

Estruturas condicionais

Prof. Ricardo Frohlich da Silva

Estrutura condicional

- Permite que uma condição seja testada
- O resultado do teste desta condição pode ser:
 - Verdadeiro;
 - Falso;
- Dependendo do resultado, um conjunto de instruções é executado.

Comando if-else - Sintaxe

- sem opção para o caso da condição ser falsa:

```
if (condição){  
    <conjunto de instruções>  
}
```

- com opção para o caso da condição ser falsa:

```
if (condição){  
    <conjunto de instruções 1>  
}  
else {  
    <conjunto de instruções 2>  
}
```

Exemplo 1: informar se número é negativo

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());

    if (num > 0)
    {
        Console.WriteLine("Positivo!");
    }
}
```

Exemplo 1: informar se número é negativo

- Problemas do código anterior???
- E se digitarmos um número negativo?

Exemplo 2: informar se número é positivo ou negativo

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());

    if (num > 0)
    {
        Console.WriteLine("Positivo!");
    }
    else
    {
        Console.WriteLine("Negativo!");
    }
}
```

Exemplo 2: informar se número é negativo

- Problemas do código anterior???
- E se digitarmos ZERO?

Exemplo 3v2: informar se número é positivo, negativo ou zero

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());

    if (num > 0)
    {
        Console.WriteLine("Positivo!");
    }
    else if (num < 0)
    {
        Console.WriteLine("Negativo!");
    }
    else
    {
        Console.WriteLine("Zero!");
    }
}
```


Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
}
```

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());

    Como testar se
      num ele é
    divisível por 2?

}
```

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
}
```

$$\begin{array}{r} 15 \overline{) 2} \\ 14 \\ \hline 1 \end{array}$$

Não é divisível!

$$\begin{array}{r} 24 \overline{) 2} \\ 24 \\ \hline 0 \end{array}$$

É divisível!!

conclusão



Se o resto da divisão do número por **2** for igual a 0, então é divisível!

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("Divisível por 2");
    }
}
```

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("Divisível por 2");
    }
}
```

- E se for divisível por 4?
- E se for divisível por 8?
- Um número pode ser divisível por 4 e por 8 ao mesmo tempo?
- Um número pode ser divisível por 2, por 4 e por 8 ao mesmo tempo?
 - Como completamos o código??

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("Divisível por 2");
    }
    else if (num % 4 == 0)
    {
        Console.WriteLine("Divisível por 4");
    }
    else if (num % 8 == 0)
    {
        Console.WriteLine("Divisível por 8");
    }
}
```

OPÇÃO 1?

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("Divisível por 2");
    }
    else if (num % 4 == 0)
    {
        Console.WriteLine("Divisível por 4");
    }
    else
    {
        Console.WriteLine("Divisível por 8");
    }
}
```

OPÇÃO 2?

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("Divisível por 2");
    }
    if (num % 4 == 0)
    {
        Console.WriteLine("Divisível por 4");
    }
    if (num % 8 == 0)
    {
        Console.WriteLine("Divisível por 8");
    }
}
```

OPÇÃO 3?

• • •

Exemplo 4: informar se número é divisível por 2, 4 ou 8.

```
static void Main(string[] args)
{
    int num;
    Console.WriteLine("Digite um numero:");
    num = int.Parse(Console.ReadLine());
    if (num % 2 == 0)
    {
        Console.WriteLine("Divisível por 2");
    }
    if (num % 4 == 0)
    {
        Console.WriteLine("Divisível por 4");
    }
    if (num % 8 == 0)
    {
        Console.WriteLine("Divisível por 8");
    }
}
```

OPÇÃO 3!!

Operadores relacionais

Operador	Ação
>	Maior que
<	Menor que
>=	Maior ou igual a
<=	Menor ou igual a
==	Igual a
!=	Diferente de

Estruturas de repetição

Estruturas de repetição

- As estruturas de repetições são bastante importantes pois muitas vezes necessitamos executar um mesmo procedimento diversas vezes.
- A linguagem C basicamente possui três tipos de estrutura de repetição:
 - **for**, **while** e **do... while**

For

- O comando **for** permite implementar laços de repetição, ou seja, definir trechos do código de um programa que se repetem.
- O **for**, como qualquer iteração, necessita de uma variável para controlar a quantidade de loops (voltas) a serem executadas.
- O comando **for** inclui na sua própria definição uma variável contadora:
 - A variável que vai definir quantas vezes aquele trecho de código será executado.

For

- No comando **for**, a variável contadora deverá ser inicializada com um valor, e ser incrementada ou decrementada a cada loop;
- Além disso, deveremos ter uma condição, testando se a quantidade de vezes que o laço deveria ser executado chegou ao fim ou não (dependendo do valor atual da variável contadora);

For

- Exemplo:
 - executando um laço de repetição 4 vezes.
 - No **for**, inicializamos uma variável contadora **int i**, cujo valor inicial é **0**;
 - A cada loop/laço a variável é incrementada em **1**;
 - Após cada incremento, é testada uma condição para verificar se a variável chegou ao fim, ou seja, se o seu valor é igual a **4**:
 - Caso o valor tenha chegado a **4**, o laço termina;
 - Caso contrário, o laço se repete novamente.

For

- Portanto, o comando **for** necessita de 3 partes:

