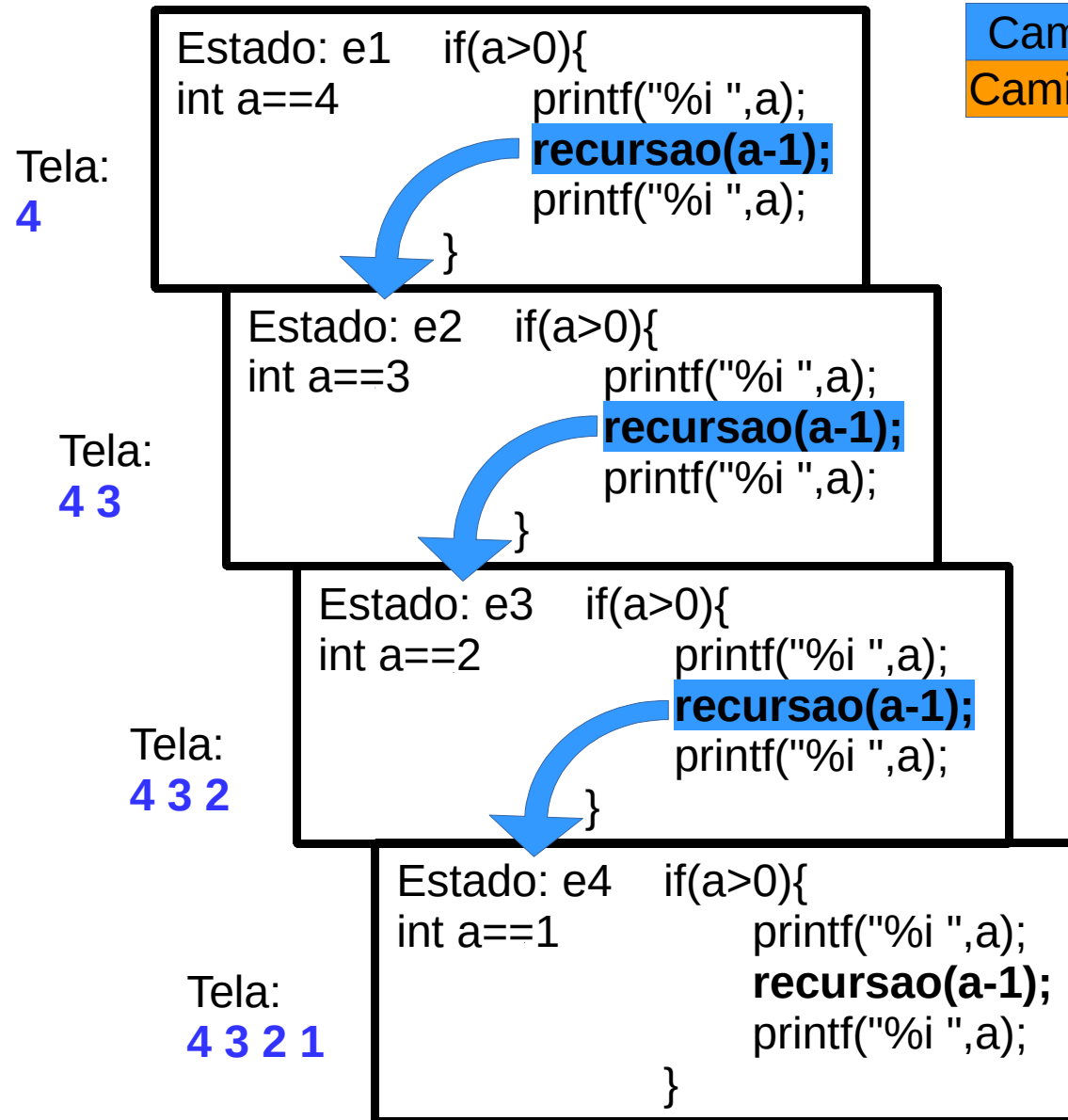


Caminho Ida
Caminho Volta

```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

```
int main() {
    recursao(4);
    return 0;
}
```

SAÍDA: 4 3 2 1
1 2 3 4

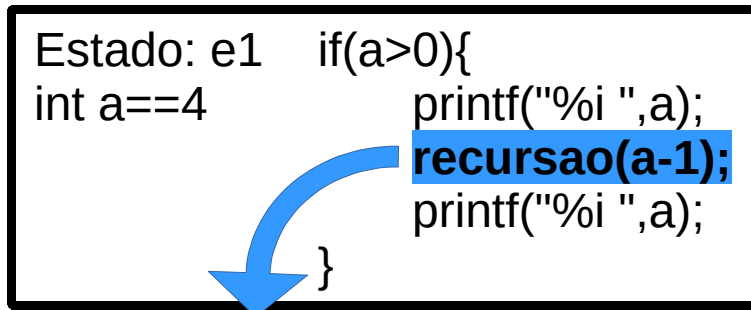



```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

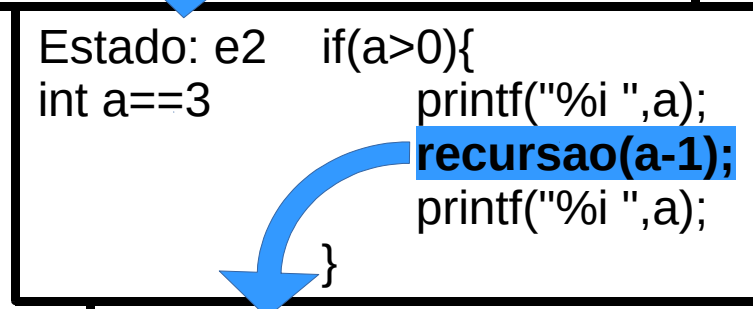
```
int main() {
    recursao(4);
    return 0;
}
```