```
for(valor inicial; condição; incremento ou decremento){  
    <bloco de instruções>;  
}
```

- Em que:
 - **valor inicial**: definimos um valor inicial para nossa variável contadora, que no caso do exemplo anterior, definimos que *i* vale 1 (*i*=0);
 - **condição**: testamos se o laço chegou ao fim, que no caso do exemplo anterior, se *i* chegou a 4 (*i* < 4);
 - **incremento**: *i* é incrementado em 1, para contar o número de vezes que o laço será repetido (*i*++)
 - **<bloco de instruções>**: bloco que será executado 4 vezes, neste exemplo.

For

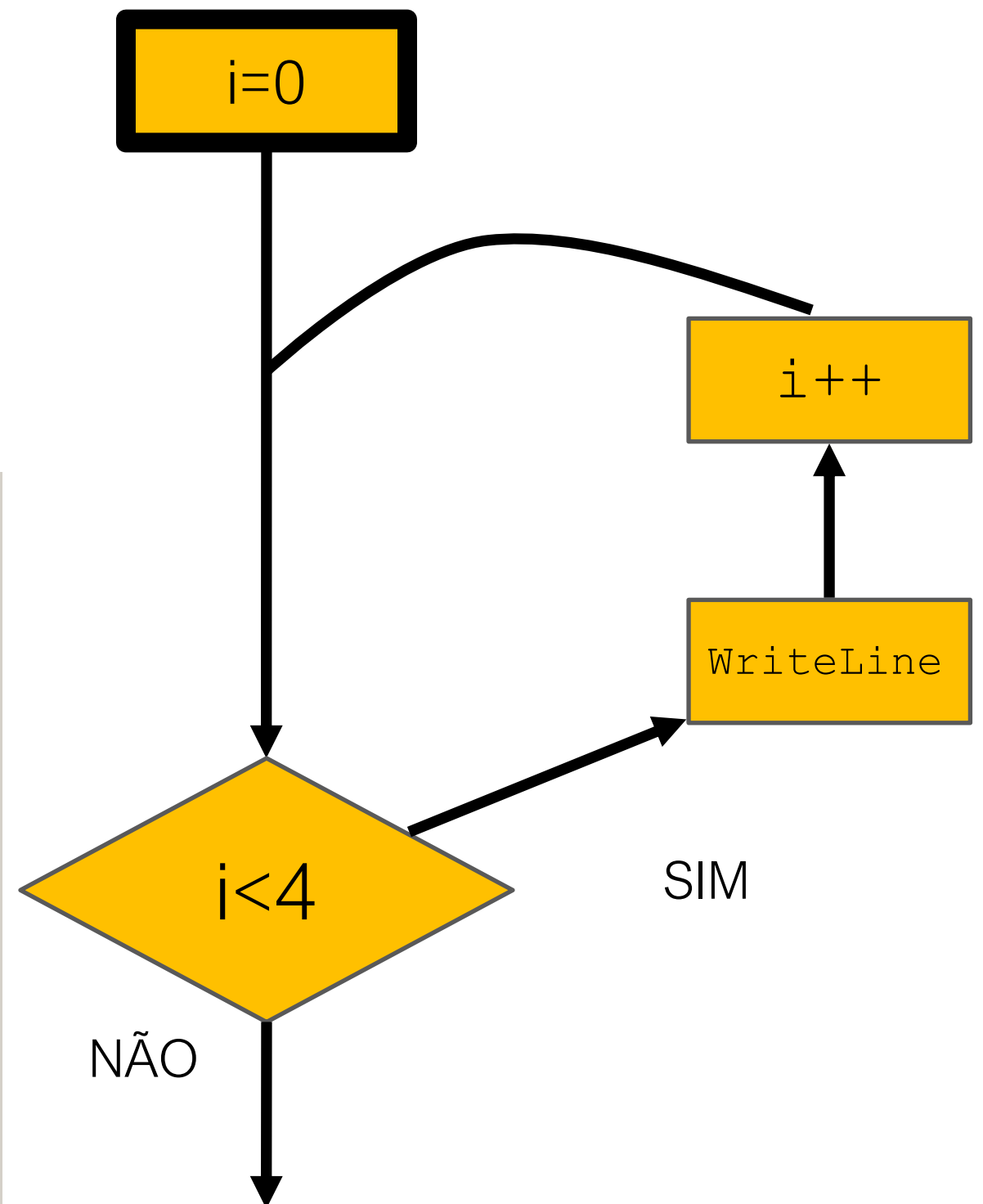
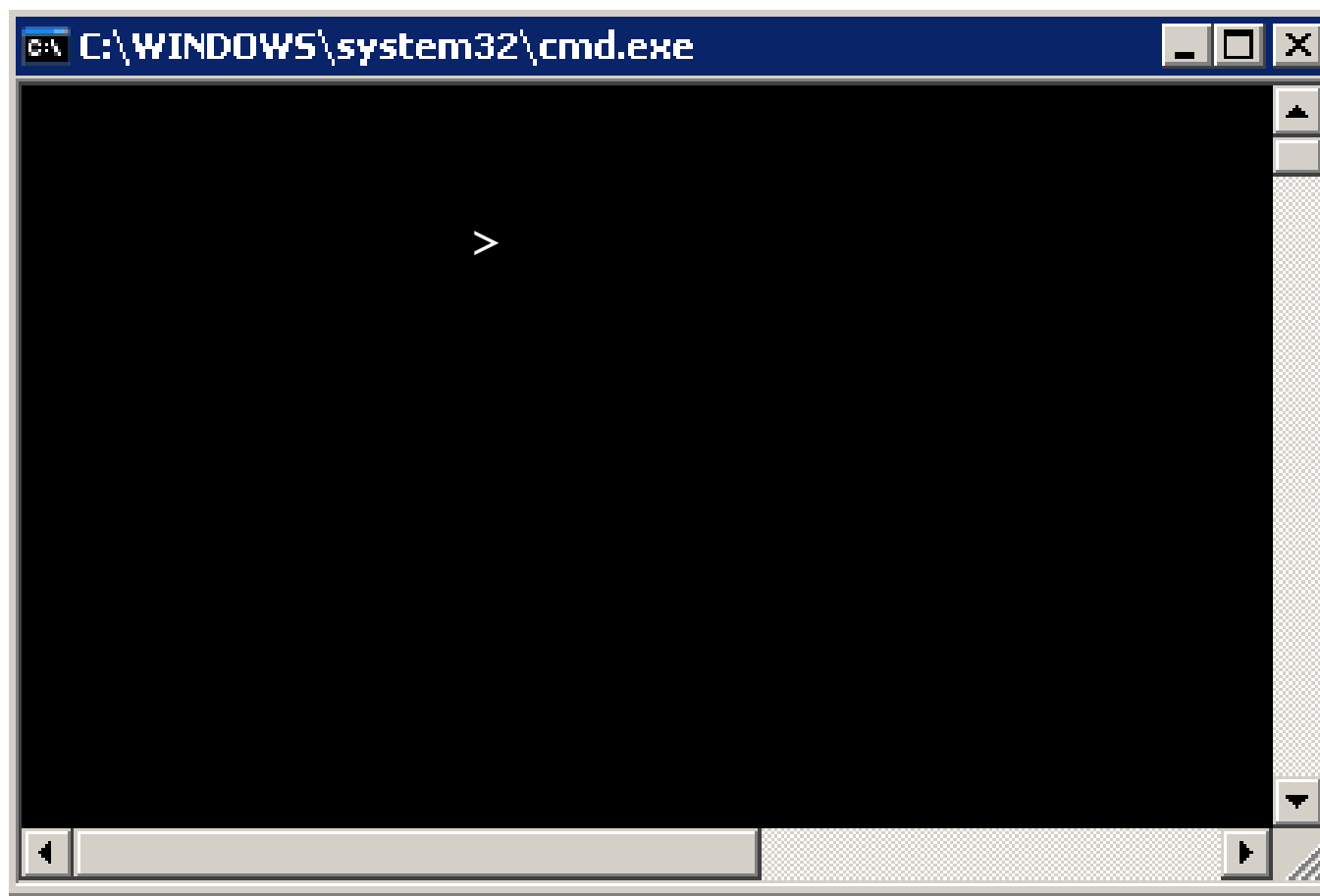
- O comando **for** funciona assim:
 - na primeira iteração, a variável contadora assume o **valor inicial**;
 - Em seguida é testado se a **condição** for verdadeira:
 - Caso seja verdadeira, o **bloco de instruções** é executado.
 - Caso seja falsa, o algoritmo não executa a iteração do laço e continua a execução a partir da primeira instrução após o final **for**.
 - Quando o bloco de instruções chegar ao fim, o controle retorna para a linha do **for**.
 - Daí é executado o **incremento ou decremento**;
 - Em seguida a condição é testada novamente:
 - Caso seja verdadeira, o bloco de instruções é executado.
 - Caso seja falsa, o algoritmo não executa a iteração do laço e continua a execução a partir da primeira instrução após o final **for**.
 - E assim vai...

For

- Desta forma, o exemplo anterior ficaria assim:
- ```
static void Main(string[] args)
```
- ```
{
```
- ```
 int i;
```
- ```
    for (i = 0; i < 4; i++)
```
- ```
 { //i=i+1;
```
- ```
        Console.WriteLine("i vale "+i);
```
- ```
 }
```
- ```
}
```

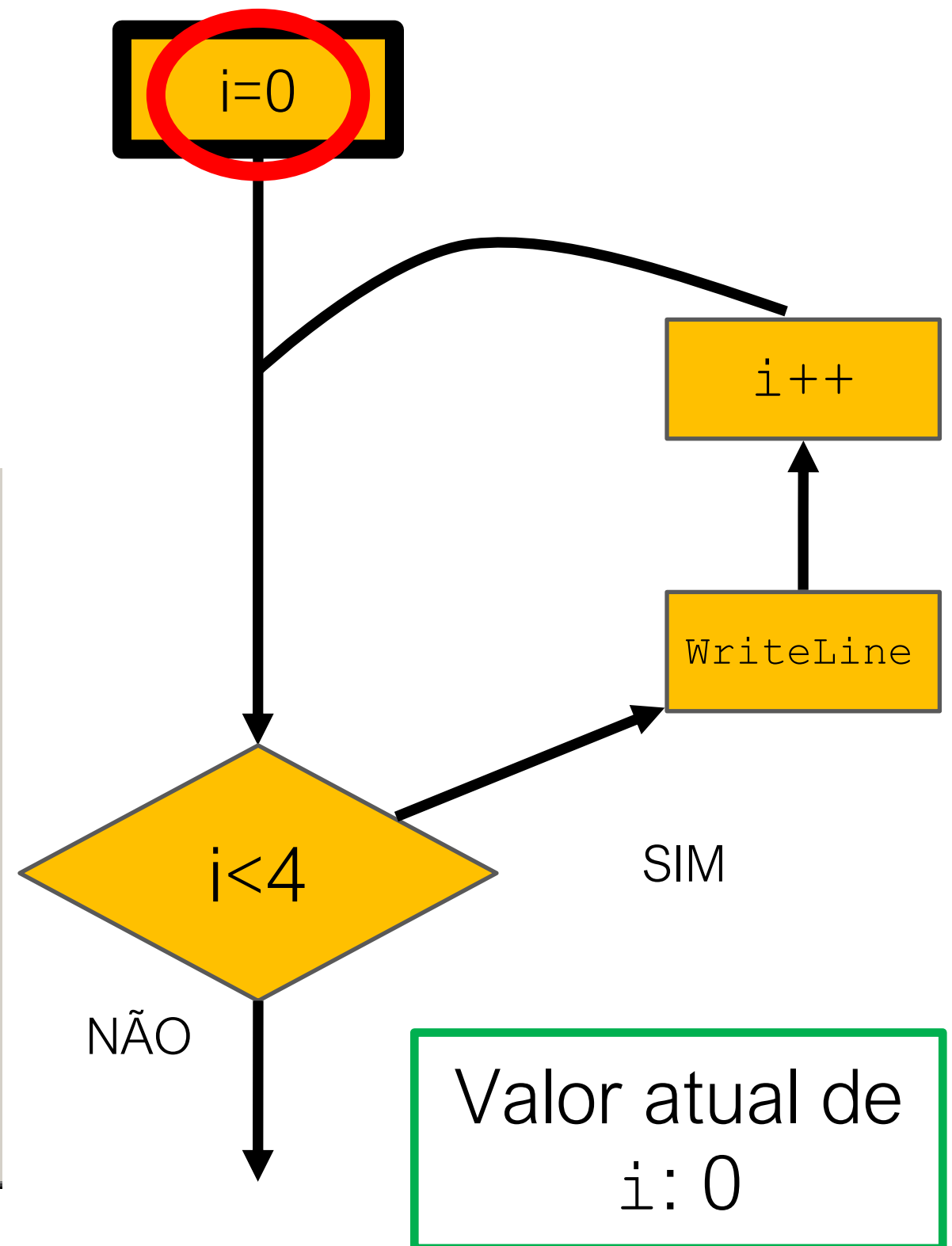
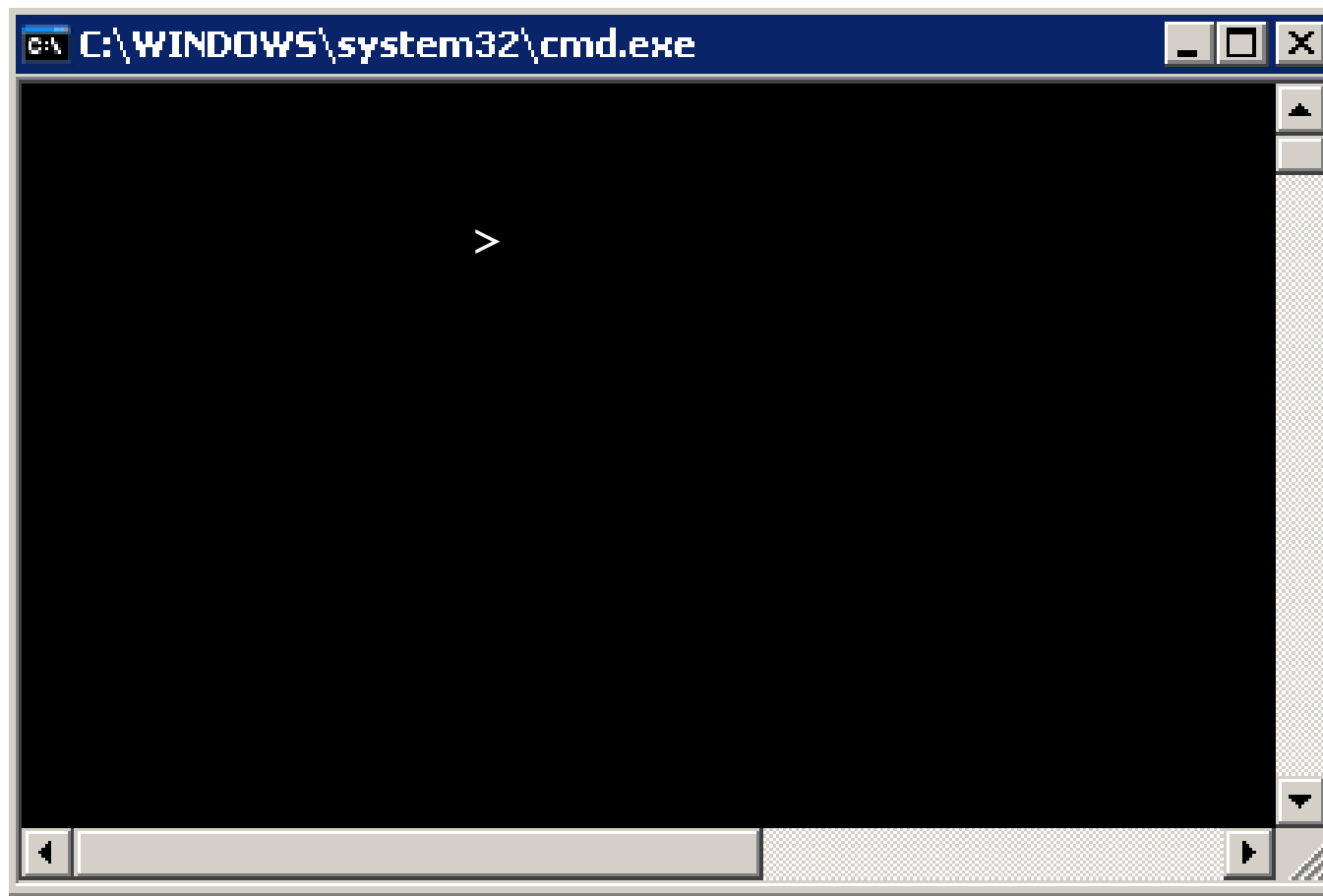
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4; i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



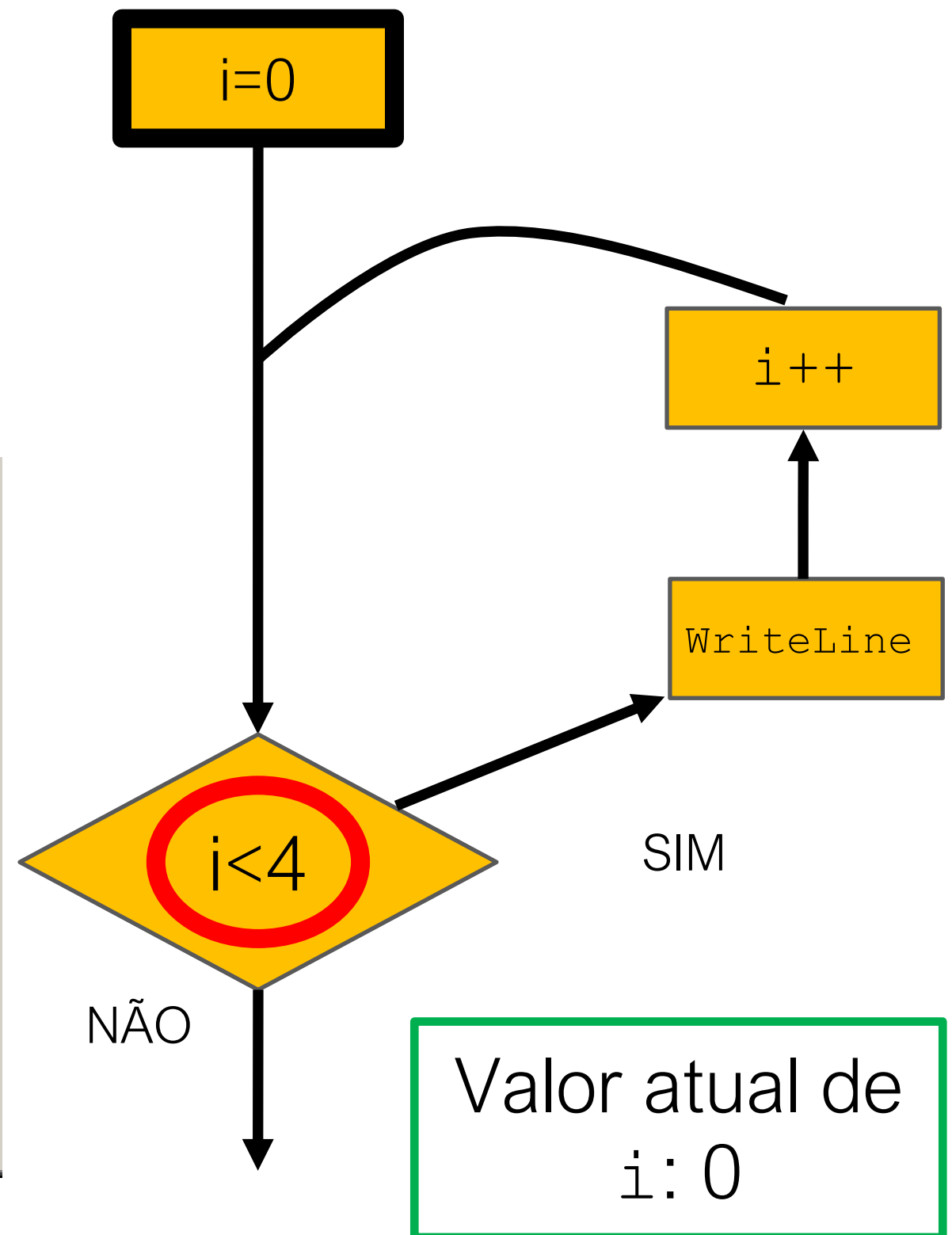
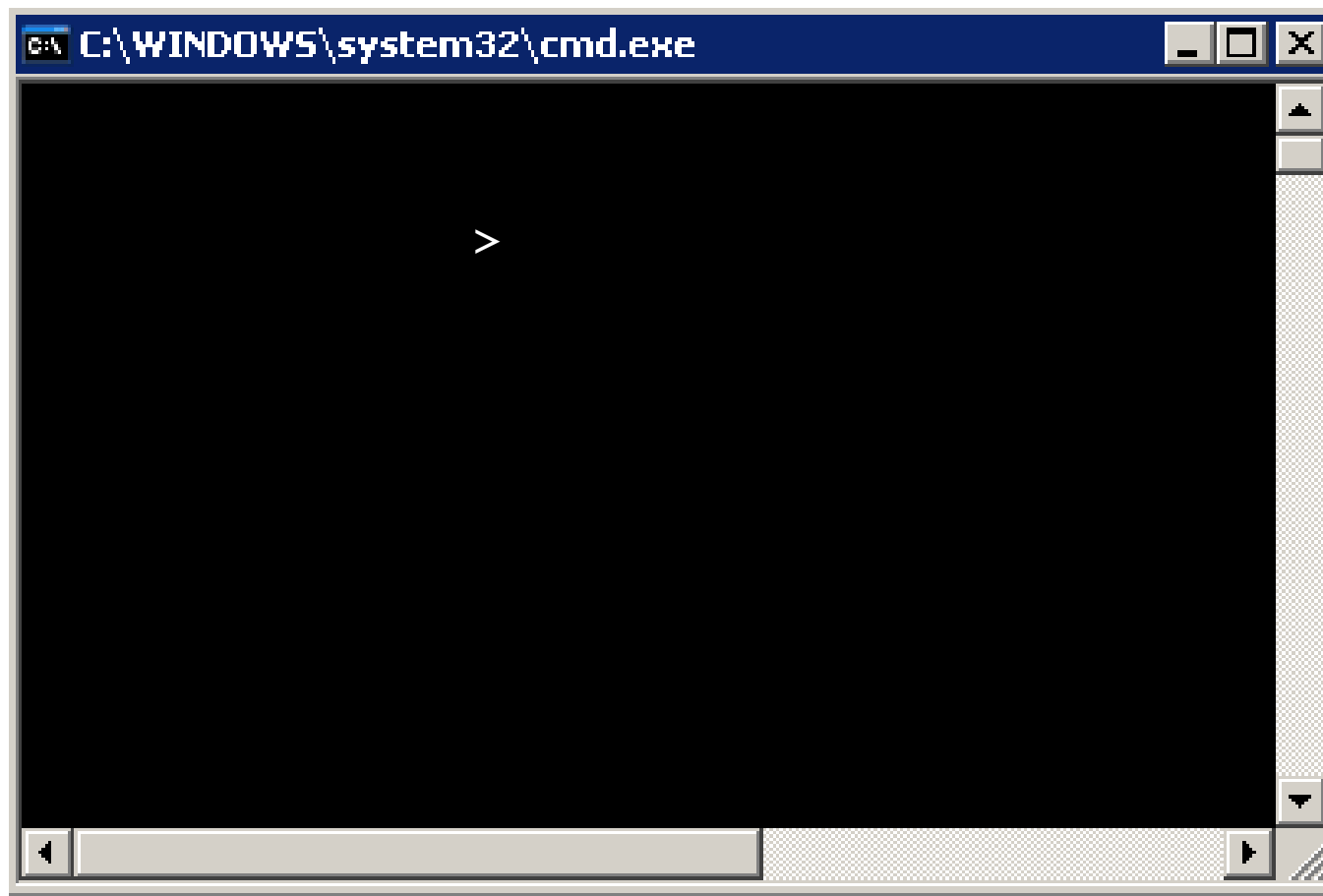
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4; i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



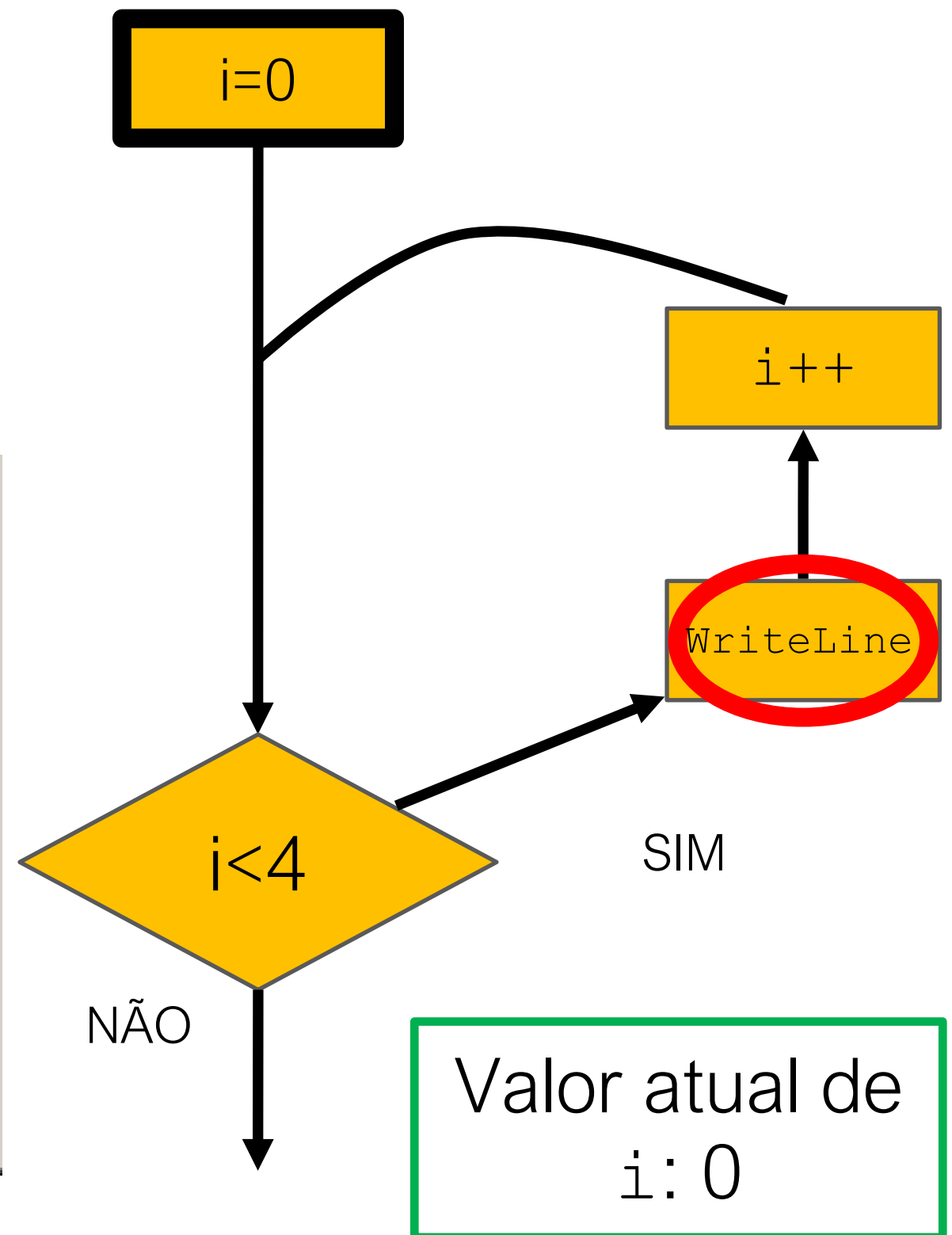
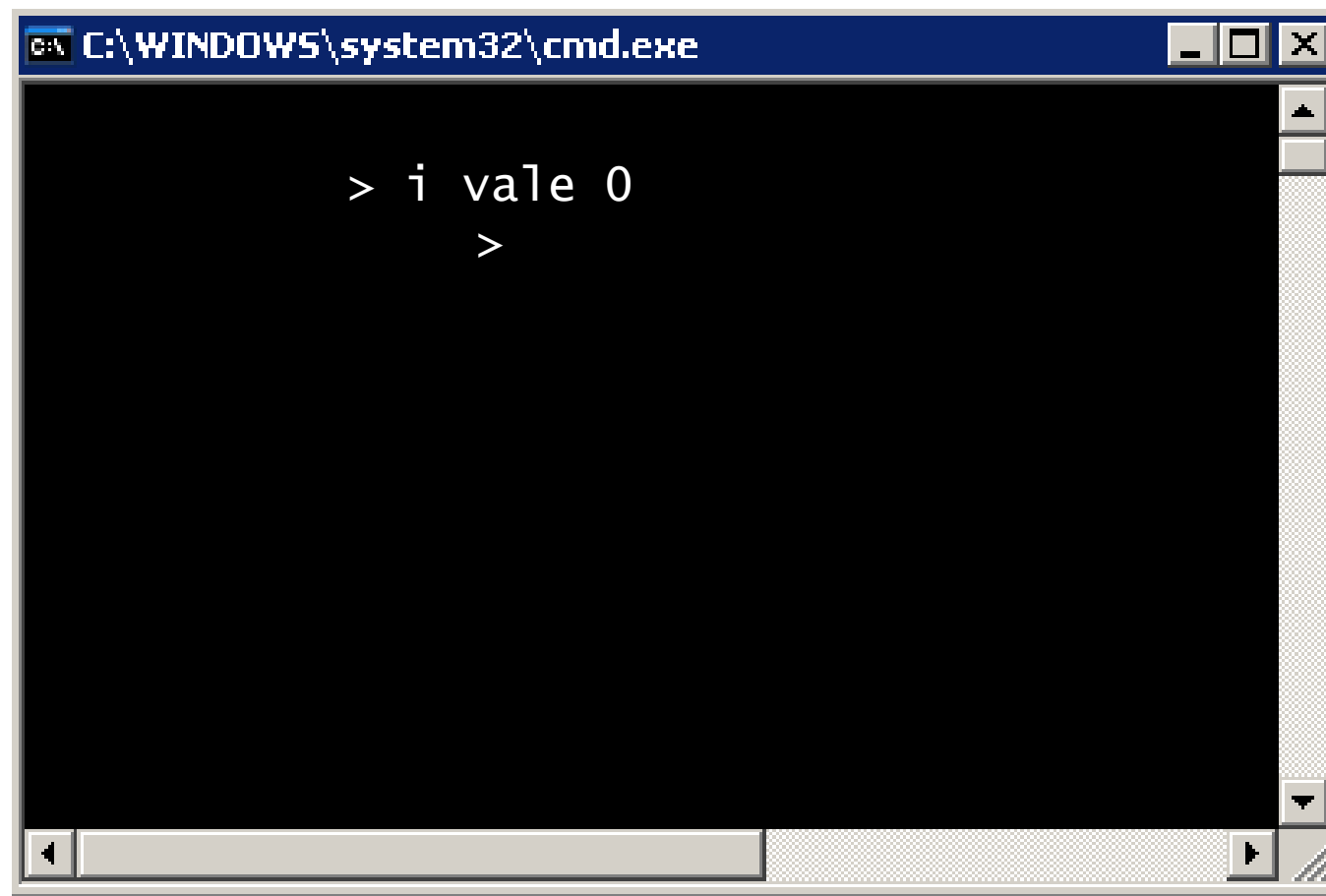
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4 i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



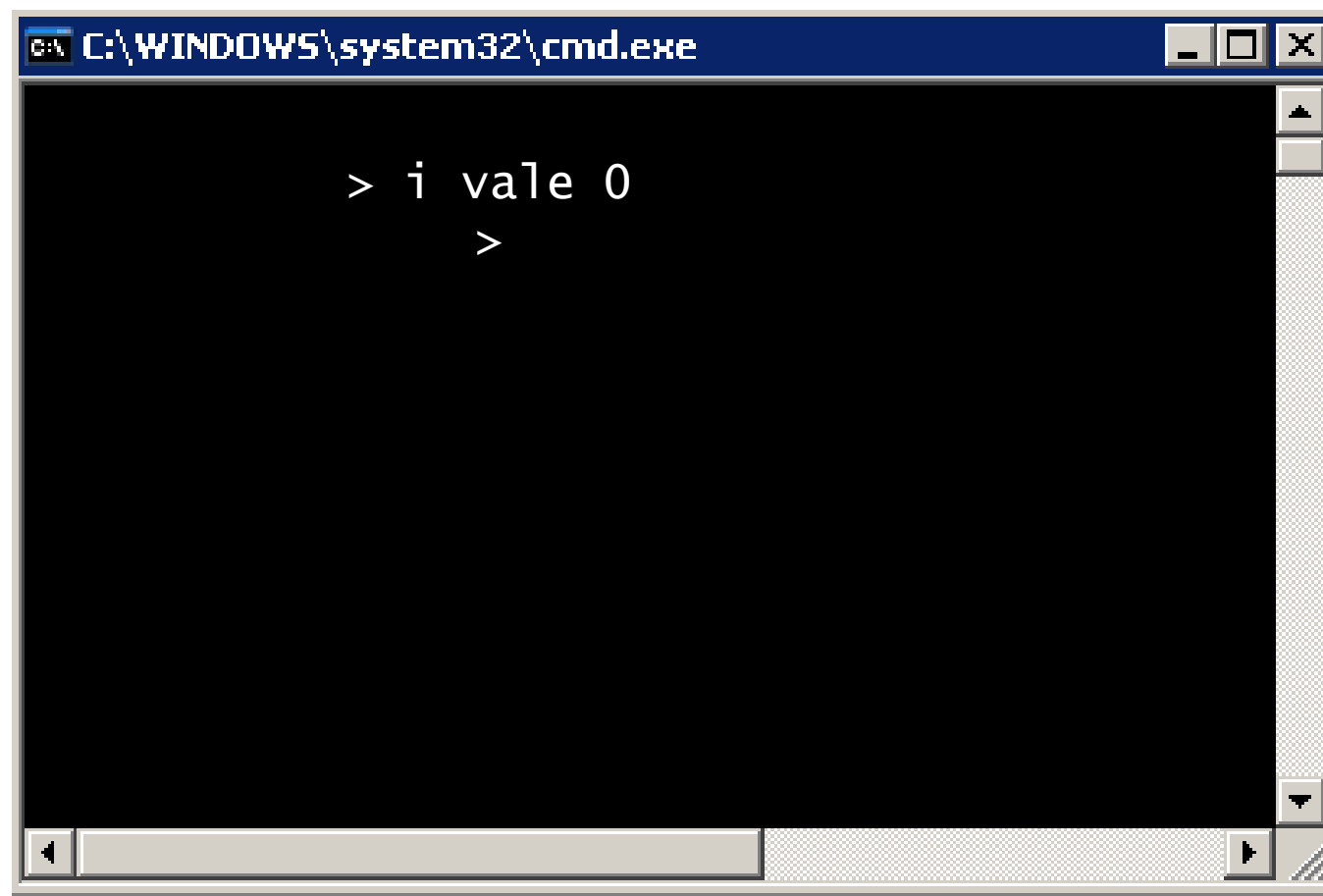
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4; i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```

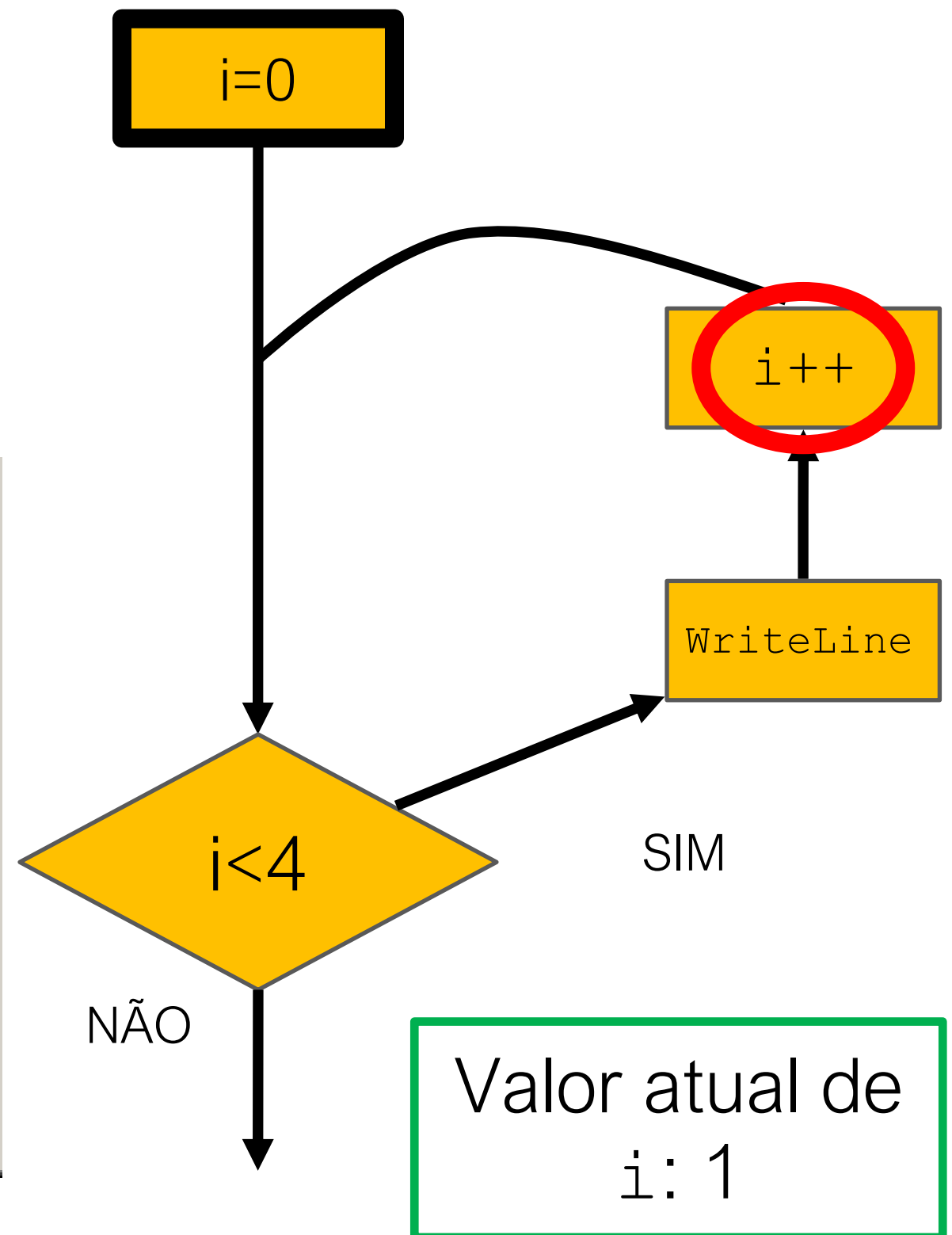


Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4, i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```

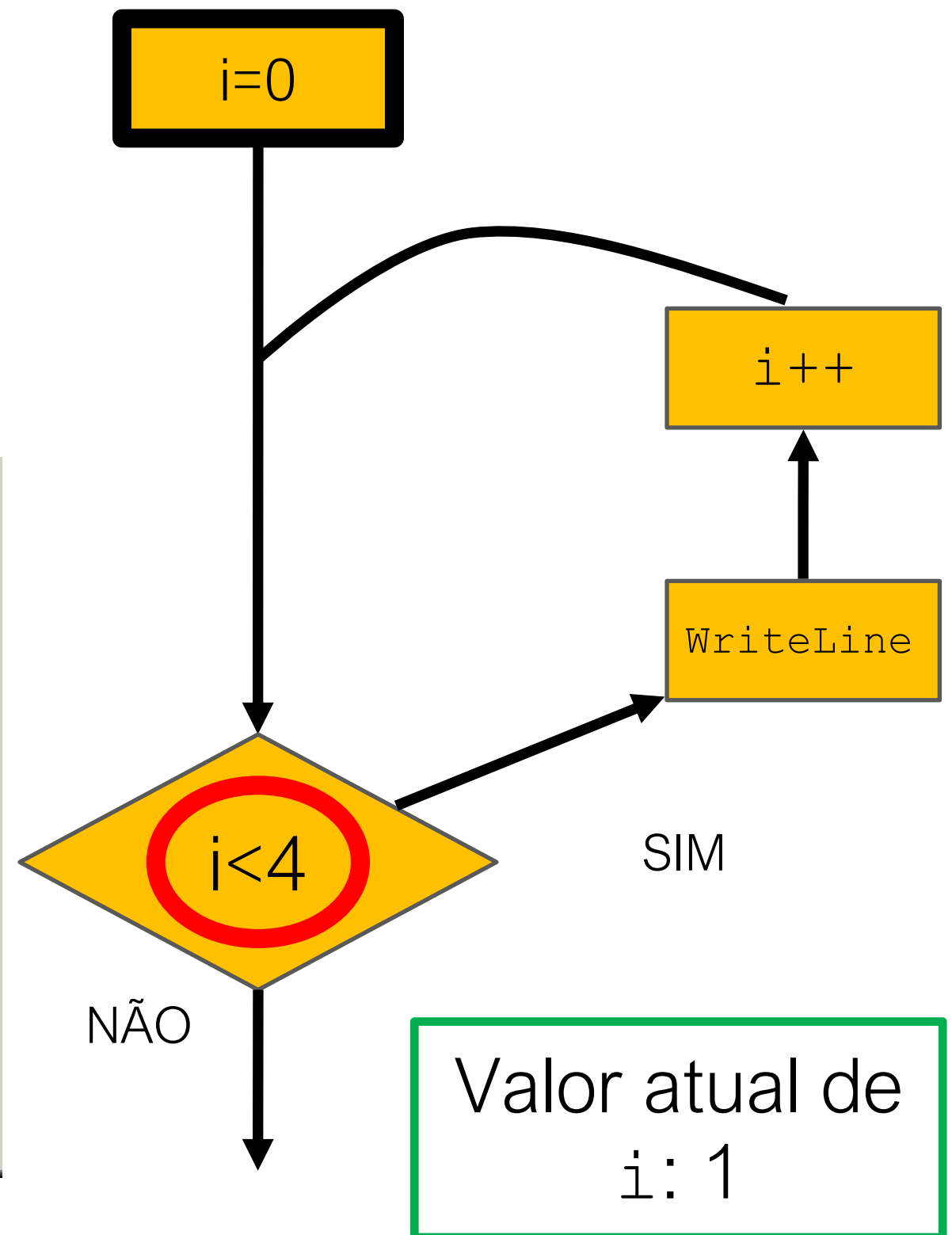
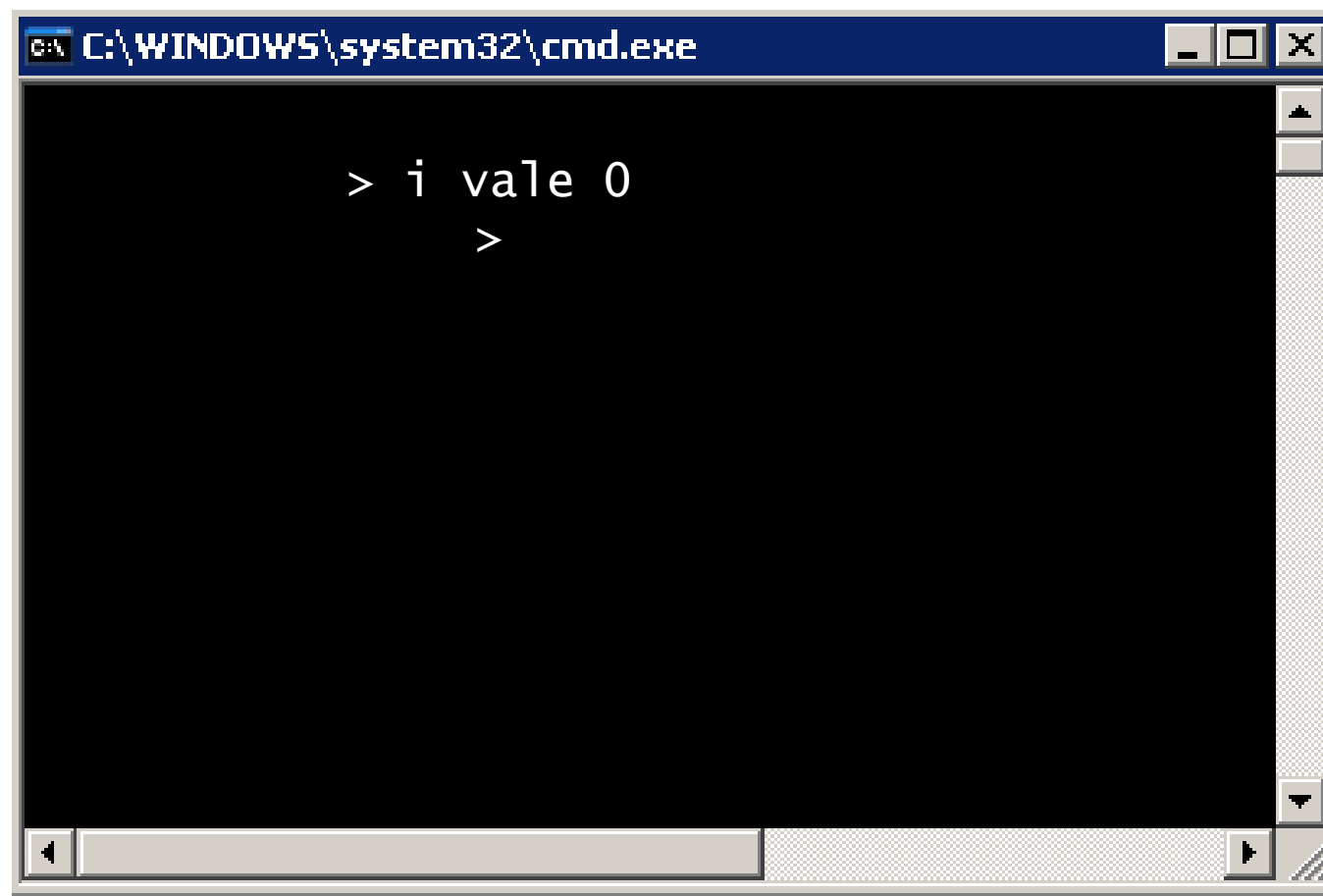


A screenshot of a Windows command prompt window. The title bar shows the path "C:\WINDOWS\system32\cmd.exe". The command prompt displays the output of the program: "> i vale 0" followed by a new line and a prompt ">".



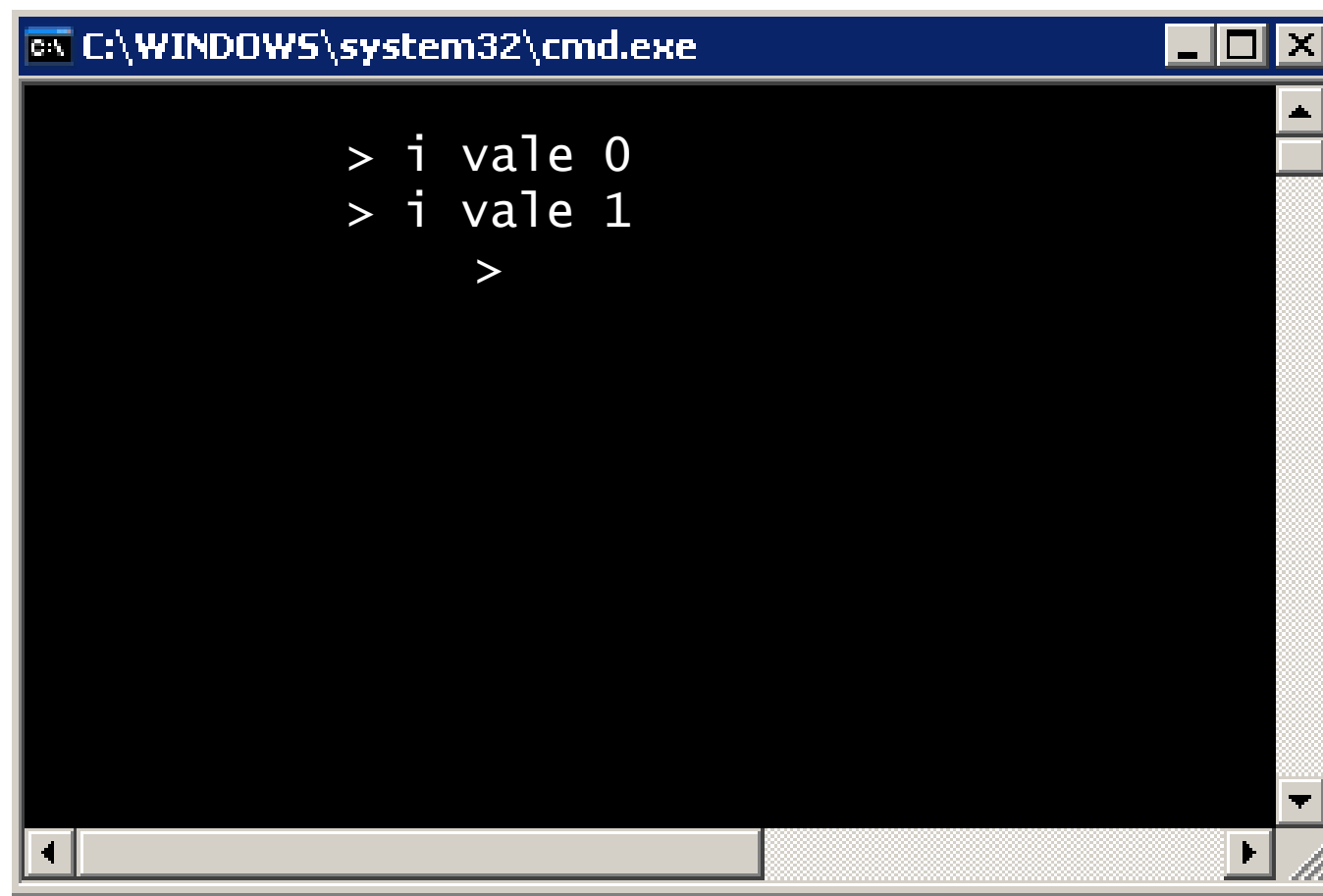
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4 i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



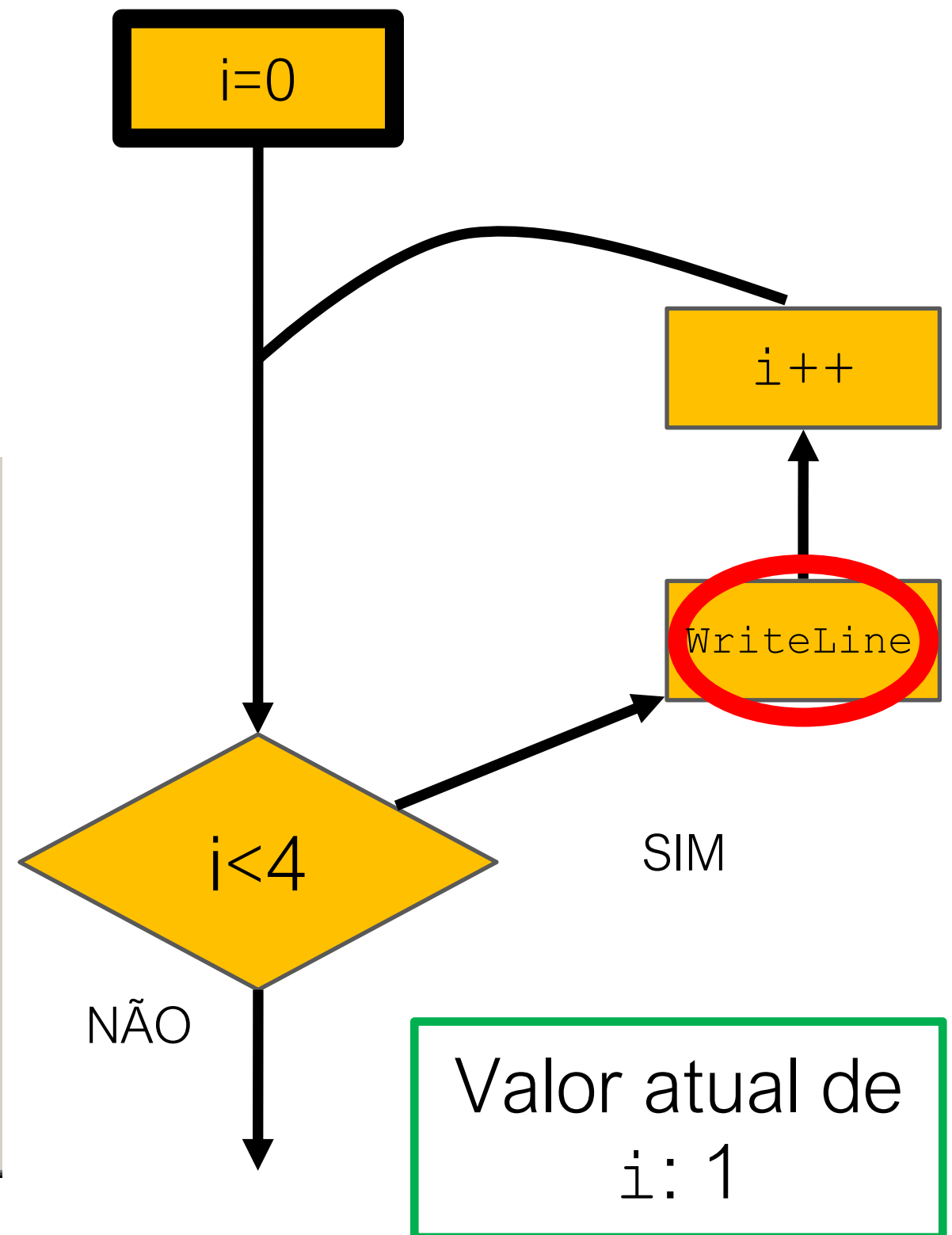
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4; i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



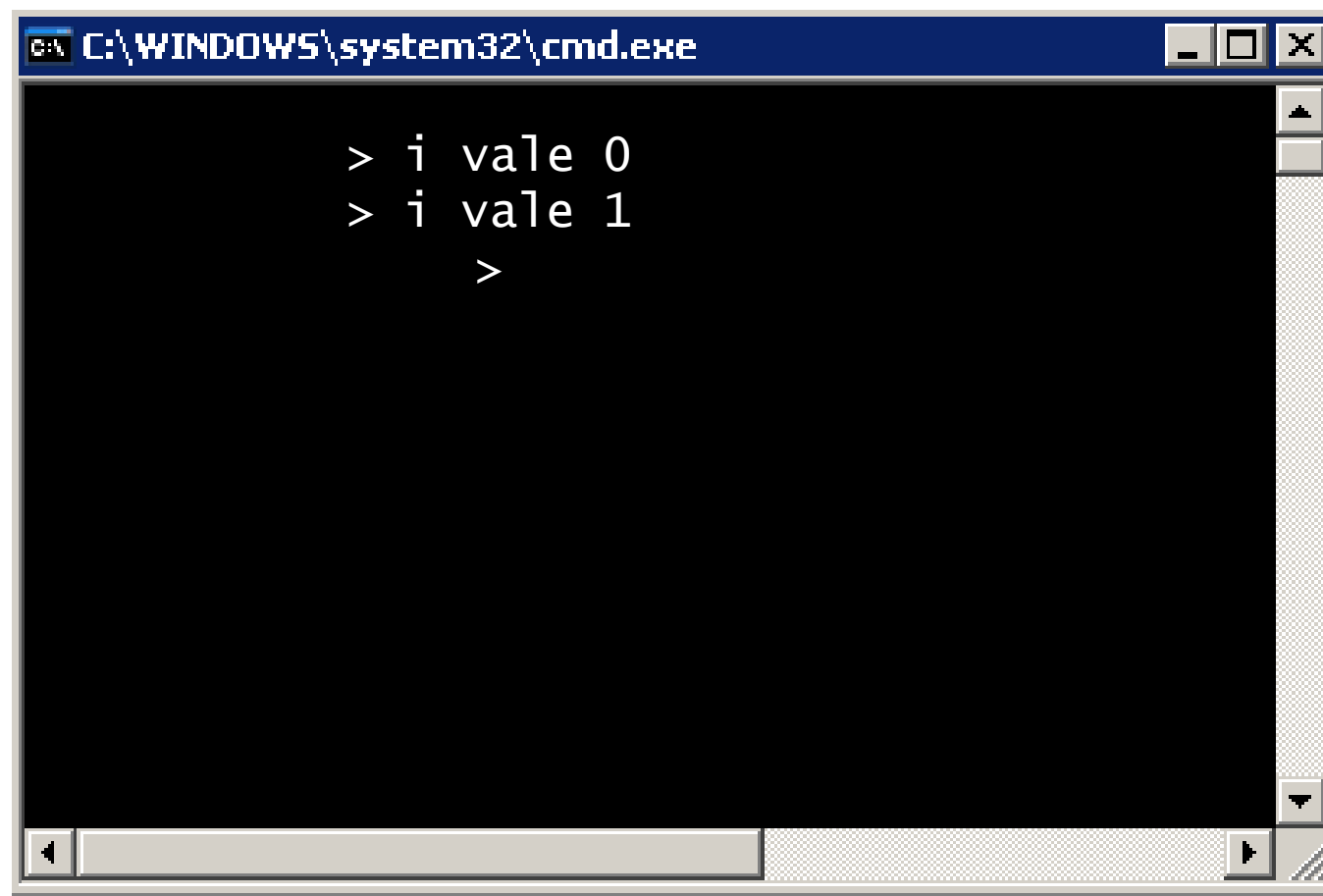
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
>
```