SAÍDA: 4 3 2 1
1 2 3 4

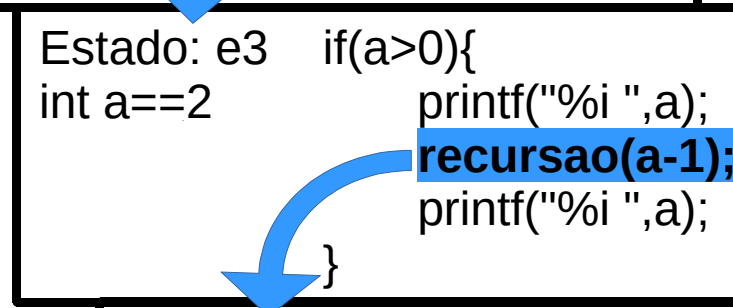
Tela:
4



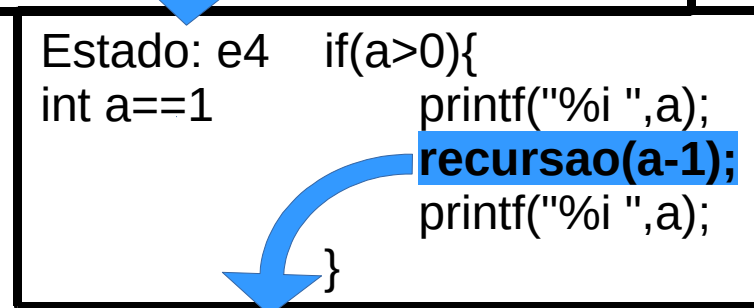
Tela:
4 3



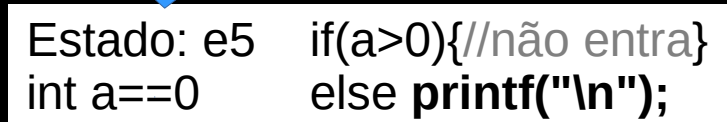
Tela:
4 3 2



Tela:
4 3 2 1



Tela:
4 3 2 1 '\n'

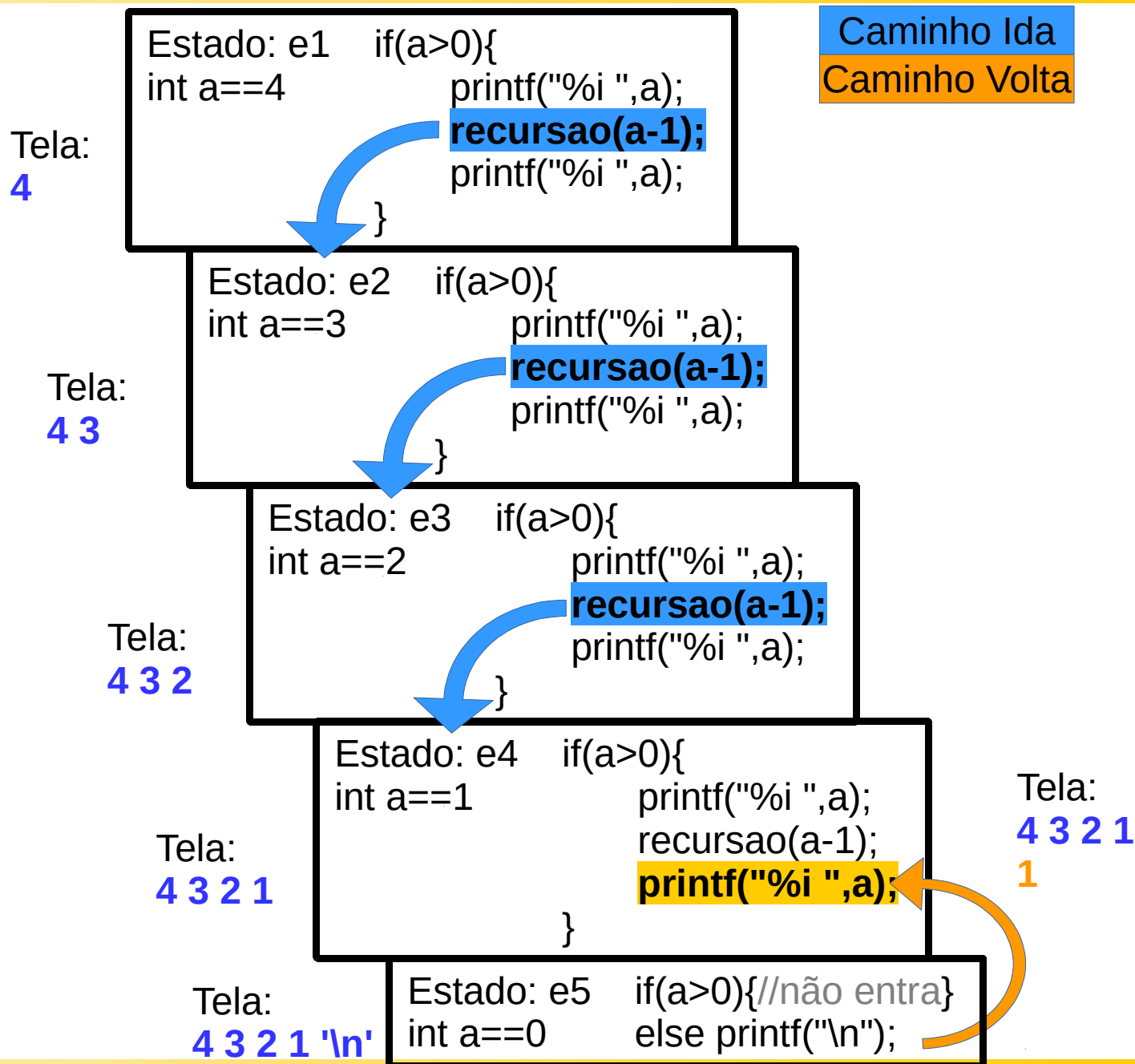


Caminho Ida
Caminho Volta

```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

```
int main() {
    recursao(4);
    return 0;
}
```

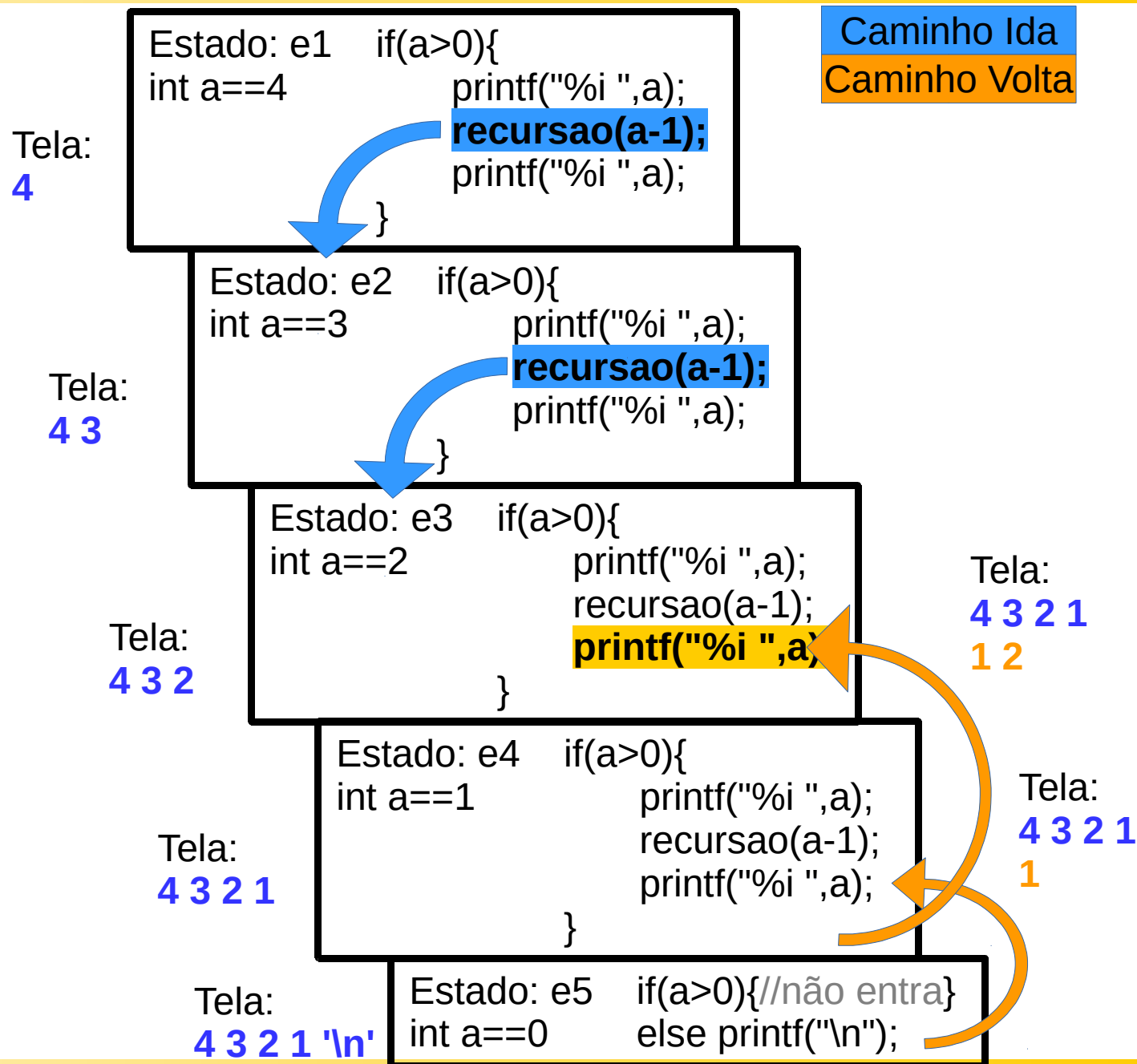
SAÍDA: 4 3 2 1
1 2 3 4



```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

```
int main() {
    recursao(4);
    return 0;
}
```