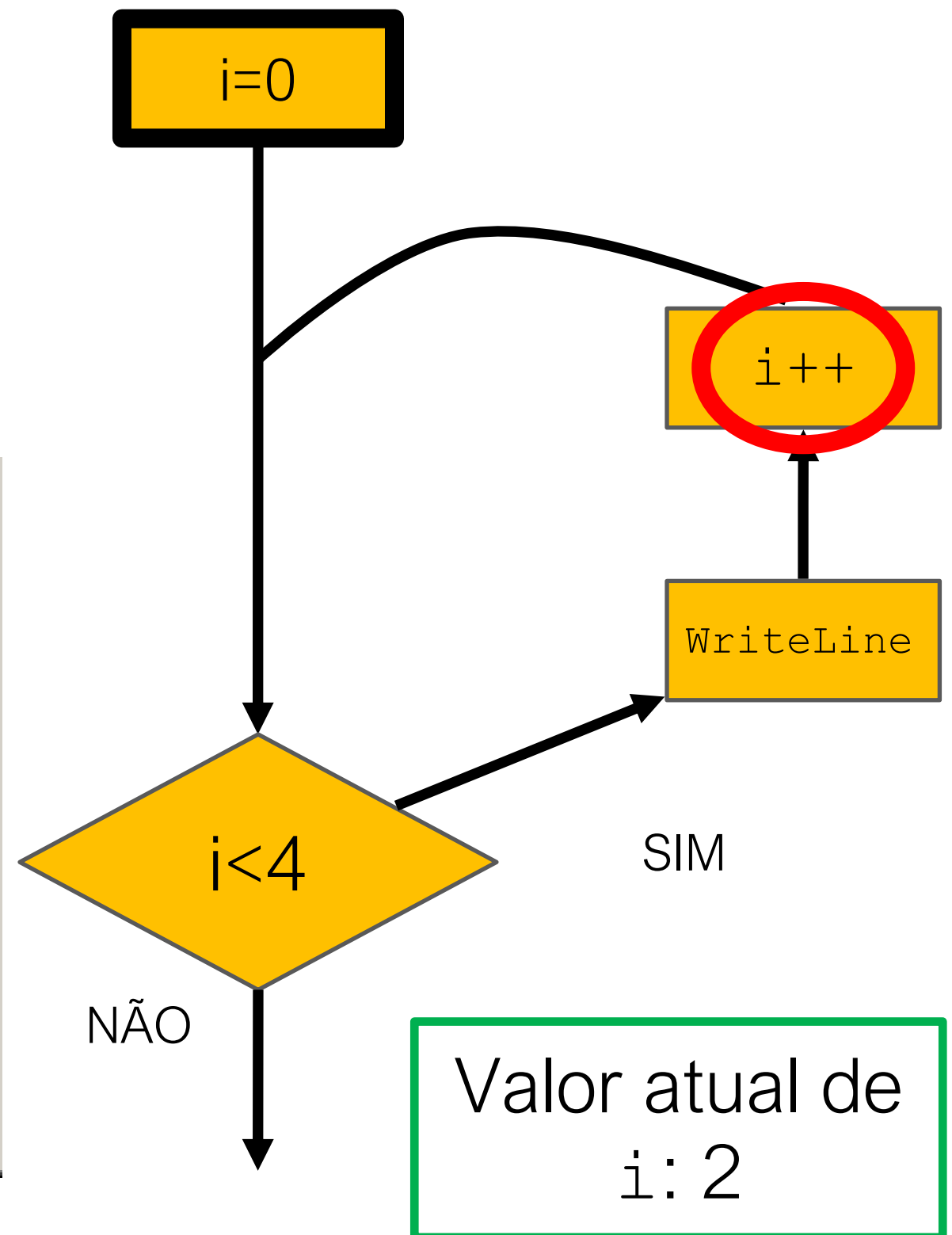
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4, i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