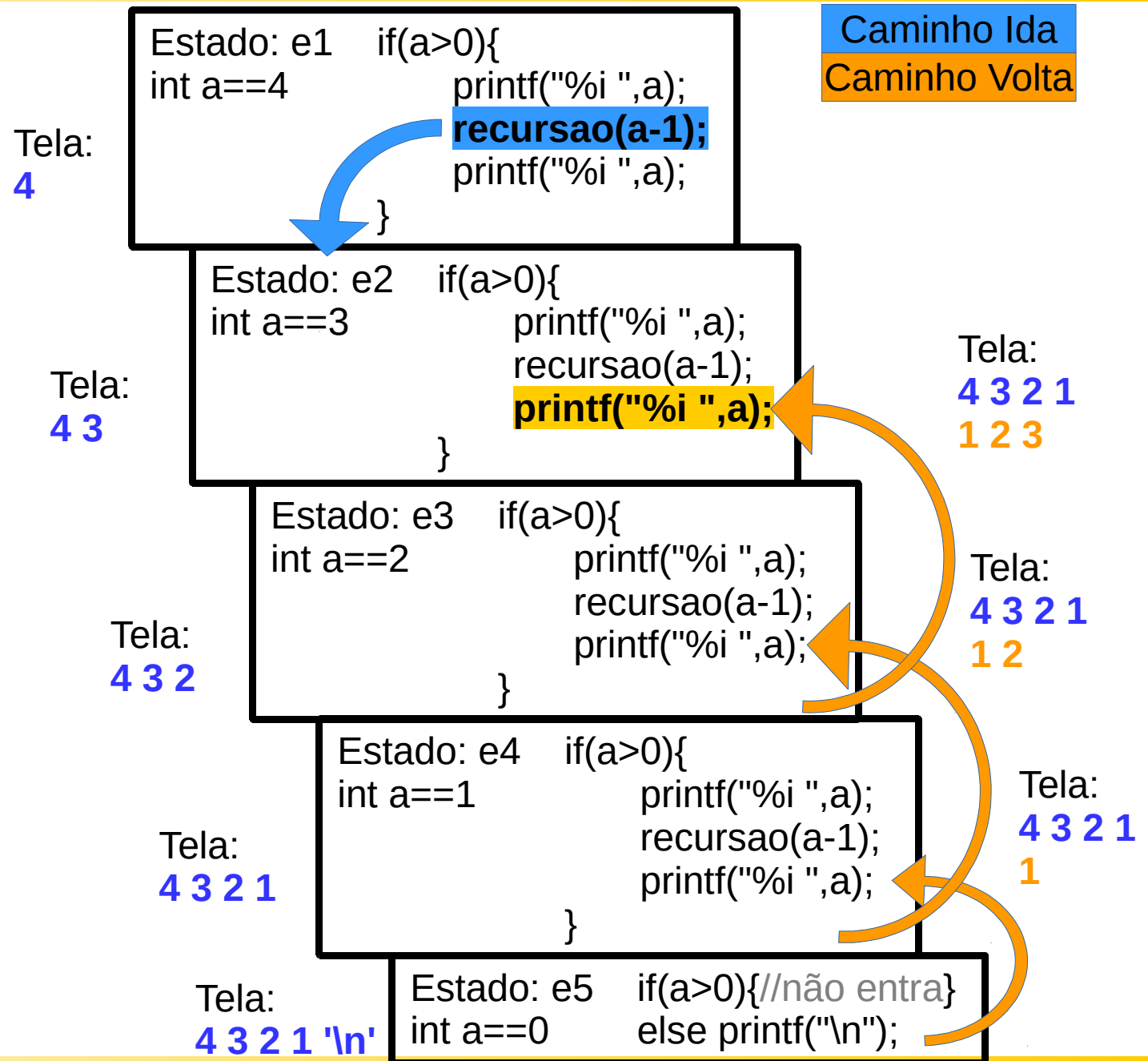
SAÍDA: 4 3 2 1
1 2 3 4



```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

```
int main() {
    recursao(4);
    return 0;
}
```

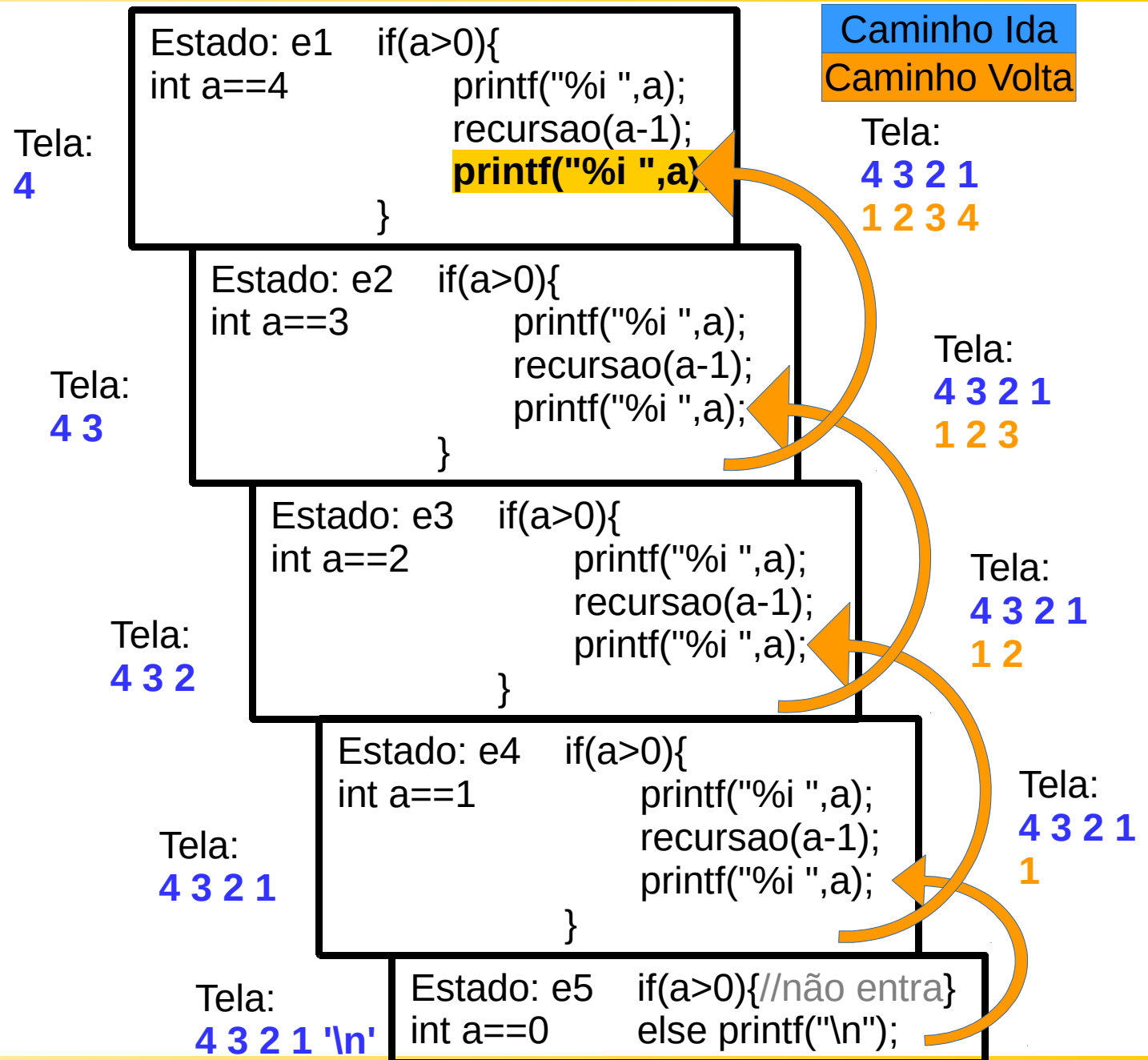
SAÍDA: 4 3 2 1
1 2 3 4



```
void recursao(int a) {
    if (a>0) {
        printf("%i ",a);
        recursao(a-1);
        printf("%i ", a);
    }
    else printf("\n");
}
```

```
int main() {
    recursao(4);
    return 0;
}
```

SAÍDA: 4 3 2 1
1 2 3 4



main() com x==5
mostra_crescente(x)

Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {  
    if (inicio > 0) {  
        printf("%i ", mostra_crescente(inicio-1));  
        return inicio;  
    }  
    else return 0;  
}
```

```
int main( ) {  
    int x=5;  
    printf("%i\n", mostra_crescente(x));  
    return 0;  
}
```

SAÍDA: 0 1 2 3 4 5

Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {  
    if (inicio > 0) {  
        printf("%i ", mostra_crescente(inicio-1));  
        return inicio;  
    }  
    else return 0;  
}
```

```
int main( ) {  
    int x=5;  
    printf("%i\n", mostra_crescente(x));  
    return 0;  
}
```

SAÍDA: 0 1 2 3 4 5

main() com x==5
mostra_crescente(x)



Estado: e0 , int inicio==5
mostra_crescente(inicio-1)
return inicio

Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}
```

```
int main( ) {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5

main() com x==5
mostra_crescente(x)

Estado: e0 , int inicio==5
mostra_crescente(inicio-1)
return inicio

Estado: e1 , int inicio==4
mostra_crescente(inicio-1)
return inicio

Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}
```

```
int main( ) {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5

main() com x==5
mostra_crescente(x)

Estado: e0 , int inicio==5
mostra_crescente(inicio-1)
return inicio

Estado: e1 , int inicio==4
mostra_crescente(inicio-1)
return inicio

Estado: e2 , int inicio==3
mostra_crescente(inicio-1)
return inicio

Função recursiva com valor de retorno