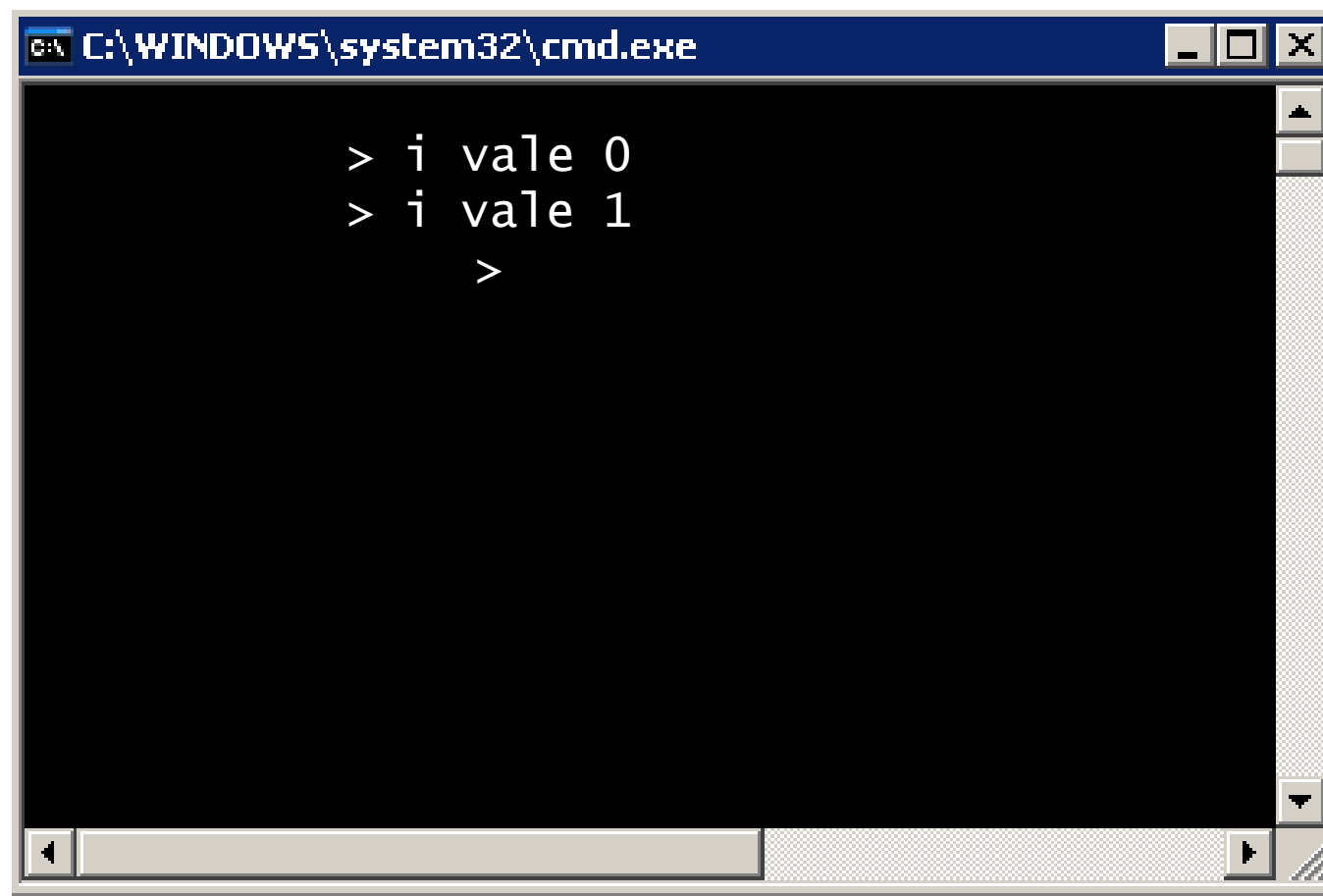
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
>
```



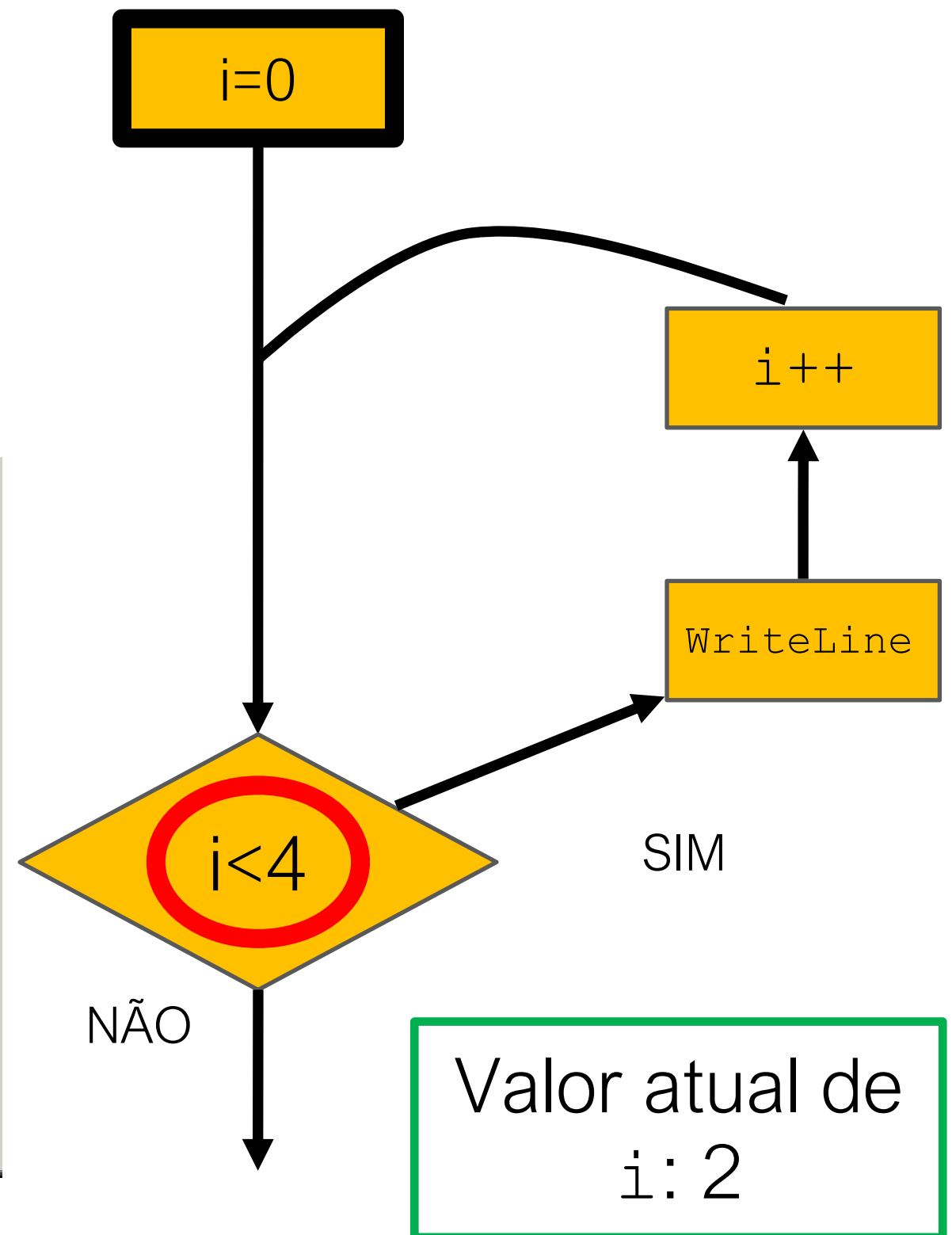
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4 i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



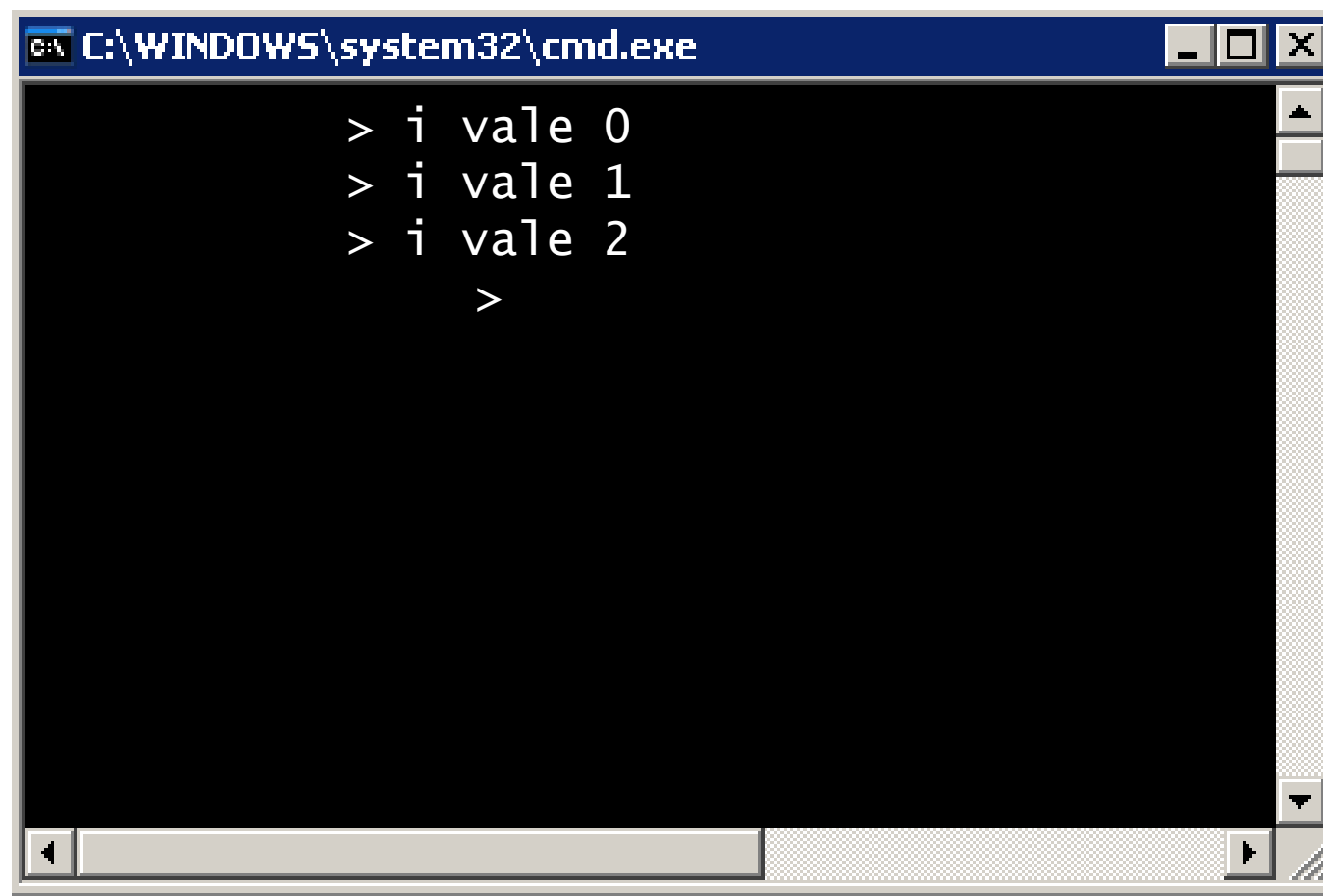
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
>
```