```

int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
    
```

SAÍDA: 0 1 2 3 4 5

main() com x==5
mostra_crescente(x)

Estado: e0 , int inicio==5
mostra_crescente(inicio-1)
return inicio

Estado: e1 , int inicio==4
mostra_crescente(inicio-1)
return inicio

Estado: e2 , int inicio==3
mostra_crescente(inicio-1)
return inicio

Estado: e3 , int inicio==2
mostra_crescente(inicio-1)
return inicio

Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5

main() com x==5
mostra_crescente(x)

Estado: e0 , int inicio==5
mostra_crescente(inicio-1)
return inicio

Estado: e1 , int inicio==4
mostra_crescente(inicio-1)
return inicio

Estado: e2 , int inicio==3
mostra_crescente(inicio-1)
return inicio

Estado: e3 , int inicio==2
mostra_crescente(inicio-1)
return inicio

Estado: e4 , int inicio==1
mostra_crescente(inicio-1)
return inicio

Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5

main() com x==5
mostra_crescente(x)

Estado: e0 , int inicio==5
mostra_crescente(inicio-1)
return inicio

Estado: e1 , int inicio==4
mostra_crescente(inicio-1)
return inicio

Estado: e2 , int inicio==3
mostra_crescente(inicio-1)
return inicio

Estado: e3 , int inicio==2
mostra_crescente(inicio-1)
return inicio

Estado: e4 , int inicio==1
mostra_crescente(inicio-1)
return inicio

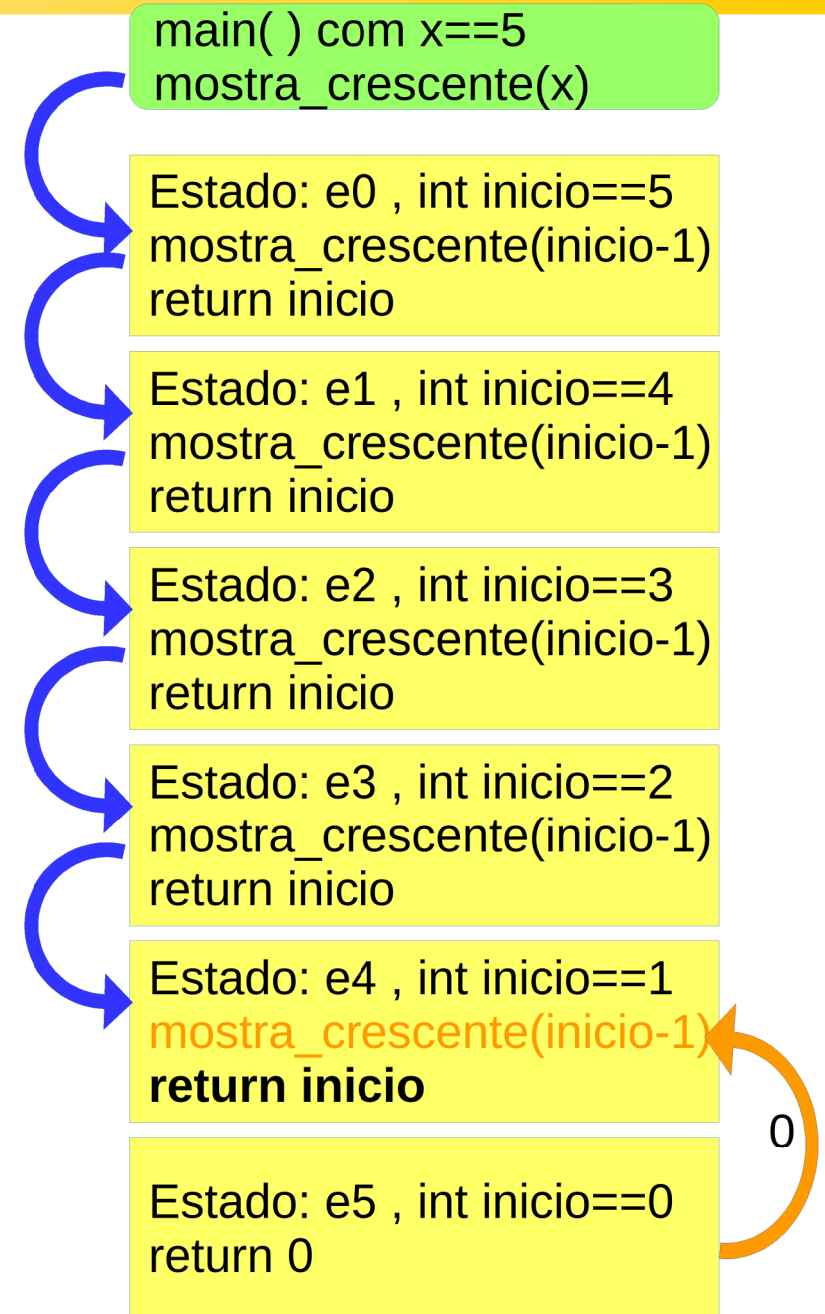
Estado: e5 , int inicio==0
return 0

Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5



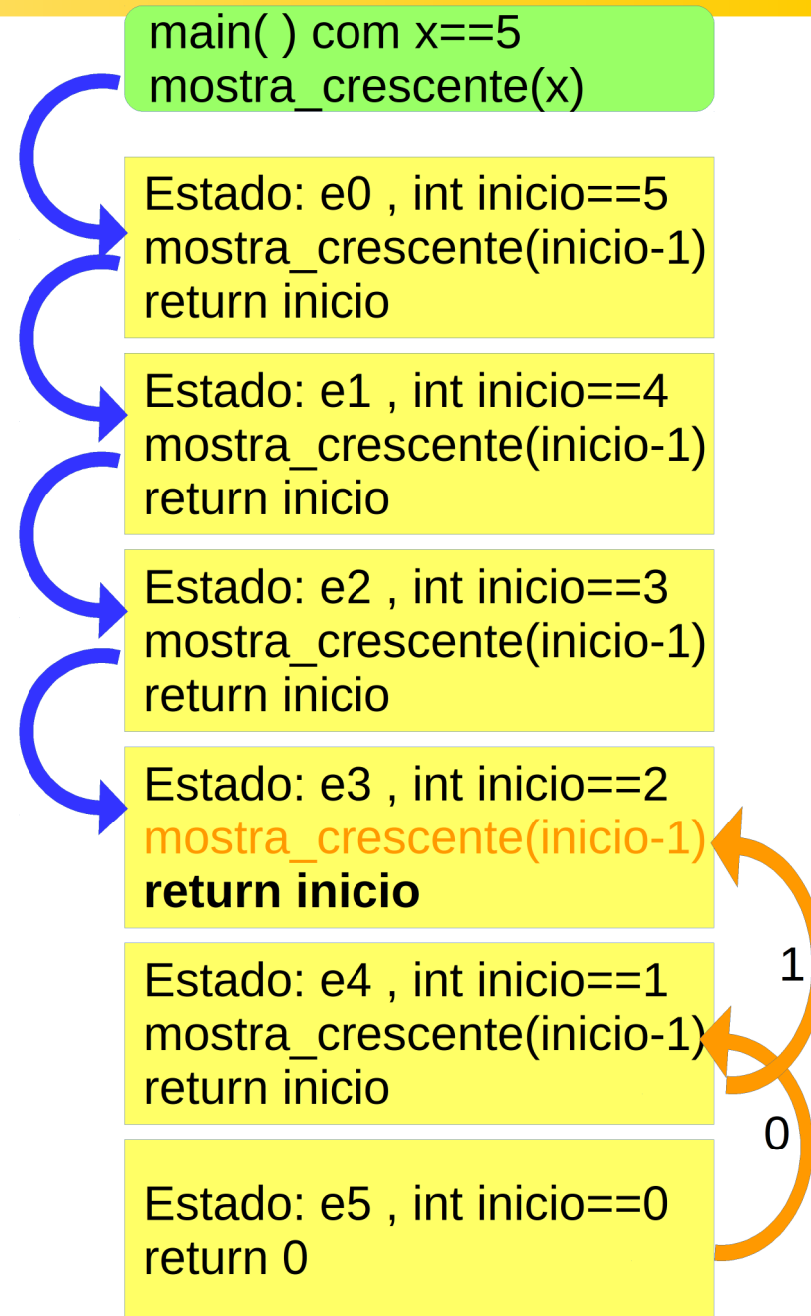
Função recursiva com valor de retorno

```

int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
  
```

SAÍDA: 0 1 2 3 4 5

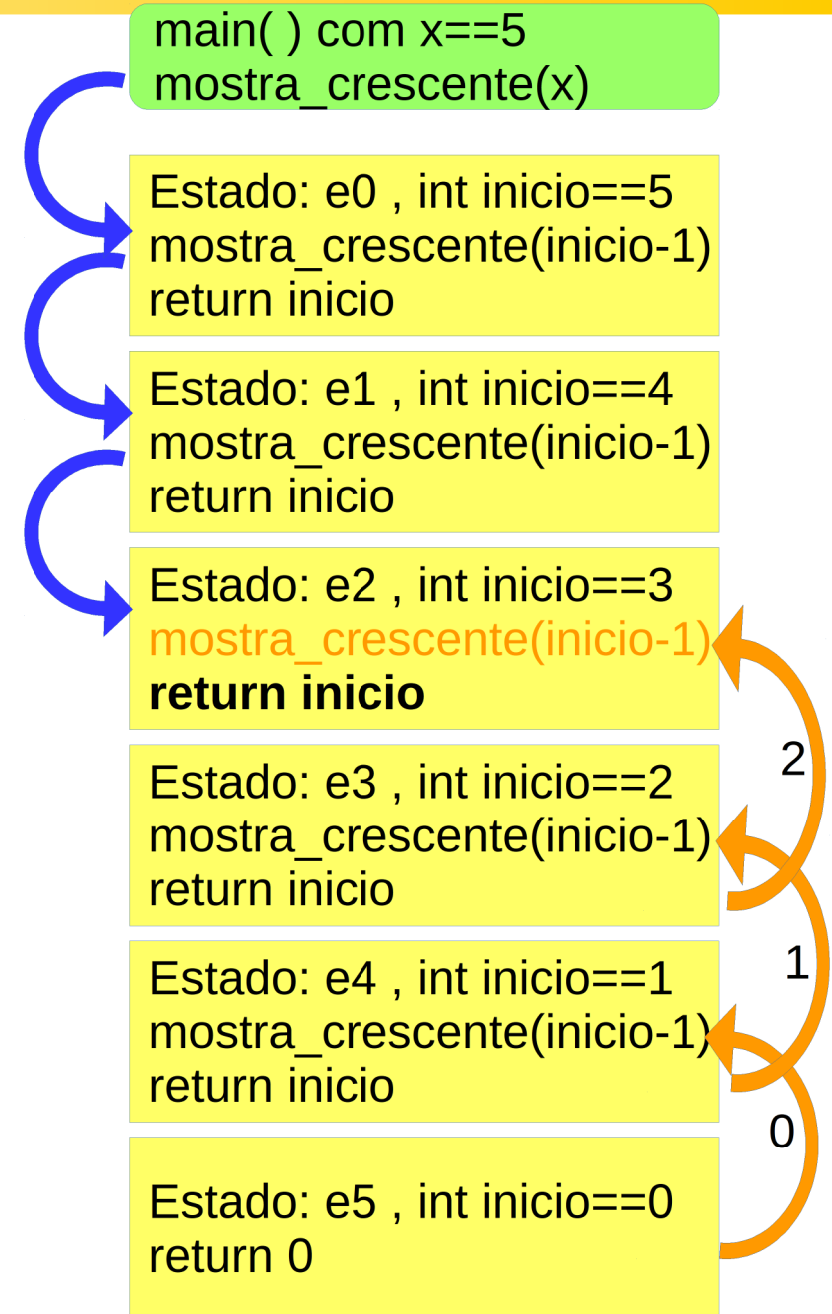


Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5

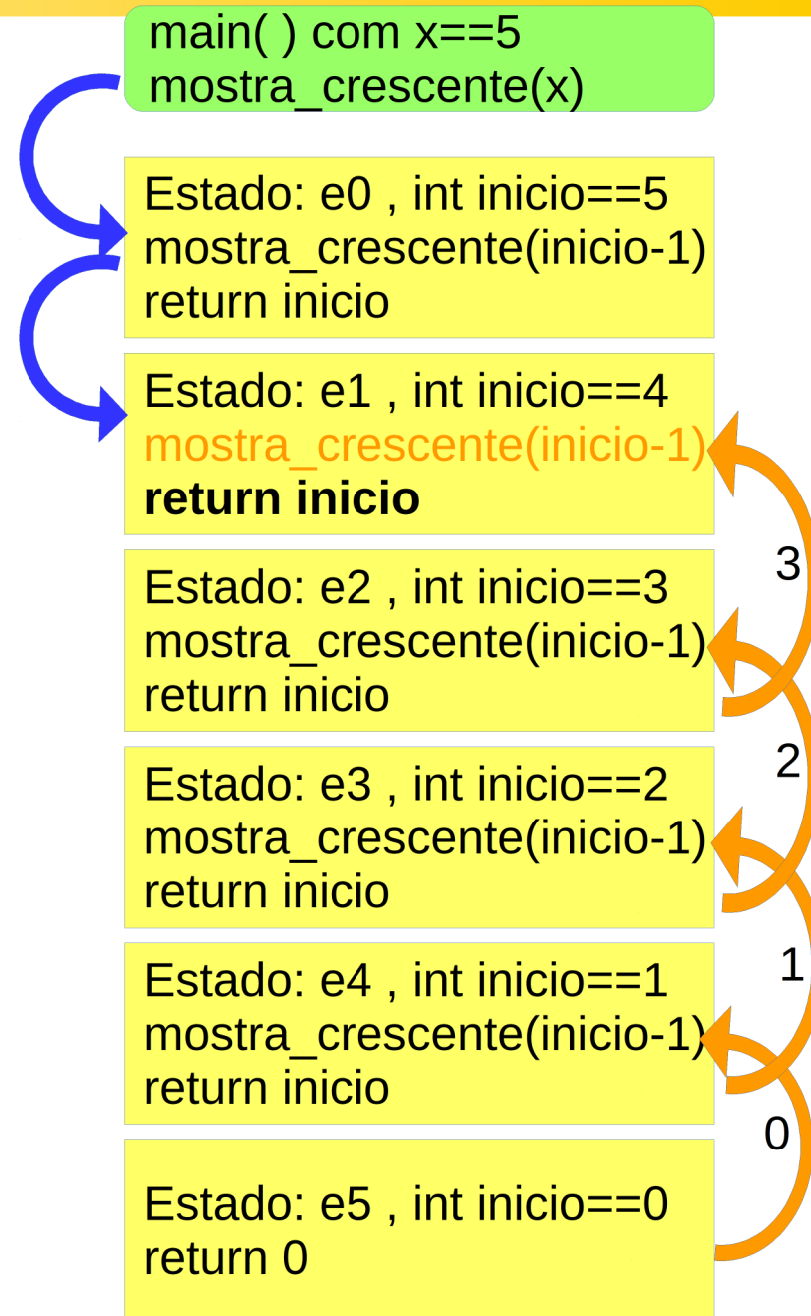


Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5

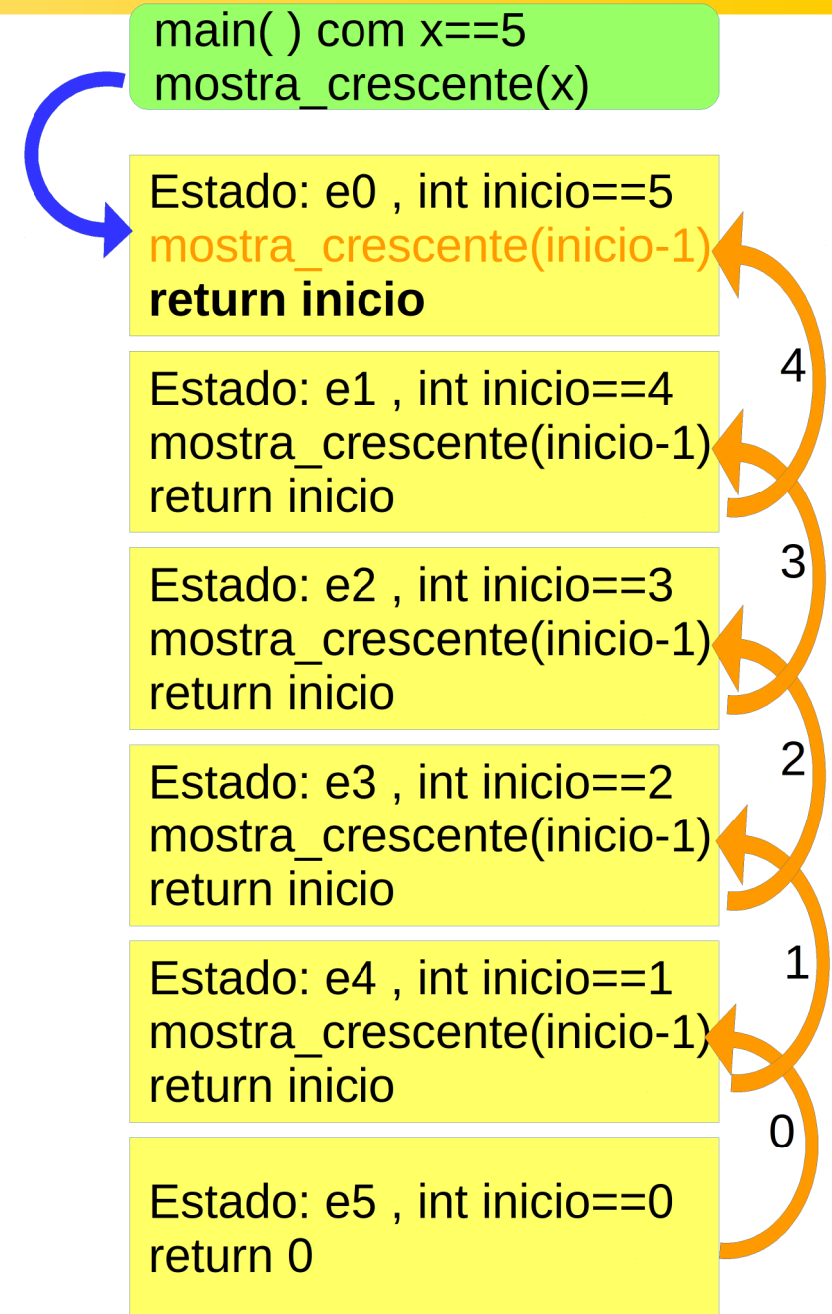


Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5



Função recursiva com valor de retorno