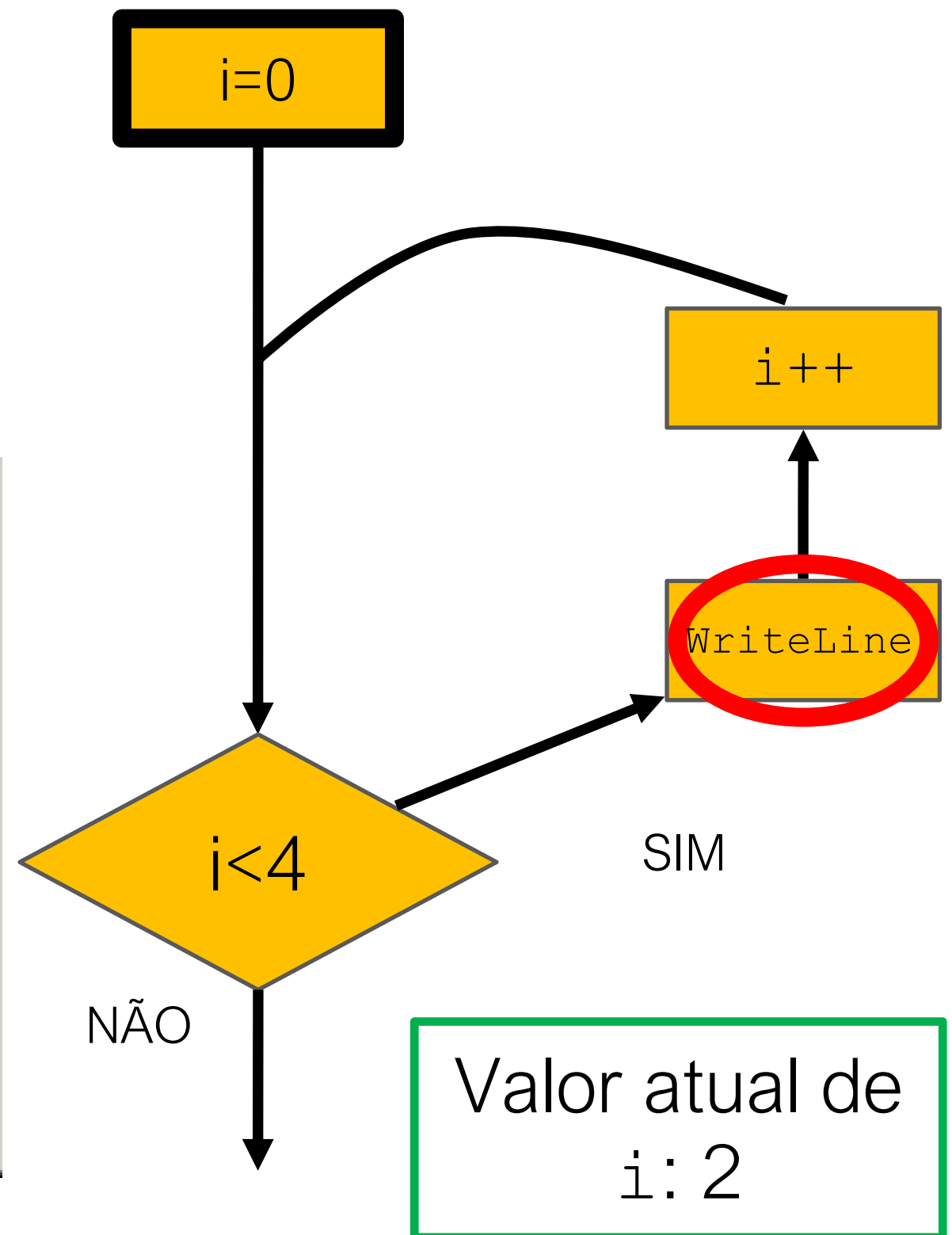
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4; i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



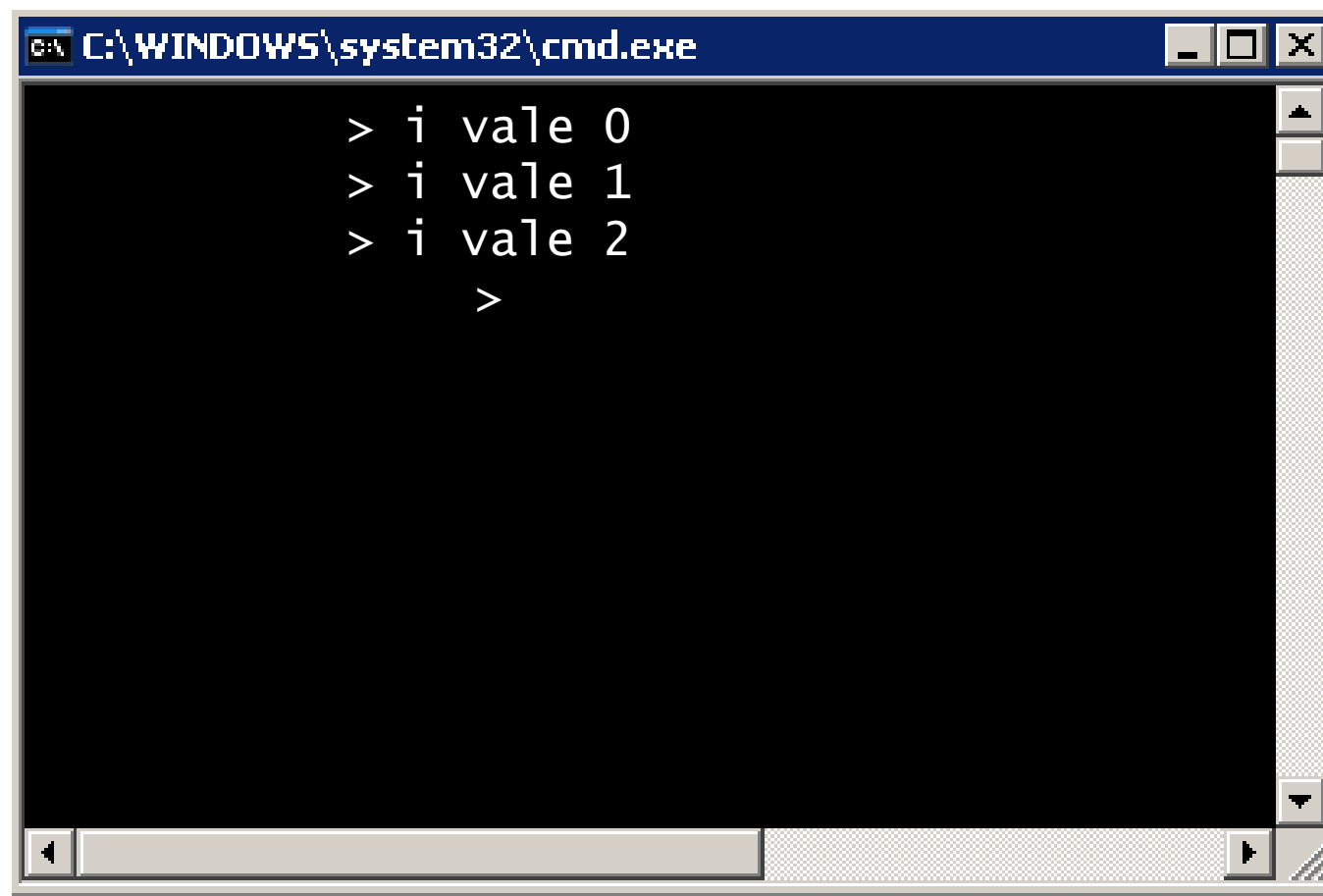
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
> i vale 2  
>
```



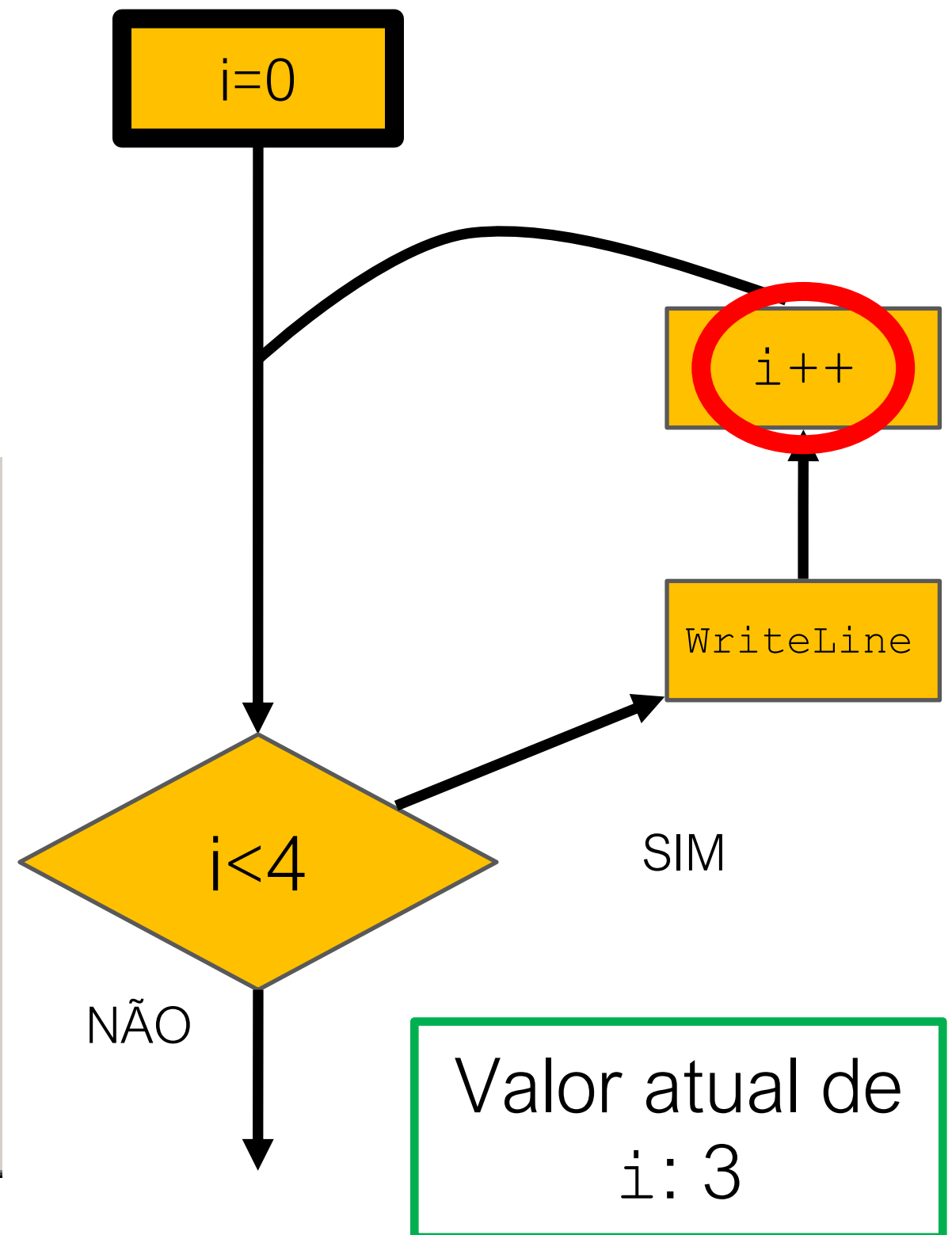
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4, i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



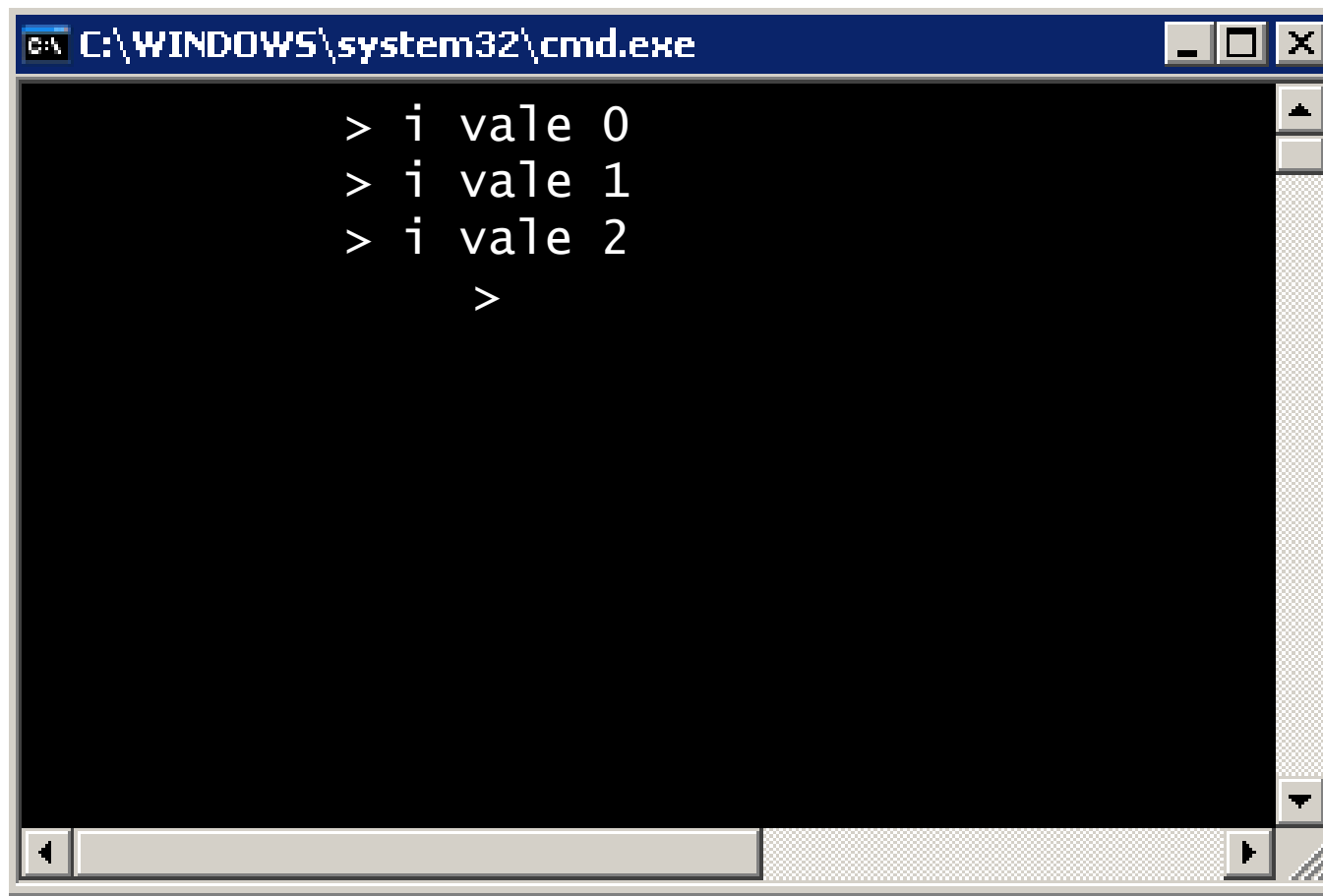
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
> i vale 2  
>
```



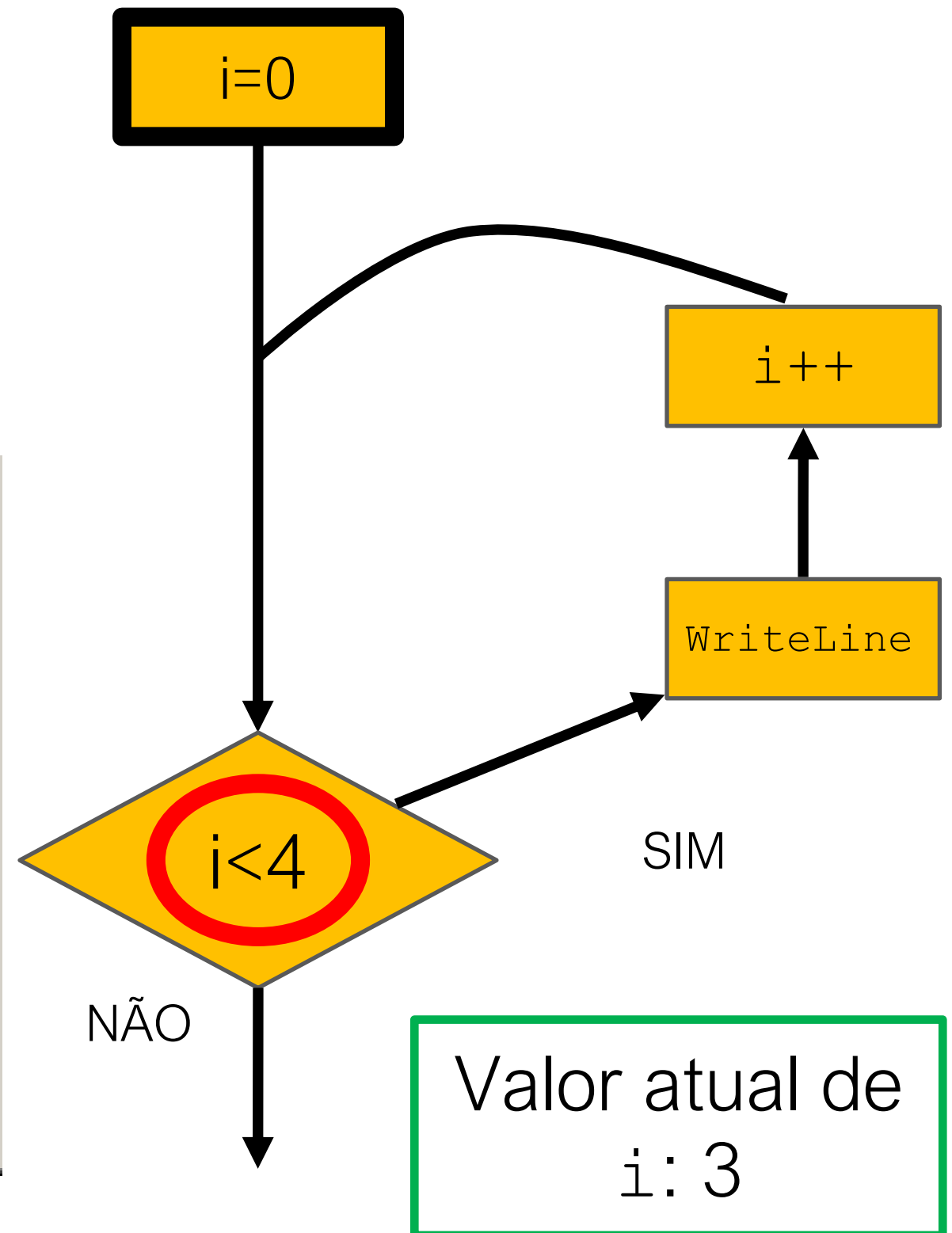
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4 i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