```
int mostra_crescente(int inicio) {
    if (inicio > 0) {
        printf("%i ", mostra_crescente(inicio-1));
        return inicio;
    }
    else return 0;
}

int main() {
    int x=5;
    printf("%i\n", mostra_crescente(x));
    return 0;
}
```

SAÍDA: 0 1 2 3 4 5

main() com x==5
mostra_crescente(x)

Estado: e0 , int inicio==5
mostra_crescente(inicio-1)
return inicio

Estado: e1 , int inicio==4
mostra_crescente(inicio-1)
return inicio

Estado: e2 , int inicio==3
mostra_crescente(inicio-1)
return inicio

Estado: e3 , int inicio==2
mostra_crescente(inicio-1)
return inicio

Estado: e4 , int inicio==1
mostra_crescente(inicio-1)
return inicio

Estado: e5 , int inicio==0
return 0

5

4

3

2

1

0

- Qual o resultado do programa abaixo e por que?

```
int analisar(int num) {  
    if (num == 1)  
        return 1;  
    if (num % 2 == 0)  
        return analisar(num/2);  
    return analisar((num-1)/2) + analisar((num+1)/2);  
}  
  
int main() {  
    printf("%i", analisar(7));  
    return 0;  
}
```

- Utilizando Recursão:

- Crie uma função que receba um número inteiro e gere todo os números em ordem decrescente até 0
- Crie uma função que retorne $x*y$ através de operação de soma. A função recebe x e y por parâmetro
- Crie uma função que retorne x^y através de operação de multiplicação. A função recebe x e y por parâmetro
- Crie uma função que retorne o fatorial de um número passado por parâmetro. A ideia do fatorial está abaixo:

$$F(n) = \begin{cases} 1 & \text{if } n = 0, n = 1 \\ nF(n-1) & \text{if } n > 1 \end{cases}$$

- Faça uma função recursiva que retorne o n-ésimo termo da sequência de Fibonacci, sendo que n é recebido por parâmetro. Utilize essa função para desenvolver um programa que mostre no main() os n termos dessa sequência na tela, a partir do valor de n recebido pelo teclado. Sabe-se que o 1º termo é 0 e o 2º termo é 1.