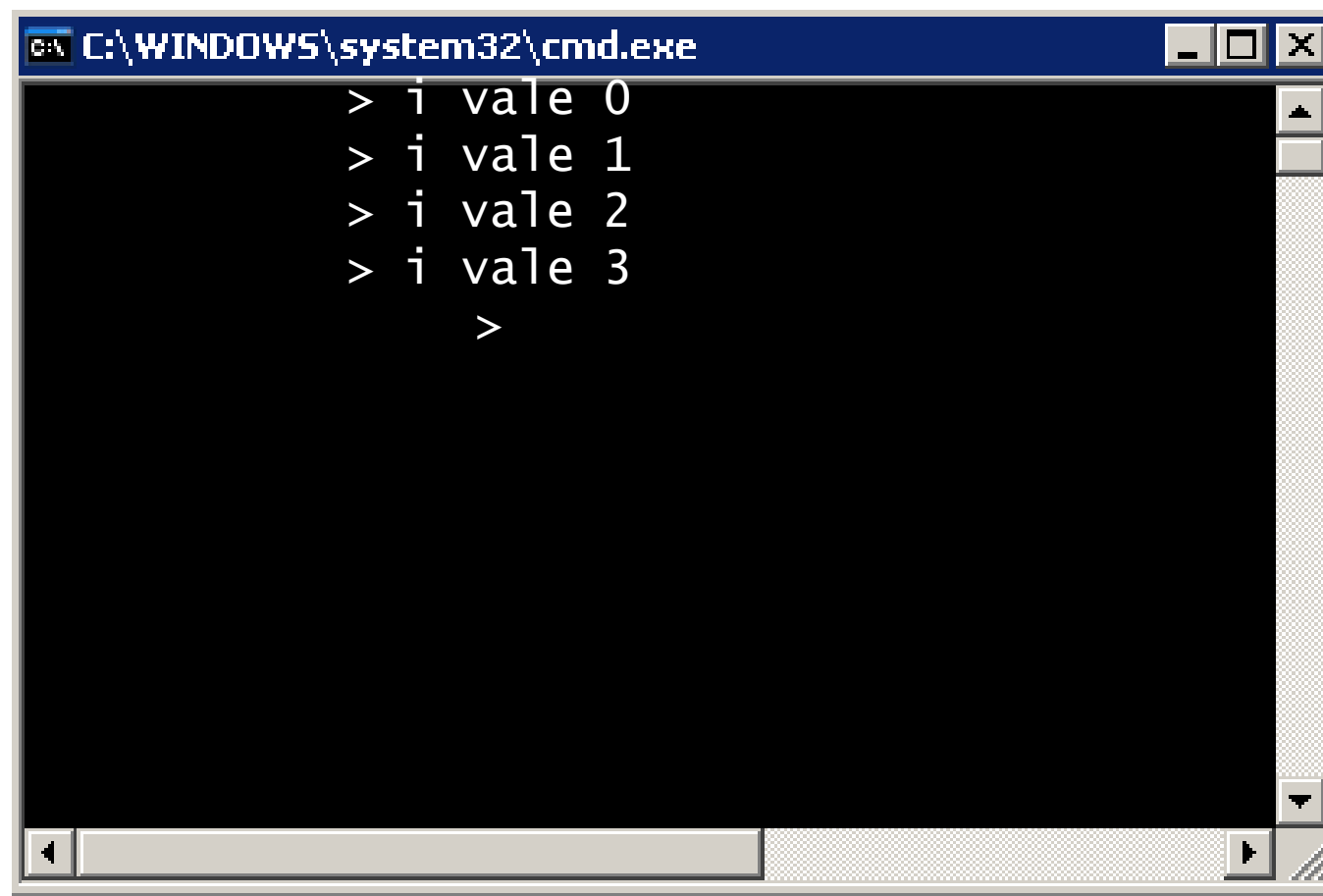
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
> i vale 2  
>
```



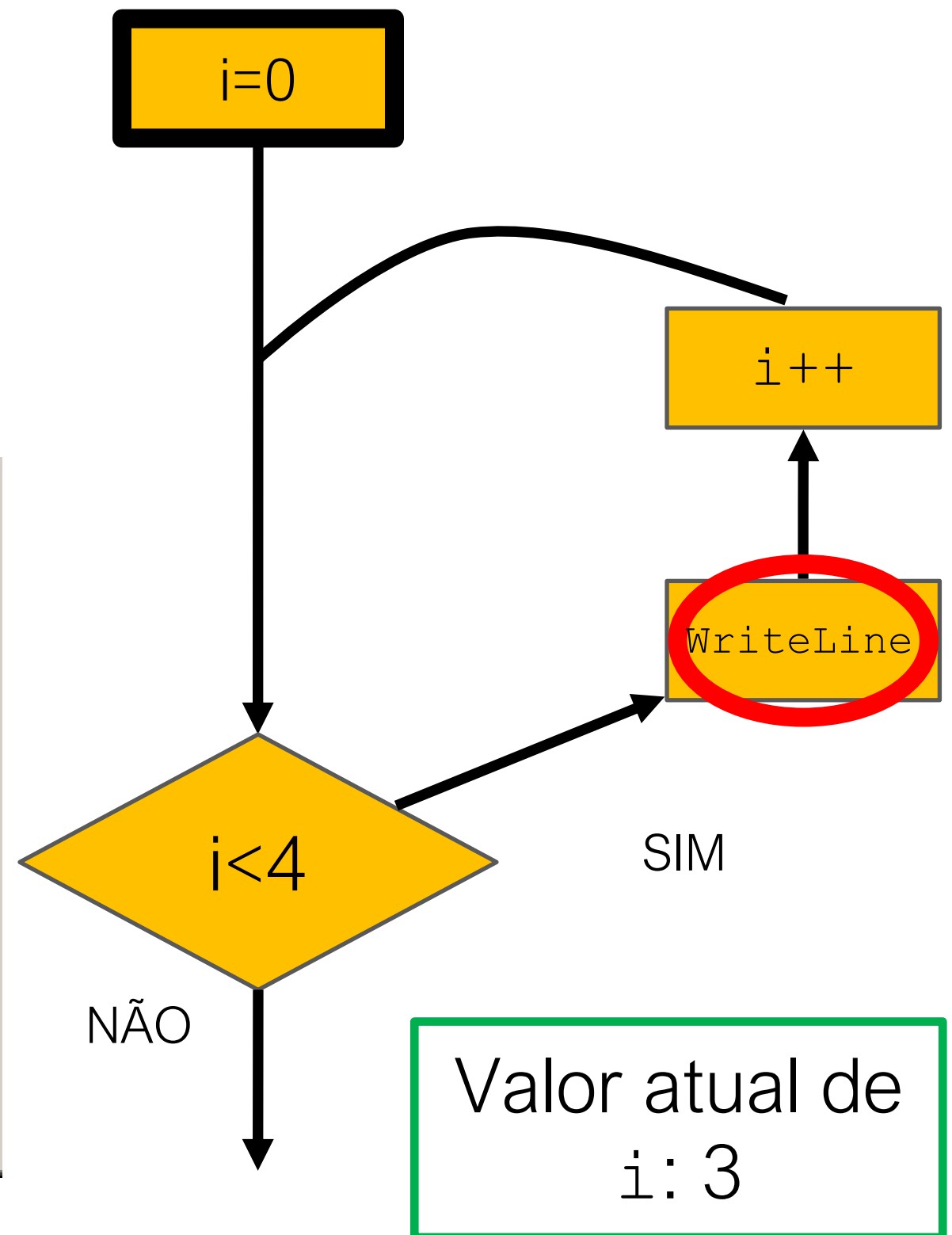
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4; i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



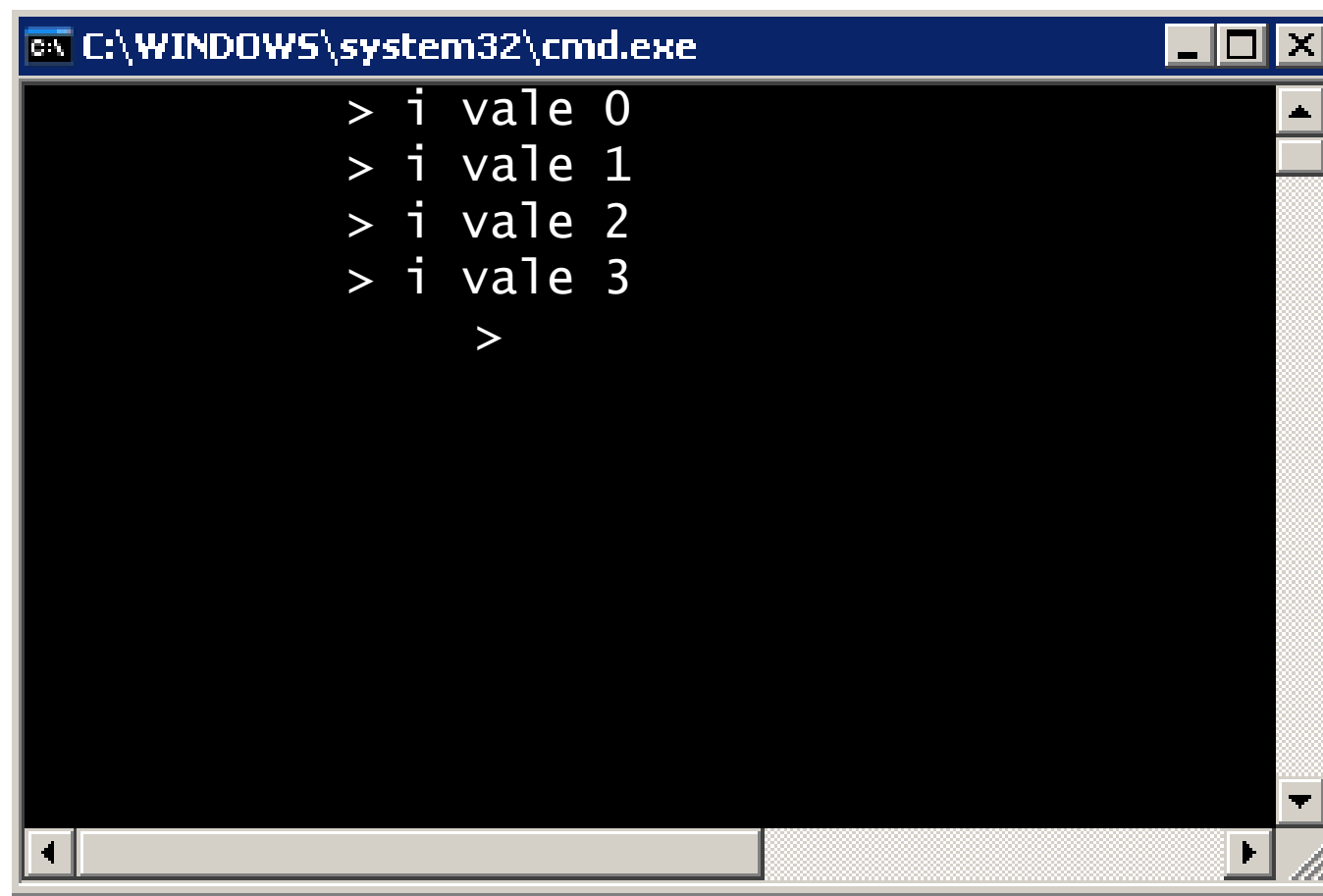
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
> i vale 2  
> i vale 3  
>
```



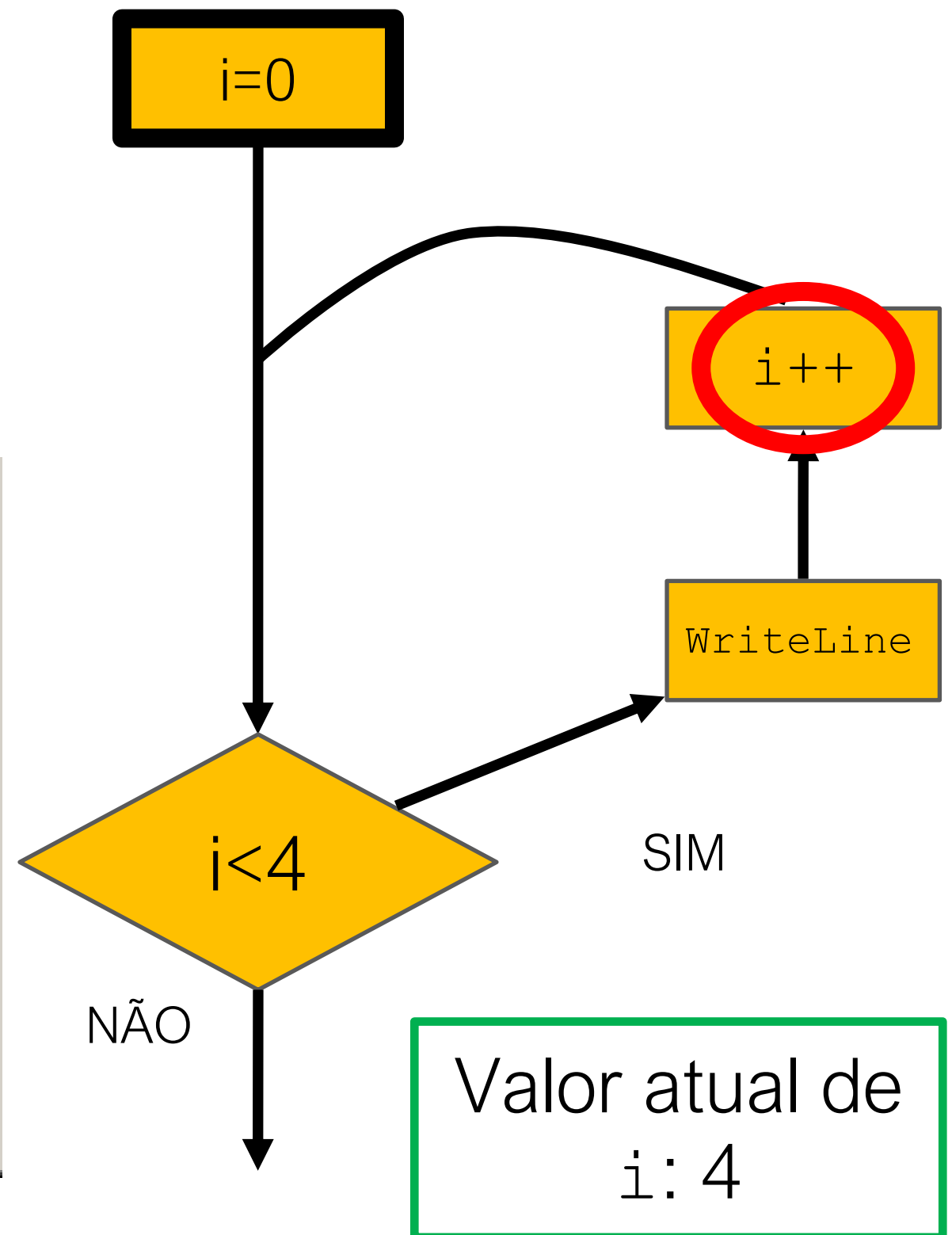
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4, i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



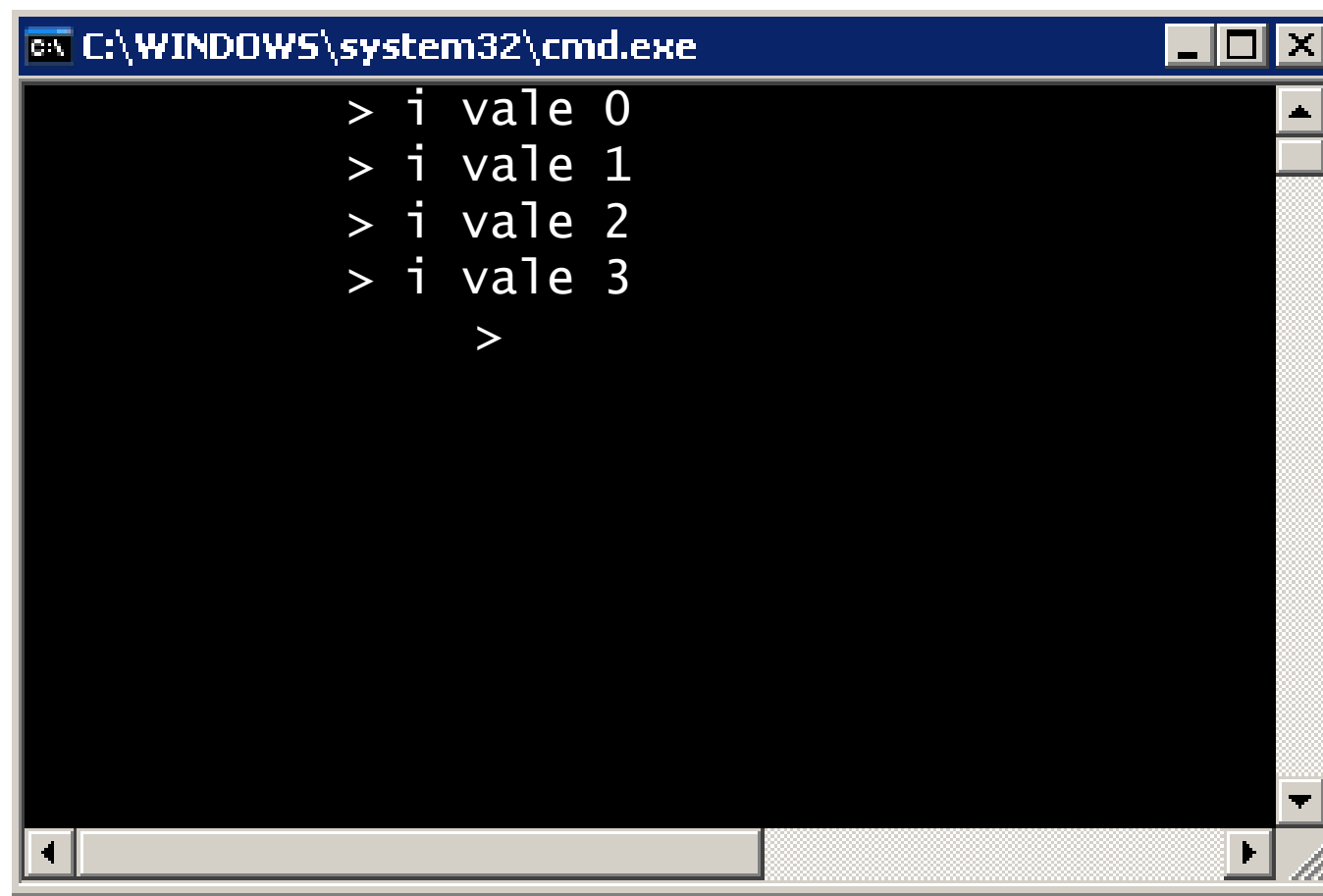
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
> i vale 2  
> i vale 3  
>
```



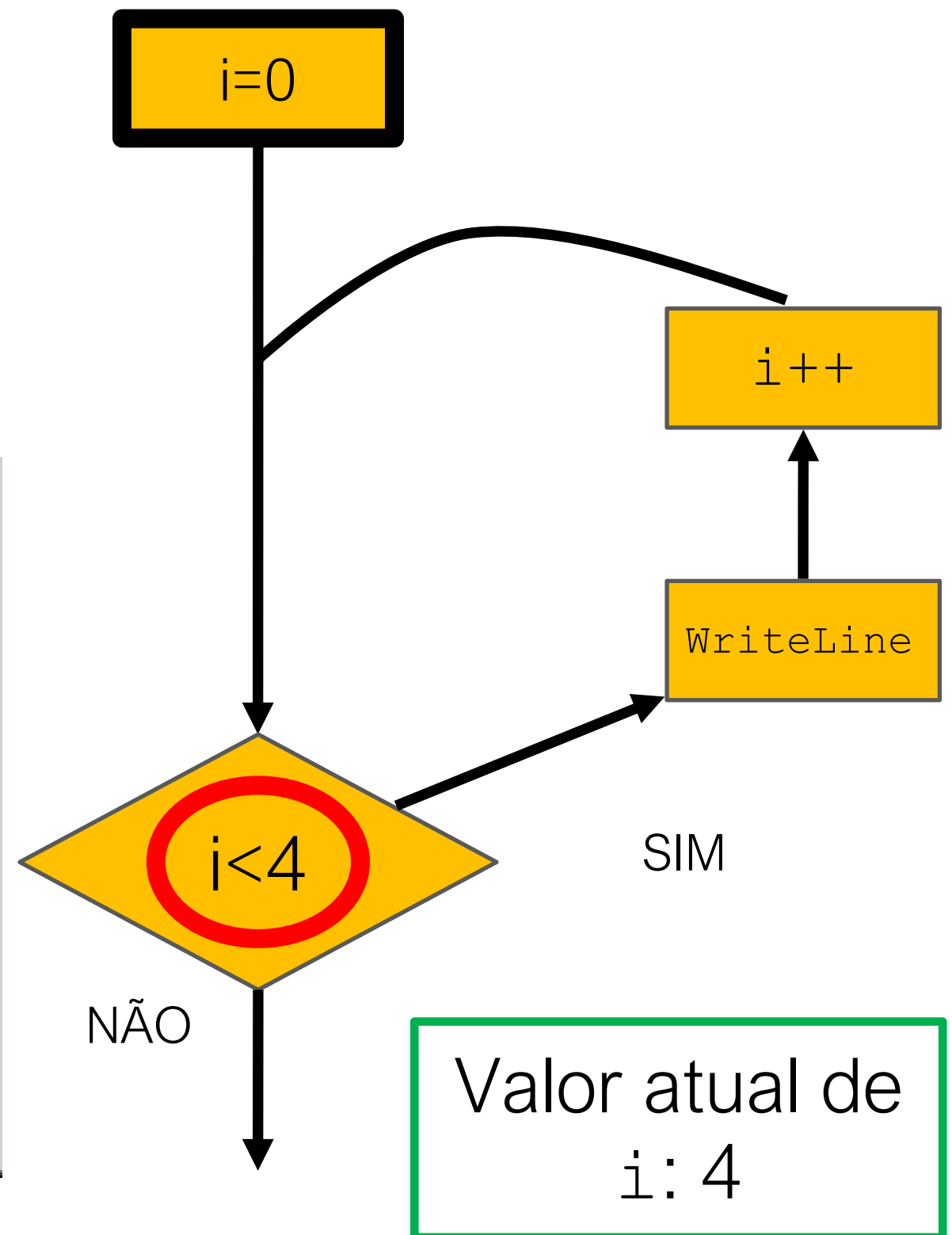
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4 i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



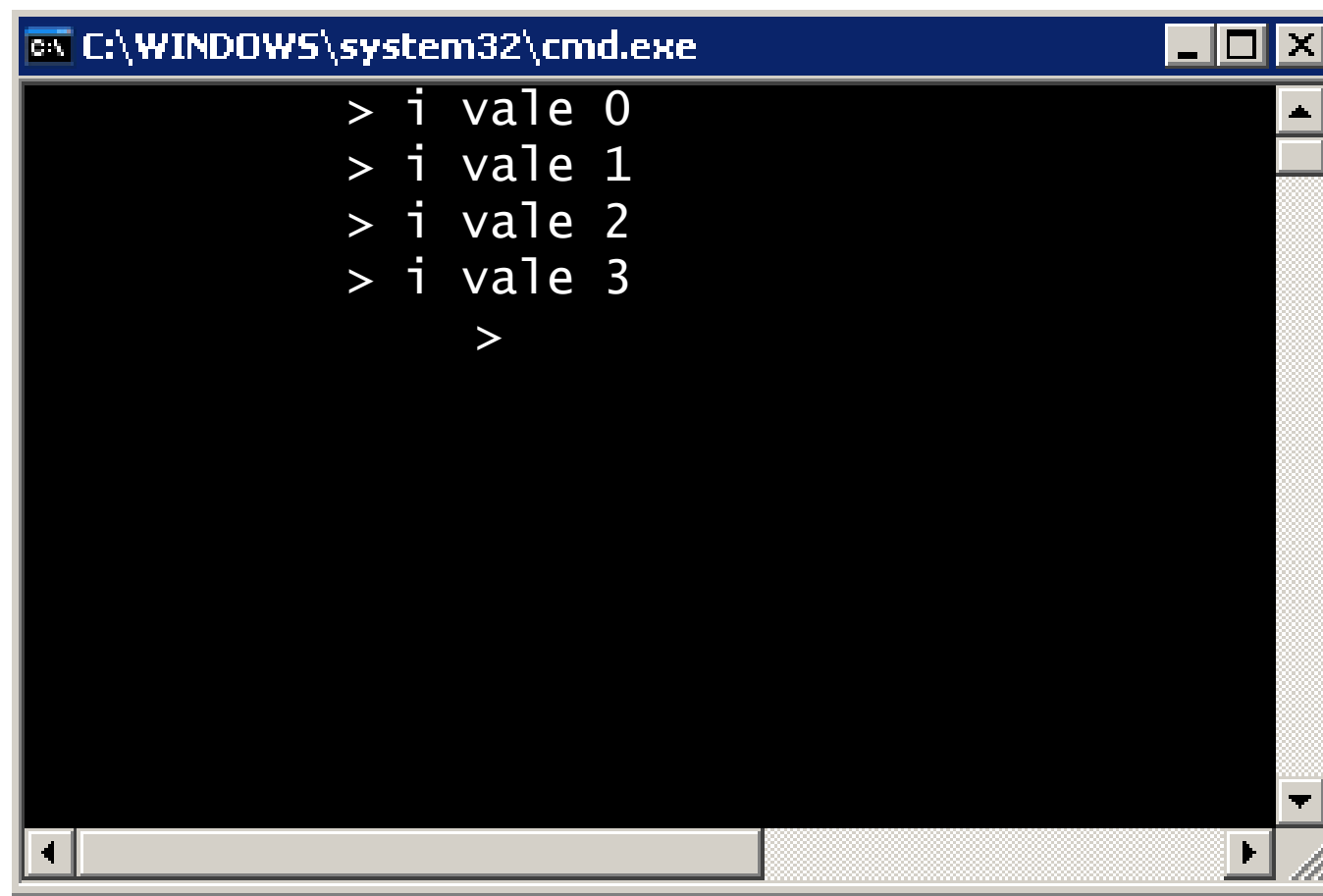
C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
> i vale 2  
> i vale 3  
>
```



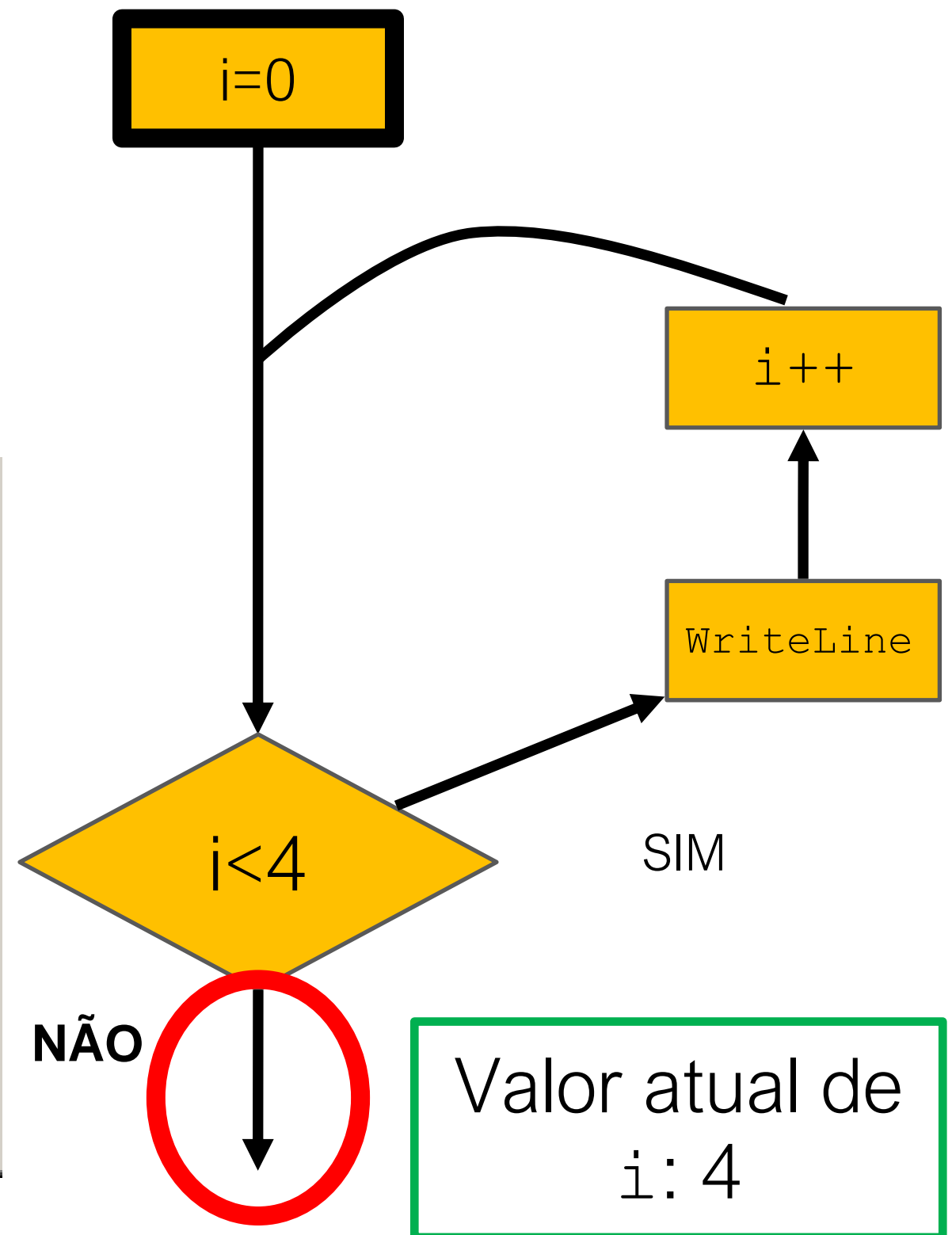
Entendendo o Exemplo 1

```
int i;  
for (i = 0; i < 4 i++)  
{  
    Console.WriteLine("i vale "+i);  
}
```



C:\WINDOWS\system32\cmd.exe

```
> i vale 0  
> i vale 1  
> i vale 2  
> i vale 3  
>
```



Exemplo 2

- Escreva um algoritmo que mostre todos os números de 0 a 10.

Exemplo 3

- Escreva um algoritmo que mostre todos os números de 20 a 5.

Exemplo 4

- Escreva um algoritmo leia pelo teclado os dois valores inteiros chamados inicio e fim, e então mostre todos os números entre inicio e fim.

Exemplo 5

- Escreva um algoritmo leia pelo teclado um valor inteiro chamado x , e então mostre na tela todos os números de 0 a 100, pulando de x em x .

Exemplo 6

- Escreva um algoritmo leia pelo teclado um valor inteiro chamado x , e então mostre na tela todos os números de 0 a 100 que são divisíveis por x .

Exemplo 7

- Escreva um algoritmo leia pelo teclado a quantidade de alunos de uma turma, e então, para cada aluno, leia suas 3 notas do semestre e mostre na tela sua média final, e se está aprovado ou não.

For – controle do laço

- O comando **break** faz com que todo laço de repetição seja finalizado, antes mesmo da condição do for se tornar falsa.
- O comando **continue** faz com que a interação atual do laço de repetição seja finalizada, e então seguir para a próxima interação.

Exemplo 8

- Escreva um algoritmo leia continuamente pelo teclado o valor de uma variável inteira chamada op, e só finalize a execução do programa quando o valor lido para op seja zero.

Exemplo 8 – comando **break**

```
static void Main(string[] args)
{
    int i, op;
    for (i = 0; true; i++)
    {
        Console.WriteLine("Quer continuar? \nDigite 0 para sair!");
        op = int.Parse(Console.ReadLine());
        if (op == 0)
        {
            Console.WriteLine("Obrigado pela presença neste lindo laço de repetição");
            break;
        }
    }
}
```

O comando **break** quebra o laço, indo para a próxima linha após o bloco **for**, finalizando a sua execução

Exemplo 9

- Escreva um algoritmo leia pelo teclado o valor de uma variável inteira chamada opcao, porém, o programa só pode seguir sua execução caso o valor de opcao seja entre 1 e 4. Se o valor digitado não for entre 1 e 4, a variável opcao deverá ser lida novamente pelo teclado.

Exemplo 9 – comando **continue**

```
static void Main(string[] args)
{
    int i, op;
    for (i = 0; true; i++)
    {
        Console.WriteLine("Digite sua opção:");
        op = int.Parse(Console.ReadLine());
        if (op > 4 || op < 1)
        {
            Console.WriteLine("Digite um valor entre 1 e 4!!! \n");
            continue;
        }
        else
        {
            Console.WriteLine("Opção "+op+" escolhida!");
            break;
        }
    }
    switch (op)
    {
        case 1:
            Console.WriteLine("Primeira opção");
            break;
        case 2:
            Console.WriteLine("Segunda opção");
            break;
        case 3:
            Console.WriteLine("Terceira opção");
            break;
        case 4:
            Console.WriteLine("Quarta opção");
            break;
    }
}
```

E comando **continue** manda seguir no laço para a próxima interação direto, ou seja, **vai direto pro incremento/decremento (i++)**, ignorando todo o resto do bloco!

Exemplo 10 – media de notas da turma

```
static void Main(string[] args)
{
    int i, n;
    double media = 0, nota;
    Console.WriteLine("Cálculo da média de notas da turma em uma avaliação \n");
    Console.WriteLine("Quantos alunos realizaram a avaliação? ");
    n = int.Parse(Console.ReadLine());
    for (i = 1; i <= n; i++)
    {
        Console.WriteLine("Digite a nota do aluno "+i);
        nota= int.Parse(Console.ReadLine());
        media += nota;
    }
    media = media / n;
    Console.WriteLine("\nA média final da turma é "+media);
}
```

Exemplo 11 – `for` dentro de um `for`

```
static void Main(string[] args)
{
    int i, j;
    for (i = 0; i < 20; i++)
    {
        Console.WriteLine("i: " + i);
        for (j = 0; j < 10; j++)
        {
            Console.WriteLine("    j: " + j);
        }
    }
}
```


Exemplo 12 – `for` dentro de um `for` dentro de um `for`

```
static void Main(string[] args)
{
    int i, j, k;
    for (i = 0; i < 20; i++)
    {
        Console.WriteLine("i: " + i);
        for (j = 0; j < 10; j++)
        {
            Console.WriteLine("    j: " + j);
            for (k = 0; k < 15; k++)
            {
                Console.WriteLine("        k: " + k);
            }
        }
    }
}
```